IBM

IBM Application Development and Problem Determination Tools V7 for System z

Application Performance Analyzer, Debug Tool Utilities and Advanced Functions, Fault Analyzer, File Export, File Manager, and Workload Simulator

- With features and functionality from the latest versions of each product
- New scenarios and practical advice
- Completely new File Export sections

Amintore de Nardis
Mark Duckworth
Jennifer Nelson
Adrian Simcock

Redbooks

ibm.com/redbooks

International Technical Support Organization

IBM Application Development and Problem Determination Tools V7 for System z

Application Performance Analyzer, Debug Tool Utilities and Advanced Functions, Fault Analyzer, File Export, File Manager, and Workload Simulator

June 2007

SG24-7372-00

Note: Before using this information and the product it supports, read the information in “Notices” on page xvii.

First Edition (June 2007)

This edition applies to:

- Version 7, Release 1, Modification 0 of IBM Application Performance Analyzer for z/OS (5697-N53)
- Version 7, Release 1, Modification 0 of IBM Debug Tool for z/OS (5655-R44)
- Version 7, Release 1, Modification 0 of IBM Debug Tool Utilities & Advanced Functions for z/OS (5655-R45)
- Version 7, Release 1, Modification 0 of IBM Fault Analyzer for z/OS (5655-R46)
- Version 1, Release 2, Modification 0 of IBM File Export for z/OS (5697-I12)
- Version 7, Release 1, Modification 0 of IBM File Manager for z/OS (5655-R47)
- Version 1, Release 1, Modification 0 of IBM Workload Simulator for z/OS (5655-I39)

Contents

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX®
BookManager®
CICS®
CUA®
DB2®
DB2 Universal Database™
DFSMS™
DFSORT™
IBM®
IMS™
Language Environment®
MQSeries®
MVS™
NetView®
OMEGAMON®
OS/390®
Parallel Sysplex®
RACF®
Rational®
Redbooks®
Redbooks (logo) ®
RETAIN®
REXX™
System z™
VisualAge®
VTAM®
WebSphere®
z/OS®
zSeries®

The following terms are trademarks of other companies:

EJB, Java, JVM, J2SE, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Windows NT, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Pentium, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Preface

This IBM® Redbooks® publication introduces the IBM Application Development and Problem Determination Tools V7. The tools covered are:

- IBM Application Performance Analyzer for z/OS® Version 1 Release 1
- IBM Debug Tool for z/OS Version 7 Release 1
- IBM Debug Tool and Advanced Facilities for z/OS Version 7 Release 1
- IBM Fault Analyzer for z/OS Version 7 Release 1
- IBM File Manager for z/OS Version 7 Release 1
- IBM File Export for z/OS Version 1 Release 2
- IBM Workload Simulator for z/OS and OS/390® Version1 Release 1

This comprehensive suite of powerful yet easy-to-use tools helps developers efficiently build, test, and service applications while minimizing errors and increasing cost-effectiveness. The tools allow users to quickly identify and resolve problems that occur in batch, IMS™, DB2®, CICS®, and UNIX® System Services applications.

For each tool, this book provides step-by-step instructions for installation and customization, a detailed discussion of the features and capabilities, and guidance for using the tool in your environment.

Practical scenarios demonstrate how to apply the tools to monitor, manage, debug, and analyze applications in a real-world environment.

The team that wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

Mark Duckworth is a Software Engineer from Seattle, Washington. He has 27 years of experience with a variety of IBM tools and languages on multiple platforms. During his IBM career, he has worked in Language Environment® development, as a member of the Y2K Worldwide Technical Support Center, as an external consultant and trainer, and most recently, as a Technical Support Specialist for Fault Analyzer and File Manager. He holds a BSBA degree in Finance and Economics from the University of Missouri.

Amintore de Nardis joined IBM in 2001. He currently works as a software pre-sales IT specialist on the WebSphere® for z/OS team in Rome, Italy. He has over 20 years of experience in application programming with COBOL, IMS, DB2, CICS, and MQ, along with CICS and DB2 systems programming. During these years, he provided assistance in creating and integrating new applications using IBM methodologies and standards for customers in the Finance, Economics, Robotics, and Insurance industries.

Jennifer Nelson is a product specialist of the AD Tools with Rocket Software in Austin, Texas. She has 10 years experience in the mainframe industry working with DB2 and mainframe storage with large mainframe vendors. She is currently working as a Product Specialist with Rocket Software as a business partner with IBM assisting Technical Sales with potential and existing customers. She holds a BA in Political Science from the University of Texas at Austin.

Adrian Simcock is a Software Engineer working at the IBM Australian Development Laboratory in Perth, Australia. He has 20 years of experience working with CICS and related products for a variety of IBM internal and external customer accounts. He is currently working as a developer for IBM Fault Analyzer. He holds a degree in Electrical and Electronic Engineering from Plymouth University.

From left to right: Adrian Simcock, Jennifer Nelson, Mark Duckworth, and Amintore de Nardis

Thanks to the following people for their contributions to this project:

Eric Chabert, IBM, Software Sales IT Specialist, Montpellier, France

James Conover, IBM, System Programmer, Buffalo, NY

Peter Costigan, IBM, Software Engineer, File Manager, Silicon Valley Laboratory, CA

Kevin Gasiorowski, IBM, Software Engineer, File Manager, Silicon Valley Laboratory, CA

Graham Hannington, Fundi Software, Perth Australia, for the permission to use the 3270 screen capture software used through much of this book

Neil Kenyon, IBM, Software Sales Senior IT Specialist, Portsmouth, UK

Franka Reuter, IBM, Software Engineer, Debug Tool, Silicon Valley Laboratory, CA

Michael Rotter, IBM, Senior Software Engineer, Piscataway, NY

Ivan Shepherd, IBM, Software Sales pan-IOT ISV Migration Specialist, Hursley Park, UK

William Deason, IBM, AD Tools Solution Test Manager, Silicon Valley Laboratory, CA

Lisa Ready, IBM, COBOL and PL/I Project Manager, Silicon Valley Laboratory, CA

Marty Shelton, IBM, AIM PD Tools manager, Silicon Valley Laboratory, CA

Yvonne Lyon, ITSO Editor, International Technical Support Organization, San Jose, CA

Debbie DeCarlo, Event Planner, Social and Culinary Support, San Jose, CA

and

Joe DeCarlo, Manager, Special Projects, International Technical Support Organization, Poughkeepsie, NC via San Jose, CA

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbooks publication dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners, and customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Learn more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our Redbooks® to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an E-mail to:

  redbook@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Part 1

Overview of the problem determination and deployment tools

In this part of the book, we provide an overview of the IBM problem determination and deployment tools.

1

Overview of the IBM problem determination and deployment tools

In this chapter we present an overview of the IBM problem determination tools:

- Application Performance Analyzer for z/OS
- Debug Tool for z/OS and Debug Tool Utilities and Advanced Functions
- Fault Analyzer for z/OS
- File Export for z/OS
- File Manager for z/OS
- Workload Simulator for z/OS and OS/390

1.1 Application Performance Analyzer for z/OS

In an increasingly complex and competitive business environment, the managers, system programmers, application developers, and performance specialists are under pressure to operate their systems at peak level. Adopting on demand business strategies and Implementing Web services (for example, creating more-complex, business-critical applications) increases the risk of missing your service-level commitments. To mitigate this risk, you must carefully consider performance and, ultimately, customer satisfaction as part of your on demand business strategy. Historically, traditional performance tools have not provided the application developer with comprehensive solutions.

Application Performance Analyzer is an easy-to-use and a non-intrusive performance measurement and analysis system that helps to resolve z/OS and OS/390 job performance issues, whether those jobs are applications, subsystems, or tasks. It provides performance statistics on any job that you want to monitor. These statistics can be the current system data, data for scheduled future jobs or data collected over a certain time period. By using Application Performance Analyzer, you can immediately focus your activities on tuning specific areas of an application, thereby improving productivity and meeting the challenges demanded by e-business applications.

1.1.1 Version 7.1 enhancements

- Threshold monitoring improvements:

  There are a number of new enhancements and features for setting up the automatic monitoring of jobs when selected threshold criteria are met. You can now automatically start APA when EXCPs, CPU time, or job duration reach specified values. As soon as a job is started, APA checks each second to see if these criteria are met.

- Java™ support for J2SE™:
  - CICS
  - IMS
  - DB2 stored procedures
- Java support for JZOS:
  - RACF® authorization required to spawn
  - Specific invocation options required
- DB2 enhancements:
  - Host variables are now displayed
- C/C++ Support available (only for TSO address spaces):
  - All subsystems are supported
  - Supports CICS TS 3.2
- API for invocation from within a program:
  - Sample programs are provided for C, C++, COBOL, PL/I and Assembler
  - No longer requires JCL - but it is still available if desired.

1.1.2 Use of Application Performance Analyzer

The next sections discuss when an application programmer can use Application Performance Analyzer.

During application development

The most straightforward use of the profiler is in situations where your code is not meeting its intended performance targets. You can use a profiler to determine which routines are used the most, and which would benefit from an examination of performance tuning.

The next obvious use is for determining the location of a supposedly infinite loop or performance bottleneck. You can use a profiler such as Application Performance Analyzer to actively monitor the application while it is running and *drill down* through the list of modules to the assembler instruction or the COBOL/PLI/C/C++ source line.

There is yet another use for a profiler during development. While programming an application, a programmer often uses code developed by another developer. This can leave the programmer dangerously lacking in information about the performance of the code. If two implementations are possible, you can use a profiler to determine which one can be expected to perform best. In this case, you might want to compare multiple reports.

During application test

Most application development projects have performance targets. A regimen of performance test scenarios is especially common. You can use the profiler tool to provide more information, for example, suppose that an application runs 25% slower than expected. The tool could be used to determine which part of the application has degraded or if it was an overall degradation. Comparative reports could also be used to monitor changes in the application's performance.

During application service

If a customer reports a problem with performance, a profiler is the perfect tool to help the service professional narrow down the cause and pinpoint the solution.

During application maintenance

It is a fact of modern development that the person who maintains a given application is not necessarily the person who wrote it. It is also true that the performance characteristics of an application can change during its service lifetime.

A profiler such as Application Performance Analyzer can help stop the performance loss that often occurs during maintenance. An application programmer can use this tool to determine where the hot spots or heavy loops are in the application. The information about where the performance bottleneck is located can then be taken into consideration when putting in a fix. For example, the developer could learn where the heavy loop is, and then do whatever is possible to avoid adding code to that loop when fixing the problem.

Application Performance Analyzer could also be used after a tentative fix is made that seems to degrade performance, to help find the cause of that degradation.

Other uses

These are some other uses for Application Performance Analyzer:

- To provide information to a software provider when you run into a performance problem with their code
- To help system programmers determine what parts of a given application are causing a system performance problem

1.1.3 What Application Performance Analyzer can do

Application Performance Analyzer is a performance measurement and analysis tool that:

- Optimizes the performance of existing application resources
- Provides maximum flexibility, with support for Assembler, C/C++, COBOL, PL/I, DB2, CICS, IMS, and Web Sphere MQ technologies. APA also provides support for Stored Procedures written in JAVA and for JAVA running in a CICS region.
- Eliminates excessive I/O activity and CPU time to increase response times
- Improves response time of online transactions and batch turnaround times
- Identifies code bottlenecks during initial testing and isolates performance problems in existing applications

Application Performance Analyzer collects samples from the monitored address space and analyzes the system or resource application in the following categories:

- An at-a-glance summary of various aspects of the measurement data to help you to choose (using *hot spot*) which other reports to concentrate on
- Total address space utilization of all modules in the address space
- CSECT utilization within each load module
- Instruction or statement utilization within each CSECT
- Assembler, COBOL, C/C++ and PL/I statement usage within each module.
- Direct access storage device (DASD statistics)
- Coupling facility usage during the observation session
- JAVA support for J2SE and JZOS
- Processor usage

If the IMS measurement option has been selected, Application Performance Analyzer allows you to display detailed information about:

- An at-a-glance summary of various aspects of the measurement data to help you to choose (using *hot spot*) which other reports to concentrate on.
- IMS environment parameters including the region type and its status, the PSB, and the transaction code.
- DL/I calls in the target address space with all of the call parameters, SSAs.
- If the IMS+ measurement option has been selected, the precise number of DL/I calls is counted, as well as the exact DL/I service time and CPU time by DL/I call.

Notes: Many of the IMS reports require that the IMS+ extractor be used.

Running measurements with the IMS+ data extractor turned on causes each IMS call to be intercepted to collect additional data. This could have a small impact on the performance of the target address space. Care should be taken when using this feature with other products that also intercept IMS calls, as unpredictable results might occur. Your installer might have chosen to limit access to this feature.

If the CICS measurement option has been activated, Application Performance Analyzer allows you to display detailed information about two distinct types of data that it gathers:

- **Session statistics:** This data is a summary of how much CICS-related activity occurred in the region during the observation session.

- **Transaction measurement data:** There is one sample record created for each in-flight transaction, allowing you to drill down to the statement level and display detailed information about:
  - CICS, DB2, DLI, and MQ Calls in a transaction
  - CICS transaction executions
  - CICS service executing on behalf of the application
  - CICS calls in a unit of work
- Users can now select observations in CICS by termID selection which provides another powerful way to isolate transactions to monitor in CICS.

If the DB2 option is activated, Application Performance Analyzer analyzes the following DB2 categories for any application (batch or online) using DB2:

- An at-a-glance summary of various aspects of the measurement data to help you to choose (using hot spot) which other reports to concentrate on.
- SQL activity by statement or DBRM or plan.
- SQL statement full text.
- SQL statement attributes.
- SQL explain analysis.
- Threads analysis.
- View DB2 host variables
- CPU by plan or stored procedure.

f the DB2+ measurement feature has been selected, Application Performance Analyzer collects the data required to report exact SQL request counts, SQL CPU time, and SQL service time.

Note: Many of the DB2 reports require that the DB2+ extractor be used.

Running measurements with the DB2+ data extractor turned on causes each DB2 call to be intercepted to collect additional data. This might have a small impact on the performance of the target address space. Care should be taken when using this feature with other products that also intercept DB2 calls, as unpredictable results might occur. Your installer might have chosen to limit access to this feature.

If the MQSeries® measurement feature has been selected, Application Performance Analyzer allows you display detailed information about:

- MQSeries call (dynamic and static) in batch, IMS, and CICS programs
- CPU and wait time spent in MQSeries interface call by queue, request, or transaction

1.1.4 The major functions of Application Performance Analyzer

Application Performance Analyzer for z/OS provides analysis information throughout the cycle of application design, development, and maintenance. It evaluates application prototypes in the design phase, reviews the impact of increased data volume or changes in business requirements on performance, and generates historical data and reports to analyze performance trends and evaluate program changes for cost effectiveness.

A high-level overview of the Application Performance Analyzer's functions follows.

Real-time monitor

This function provides dynamic displays of job performance. You can check job execution online without having to first record the data and then route it to display or print.

Scheduled measurement

When you do not know exactly when some jobs will begin executing, you can set up scheduled measurement entries. In the scheduled measurement entries, you specify a start time for Application Performance Analyzer to watch for the job to become active, how many times this measurement has to be done (for example, every Friday for 10 weeks), or how many times the job must run before the measurement begins. The scheduled measurement entries allow you to monitor jobs that run at night or at other times when you are not available to initiate a monitoring request online.

Comparison of reports

As a companion function to scheduled measurements, Application Performance Analyzer now permits comparison of observation reports to show the relevant differences.

Analysis report

This function lets you view data collected during a previous analysis session. The data, collected over a period of time, resembles the information available in real-time monitor analysis.

Statistics and storage analysis reports

These reports provide a general overview of the measurement, but also information about:

- Each of the load modules for which activity was measured during the observation session (including name, location, address, count, size, attributes, and so on)
- Each of the TCBs that were active at any time during the observation session (including name, address, CPU type, and so on)
- Timeline memory and dataspace usage to illustrate any progressive resource usage trends

CPU analysis reports

These reports provide information about measured CPU consumption, attributing CPU consumption by:

- Categories (application code, system/OS services, SQL processing, DL/I call processing, or data management processing)
- Load modules
- Procedure within programs
- Task Control Blocks (TCBs)
- Code slice (range of storage addresses containing executable object code)
- Chronological time intervals

Wait time analysis reports

These reports provide information about potential or existing bottlenecks that might impact job performance.

Data set analysis reports

These reports display I/O allocations for which activity was measured for device (tape or DASD), DD cards, Data Set Name (DSName), and for each file as to how I/O activity was distributed over the measurement interval.

These reports display VSAM data set information and information about the local shared resource pools.

The reports also examine delays resulting from waits during DASD I/O operation, data set attributes, and VSAM statistics.

Coupling facility analysis reports

These reports provide the information that you must have for coupling facility performance analysis and tuning. In addition to a coupling facility report line, you have a breakdown by structure name within the facility name to display the number of requests (synchronous, asynchronous, and queued) and the average and total service time per operation (synchronous, asynchronous, queued, and delays) for operation.

CICS performance analysis reports

Application Performance Analyzer understands information associated with CICS Transaction Server regions. You can monitor a CICS region as a whole or display detailed information about a transaction identification (TRANSID) and task identification (TASKID) that is running in the region.

IMS analysis reports

You can use Application Performance Analyzer to monitor the activity in IMS-dependent regions, including batch, Batch Message Processing (BMP), Message Processing Program (MPP), and Fast Path (FP) dependent regions. You can monitor an IMS region as a whole or select a particular message or batch program to analyze in detail either at PSB, transaction, or DL/I call level.

DB2 analysis reports

These reports provide all of the information that you must have for DB2 performance analysis and tuning. In addition to CPU utilization by plan, DBRM, or SQL statement, you see the EXPLAIN data and object analysis for tables and indexes. The same information is reported for each stored procedure measured during the sampling interval.

Application Performance Analyzer also provides information for each DB2 thread observed.

MQSeries analysis reports

These reports provide the ability to observe, sample, and report on MQSeries interface calls (both dynamic and static) in batch, IMS, and CICS programs. They show the CPU and wait time spent in MQSeries interface calls and attribute the time spent to a particular MQSeries call interface.

Reports

Printed reports are useful for comparing job performance from a daily, weekly, or monthly perspective. You can produce Application Performance Analyzer performance analysis reports suitable for printing in a PDF format.

You can also use the EXPORT and IMPORT functions to share and analyze reports in several sites where Application Performance Analyzer is installed.

1.2 Debug Tool for z/OS and Debug Tool Utilities and Advanced Functions

Debug Tool helps the developer to test programs and examine, monitor, and control the execution of programs written in assembler, C/C++, COBOL, or PL/I on a z/OS or OS/390 system.

Applications can also include other languages. For such applications, Debug Tool provides a disassembly view that lets the developer debug, at the machine code level, the corresponding portions of applications. Of course, in the disassembly view, debugging capabilities are limited.

Debug Tool works in the following environments:

- DB2
- CICS
- TSO
- JES batch
- UNIX System Services in remote debug mode, or full-screen mode through a VTAM® terminal
- IMS

To use Debug Tool, an application must be compiled with the TEST compile-time option, link-edited, and then run with the TEST runtime option.

Debug Tool Utilities and Advanced Functions, which includes Debug Tool, provides a complete solution that enables you to increase debugging efficiencies and to reduce application development life-cycle times.

1.2.1 What Debug Tool can do

In the following sections, we present an overview of how Debug Tool can assist you.

Source-level debugging

Monitor source code for C/C++, COBOL and PL/I and Assembler. Compiler listings are an expanded source listing that, in addition to program source, contain copy book source for COBOL or include file source for PL/I. Non-LE (Language Environment) programs are supported at the assembler (and disassembler) statement level only.

Mixed-language applications debugging

Debug Tool supports the debugging of mixed-language applications within the same session, and recognizes when the current programming language changes (see the following section). Language modules not supported by Debug Tool are tolerated, but no debugging support is provided for these modules.

Dynamic patching

For each supported programming language, there is a set of interpreted commands that you can use to specify actions to be taken. These commands are subsets of the programming languages, so they are easy to learn, and allow you to make adjustments to your application while you debug it.

You can use the commands to alter the value of variables and structures, and to control the flow of an application. For example, a programmer can declare a new variable and use the variable to patch a program as it executes.

Version 7.1 enhancements

With Version 7.1, you can now view source code prior to loading the compile unit. You can also set breakpoints before debugging begins.

Debug Tool now is able to display variable values in columnar format. This includes members of an array or structure.

It provides new commands:

- Set Monitor Column, to direct Debug Tool to format the monitor window display in column format.
- Set Monitor Wrap, to direct Debug Tool to display the value of a monitored or automonitored variable either wrapped in the monitor window or on a single scrollable line.
- Set Monitor Datatype, to direct Debug Tool to display the data type of a variable for monitored and automonitor variables.

The Coverage Utility Annotated Listing report is updated to be HTML showing colored lines, which indicates the statements not executed as well as recomputed statistics based on the annotations in the listing rather than the raw data.

A new option is added to the DTU & AF panel to be able to invoke File Manager's base function.

Dynamic breakpoint setting

You can set breakpoints in an application program, monitor variables for changes, and watch for specified exceptions and conditions during program execution. For example, you can cause an application to break when a specific variable or location in storage is changed. You can set, change, and remove breakpoints when going through the application. It is not necessary to know where to break before the start.

Single-step debugging

To focus on a problem area, you can step line-by-line through the execution of an application. For example, when an application stops for a breakpoint, you can examine each line that follows. Single-step debugging, along with the ability to set dynamic breakpoints, allows you to monitor, interrupt, and continue through the flow of the program to identify errors easily.

Program frequency counting

Debug Tool counts how many times a statement or verb has been processed in an application program. This allows you to verify the coverage of code paths.

Program information gathering

Debug Tool can display program and environment information enabling you to display, monitor, and alter program variables or storage in real time. You can also check how the application was compiled, and look at its structure.

Session logging

Each debug session can be recorded in a log file for reviewing, editing, or replaying. This allows you to replay the actions taken in a session to pinpoint errors in an application.

General testing

You can also use Debug Tool as a test tool. By using the session logging feature as code is being debugged, you can save the results of the session for use as input to a future Debug Tool session. As changes are made to the code, you can use the saved log file as input to Debug Tool in order to verify that no unexpected behavior occurs as a result of these changes. Session logging allows you to create suites of regression test cases that you can use to minimize the number of new bugs introduced during the normal application development process.

1.2.2 What Debug Tool provides

You can start Debug Tool in the following ways:

- The application can start Debug Tool when it terminates normally, abends, or generates errors or conditions above a chosen severity, if the pertinent suboptions are specified on the TEST runtime option.
- The application can start Debug Tool directly using a library service call such as CEETEST, PLITEST, or the ctest() function.
- For CICS applications, you can use Debug Tool in one of the following ways:
  - **Single-terminal mode:** A single 3270 session for Debug Tool and the application. When the application is running Debug Tool is hidden, and vice versa.
  - **Dual-terminal mode:** One 3270 session is used to display the application, and the other is used for Debug Tool.

Debug Tool enables you to perform interactive or batch debugging:

- Using the full-screen interface, you can interactively debug an application as it runs (including batch applications).
- You can also debug batch applications with Debug Tool in batch mode, using a predefined command file.

The full-screen interface of Debug Tool is divided into three windows:

Monitor window	Displays status of items chosen to be monitored, such as variables, registers, programs, the execution environment, and Debug Tool settings. For example, this window is used to watch how the content of variables changes during program execution.
Source window	Displays the program source, with the current statement highlighted. The prefix area at the left of this window is used to enter commands to set, display, and remove breakpoints.
Log window	Records and displays user interactions with Debug Tool and, optionally, shows program output. This window contains the same information as the log file.

Debug Tool has an extensive set of commands to assist with the debug process.

1.2.3 Debug Tool Utilities and Advanced Functions

Debug Tool Utilities and Advanced Functions adds tools to help you perform the following tasks:

- Manage and use setup files that contain file allocation statements, runtime options, program parameters, and the name of your program.

- Prepare high-level language programs for debugging by helping compile, create debug information, and link.
- Prepare assembler programs for debugging by helping assemble, create debug information and link.
- Conduct analysis on test cases to determine how thoroughly test cases are exercised (also called code coverage). The Coverage Utility's Annotated Listing report is updated to be an HTML version that contains colored lines indicating statements not executed and recomputed statistics based on the annotations in the listing instead of the raw coverage data.
- Start and run a program in the foreground or batch by storing and using setup information. Setup information can be the runtime parameters, libraries, and names of input and output data sets.
- For IMS Version 8, browse and edit the Language Environment runtime parameters table.
- Create a batch job for a private IMS message region with customized load libraries and region attributes.
- Analyze load modules or program objects to determine the language translator (compiler or assembler) that was used to generate them.
- Convert old COBOL source code and copybooks to new versions of COBOL by using the Convert and Compile option.
- Invoke File Manager (base function only) directly.

Debug Tool Utilities and Advanced Functions enhances the base set of Debug Tool commands by adding more than 15 additional commands, such as **`allocate`**, **`free`**, **`query automonitor`**, and others.

1.2.4 Debug Tool interfaces

You can interact with Debug Tool in the following ways:

- Full-screen mode:

  An interactive full-screen interface on a 3270 device, with debugging information displayed in three windows.

- Remote mode:

  A debugger on your Windows® work station interacts, through a graphical user interface, with Debug Tool running on a z/OS system.

- Batch mode:

  You specify a predefined series of Debug Tool commands to be performed on a running batch application. No user interaction is possible. The results of the debug session are saved to a log for later review.

DTU's primary panel now has a selection item to invoke File Manager base functions directly.

1.3 Fault Analyzer for z/OS

In today's fast-paced environment, application developers have to work quickly, with as few errors as possible, to keep up with increasing business demands.

The slightest application or system failure can jeopardize goals crucial to business success. To meet these challenges, users require greater visibility into application events to help analyze and resolve problems before they can pose a serious threat to business objectives.

IBM Fault Analyzer Version 7.1 is a robust problem determination tool that helps ascertain why applications fail.

By gathering information about an application and its environment at the time of failure, Fault Analyzer software can help users repair failures quickly, develop and test new and existing applications more productively, and reduce costs along the way. Fault Analyzer also analyzes IBM CICS system failures, and can help analyze IBM WebSphere Application Server for z/OS system failures.

1.3.1 Locating application failures quickly to reduce development costs

When an application abend occurs, Fault Analyzer captures and analyzes real-time information about the application and its environment, then generates an analysis report detailing the cause of the failure.

The report describes the failure in terms of the application code, so developers no longer lose time reviewing cumbersome, low-level system error messages. Fault Analyzer allows you to choose a report format to locate the information more easily.

Each application abend is recorded by Fault Analyzer in a fault-history file by job name, failure code, and other details, along with the analysis report and storage pages referenced during the analysis. This information can later be retrieved to reanalyze the failure, helping to speed future application development.

You can logically group fault-history files, and restrict each user's access to particular groups. You can also transfer fault records between history files for analysis on remote systems.

Designed for ease of use, Fault Analyzer includes soft-copy versions of selected manuals from the IBM z/OS Online Library. It extracts message and failure code descriptions from these manuals and inserts them into the analysis report where applicable, freeing your developers from researching the explanation of message and failure details. You can also provide your own descriptions for messages (including those issued by your own applications).

1.3.2 Maximizing development assets

Fault Analyzer starts only after an application abend occurs, hence you do not waste valuable processing resources during normal application execution.

Fault Analyzer does not require applications to be recompiled. However if you store compiler listings or side files in the appropriate repository Fault Analyzer is then able to identify the source statement of the abending program.

1.3.3 Analyzing application failures to improve developer productivity

Fault Analyzer provides three modes to help you better analyze application and system failure information.

Real-time analysis

Fault Analyzer includes exit programs for CICS, IBM Language Environment, and z/OS systems. If installed, these exits do Fault Analyzer real-time analysis as part of normal failure processing. After analysis you can view the analysis report in the job output or through the Fault Analyzer Interactive System Productivity Facility (ISPF) interface.

Batch reanalysis

The Fault Analyzer batch reanalysis mode generates a new analysis report. This report is based on the dump and information gathered in real time, but with potentially different options specified, or with compiler listings or side files made available. You can submit a Fault Analyzer batch mode job using either the Fault Analyzer full-screen ISPF interface or your own job control language (JCL).

Interactive reanalysis

Interactive reanalysis provides several advantages over batch reanalysis:

- The sections of the report that are of interest can be selected and examined separately.
- Any storage area that is included in the associated minidump or SYSMDUMP can be displayed, regardless of whether it is included in the Fault Analyzer real-time report.
- Source code information (if provided via compiler listing or side file) can be viewed in its entirety.
- This is the only way to analyze CICS system abends.

The ISPF interface has many point-and-shoot fields for easy navigation through the interactive reports. For example, when viewing an analysis report on screen in interactive mode, you can cursor-select a storage address to view the contents of storage at the time of the failure.

You can also associate addresses with notes that might assist with solving a problem. The notes are saved in the fault entry and are automatically displayed whenever the storage for the associated address is displayed.

1.3.4 Fault Analyzer V7.1 enhancements

Fault Analyzer Version 7.1 includes a number of enhancements.

Improved functions

There have been enhancements to the following functions:

- Additional CICS domains to help you debug CICS system problems
- Improved logic that detects and analyzes CICS storage violations
- Additional program data checking to improve the process of selecting the compile listing or side file that provides source code information for the abend
- A new *NoDup CICSfast* option that makes explicit dump calls eligible for fast duplicate processing when repeated requests come from the same program call point

- A new IMS NoDup ImageFast option that provides fast dispatching of repeat abends for an IMS transaction environment that is similar to the NoDup CICSfast processing available for CICS
- WebSphere MQ Application PD Enhancements that improve API information and WebSphere MQ return code diagnostics
- Additional flexibility for the PL/I compile options that are accepted by the parsing utility that generates Fault Analyzer side files

Improved performance

There are several analysis processing performance enhancements in Fault Analyzer for z/OS V7.1:

- The new function, *DeferredReport*, gives you the option of sending excess abends straight to CICS transaction dump processing so that they bypass Fault Analyzer. In busy CICS systems, this can prevent an unexpected inrush of abends from causing processing delays and over-commitment.
- The interactive reanalysis of fault entries now uses a smaller TSO region size.
- Fault Analyzer for z/OS V7.1 can now call a new function that creates an MVS™ dump if there is not enough storage space to capture and analyze an abend.

SOA support

To support SOA, Fault Analyzer for z/OS V7.1 now has:

- CICS Web services awareness: CICS Web Services data areas are formatted to assist the CICS Web services programmer.
- Initial integration with WebSphere Developer for zSeries® technology preview: This preview will be provided in conjunction with the next version of WebSphere Developer for zSeries.

New hardware and software support

Fault Analyzer for z/OS V7.1 now supports:

- CICS v3.2
- XPLINK support
- Analysis of 31-bit addressing XPLINK calling convention program abends
- New versions of WebSphere and Java
- Enterprise PL/I side files
- Assembler DSECT variable mapping of the storage areas for an assembler program when the assembler side files are available

1.3.5 For more information

To learn more about IBM Fault Analyzer for z/OS, visit:

`http://www-306.ibm.com/software/awdtools/faultanalyzer`

1.4 File Export for z/OS

Data at a customer site often exists in several formats, such as sequential files, partitioned files, VSAM files, DB2 tables, IMS segments, and so on. To test new or modified applications, developers and testers require test data that mimics production data. Creating a test environment that contains a complete set of related data derived from production is imperative for adequately testing an application. File Export for z/OS is the product to accomplish this task in an easy-to-learn tool.

1.4.1 What is File Export?

File Export is a powerful and versatile tool that simplifies test data generation. It facilitates the selection of related data from traditional data sources as well as transforms the data, that is, it scrambles sensitive data, ages date-related data and so on.

IBM File Export for z/OS allows you to extract related subsets of data, typically from your production environment, to create test database with meaningful data quickly and efficiently. Some of the File Export product highlights include:

- Extract related sets of data from similar sources, including DB2, IMS, VSAM and sequential files.
- Optionally transform the data, including scrambling to protect sensitive data.
- Create multiple targets for different environments, such as QA and development.
- Define relationships between data objects to reflect application logic.

File Export expedites the testing process by eliminating the requirement to write one-off extract programs, and enabling each tester or programmer to have their own, and appropriately sized test environment, eliminating the confusion caused by a shared test environment.

The objects defined to File Export (e.g. Application Relationships, Record Layouts, etc.) are stored in its Repository, which facilitates the sharing of these objects with other users.

IBM File Export for z/OS has the following hardware and software requirements:

- z/OS V1R4 or later
- XML Toolkit V1R6M0
- For IMS Support, either of:
  - IMS V7.01.00 (5655-B01)
  - IMS V8.01.00 (5655-C56)
  - IMS V9.01.00 (5655-J38)
- For IMS Fast-Path DEDB Support: IMS Fast-Path Basic Tools for z/OS V1.02.00 (5655-F78) or later
- For DB2 Support, either of:
  - IBM DB2 for z/OS V6 or V7
  - IBM DB2 for z/OS V8 in Compatibility Mode
- IBM DB2 Utilities Suite for z/OS and OS/390

1.4.2 Key File Export components

The File Export product components are:

- **User Interface:**

  The File Export ISPF-based UI is used to create the objects stored in the Repository that are used to create the Extract and Load Requests, as well as allowing you to specify the parameters that dictate how the Extracts and Loads perform. In addition, the UI generates the JCL statements that specify for the extract engine what is to be done.

- **Repository:**

  The Repository is where the File Export definitions are stored. The Repository is a set of VSAM datasets that can be shared among users.

- **Parser:**

  File Export contains an internal parser that can parse COBOL and PL/I source copybooks into File Export-format data definitions that are stored in the Repository.

- **Extract Engine:**

  The File Export extract engine extracts source data, optionally transforms it, and loads it into the specified target environments. The internal name for this program is ABXUTIL.

1.4.3 File Export terminology

Here we define the terminology we used:

- **Record Layout (RL):**

  This describes the structure of a record, that is, the sequence and format of the fields that comprise the record. This object is typically utilized for non-relational data.

- **Data Store Definition (DSD):**

  This is used to group all related Record Layouts for a given source and enables them to be treated as a single entity. For simple data records that require only a single Record Layout, a DSD is not required and the Record Layout can be used anywhere a DSD is used.

- **Application Relationships (AR):**

  This is a correlated set of one or more fields from records in two different Data Store Definitions that are used as the basis for finding related data records.

- **Related Subset (RSS):**

  This defines the set of DSDs (including RLs and DB2 tables) from which data is extracted, including the rules that control the extract such as, the starting objects, selection criteria, relationship usage, scrambling, etc.

- **Data Store Definition-to-Data Store Map (DSD-to-DS Map or DDM):**

  This is used to map “logical” DSDs to their “physical” counterparts (called DSs), that is, the underlying files. DSD-to-DS Maps are used for both the Extract process and the Load process.

1.4.4 How to use File Export

Using the ISPF-based user interface, File Export allows users to set options to define what source objects to extract data from, what optional parameters to use to transform the extracted data, and where to load the data. The following scenarios are used to describe the steps necessary to perform an Extract (and optional Load), starting with the simplest example and building from that.

Scenario 1: DB2 Extract utilizing only DB2 Referential Integrity

This scenario involves DB2 Extract utilizing only DB2 Referential Integrity as defined in the DB2 Catalog:

1. Create a Related Subset by specifying which object (such as a table) starts the extract, and pulling in related objects based on the DB2 RI. Optionally, you can specify a maximum for the number of relationship levels to use.
2. Create a DSD-to-DS Map, based on the Related Subset that specifies the actual tables to extract from. That is, the tables included in the RSS describe a logical data model, and the DDM specifies the actual tables to use.
3. Create an Extract Request which includes the RSS, DDM, Extract File into which data is extracted, and optionally, data transformation rules.

Scenario 2: Supplement DB2 Extract with Application Relationships

This scenario involves supplementing DB2 Extract with Application Relationships:

1. Since in this scenario not all the required RI is defined to DB2, create the one or more Application Relationships required to complete the description of the data model.
2. Repeat steps 1 though 3 from scenario 1. The only difference is that the RSS utilizes both DB2 RI and File Export Application Relationships to pull in the related set of tables.

Scenario 3: Extracting related data from a non-relational data source

This scenario involves extracting related data from a non-relational data source (that is, IMS):

1. Create the one or more Record Layouts required to describe the data structure of each object to be extracted from. This can be accomplished by defining each Record Layout from scratch (that is, defining the attributes of each field), or if you have COBOL or PL/I copybooks, by parsing those to glean field information.
2. If any of the data sources (IMS databases) requires more than one Record Layout to fully define it, then create a Data Store Definition for each data source.
3. Repeat steps 1 and 2 from scenario 2. That is, define the Application Relationships required to relate the IMS databases, then create a RSS, DDM, and Extract Request. In this scenario, the DDM maps the logical Data Store Definitions to their corresponding physical Data Stores.

Scenario 4: Extracting data from a single Data Source

This scenario involves extracting data from a single Data Source (such as unrelated data). It utilizes a different set of simple panels because the scenario is simpler. That is, since you are only interested in a single Data Store, there is no requirement for an RSS or DDM. Instead, you:

1. Utilize steps 1 and 2 from scenario 3, that is, define the one or more Record Layouts and Data Store Definitions required.
2. Define a Single-Source Extract Request in which you name (a) the DSD required to understand the record structure, (b) the source and destination Data Stores, and (c) some remaining parameters, optionally including any data transformation rules.

In each scenario, the generated JCL contains Control Syntax that the program ABXUTIL uses to process the extract request. The JCL can be submitted from within File Export, put into a job scheduler to run at a later date, or saved for submission by you at another time.

Note: Users familiar with the test environment data can define the Repository metadata, such as the Record Layouts, Application Relationships, Data Store Definitions, and so on, as well as the extract request JCL. Those less familiar with the data can then use the data in the Repository to create their own extract requests, or copy the extract request JCL.

1.4.5 For more information

To learn more about file Export for z/OS, refer to the User's Guide and Reference Manual:

http://www-306.ibm.com/software/awdtools/fileexport/

File Export for z/OS also comes with training labs and sample data to utilize in the training labs. The sample data can be unpacked from the dataset ABX.SABXSAMP, member ABXUNPAK. The labs can be found in dataset ABX.SABXSAMP, member ABXLABS.

1.5 File Manager for z/OS

File Manager is a member of the IBM suite of Problem Determination Tools (PD Tools) part of the IBM Enterprise Platform Software Portfolio. File Manager provides flexible, easy-to-use application-development tools designed to enhance file processing. IBM File Manager is an integrated suite of functions to help OS/390 Application Developer Works with data access a wide range of format and storage media. This product is a data management tool that includes support for the most important IBM files Systems, including DB2, IMS and CICS VSAM files. It is designed to provide an high level of user flexibility for developers and improve the performance of operations with a single Management Tool. Application developers working with structured data files, can take advantage of extra features without having to re-learn fundamental skills. Their productivity increases and the knowledge to manage "host data" becomes more simple.

Through its comprehensive, user-friendly tools, IBM File Manager for z/OS helps and support development and production through extensive editing, browsing, with batch and interactive utility capabilities for mission-critical CICS (Customer Information Control System) Applications. IBM File Manager for z/OS:

- Finds, copies, edits, prints, compares, and displays data quickly and easily
- Defines data set records efficiently by using COBOL and PL/I copybooks stored in partitioned data sets or library management systems
- Enhances usability by enabling you to customize which fields to display, copy, or print

1.5.1 Enhancing development productivity through structured data support

In a volatile economic climate, organizations are faced with constant pressure to create new applications that help reduce costs, engender customer loyalty, and earn competitive advantage. The requirement to tighten budgets and do more with less only increases the challenge. To succeed in a fiercely competitive marketplace, companies must be able to make the most of existing investments and manage application development effectively.

Typically, records in application data sets have a well-defined structure consisting of many fields. Each field can contain a discrete item of data, such as a person's name, a product code, or a currency amount. To work efficiently with these data sets, you must have utilities that recognize the record structure and segment layout, so you can selectively process individual fields.

Within this scenario, File Manager enhances and provides comprehensive user-friendly tools for working with OS/390 data sets such as VSAM, DB2 data, IMS data, or CICS resources. These tools include the familiar browse, edit, copy, and print utilities found in ISPF, enhanced to meet the requirements of application developers.

File Manager utilizes COBOL or PL/I copybooks stored in partitioned data sets and external library management systems to define data set records more efficiently, or you can dynamically define your own record structure in a template. A COBOL compiler has been added inside FM to support customers that do not have a COBOL license on their production LPAR. In order to simplify and distinguish the various components, we use the abbreviations FM/Base, FM/DB2, FM/IMS, and FM/CICS in this manual. Guidance and related information are also provided inside the *FM User's Guide* and reference manuals for DB2, IMS and CICS. The four components are:

- *Base component* for managing z/OS data sets, including queued sequential access method (QSAM) data sets, virtual storage access method (VSAM) data sets, and partitioned data sets (PDS)
- *DB2 component* for managing DB2 data
- *IMS component* for managing IMS data
- *CICS component* for accessing and managing CICS resources

File Manager base component

To maximize flexibility, each of the File Manager components allows you to identify and map your definitions in ways suited to each data type. Here we describe the FM requirements and components in more detail:

- Browse, edit, copy, and compare data using copybooks or dynamic templates.
- Find, edit, print, and display data quickly and easily to reduce development time.
- Edit record and field selection criteria on the fly.
- Leverage multiple selection criteria to make data more accessible.
- Work with files containing multiple record and consecutive "segmented" structures.
- Work with data with data in IAM data sets as well as data in QSAM and VSAM data sets.
- Generate eXtensible Markup Language (XML) data from an input dataset.
- Support Large and Extended Sequential Dataset (ESDS), zFS in addition to HFS files
- Support the Large Block Interface (LBI) for tapes. In full-volume utilities, for disk volumes with more than 64K tracks.
- Sort data during an edit or browse session.
- Perform sophisticated data set comparisons, using the new Data Set Compare utility.
- Use multiple search strings in the Find/Change utility.
- Edit entire files, regardless of size.
- Copy data between fields of different data types and lengths.
- Enhance File Manager with your own custom REXX™ procedures (such as TALLY for tallying field values).
- Use the attributes of existing QSAM data sets to define attributes for new data sets.
- Automate tasks in batch jobs, REXX procedures, or CLISTs.

File Manager DB2 component

This is an ISPF application. It provides panels that you can use to select options and to specify parameters, commands, and program function keys to simplify requests for common functions in a full-screen format for information display and editing. FM/DB2 is a powerful set of utility functions for editing, browsing, printing, copying, and maintaining DB2 data. It also provides support for the new DB2 V9 utilities in:

- Listing DB2 objects
- Managing DB2 privileges
- Generating JCL to run DB2 standalone utilities
- Exporting and importing DB2 tables to/from QSAM or VSAM data sets using many export format separator characters and delimited variables (CSV)
- Creating data to populate DB2 tables
- Prototyping and generating standard SQL SELECT statements

File Manager IMS component

This is an ISPF application. It is used for manipulating data stored in IMS databases. Using FM/IMS, you can:

- Display data from one or more IMS segment occurrences, and see their relationship to other segment types within the database.
- Edit and update data in IMS segment occurrences.
- Copy, modify and reuse of templates, views and criteria sets. Using the 'Template Update' option, you can:
  - Update templates (that can be reused) with the latest copybook or DBD definition, using foreground or batch processes.
  - Create new templates based upon existing Templates.
  - Insert segment occurrences into an IMS database.
  - Delete segment occurrences from an IMS database.
  - Extract a subset of IMS data to a sequential file.
  - Load data into an IMS database.
  - Print selected data or entire databases.

For many tasks, you can use the FM/IMS elements called templates and views to define a logical view of a database based upon field definitions from COBOL or PL/I copybooks. When you associate a view with a database, you can:

- Format data according to segment layouts defined in COBOL or PL/I copybooks.
- Select the segment types and fields that you want to display, edit, or extract.
- Change the order in which fields are displayed and adjust field headings.
- Create criteria to identify and select the data that you want to use.

FM/IMS provides support for the new IMS Version 10 and allows you, with a number of flexible ways, to connect to your IMS databases using two different modes:

- In BMP mode, you can connect to an online multi-user database and manipulate the data.
- In DLI mode, you can work with data offline as a single user or you can share the data with others.

In addition, FM/IMS provides two functions that you can use in batch jobs. FM/IMS Edit Batch (IEB) runs a REXX procedure that can insert, update, retrieve, delete, or print segments and create views. FM/IMS Batch Print (IPR) can print the entire database in one of several available display formats, or a selected sub-set of the database, based on a view addition.

File Manager CICS component

FM/CICS provides specific panels and functions to access CICS resources under a CICS transaction. A simple selection from the “Primary option Menu” of FM/CICS invokes the FM (FM/Base) or FI (FM/IMS) component within the CICS environment. This means that all the File Manager (Base and IMS) utilities are available without ISPF support.

FM/CICS uses an address space together with an Interactive Panel Viewer (IPV) to display panels on CICS. Further information related to this process is provided in 27.2, “Architecture overview” on page 832.

The main functions provided by FM/CICS are:

- Edit, Browse, Print and alter the status of the following CICS resources:
  - VSAM files
  - Temporary Storage (TS) queues
  - Transient Data (TD) queues
- Facility to list CICS resources and change their attributes
- Support CICS VSAM files (KSDS,ESDS,RRDS,URRDS) and CICS Table
- Process - CICS TD Intrapartition (INT), extrapartition (EXT) - and TS Queued
- Select Item from a display list using a Pattern (S *Z)
- Print table in single, hex, or char Format
- Sort a CICS selection list

FM/CICS includes and incorporates much of the functionality of File Manager for z/OS into the CICS environment. In addition, apart from the way in which File Manager is invoked, there are no differences between the FM (IMS and Base) functions running under ISPF and running under CICS. The FM/CICS interface is based upon the ISPF model, using the Interactive Panel Viewer (IPV). The IPV processes and translates the data stream through the FM address space and sends the 3270 data stream to FM Transaction component to complete the process and display the FM/CICS Panel. Chapter 27, “File Manager/CICS” on page 831 explains the process more in detail.

Navigating between the FM/CICS Panels reflects the same way as you do under ISPF. You can launch processing screens from the menu panel by entering the appropriate menu option number on the command line.

Within the FM/CICS panels, you can:

- Use primary and prefix commands to instruct FM/CICS to perform an action
- Provide specific functions to process the CICS resources.
- Invoke FM Base and FM/IMS through a CICS terminal.

Important: IBM File Manager for z/OS Version 7.1 software at a glance:

- Hardware requirements:
  - zSeries system
  - Any hardware that runs IBM z/OS Version 1.4 software or later
- Software requirements:
  - z/OS Version 1.4 or later(5694-A01)
- Additional requirements (one of these):
  - IBM Enterprise COBOL for z/OS and OS/390 Version 3.3 (5655-G53)
  - IBM COBOL for OS/390 & VM V2.2 (5648-A25)
- For DB2 component (one of these):
  - FM V7 RFA announced support for DB2 V9 when available
  - IBM DB2 Universal Database™ for z/OS Version 8(5645-DB2)
  - IBM DB2 Universal Database for OS/390 and z/OS Version 7(5675-DB2)
  - IBM DB2 Universal Database for OS/390 Version 6 (5645-DB2)
- For IMS component (one of these):
  - IMS Version 9(5655-J38) or later
  - IMS Version 8(5655-C56)
- For CICS component (one of these):
  - CICS Transaction Server for z/OS V3.1 (5655-M15)
  - CICS Transaction Server for z/OS V2.2 and V2.3 (5697-E93)
  - CICS Transaction Server for OS/390 V1.3 (5655-147)
- Optional License Programs. Depending on the functions used, one or more of the following programs might be required:
  - IBM Enterprise COBOL for z/OS and OS/390 Version 3.3 (5655-G53)
  - IBM COBOL for OS/390 & VM V2.2 (5648-A25)
  - IBM VisualAge® PL/I for OS/390 V2 (5655-B22)
  - IBM Enterprise PL/I for z/OS V3 (5655-H31)
  - WebSphere Studio Enterprise Developer V5.1.2 (5724-B67)
  - IBM z/OS V1 C/C++ Element (5694-A01)
  - IBM OS/390 V2.10 C/C++ Element (5647-A01)
  - High Level Assembler/MVS & VM & VSE, V1.4 and V1.5 (5696-234)
  - WebSphere MQ for z/OS V6 (5655-L82)
  - WebSphere MQ for z/OS V5 (5655-F10)

Note: z/OS releases prior to V1.4 are out of service and the support for IMS V7 has been removed.

1.5.2 Maximizing flexibility with the File Manager base component

The File Manager base component helps speed the application-development process by identifying the data structure and displaying each field according to its data type. This component supports QSAM, VSAM, PDS, HFS, and indexed access method (IAM) data sets, including support for double-byte character set (DBCS) data in these data sets. You can edit entire files (regardless of size) and sort data during an edit or browse session. You can use the separately available IBM CICS VSAM Copy tool in conjunction with File Manager capabilities to produce ad hoc copies of VSAM data sets for testing purposes without affecting CICS activity.

Using templates, you can select the records you want based on field values, find and change data in individual fields, display fields with headings showing name, data type, and length — and then print or copy only the fields you must have. You can quickly and easily work with files containing multiple record structures and copy data between fields of different data types and lengths. FM/Base offer template support for segmented records. A segmented record is a record that contains one or more logical segments. Typically, a Record structure could have one or more segment types, and each of which could be displayed separately. It could be convenient while we are looking at a particular segment type.

In the base component, you can:

- Create multiple templates from a PDS of copybooks by using the Template Build Utility.
- Generate templates from a copybook that contains either COBOL data-description entries or PL/I DECLARE statements
- Update one or more Templates in foreground or batch.
- Define your own fields in a dynamic template.

You can then add record-selection criteria and other formatting information, and save the templates for reuse with different data sets that have the same record structure, or for reuse with different File Manager utilities. For example, while browsing through data sets, you can create, refine, and save a template that displays records meeting certain criteria, such as records with particular field values. Later, you can reuse that template in the File Manager copy utility to extract the records that meet those criteria and copy them to another data set.

The base component processes most record identification and selection criteria internally, while providing fast access to Restructured Extended Executor (REXX) for complex criteria statements. The copy and print utilities use the IBM DFSORT™ utility for supported data sets to help you gain significant performance improvements. Support is also available for DFSORT statements.

The copy utility also offers various PDS-member copying capabilities, including the ability to copy members based on finding a particular condition to be true from records in the member. The find/change utility offers flexible options to increase efficiency for batch and online modes. With the data-set compare utility, you can use a number of different synchronization types. You can also enhance File Manager with your own custom REXX procedures (such as tallying field values) and automate tasks in batch jobs, REXX procedures, or command lists (CLISTs). A high-performance subset of REXX and File Manager REXX functions, called FASTREXX, is also available. The FASTREXX subset is broad enough to handle most normal processing tasks.

The File Manager base component also includes other utilities, such as:

- Tape utilities that enable you to perform actions such as copy, update, and compare with tape files.
- System Catalog Services (SCS) utilities that provide a user interface for working with catalog entries. This release adds the ability to define and build an alternate index (AIX®) on a VSAM data set. SCS helps to list or print catalog information as:
  - SCS VSAM files INFO showing the date/time of last update
  - SCS EXTENT display for non-VSAM data sets
  - SCS EXTENT display for VSAM data sets (more than 999) and numbering scheme
  - SCS improved INFO display for VSAM and non-VSAM data sets
  - SCS INFO volumes display limited to five volumes (PFM00195) for VSAM and non-VSAM
  - SCS INFO panels to emphasize additional information available

- SCS data set list limit on number of data sets processed/displayed and number of tracks allocated
- OAM utilities that enable you to list, browse, update, and erase OAM objects, as well as copy, back up, and restore them
- Disk utilities that allow you to list the data sets on a disk, as well as their disk extents, and search for data within a disk extent

1.5.3 Manage data more efficiently with the File Manager DB2 component

The File Manager DB2 component delivers the commands and behavior of the ISPF/PDF editor and extends the capabilities to DB2 data. You can view detailed information about DB2 objects, show the relationship between them, and select the objects you want to work with from virtually any connected location or subsystem.

With the DB2 Object List utility, you can selectively list DB2 objects (for example, by owner), then invoke commands against those objects. The File Manager DB2 component also enables you to view and change DB2 privileges through a full-screen interface, so you can eliminate the requirement to code SQL statements to list and change DB2 privileges.

The interactive SQL SELECT Prototyping utility in the File Manager DB2 component helps you build a valid SELECT statement on screen. You can interactively test, analyze, and execute SQL statements. When you run the SQL statement, you can browse or edit the formatted result table and then save the statement for use in your application code.

The DB2 component features a sort command that enables you to easily rearrange data in an edit or browse session, to export selected rows and columns to QSAM or VSAM data sets, and to generate templates based upon your export selections. You can also generate job control language (JCL) for the most commonly used DB2 utilities to save time.

Edit/Browse utilities allow specification of the number of rows and support synonyms in DB2 functions.

The File Manager DB2 component provides support of DB2 V6, V7, V8, and (RFA) has been announced support for DB2 V9.

The latest File Manager/DB2 Version provides new features and enhancements to several functions.

Various improvements have been made to the FM/DB2 regarding the default values for processing options. Many of the processing operations performed by File Manager utilize default values that can be set from within the FM/DB2 application. By adjusting these values, you can customize FM/DB2 for the following options:

- System: To select and display subsequent system options panels.
- Print Setting: To control the destination of your print output and the page format to use.
- DB2/ System: To control the translation of DB2 object names and input SQL statements.
- Job Card: To specify batch job card information to be used for generating batch job JCL.
- Compiler language selection: To select which compiler is used to generate templates.
- COBOL & PL/I compiler specifications: To control various options used by the COBOL and PLI compiler during the template generation. Used by the FM/DB2 import and export utilities.

Other options have been added for LOAD, LISTDEF, OPTIONS, and TEMPLATE utilities. Export and Object List Utilities Panels used to display lists of each object type have been expanded to allow you to scroll to display the data held in every column. In addition, improvements have been made to the way you can filter columns in order to restrict the data displayed. The FIND and LOCATE primary commands are available to search and locate data displayed.

The File Manager FIND command has been improved in order to search a string within specific portions of the data you are FINDing.

The batch submit function for FM/DB2 has been enhanced by increasing the number of JCL job cards.

1.5.4 Developing faster, less costly applications with File Manager IMS component

Using record structures defined in COBOL or PL/I copybooks, the File Manager IMS component enables you to edit IMS segments displayed as individual fields, formatted according to their data types. You can find and change data across an entire database hierarchy or in specified segments and fields. You can run a REXX procedure that can edit segments in an IMS database. The support for a tabular display of data helps improve the display of segments within the hierarchy and offers significantly improved response times. Navigation commands allow you to specify segment operands, helping to simplify navigation between different segment types. You can also initialize and insert the first segment into an empty database. The File Manager IMS component helps minimize resources held during user screen interactions to maximize concurrent access to data.

You can use flexible criteria to select IMS database segments containing specific field values, and extract the segments into a sequential data set for later use in reports. Or you can use the IMS component to load them into a different database. For example, you might create a small test database using data extracted from a large production database. Through the IMS component, you can access databases using Data Manipulation Language 1(DL/I) or use batch message processing (BMP) to access databases that are concurrently being used in IMS Transaction Manager, DBCTL, or CICS environments.

Here we describe the File Manager enhancements for the IMS component:

- Support is provided for IMS Version 9.
- Support is provided for the IMS Version 8 and Version 9 release locking (RLSE) small programming enhancement (SPE), to improve File Manager editor function while continuing to maintain concurrent access to IMS databases, in cases where the previous File Manager IMS component had to automatically commit IMS edit changes every time the user pressed Enter.
- Capability to update the template to reflect the new DBD structure if segment types are added to or deleted from the associated database, or segment layouts change (New Option 4 Templates from the Primary Options Menu panel), In addition to this option, the IMS Template Update (ITU) and IMS View Update (IVU) functions provide you with the ability to update templates and views using a batch process.
- SHOWCOB Command (on all panels) to display details about the current COBOL compiler in a window.
- Print, View, or Edit a DLI database in batch using the option 7 from the FM/IMS Utilities panel. You can generate JCL to view your IMS segments in batch using a REXX procedure you generate.

- To improve the usability and navigation between FM/IMS, a new “Advanced Member Selection” panel was added to refine the search for the modules you want to update. This panel can be selected from the following panels:
  - View Update Utility
  - Template Update Utility
  - Criteria Set Update Utility

FM/IMS added a number of new functions and enhancements to improve the usability and navigation between the panels and fields. The Parent panels section lists the names of the panels from which you can navigate to the current panel. The Child panels section lists the names of the linked panels that you can navigate to, from the current panel.

1.5.5 FM/CICS component to invoke File Manager without TSO (ISPF-based)

IBM File Manager for z/OS introduced a new component to run the FM functions and utilities from the CICS environment. The graphics and panels maintain a similar design (ISPF model) for editing, browsing, printing and altering the status of CICS resources. The 3270 screens that are displayed are not Common CICS Basic Mapping Support (BMS) — maps created using special Assembler language macros. File Manager access and displays the Panels to CICS using a TCP/IP sockets interface. FM/CICS interface (based ISPF model) uses an Interactive Panel Viewer (IPV) to translate the panels and come back to another component (FM address space) that processes the translated data stream and sends the 3270 data stream to FM Transaction component to complete the process and display the FM/CICS Panel.

Using the FM/CICS interface, you can:

- Move from panel to panel
- Show more information on a panel
- Use **`Primary`** and **`Prefix`** commands
- Describe how we specify a CICS resource to File Manager.
- Specify the data set that you wish to use for a copybook or template.
- View and enter decimal values as text characters or as hexadecimal values.

The navigations between panels in FM/CICS is the same as under ISPF. That is, you launch processing panels from menu panels by entering the appropriate menu option number on the command line. FM/CICS uses a Primary Options Menu panel to provide access to processing panels that are related to a particular function or group of functions. When you know the menu structure well, you can quickly jump to the required panel by entering the full pathway through the menu options.

These solutions allow the use of both FM/Base and FM/IMS for customers that are not licensed under TSO. To start and connect FM/CICS you do not require a TSO userid. Your authority to process any function is related to the userid that is used to connect to the CICS environment (CSSN Logon procedure).

The CICS resources supported from FM/CICS are files, temporary storage queues, and transient data queues. If you have the authority, you can also modify the status of the CICS resources. FM/CICS incorporates much of the functionality of File Manager for z/OS (base and IMS) into the CICS environment. FM/CICS panels allow you to select options and to specify parameters, commands, and program function (PF) keys to simplify requests for common functions. FM/CICS panels provide full-screen format for information display and editing.

The main functions provided by FM/CICS are:

- The ability to edit or browse CICS resources.
- The ability to list CICS resources and change their status and attributes.
- The ability to run FM/Base under FM/CICS without requiring ISPF. This includes the ability to use FM/Base tools and utilities against FM/CICS resources when running under FM/CICS.
- The ability to run File Manager IMS Component (FM/IMS) under FM/CICS, using it without requiring ISPF.

The CICS resources that are supported are:

- VSAM files (KSDS, ESDS, RRDS, URRDS) and CICS (LOAD) tables
- Temporary storage (TS) queues
- Transient data (TD) queues

The CICS TD control facility provides a generalized queuing facility. Data can be queued (stored) for subsequent internal or external processing. Selected data, specified in the application program, can be routed to or from predefined symbolic TD queues: either intrapartition or extrapartition.

Transient data queues are intrapartition when they are associated with a facility allocated to the CICS region, and extrapartition if the data is directed to a destination that is external to the CICS region. TD queues must be defined and installed before first reference by an application program.

Regarding the TD queues, FM/CICS provides a panel to manage both extrapartition as well as intrapartition. When FM/CICS reads an intrapartition (READQ TD), the entire queue is read into memory by the Browse Transient Data Entry Panel. When you exit, you are given the option of restoring (or not) all the records read back to the queue. These records are appended to the existing queue if you select to rewrite the records.

This book helps to introduce the new user to easily use and navigate within FM/CICS primary commands, panels, fields, resources, and CICS queues.

1.5.6 Flexible file processing to meet your business goals

File Manager is designed to help increase productivity and lower costs as you build and test new and existing applications. You can use it to quickly create, edit, print, and reformat data files and to build new applications easily.

FM helps developers during the test and debug within complex applications to identify wrong data. These are often the origin of indeterminate behaviors and abends.

With File Manager, you gain the flexibility you must have to build, leverage, and extend your assets to meet your business goals.

1.5.7 For more information

To learn more about IBM File Manager for z/OS, visit:

http://www-306.ibm.com/software/awdtools/filemanager/

1.6 Workload Simulator for z/OS and OS/390

Workload Simulator (WSim) is a terminal and network simulation tool that can help you test your networks and determine whether your infrastructure is capable of handling an expected additional workload. It provides the ability for the simulation of terminals and associated messages, including provision for you to alter message loads during a run.

WSim has evolved from the IBM TeleProcessing Network Simulator (TPNS) product. The addition of an interactive ISPF-based Test Manager (WTM) can significantly assist users with the development and management of test cases, automation of test runs, and analysis of results. At the same time, support for obsolete and little-used functions has been removed.

You can use the IBM Web Sphere Studio Workload Simulator (WSWS) tool to provide similar functionality for browser-based infrastructures.

1.6.1 What Workload Simulator can do

Workload Simulator enables users to perform the following functions:

- Functional testing to ensure that a new program or subsystem is performing in accordance with specification
- Regression testing to ensure that a modified program or subsystem is still performing in accordance with original specifications where no changes were required
- Performance, stress, and capacity testing to ensure that a system can handle the projected load with acceptable response times

WSim can simulate three types of resources:

- System Network Architecture (SNA) logical units running as Virtual Telecommunications Access Method (VTAM) applications
- Common Programming Interface for Communications transaction programs
- Transmission Control Protocol/Internet Protocol (TCP/IP) clients using Telnet 3270, 3270E, 5250, Network Virtual Terminal, File Transfer Protocol, or simple TCP protocols attached to TCP/IP network via the IBM TCP/IP for Multiple Virtual Storage product

Both the network to be simulated and the system to be used to run WSim must be configured before testing.

A physical configuration is a configuration of the real system used to run WSim. There are two basic physical configurations:

- VTAM and CPI-C application configuration
- TCP/IP application configuration

A logical configuration is a configuration of the network containing the resources to be simulated by WSim and the real system to be tested. There are three basic logical configurations:

- VTAM application configuration
- CPI-C application configuration
- TCP/IP application configuration

When the system is configured, the network to be simulated should be defined, which is done by creating a script. Scripts contain two parts:

- Network definition statements describing resources to be simulated by WSim
- Message generation decks defining messages to be sent by the simulated resources to the system under test

WSim provides several methods to create message generation decks:

- Writes statements directly
- Writes programs in Structured Translator Language (STL)
- Uses one of the script segregating utilities provided with WSim

Depending on your familiarity with WSim and the characteristics of the system under test, you could use one, several, or all of the methods when preparing real tests.

WSim provides several online and printed reports to analyze test results. Some reports are provided by default. Others must be specifically requested. The reports fall into the following four general categories:

- Operator reports that indicate what is happening during operation
- The complete message log
- Reports based on the message log and generated by several WSim utilities
- Online response time statistics

1.6.2 What Workload Simulator provides

Workload Simulator offers the facilities for the flexible generation of scripts, runtime support, and post-test analysis. It also provides the Test Manager, a convenient usability enhancement to help you to go through the test process.

Flexible generation of scripts

WSim offers several options for creating scripts to use in simulations:

- Structured Translator Language (STL), a high-level REXX-like language
- Interactive Data Capture (IDC)
- Network Performance Monitor/LU 2 Reformatter
- Script generator

Runtime support

WSim offers various utilities for users to use in testing their networks:

- An ISPF interface that helps you set up and start the application and its utilities
- A Display Monitor Facility that shows simulated 3270 screen images and the data streams sent and received
- Runtime reports
- ITPECHO, a VTAM application that echoes received data

Post-test analysis

WSim provides the following utilities to analyze test results:

- Loglist for formatting simulation run logs
- Log Compare to display records from two simulation runs with differences listed
- Response time to print a report with response time retrieved from logs

WSim Test Manager

The WSim Test Manager (WTM) is a usability feature that provides guidance through the test process. WTM offers selectable modes of operation, test management services, automatic script generation, and task automation. WTM guides you through the test process and helps to develop and manage test cases, automate test runs, and analyze results. We highly recommend that you use WTM, at least until you have achieved a certain level of familiarity with WSim and scripts.

1.6.3 The latest enhancements

The latest enhancements to the WSim, implemented in September 2004, include:

- WSim Adapters for Rational® Test Manager, which allow existing WTM schedules to be launched from the Rational Test Manager running on a remote workstation.
- 3270 password masking of potential passwords entered by users of the WSim data capture and script generation utilities. The passwords are masked in the generated WSim scripts, simulation data views, and output reports.

Part 2

Introduction to the problem determination and deployment tools

In this part of the book, we provide a detailed introduction to each of the IBM problem determination and deployment tools.

2

Introduction to Application Performance Analyzer for z/OS

In this chapter we describe the main functions provided by Application Performance Analyzer for z/OS, hereafter referred to as Application Performance Analyzer (APA). These functions can be used safely in both test and production environments. Application Performance Analyzer uses a low-impact sampling technology so it runs with very low processor overhead. It contains an online, instantaneous monitoring and measurement function for real-time bottleneck analysis, as well as a project-oriented performance management functionality. It provides you with the responsive and speedy information required to make your applications deliver the best performance possible.

We discuss the following functions:

- Administration/miscellaneous
- Statistics/Storage reports
- CPU Usage Analysis reports
- DASD/O Analysis reports
- CPU WAIT Analysis reports
- Coupling Facility reports
- IMS Measurement reports
- CICS Measurement reports
- DB2 Measurements reports
- MQ Measurement reports
- Real-time Analysis reports
- Java Measurement reports

2.1 How Application Performance Analyzer samples an application

Application Performance Analyzer is a sampling profiler. During an Application Performance Analyzer observation session, activity in the target address space is sampled at the frequency and for the duration that was specified when the session was requested. Each observation results in data being recorded that describes an observed System State (CPU executing, CPU waiting, queued). Each observation or *sample* interrupts and momentarily *freezes* system activity in the target address space. Information about the state of the interrupted process is recorded.

Note: It is important to understand that an observation session measures activity in a single address space. When we refer to system states such as *CPU waiting* or *CPU unavailable*, these states are with respect to the target address space only.

The essence of the analysis reports is the aggregation of system states and attribution of these aggregates to various system objects. By mapping observed system states to system objects, the analysis reports provide a meaningful picture of how resources are consumed.

Note: Systems objects are objects to which quantified observation of systems states can be attributed. These are the basic system objects:

- Load modules
- TCBs
- DD names
- DASD volumes
- SVCs
- MQSeries queue names

Because Application Performance Analyzer is a sampling profiler, it means that the profile it gives *is not* an *exact* view of the program (that is, found by following the code exactly and seeing all that is executed), but an *average*. The performance analysis reports express most quantified data as percentages. In most cases, absolute values would not, by themselves, be particularly meaningful. This is because the total number of samples chosen for an observation session is somewhat arbitrary. The percentage of activity attributed to a system object, on the other hand, provides a much better measure of the impact of that system object on performance.

Note: Furthermore, when expressed as percentages, quantification is likely to remain roughly equal if the sampling frequency and duration parameters are varied.

To get the most out of a sampling profiler, a user should ensure that they are basing their analysis on a significant number of samples.

If an application programmer has to follow an application step-by-step, Debug Tool for z/OS should be used instead.

Nevertheless, running Application Performance Analyzer measurements with the DB2+ or IMS+ data extractor turned on causes each DB2 call and each IMS call to be intercepted to collect additional data. It means Application Performance Analyzer becomes a tracing profiler for both DB2 and IMS subsystems.

Note: This could have a small impact on the performance of the target address space. Care should be taken when using this feature with other products that also intercept DB2 and IMS calls, as unpredictable results might occur.

Your installer might have chosen to limit access to this feature.

A margin of error value is displayed in various reports. The value is expressed as a percentage and represents a 95% confidence interval. What this means is that in 95% of cases, a repetition of the same measurement produces results within +/- the indicated number of percentage points.

Note: This value is based entirely on the size of a sample population and reflects only the statistical error that can be expected from the sample size. It does not take into account any effects caused by a biased sample collection.

2.2 Where Application Performance Analyzer saves reports

Next we discuss situations where Application Performance Analyzer saves reports.

2.2.1 Measurement file/sample file

When Application Performance Analyzer measures activity, it saves the observed data in memory (auxiliary storage). When the measurement process stops, Application Performance Analyzer creates a sequential file to permanently save the data. This file is named measurement file or sample file.

2.2.2 Checkpoint file

Application Performance Analyzer uses the *checkpoint file* data set to record the status of all measurement requests (one record for each measurement file). The Application Performance Analyzer started task allocates this sequential data set.

2.3 How to get help

Three types of help are always available with Application Performance Analyzer:

- Help on report navigation:

  For general help on *report navigation*, enter the Help command line and press Enter, or easier, position the cursor on the command line and press PF1. This help lists all of the available commands for the reports.

- Help on report interpretation:

  For help about a report currently displayed, position the cursor anywhere on the body of the report (not on an input field — almost all input fields are underlined) and press PF1. Pressing PF1 on an input field displays specific information about that field.

- Help linkage:

  Certain phrases appear underlined and highlighted in white (column headings) or in blue (input fields). By selecting such a field with the cursor and pressing PF1, we can jump to another help window. These are usually definitions of terms.

When selecting **Help** on the action bar menu, a pop-up window appears allowing us to display help on report navigation as well as on report interpretation.

2.4 The functions of Application Performance Analyzer

Application Performance Analyzer reports are each assigned a three-character code consisting of a one-letter prefix, followed by a two-digit number. The prefix indicates which category the report belongs to, and the number is a sequence number within that category. When we are viewing a report, we can always enter another report's three-character code on the command line, and that report is also opened. Table 2-1 lists the categories.

Table 2-1 Reports prefixes and categories

Prefix	Category
A	Administration/Miscellaneous
S	Statistics/Storage
C	CPU usage analysis
D	DASD I/O analysis
G	Coupling facility
W	CPU WAIT analysis
I	IMS measurement
E	CICS measurement
F	DB2 measurement
Q	MQseries measurement
J	Java measurement

This section describes the main categories provided by Application Performance Analyzer.

2.4.1 Administration/Miscellaneous

The Administration/Miscellaneous category contains two reports listed in Table 2-2.

Table 2-2 Admin/Miscellaneous reports

Report identification	Report description
A01	Source program mapping
A02	Request Printed Reports

A01 (Source Program Mapping) allows us to specify and manage associations between source program mapping files, hereafter referred to as SPM, and observation sessions. The SPM feature allows measured addresses to be mapped to their corresponding source program statements. An SPM file can be sequential or a member in a partitioned data set and can be one of the following file types:

- A listing produced by the compiler (COBOL only)
- An Associated Data (ADATA) file produced by HL-Assembler
- A side file member produced by the IDILANGX utility (COBOL and PL/I)
- A sysdebug file member produced by the compiler (COBOL only)

The Source Program Mapping function is described in detail in Chapter 8, “Application Performance Analyzer: Settings and customization” on page 135.

A02 (Request Printed Reports) allows us to select sections to be included in a batch report with the possibility to modify the format of some report sections by specifying formatting options. The batch can produce:

- A PDF file to create a report file in PDF format
- A SYSOUT file to create a report in a JES-managed SYSOUT file

This function is described in detail in Chapter 12, “Application Performance Analyzer: More about reports” on page 269.

2.4.2 Statistics/Storage

The Statistics/Storage category contains eight reports, listed in Table 2-3.

Table 2-3 Statistics/Storage reports

Report identification	Report description
S01	Measurement profile
S02	Load module attributes
S03	Load module summary
S04	TCB summary
S05	Memory usage timeline
S06	Data space usage timeline
S07	TCB execution summary
S08	Processor utilization summary

S01 (Measurement profile) gives a general overview of the measurement. This is a good report to examine first when analyzing a measurement. It provides an at-a-glance summary of various aspects of the measurement data and helps us choose which other reports to concentrate on.

S02 (Load module attributes) lists information about each of the load modules for which activity was measured during the observation session. Various attributes, including the External Symbol Dictionary information, of each of the modules are reported.

S03 (Load module summary) lists the load modules for which activity was measured during the observation session with, but not only, their location, address, size, and count.

S04 (TCB summary) and S07 (TCB execution summary) list all TCBs (tasks) that were active at any time during the observation session. The list is arranged in hierarchical sequence with ATTACHed subtasks indented relative to the parent tasks that performed the ATTACH functions. S04 gives the number of samples in which the TCB was observed and S07 gives the CPU time used by the task.

S05 (Memory usage timeline) and *S06 (Data space usage timeline)* break the observation session duration into a number of fixed-length, chronological time intervals. This quantifies the number of real storage (S05) or virtual storage (S06) that were allocated to the address space and illustrates any progressive memory usage trends.

S08 (Processor utilization summary) gives a breakdown of CPU states observed during the measurement.

These reports are described in detail in Chapter 10, “Application Performance Analyzer: Performance Analysis Reports” on page 173.

2.4.3 CPU usage

The CPU usage category contains nine reports, listed in Table 2-4.

Table 2-4 CPU usage reports

Report identification	Report description
C01	CPU usage by category
C02	CPU usage by module
C03	CPU usage by code slice
C04	CPU usage by timeline
C05	CPU usage by task/category
C06	CPU usage by task/module
C07	CPU usage by procedure
C08	CPU usage by referred attribution
C09	CPU usage by PSW/object code

C01, C02, C05, C06, C07, and C08 analyze measured CPU consumption. The CPU consumption is attributed by:

- *C01 (CPU usage by category)* to four general categories:
  - APPLCN - Application Code
  - SYSTEM - System/OS Services
  - DB2SQL - SQL Processing
  - DATAMG - Data Management (DASD) Requests
- *C02 (CPU usage by module)* to load modules.
- *C05 (CPU usage by task/category)* to each task (TCB), under each task. This information is shown under four general categories:
  - APPLCN - Application Code
  - SYSTEM - System/OS Services
  - DB2SQL - SQL Processing
  - DATAMG - Data Management (DASD) Requests

- *C06 (CPU usage by task/module)* to each task (TCB). Under each task, a further breakdown of CPU consumption is shown by load modules.
- *C07 (CPU usage by procedure)* to the source program procedure level.
- *C08 (CPU usage by referred attribution)* to system modules referred back to the points of invocation in application modules.

C03 (CPU usage by code slice) attributes CPU usage to code slices. A code slice is a range of storage addresses containing executable object code. We can use this report to pinpoint the exact locations of hot spots — segments of code where CPU consumption is particularly high.

C04 (CPU usage by timeline) breaks the observation session duration into a number of fixed-length, chronological time intervals. This quantifies the number of CPU usage for an interval and illustrates any progressive CPU usage trends.

C09 (CPU usage by PSW/object code) lists information about sampled CPU execution at the machine instruction level. This report is most useful when used in Subset Analysis mode to provide more detailed analysis for a particular quantification.

These reports are described in detail in Chapter 10, “Application Performance Analyzer: Performance Analysis Reports” on page 173.

2.4.4 DASD I/O

The DASD I/O category contains nine reports, listed in Table 2-5.

Table 2-5 DASD I/O reports

Report identification	Report description
D01	DASD usage by device
D02	DASD usage by DDNAME
D03	DASD usage by data set
D04	Data set attributes
D05	DASD EXCP summary
D06	DASD VSAM statistics
D07	DASD activity timeline
D08	DASD I/O wait time
D09	VSAM buffer pool usage

D01 (DASD usage by device), D02 (DASD usage by DDNAME), and *D03 (DASD usage by data set)* show how much DASD I/O time was used by each DASD device (D01), file (D02), or data set (D03) for which activity was measured during the observation session.

D04 (Data set attributes) lists information about each of the data sets that was open at some point during the observation session.

D05 (DASD EXCP summary) gives a summary of the number of EXCPs for each open dataset.

D06 (DASD VSAM statistics) gives file access statistics for each open VSAM dataset.

D07 (DASD activity timeline) gives for each file, the I/O activity distribution over the measurement interval.

D08 (DASD I/O wait time) gives delays resulting from waits during DASD I/O operations.

D09 (VSAM buffer pool usage) gives information about activity in VSAM LSR buffer pools.

These reports are described in detail in Chapter 10, “Application Performance Analyzer: Performance Analysis Reports” on page 173.

2.4.5 Coupling facility

The Coupling facility category contains three reports, listed in Table 2-6.

Table 2-6 Coupling facility reports

Report identification	Report description
G01	Coupling facility summary
G02	Coupling facility mean times
G03	Coupling facility total times

G01 (Coupling facility summary) gives a summary of the coupling facility data collected during the observation session.

G02 (Coupling facility mean times) and *G03 (Coupling facility total times)* give an analysis of how mean time (G02) or total time (G03) was spent by the coupling facility during the observation session. We can expand a coupling facility report line to see a further breakdown by structure name within the facility name.

These reports are described in detail in Chapter 10, “Application Performance Analyzer: Performance Analysis Reports” on page 173.

2.4.6 CPU WAIT

The CPU WAIT analysis category contains three reports, listed in Table 2-7.

Table 2-7 CPU WAIT reports

Report identification	Report description
W01	WAIT time by task/category
W02	WAIT time by task/module
W03	WAIT time referred attribution

W01 (WAIT time by task/category) gives for each task (TCB) the percentage of elapsed time during which the task was observed to be in a WAIT. Under each task, this information is shown under two general categories:

- APPLCN - System/OS Services
- SYSTEM - Application Code

W02 (WAIT time by task/module) gives for each task (TCB), the percentage of elapsed time during which the task was observed to be in a WAIT. Under each task, a further breakdown of wait time is shown by load modules.

W03 (WAIT time referred attribution) shows the attribution of WAIT time. WAITs issued in system modules are referred back to the points of invocation in application modules.

These reports are described in detail in Chapter 10, “Application Performance Analyzer: Performance Analysis Reports” on page 173.

2.4.7 IMS measurement

The IMS measurement category contains twenty-one reports, listed in Table 2-8.

Table 2-8 IMS measurement reports

Report identification	Report description
I01	IMS measurement profile
I02	IMS DL/I call timeline
I03	IMS transaction timeline
I04	IMS transaction activity timeline
I05	IMS DL/I CPU usage by PSB
I06	IMS DL/I CPU usage by transaction
I07	IMS DL/I CPU usage by DL/I call
I08	IMS DL/I WAIT time by PSB
I09	IMS DL/I WAIT time by transaction
I10	IMS DL/I WAIT time by DL/I call
I11	IMS DL/I activity by PSB
I12	IMS DL/I activity by transaction
I13	IMS DL/I activity by DL/I call
I14	IMS PSB/PCB attributes
I15	IMS DL/I call attributes
I16	IMS transaction service time
I17	IMS transaction DL/I call counts
I18	IMS CPU/Svc time by DL/I calls
I19	IMS CPU/Svc time by PSB
I20	IMS CPU/Svc time by transaction
I21	IMS CPU/Svc time by PCB

I01 (IMS measurement profile) gives a general overview of the IMS measurement data. This is a good report to examine first when analyzing IMS information. It provides an at-a-glance summary of various aspects of the measurement data and helps us choose which other reports to concentrate on.

I02 (IMS DL/I call timeline) and *I03 (IMS transaction timeline)* give the chronology of DLI calls (I02) or IMS transactions (I03) observed during the measurement interval. Each line shows information about one executed DLI call (I02) and IMS transaction (I03). The IMS+ feature must have been enabled when the measurement was performed.

I04 (IMS transaction activity timeline) shows for each IMS transaction, how execution of that transaction was distributed over the measurement interval.

I07 (IMS DL/I CPU usage by DL/I call) gives the distribution of CPU usage across IMS DL/I calls. I07 reports only CPU activity that occurred during processing of DL/I calls.

I10 (IMS DL/I WAIT time by DL/I call) identifies delays caused by wait conditions in specific DL/I calls.

I13 (IMS DL/I activity by DL/I call) shows how time is consumed in IMS-dependent regions by specific DL/I calls. All time is reported regardless of whether the time is CPU usage or wait.

Note: Reports I05, I08, and I11 are meaningful when measuring a region in which multiple IMS PSBs are scheduled.

Reports I06, I09, and I12 are meaningful when measuring a region in which multiple transactions are scheduled.

I05 (IMS DL/I CPU usage by PSB) and *I06 (IMS DL/I CPU usage by transaction)* give the distribution of CPU usage in an IMS-dependent region. Both IMS and non IMS CPU usage is reported.

I08 (IMS DL/I WAIT time by PSB) and *I09 (IMS DL/I WAIT time by transaction)* identify any delays caused by wait conditions in IMS regions. Only wait time observed when an IMS PSB is active is reported. Wait time is identified both within the processing of DL/I calls and outside of DL/I call processing.

I11 (IMS DL/I activity by PSB) and *I12 (IMS DL/I activity by transaction)* show how time is consumed in IMS-dependent regions. All time is reported regardless of whether the time is CPU usage or wait. Only time observed while an IMS PSB was active is reported. Time is identified both within the processing of DL/I calls and outside of DL/I call processing but only when an IMS PSB is active.

I14 (IMS PSB/PCB attributes) and *I15 (IMS DL/I call attributes)* give detailed information about PSBs and their PCBs (I14) or DL/I call (I15) referred to by detail lines in other reports.

Note: For reports I16, I17, I18, I19, I20, and I21, the IMS+ feature must have been enabled when the measurement was performed.

I16 (IMS transaction service time) gives information about IMS transaction service times.

I17 (IMS transaction DL/I call counts) gives information about the number of DLI calls issued by each of the measured IMS transactions.

I18 (IMS CPU/Svc time by DLI calls), I19 (IMS CPU/Svc time by PSB), I20 (IMS CPU/Svc time by Transaction), and *I21 (IMS CPU/Svc time by PCB)* give information about exact CPU times and service times for DLI calls (I18), for DLI call by PSB (I19), for DLI call by transaction (I20), or for DLI call by individual PCB (I21).

These reports are described in Chapter 11, “Application Performance Analyzer: Analyzing a job with subsystem” on page 187.

2.4.8 CICS measurement

The CICS measurement category contains six reports, listed in Table 2-9.

Table 2-9 CICS measurement reports

Report identification	Report description
E01	CICS Session Statistics
E03	CICS CPU usage by transaction
E04	CICS mean service time by transaction
E05	CICS total service time by transaction
E06	CICS service time by task ID
E07	CICS wait by transaction
E08	CICS Mean Service Time by Terminal ID
E09	CICS Total Service Time by Terminal ID

E01 (CICS session statistics) shows a summary of the CICS measurement data collected during the observation session.

E03 (CICS CPU usage by transaction) shows how CPU consumption was distributed across the CICS transactions that were executing during the observation session.

E04 (CICS mean service time by transaction) analyze the mean times spent, by the CICS transactions, that were measured during the observation session.

E05 (CICS total service time by transaction) analyze how time was spent, by the CICS transactions, that was measured during the observation session.

E06 (CICS service time by task ID) shows a chronology of occurrences of CICS transactions.

E07 (CICS wait by transaction) shows where CICS transactions were waiting.

E08 (CICS mean service time by terminal ID) analyze the mean times, by terminal ID, that were measured during the observation times.

E09 (CICS total service time by terminal ID) analyze how total time was spent, by terminal ID, that was measured during the observation time.

These reports are described in detail in Chapter 11, “Application Performance Analyzer: Analyzing a job with subsystem” on page 187.

2.4.9 DB2 measurement

The DB2 measurement category contains fourteen reports, listed in Table 2-10.

Table 2-10 DB2 measurement reports

Report identification	Report description
F01	DB2 measurement profile
F02	DB2 SQL activity timeline
F03	DB2 SQL activity by DBRM
F04	DB2 SQL activity by statement
F05	DB2 SQL activity by plan
F06	DB2 SQL statement attributes
F07	DB2 SQL wait time by DBRM
F08	DB2 SQL wait time by statement
F09	DB2 SQL wait time by plan
F10	DB2 SQL/Svc time by DBRM
F11	DB2 SQL/Svc time by statement
F12	DB2 SQL/Svc time by plan
F13	DB2 SQL threads analysis
F14	DB2 CPU by plan/stored procedure
F15	DB2 SQL CPU/Svc time by requestor location

F01 (DB2 measurement profile) gives a general overview of the DB2 measurement data. This is a good report to examine first when analyzing DB2 information. It provides an at-a-glance summary of various aspects of the measurement data and helps us choose which other reports to concentrate on.

F02 (DB2 SQL activity timeline) gives information about the chronology of SQL requests that were sampled over the duration of the measurement and to identify any calls with excessive service times.

F03 (DB2 SQL activity by DBRM), F04 (DB2 SQL activity by statement), and *F05 (DB2 SQL activity by plan)* show how time was consumed by SQL request processing. The percentage of time is reported by each module (F03), each statement (F04), or each DB2 plan (F05) that issued SQL requests.

F06 (DB2 SQL statement attributes) shows detailed information about each of the measured SQL statements. This is useful as a reference report when working with printed copies of other DB2 reports that do not show full SQL statement details.

F07 (DB2 SQL wait time by DBRM), F08 (DB2 SQL wait time by statement), and *F09 (DB2 SQL wait time by plan)* give information about WAIT time that occurred during the processing of SQL requests. The percentage of time is reported for each DBRM (F07), each statement (F08), or each DB2 plan (F09) that issued SQL requests that were issued.

Note: A prerequisite for reports F10, F11, and F12 is activation of the DB2+ option during the measurement. This option records exact SQL call counts, total SQL service time, and total SQL processing CPU time by embedded SQL statements.

F10 (DB2 SQL/Svc time by DBRM), F11 (DB2 SQL/Svc time by statement), and *F12 (DB2 SQL /Svc time by plan)* show quantification of service time by DBRM (F10), by statement (F11), or by DB2 plan (F12).

F13 (DB2 SQL threads analysis) shows information about DB2 threads observed during the sampling of SQL call activity.

F14 (DB2 CPU by plan/stored procedure) shows how CPU resource was consumed by stored procedures within each plan measured during the sampling interval.

F15 (DB2 SQL CPU/Svc time by requestor location) shows the exact SQL call counts, total SQL service time and total SQL processing SCP time by embedded SQL statements. It shows quantification by Requestor Location. This report is for DDF measurements only.

These reports are described in detail in Chapter 11, “Application Performance Analyzer: Analyzing a job with subsystem” on page 187.

2.4.10 MQ measurement

The MQSeries measurement category contains ten reports listed in Table 2-11.

Table 2-11 MQSeries measurement reports

Report identification	Report description
Q01	MQSeries activity summary
Q02	MQSeries CPU usage by queue
Q03	MQSeries CPU usage by request
Q04	MQSeries CPU usage by transaction
Q05	MQSeries service time by queue
Q06	MQSeries service time by request
Q07	MQSeries service time by transaction
Q08	MQSeries wait time by queue
Q09	MQSeries wait time by request
Q10	MQSeries wait time by transaction

Q01 (MQSeries activity summary) shows a summary of the MQSeries requests (Calls) issued during the observation session and a list of the MQSeries objects referenced by these requests.

Q02 (MQSeries CPU usage by queue), Q03 (MQSeries CPU usage by request), and *Q04 (MQSeries CPU usage by transaction)* show how CPU resources were consumed by MQSeries Requests. The percentage of CPU usage is reported by MQSeries Queue Name (Q02), by MQSeries Request (Q03), or by CICS or IMS transaction (Q04).

Q05 (MQSeries service time by queue), Q06 (MQSeries service time by request), and *Q07 (MQSeries service time by transaction)* show how time was consumed by MQSeries Requests. The percentage of time is reported by MQSeries Queue Name (Q05), by MQSeries Request (Q06), or by CICS or IMS transaction (Q07).

Q08 (MQSeries wait time by queue), Q09 (MQSeries wait time by request), and *Q10 (MQSeries wait time by transaction)* show how much wait time occurred during processing of MQSeries Requests. The percentage of time is reported by MQSeries Queue Name (Q08), by MQSeries Request (Q09), or by CICS or IMS transaction (Q10).

These reports are described in detail in Chapter 11, “Application Performance Analyzer: Analyzing a job with subsystem” on page 187.

2.4.11 Another approach to sort the APA’s reports

We can also group reports in other categories such as:

- Global information:
  - S01: Measurement profile
  - G01: Coupling Facility summary
  - I01: IMS measurement profile
  - E01: CICS session statistics
  - F01: DB2 measurement profile
  - Q01: MQSeries activity summary
- Load module:
  - S02: Load module attributes
  - S03: Load module summary
  - C02: CPU usage by module
  - C03: CPU usage by code slice
  - C06: CPU usage by task/module
  - C07: CPU usage by procedure
  - W02: WAIT time by task/module
- Task:
  - S04: TCB summary
  - S07: TCB execution summary
  - C05: CPU usage by task/category
  - C06: CPU usage by task/module
  - W01: Wait time by task/category
  - W02: Wait time by task/module
- Timeline:
  - S05: Memory usage timeline
  - S06: Data space usage timeline
  - C04: CPU usage by timeline
  - D07: DASD activity timeline
  - I02: IMS DL/I call timeline
  - I03: IMS transaction timeline
  - I04; IMS transaction activity timeline
  - F02: DB2 SQL activity timeline

- CPU time:
  - All CPU reports (C01 to C09)
  - I18: IMS CPU/Svc time by DL/I calls
  - I19: IMS CPU/Svc time by PSB
  - I20: IMS CPU/Svc time by transaction
  - I21: IMS CPU/Svc time by PCB
  - E08: CICS mean Svc time by terminal ID
  - E09: CICS total Svc time by terminal ID
  - F10: DB2 SQL/Svc time by DBRM
  - F11: DB2 SQL/Svc time by statement
  - F12: DB2 SQL/Svc time by plan
  - F14: DB2 CPU by plan/sorted procedure
  - F15: DB2 SQL CPU/Svc time by requestor location
  - Q02: MQSeries CPU usage y queue
  - Q03: MQSeries CPU usage by request
  - Q04: MQSeries CPU usage by transaction
- Wait time:
  - D08: DASD I/O wait time
  - W01: Wait time by task/category
  - W02: Wait time by task/module
  - W03: Wait time referred attribution
  - I08: IMS DL/I wait time by PSB
  - I09: IMS DL/I wait time by transaction
  - I10: IMS DL/I wait time by DL/I call
  - E07: CICS wait by transaction
  - F07: DB2 SQL wait time by DBRM
  - F08: DB2 SQL wait time by statement
  - F09: DB2 SQL wait time by plan
  - Q08: MQSeries wait time by queue
  - Q09: MQSeries wait time by request
  - Q10: MQSeries wait time by transaction

2.4.12 Real-time analysis

The Realtime Monitor facility allows you to view information about an in-progress measurement. Start this facility by selecting an active measurement from the Observation Session List (information about creating a monitoring request is in Chapter 9, “Application Performance Analyzer: Creating a monitoring request” on page 145.

Auto-refresh mode

In this mode the panel is refreshed automatically to show changing data as it is measured. In auto-refresh mode the keyboard is locked.

Note: To halt auto-refresh mode and unlock the keyboard, press the Attention key. We can then refresh the panel manually by repeatedly pressing Enter. To re-activate the auto-refresh mode, enter the PULSE (or P) primary command.

ACCUM and CURRENT modes

Some monitor views display data based on all of the accumulated (ACCUM) data for the measurement, or for the last measured *time slice* (CURRENT). `ACCUM` or `CURRENT` is displayed on the heading line of views affected by this mode setting.

Note: Enter the ACCUM (or A) command to set the mode to report accumulated data.

Enter the CURRENT (or C) command to set the mode to report on the most recent time slice.

Real-time analysis view 1: measurement overview

The measurement overview shows an at-a-glance summary of the measurement status and shows a very high-level overview of observed resource usage.

Real-time analysis view 2: CPU utilization

CPU utilization quantifies distribution of CPU usage. The quantifications are reported in two modes: overall and current.

Overall mode appears on the left side of the screen and shows accumulated quantifications based on the overall measurement. Each quantity is an overall sample count. It is also expressed as a percentage and illustrated by a histogram.

Current mode appears on the right side of the screen and shows quantifications representing the last measured time slice. Each quantity is a sample count for the time slice and is also illustrated by a histogram.

Real-time analysis view 3: measurement environment

The data reported here is static and shows the measurement request parameters and information about the measurement environment.

Real-time analysis view 4: CPU active modules

CPU active modules quantify distribution of CPU usage by module. The quantifications are reported in two modes: overall and current.

Real-time analysis view 5: data management service time

The quantifications are reported in two modes: overall and current.

In ACCUM mode, files for which EXCPs were processed during the measurement are shown.

In current mode, files for which EXCPs were processed since the last data refresh are shown. In both cases the detail lines are sorted in descending sequence by EXCP count since the last data refresh.

3

Introducing Debug Tool, Debug Tool Utilities, and Advanced Functions

Debug Tool provides interactive debugging capabilities for debugging compiled zSeries applications. Its program debugging, testing, and analysis capabilities include the ability to let application developers examine, monitor, and control the execution of application programs written in COBOL, PL/I, C, C++, and Assembler on a z/OS system.

Available interfaces, along with program compile, link-edit, and runtime options, plus frequently used Debug Tool commands, are described in this chapter.

This chapter is organized in two parts:

- Debug Tool
- Debug Tool Utilities and Advanced Functions

Debug Tool is included in Debug Tool Utilities and Advanced Functions.

3.1 Debug Tool

Debug Tool helps you test programs by allowing you to examine, monitor, and control the execution of programs written in assembler, C/C++, COBOL, or PL/I on a z/OS system.

Applications can also include other languages. For such applications, Debug Tool provides a disassembly view that lets you debug, at the machine code level, the corresponding portions of applications. Of course, in the disassembly view, debugging capabilities are limited.

Debug Tool can be used to debug programs in a batch mode, interactively in a full-screen mode, or in a remote debug mode. You must use the correct compiler options to be able to use Debug Tool.

3.1.1 Debug Tool interfaces

The terms batch mode, remote debug mode, and full-screen mode identify the types of debugging interfaces provided by Debug Tool.

Batch mode

To run Debug Tool in batch mode, a command file is prepared in advance, with a predefined series of Debug Tool commands that are to be performed on a running batch application. The TEST runtime option gives control of your program to Debug Tool.

No terminal input or user interactive intervention is possible in a batch debugging session.

The results are saved in a log data set and can be reviewed and analyzed when a batch debugging session is finished.

There are several ways to define the TEST runtime option with the commands file specified, including:

- In the PARM parameter of the JCL EXEC statement
- Using the PLIXOPT string in the PL/I source application
- Using CEEUOPT CSECT linked with an application

Remote debug mode

In remote debug mode, the host application starts Debug Tool, which then uses a TCP/IP connection to communicate with a remote debugger on your Windows workstation. Not all compilers are compatible with this mode.

Debug Tool, in conjunction with a remote debugger, provides users with the ability to debug host programs, including batch programs, through a graphical user interface (GUI) on the workstation.

The following remote debuggers are available:

- Compiled Language Debugger component of WebSphere Developer for zSeries (WDz):

  This remote debugger is the recommended choice since it offers more functionality than the IBM Distributed Debugger.

- Compiled Language Debugger component of WebSphere Developer Debugger for zSeries (WDDz):

  This remote debugger offers similar functionality to WDz, but without the comprehensive application development support provided with WDz. Both are built on an Eclipse platform.

- IBM Distributed Debugger:

  This remote debugger is available through several products, for example:

  - OS/390 C/C++ Productivity Tools
  - VisualAge for Windows family of products

All remote debuggers run on Windows NT® 4.0, Windows 2000, and Windows XP.

WebSphere Developer for zSeries Debugger

This debugger is available as a function of the IBM WebSphere Developer for zSeries (WDz). When initially started, the screen is formatted as shown in Figure 3-1.

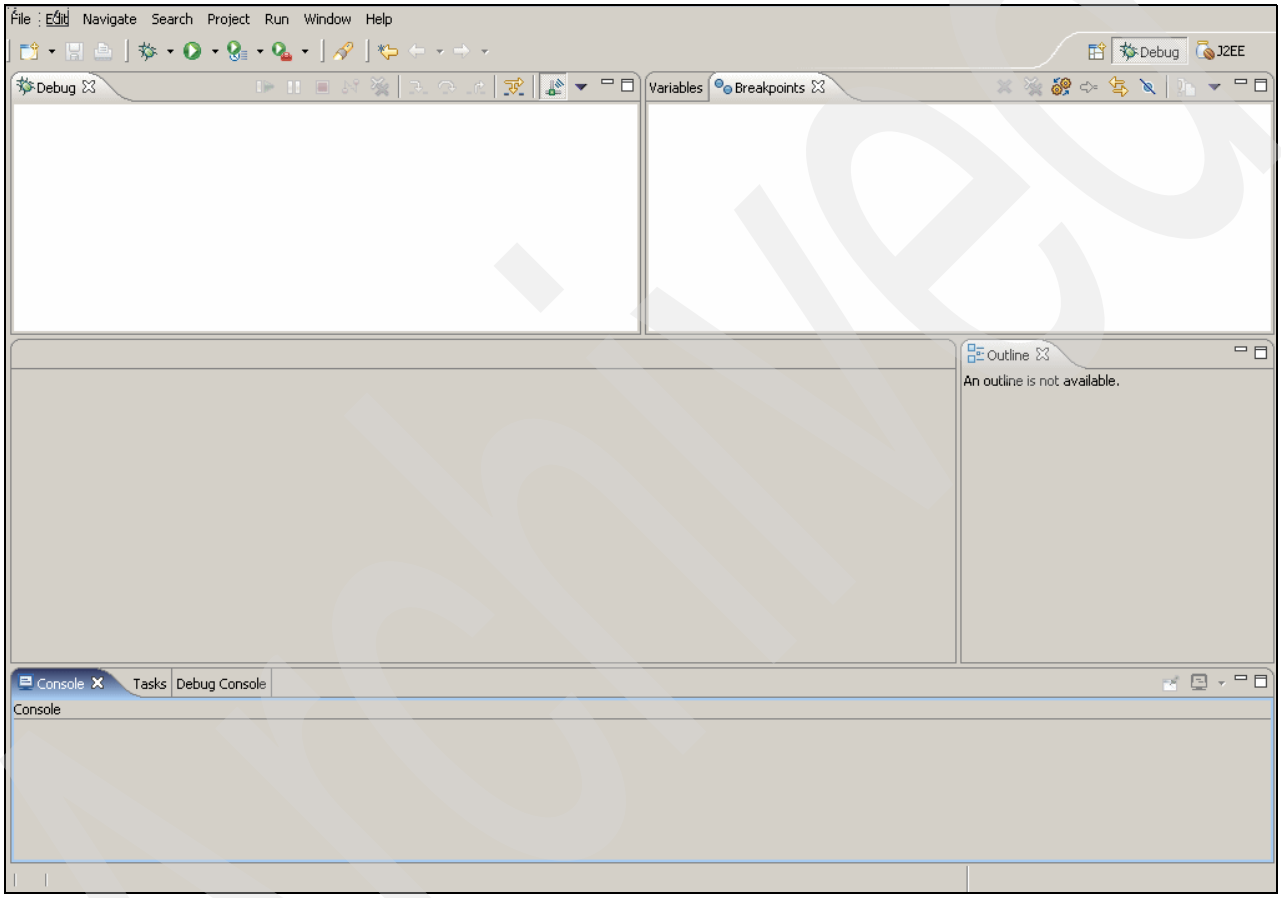


Figure 3-1 The WebSphere Developer for zSeries initial Debug perspective

At the top-right of the Debug window (upper left-hand corner), there is a series of icons. The Listener button is shown in Figure 3-2. This must be selected in order for the remote debugger to work as expected.

Figure 3-2 The Listener button

By default, WDz begins to listen on port 8001 at startup. The next step in debugging is going back to your TSO, batch, IMS, or CICS session and starting, with a suitable TEST runtime option, the job you wish to run. The TEST runtime option must specify the TCP/IP address of the machine that is running WDz. In this case, an extract of the JCL used to invoke a debug session and initiate contact with WDz is shown in Figure 3-3.

```
EDIT - Edit Setup File 'MLDUCKW.COVERAGE.DTCU'                 Row 1 to 5 of 5
Command ===> ____________________________________________________ Scroll ===> CSR

Load Module Name _________
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs / OS/VS COBOL - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,*,PROMPT,TCPIP&9.49.137.66%8001:*)'

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)              DISP
___ _______ ___ _ ********** Top of Data **********                    _____
___ COMPFIL  1   _ 'CHABERT.TRADER.COMPFILE'                           SHR
___ CUSTFILE 1   _ 'CHABERT.TRADER.CUSTFILE'                           SHR
___ INSPPREF 1   _ 'CHABERT.DEBUG.PREF(TRAD1)'                         SHR
___ REPOUT   1   _ SYSOUT=*                                            _____
******************************** Bottom of data ********************************
```

Figure 3-3 Invoking the program to be debugged on the mainframe

In this example we have invoked the program in batch, but we could have invoked it in TSO just as easily. The TEST runtime option contains enough information so that the runtime on the mainframe can contact WDz on our PC.

The WDz screen pops up, along with a dialog box to warn about the state of the program, as shown in Figure 3-4.

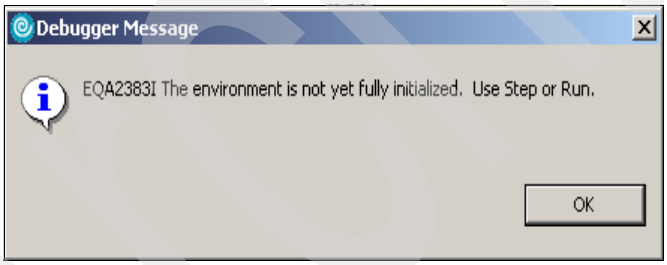


Figure 3-4 WDz Initial Program Status message

Once we have clicked **OK** we can STEP into the program. This is done using the left-hand button in the following series of three buttons, extracted from the row of icons in the Debug window, as shown in Figure 3-5.

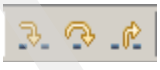


Figure 3-5 WDz STEP buttons

We now have a fully functional point-and-click debugging environment, with many different views such as breakpoints, variables, registers, and so forth, as shown in Figure 3-6.

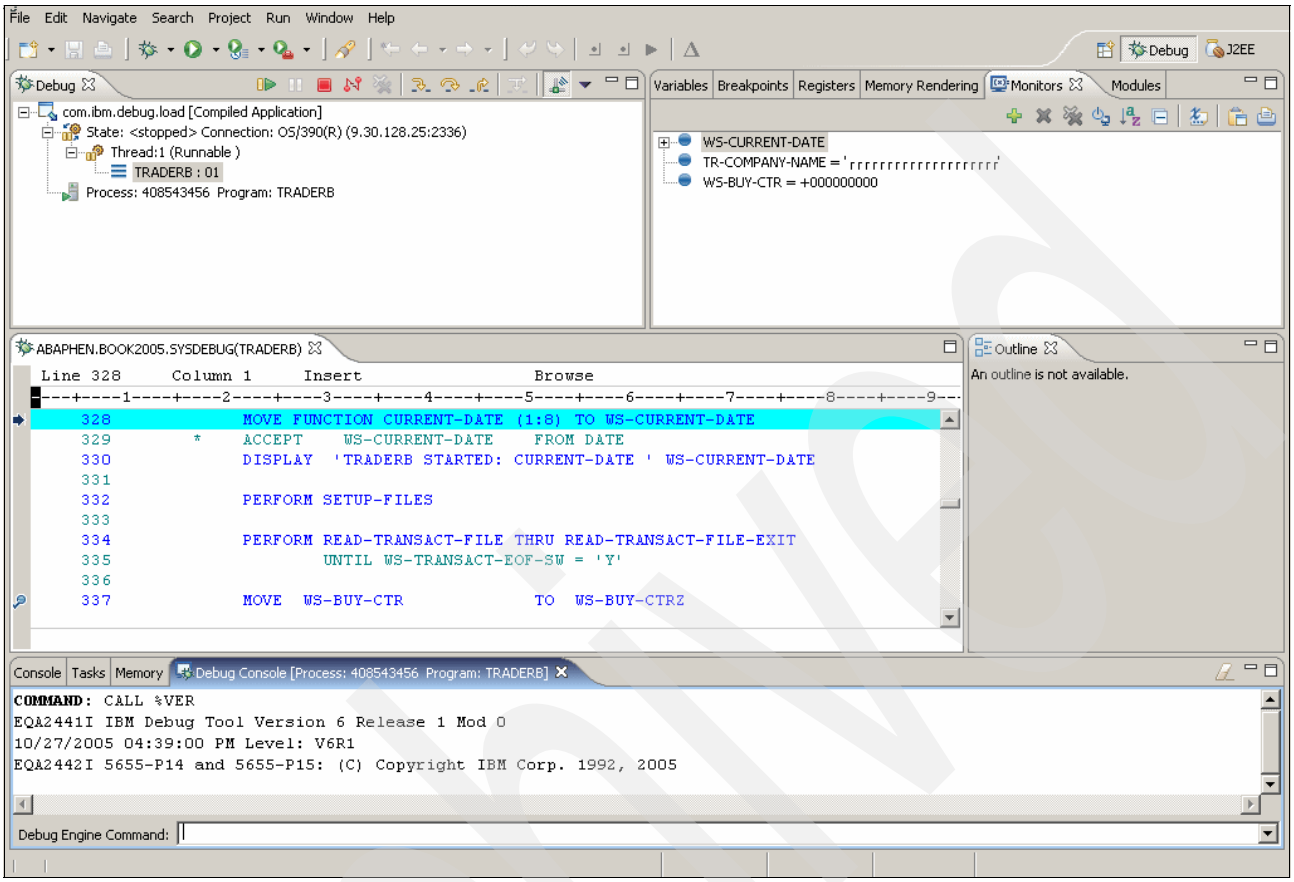


Figure 3-6 The WDz Debug screen

WebSphere Developer Debugger for zSeries Debugger

This debugger is a subset of WDz functions that still provides a workstation graphical interface to Debug Tool and Debug Tool Utilities and Advanced Functions.

WDDz, integrated with Debug Tool or Debug Tool Utilities and Advanced Functions, provides Eclipse-based interactive source-level debugging capabilities for debugging compiled zSeries applications. Its program debugging, testing, and analysis capabilities include the ability to let developers examine, monitor, and control the execution of application programs written in COBOL, PL/I,C, C++, and Assembler on a z/OS system.

IBM Distributed Debugger

Any version of the IBM Distributed Debugger should work with Debug Tool and all of its supported languages.

Using this interface, your host application running under the control of Debug Tool starts a TCP/IP connection to a Windows NT, 2000, or XP workstation on which the IBM Distributed Debugger program was started and listens for requests on a specified port (default is 8000). This interface is supported by all environments including CICS, IMS, and UNIX System Services (USS).

Note: In some TCP/IP installations, a SYSTCPD DD card might be required to point to your installation's TCPIPDATA.

Debugging with the IBM Distributed Debugger is a two-step process:

1. Start the remote debugger on your workstation. For the IBM Distributed Debugger on a workstation, this involves a command like:

   ```
   C:\IBMDebug\bin\idebug.exe -qdaemon -quiport=8000
   ```

2. Run your program using the TEST runtime option. The runtime option to be used must define your workstation's TCP/IP address. With a COBOL program, this might be:

   ```
   /TEST(ALL,*,PROMPT,VADTCPIP&9.30.40.117%8000:*)
   ```

Full-screen mode

Debug Tool provides an interactive full-screen interface on a 3270 device in one of the following ways:

- Full-screen mode through a VTAM terminal:

  Debug Tool full-screen mode through a VTAM terminal requires that you know the LU name of the terminal that your debug session will run on.

- Full-screen mode through the Terminal Interface Manager:

  Debug Tool Terminal Interface Manager enables you to associate a user ID with a VTAM terminal, removing the necessity to update your runtime parameter string whenever the VTAM terminal LU name changes. The Terminal Interface Manager is a VTAM application that must be started before users can access it.

Both full-screen modes require that the correct VTAM customization is completed to ensure that basic full-screen mode is operational. Once you have your debug session started in either full-screen manner, the visual interface is the same. The debugging information can be displayed in three windows on the screen:

- Source window: Displays the program source or listing
- Log window: Displays a record of commands and other interactions between Debug Tool and the program
- Monitor window: Indicates changes in the program

Each of these windows can be expanded to full screen, re-sized, or hidden.

Programs written in any of the languages supported by Debug Tool can be debugged in full-screen mode.

Source window

The Source window displays the source file or listing. It has four parts:

Header area Identifies the window, shows the compile unit name, and shows the current position in the source or listing.

Prefix area Occupies the left-most eight columns of the Source window. Contains statement numbers or line numbers that can be used when referring to the statements in your program. The prefix area can be used to set, display, and remove breakpoints with the prefix commands AT, CLEAR, ENABLE, DISABLE, QUERY, and SHOW.

Source display area Shows the source code (for a C/C++ program), the source listing (for a COBOL or PL/I program), a pseudo assembler listing (for an assembler program), or the disassembly view (for programs without debug information) for the currently qualified program unit. If the current executable statement is in the source display area, it is highlighted.

Suffix area A narrow, variable-width column at the right of the screen that Debug Tool uses to display frequency counts. It is only as wide as the largest count it must display. The suffix area is optional. To show the suffix area, enter `SET SUFFIX ON`. To hide the suffix area, enter `SET SUFFIX OFF`. It can also be set on or off with the Source Listing Suffix field in the Profile Settings panel.

Log window

The Log window records and displays user interactions with Debug Tool. All commands that are valid in line mode, and their responses, are automatically appended to the Log window. The following commands are not recorded in the Log window:

- PANEL
- FIND
- CURSOR
- RETRIEVE
- SCROLL
- WINDOW
- IMMEDIATE
- QUERY prefix command
- SHOW prefix command

If SET INTERCEPT ON is in effect for a file, that file's output also appears in the Log window. You can exclude STEP and GO commands from the log by specifying SET ECHO OFF.

By default, the Log window keeps 1000 lines for display. To change this value, enter `SET LOG KEEP` *n*, where *n* is the number of lines you want kept for display. The maximum number of lines is determined by the amount of storage available.

Monitor window

The Monitor window is used to continuously display output from the MONITOR LIST, MONITOR QUERY, MONITOR DESCRIBE, and SET AUTOMONITOR commands.

If this window is not open, Debug Tool opens it when the MONITOR or SET AUTOMONITOR commands are entered. Its contents are refreshed whenever Debug Tool receives control and after every Debug Tool command that can affect the display.

When the MONITOR command is issued, it is assigned a reference number between 1 and 99, then added to the monitor list. You can specify the monitor number. However, it must either replace an existing monitor number or be the next sequential number.

When you issue the SET AUTOMONITOR ON command (if available), the following line is displayed at the bottom of the list of monitored variables:

```
********** AUTOMONITOR **********
```

Variables that are added to the Monitor window as a result of the SET AUTOMONITOR command are displayed underneath this line.

While the MONITOR command can generate an unlimited amount of output, bounded only by your storage capacity, the Monitor window can display a maximum of only 1000 scrollable lines of output.

If a window is not wide enough to show all the output it contains, either scroll right (to scroll the window to the right) or zoom (to make it fill the screen) can be used.

In most cases, you can update the values of monitored variables by typing new values over the displayed values.

A sample of Debug Tool screen in the full-screen mode is shown in Figure 3-7.

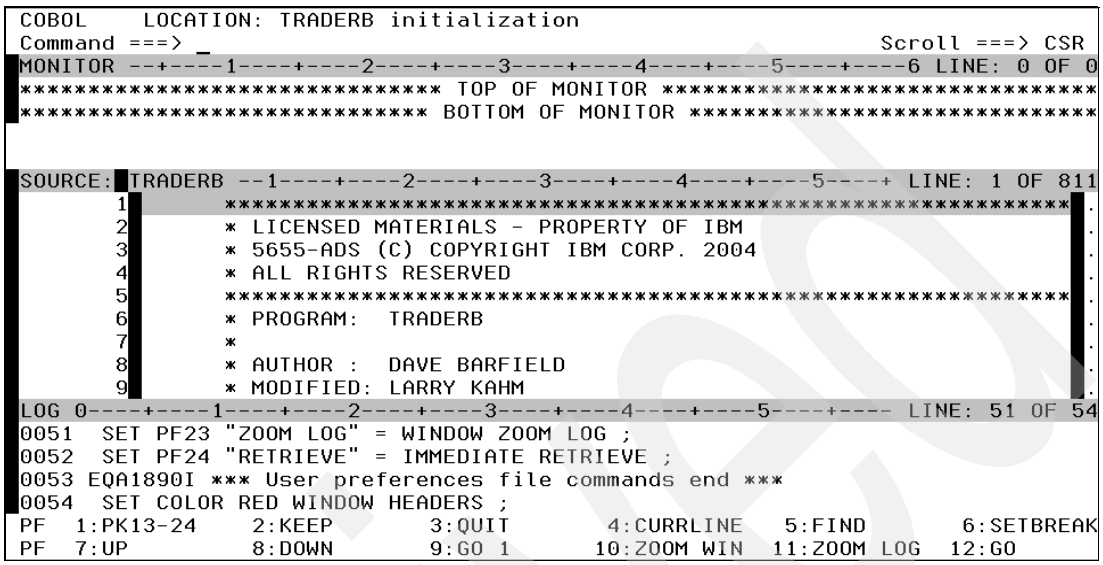


Figure 3-7 Debug Tool full-screen session

3.1.2 Compiler options

Each programming language has a comprehensive set of compiler options. It is very important to use the correct compiler options to debug the program:

C/C++ TEST(ALL) provides maximum debugging capability. There are suboptions to refine debugging capabilities.

COBOL TEST(ALL,SYM) provides maximum debugging capability. There are suboptions to refine debugging capabilities. Some suboptions are used only with a specific version of COBOL. When using Enterprise COBOL for z/OS and OS/390 Version 3 or COBOL for OS/390 and VM Version 2 Release 2 compilers, the TEST(NONE,SYM,SEPARATE) compiler option retains most of Debug Tool's capabilities. The suboption SEPARATE instructs the compiler to store debugging information and symbol tables in a separate file. The suboption NONE specifies that there is no compiled-in hooks, so the Dynamic Debug facility must be activated during a debug session.

PL/I TEST(ALL,SYM) provides maximum debugging capability. Programs compiled with the PL/I for MVS or OS PL/I compilers must specify the SOURCE suboption. The suboptions BLOCK, STMT, PATH, and ALL regulate the points in which compiler inserts hooks. The suboption SYM controls the insertion of symbol tables into the object file. These tables are used by Debug Tool to obtain information about program variables. The syntax for the TEST compiler option of the Enterprise PL/I compilers is slightly different. Refer to the documentation that corresponds to the version of the compiler you are using for a description of the TEST compiler option.

Assembler The ADATA option must be specified. This option generates a SYSADATA file, which the EQALANGX postprocessor requires to create a debug file (also called the EQALANGX file).

3.1.3 Link-edit options

In most cases, Debug Tool does not require specific link-edit options for application programs.

When using the DTCN transaction to manage debugging profiles in CICS, the main programs to be debugged should be link-edited with the object module EQADCCXT if they are written in PL/I or C/C++. When using the CADP transaction, which is available with CICS Transaction Server for z/OS V2.3 and later, this is not required.

3.1.4 Runtime TEST option

Note: The following information is based on the description of the TEST runtime option provided in *Debug Tool V7R1 Reference and Messages*, GC18-9536.

About runtime TEST option

The TEST runtime option is used to specify the conditions under which Debug Tool will assume control of an application. The basic format of the instruction is as follows:

NOTEST Specifies that Debug Tool is not started at program initialization. However, it can still be started with CEETEST, PLITEST or __ctest(). The suboptions specified with NOTEST are used when Debug Tool is started (if it is started). Note that if the TEST option is specified somewhere that has a higher precedence than where the NOTEST option is, the values on the NOTEST option are taken as defaults.

TEST Indicates that Debug Tool is given control according to the specified sub-options.

test_level

The test_level suboption has three possible values:

ALL This default value specifies that Debug Tool gains control, even without defined breakpoints, at:

The attention function
Any Language Environment condition of severity 1 or above
Application termination

ERROR Without a defined AT OCCURRENCE for a particular condition, Debug Tool only gets control at the occurrence of:

The attention function
Any Language Environment condition of severity 1 or above
Application termination

NONE This specifies that no condition will cause Debug Tool to gain control without a defined AT OCCURRENCE for a particular condition or AT TERMINATION.

commands_file

The commands_file designator is a valid DD name or file name that gives the name of the primary commands file for this program run. If this parameter is empty, requests for commands go to the user terminal.

If an asterisk (*) is specified instead of a commands file, then no commands file is expected.

prompt_level

The prompt_level suboption is used to specify whether an initial commands list is unconditionally executed during program initialization or to specify particular Debug Tool commands:

PROMPT	This default value specifies that Debug Tool is invoked at Language Environment initialization.
NOPROMPT	Specifies that Debug Tool is not invoked at Language Environment initialization.
*****	Equivalent to NOPROMPT.
;	Equivalent to PROMPT.
command_list	A character string that specifies a valid Debug Tool command. Maximum of 250 characters. It should be enclosed in single or double quotation marks whenever it contains embedded blanks, commas, or parenthesis. The use of a preferences file is recommended rather than putting a command list in the third suboption.

preferences_file

The final suboption is the preferences file. This suboption controls the interface and location of the debugger as well as the location of a preferences file that becomes the first source of Debug Tool commands after Debug Tool has started. This suboption has a complex format as shown in Figure 3-8.

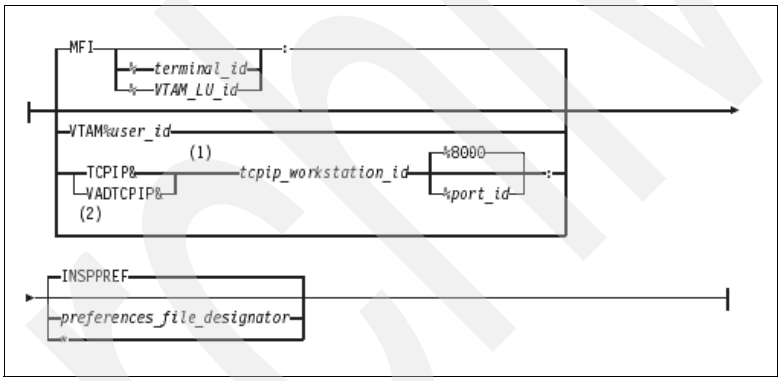


Figure 3-8 Preferences file syntax

Notes on Figure 3-8:

- (1) Specifies remote debug mode using WebSphere Developer for zSeries (WDz) or WebSphere Developer Debugger for zSeries (WDDz)
- (2) Specifies remote debug mode using the VisualAge remote debugger or the IBM Distributed Debugger

The terms used are as follows:

- MFI (main frame interface): Specifies that Debug Tool should be started in full-screen mode through a VTAM terminal for your debug sessions.
- *terminal_id* (CICS only): Specifies up to a four-character terminal ID to receive Debug Tool screen output during dual terminal debugging of a CICS application.
- *VTAM_LU_id* (full-screen mode using a VTAM terminal only): This is a VTAM logical unit (LU) name for a terminal. This cannot be used to debug CICS applications. It is used for full-screen debugging. The application continues in its original location and the new

unused VTAM terminal is used for full screen mode through a VTAM terminal. This can be used for batch or IMS debugging as well as traditional debugging.

- *user_id* specifies the user ID that was used to log on to Debug Tool Terminal Interface Manager.
- INSPPREF (or blank) is the default DD name for the preference file.
- *preferences_file_designator*: This is a valid DD name, data set name, or file name specifying the name of the preference file. The preference file is a command file that can be used to specify Debug Tool commands to be executed on entry to your environment.
- *: Instead of a preference file, the asterisk indicates that no preferences file is supplied.
- *%port_id*: Specifies the TCP/IP port on your workstation to be used by the remote debugger. The default port for the WDz and WDDz Compiled Language debuggers is %8001. The default port for the IBM Distributed Debugger or the VisualAge remote debugger is %8000.

Sample runtime options

Sample runtime options are:

- NOTEST

  Debug Tool is not started unless there is a call to CEETEST, PLITEST, or __ctest().

- NOTEST(ALL,MYCMDS,*,*)

  Debug Tool is not started at initialization. If it is started by a call to CEETEST, PLITEST, or __ctest(), the suboptions are used and the instructions in the file are allocated to DD name MYCMDS.

- TEST

  Searches through runtime options specified in any level (CEEUOPT, pragma runopts, CEEROPT) and brings up the debugger using the options found on a prior TEST instruction or the default TEST values.

- TEST(ALL,*,*,*)

  Debug Tool is not started initially, but any condition in the code (or a call to CEETEST, PLITEST, or __ctest) causes it to be started. No preference file or command file is expected.

- TEST(NONE,,*,*)

  Debug Tool is not started until a call to CEETEST, PLITEST, or __ctest() occurs.

- TEST(ALL,test.scenario,PROMPT,prefer)

  Debug Tool is started after environment initialization but before program initialization. The first commands executed are found in the file referred to by the DDNAME prefer.

- TEST(ALL,,,MFI%F100:*)

  For CICS Dual Terminal and Batch CICS transactions, Debug Tool is started at CICS terminal F100 after initialization. Alternatively, F100 could be a very short VTAM LU ID that is used to define a terminal for debugging a batch or IMS transaction. No preference file is expected.

- TEST(ALL,,,MFI:foo) or TEST(ALL,,,foo)

  These are equivalent instructions when used in TSO that cause the full screen interface to be used in single screen mode (the same place that the program was called from). In both cases, the preference file is the file referenced by the foo DD.

- TEST(ALL,,,MFI%LU000001:*)

  For environments other than CICS. The full screen mode through a VTAM terminal debugger is started on the VTAM LU whose ID is LU000001. The LU must be known to VTAM and not be in session when the debugger is started.

- TEST(,,,VADTCPIP&9.30.62.149%8000:*)

  Bring up the IBM Distributed Debugger or the VisualAge remote debugger listening on channel 8000 on the terminal at TCP/IP address 9.30.62.149.

- TEST(,,,TCPIP&cello%8001:*)

  Bring up the WDz or WDDz remote debugger listening on port 8001 on the machine named cello. This method does not work in all environments and requires an up-to-date name server to be available. It was found that using the actual machine address is more consistently effective.

Specifying TEST runtime option

Language Environment has several ways of specifying the runtime options for a program. The following list gives them in ascending order of precedence (that is, things lower in the list can override things higher in the list):

- CEEDOPT: Language Environment options specified at installation
- CEEROPT: Region-wide CICS or IMS default options
- CLER: Transaction under CICS
- CEEUOPT: Also #pragma runopts, or PLIXOPTS
- Command Line, or _CEE_RUNOPTS
- CEEBXITA: Used by IMS Single Point of Control (SPOC) and Debug Tool IMS utilities
- Language Environment Storage Tuning User Exit
- Options defined at installation time that have the non-overridable attribute

For more information about how to specify a Language Environment runtime option, see the *Language Environment Programming Guide*.

In some cases, where runtime options cannot be passed to the Language Environment, a CEEUOPT must be generated including a specific TEST runtime option. In Example 3-1 the original sample can be found as member CEEUOPT on your SCEESAMP.

Example 3-1 Defining runtime options using CEEUOPT

```
*/*****************************************************************/
*/* LICENSED MATERIALS - PROPERTY OF IBM                          */
*/*                                                               */
*/* 5694-A01                                                      */
*/*                                                               */
*/* (C) COPYRIGHT IBM CORP. 1991, 2001                            */
*/*                                                               */
*/* US GOVERNMENT USERS RESTRICTED RIGHTS - USE,                  */
*/* DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP               */
*/* SCHEDULE CONTRACT WITH IBM CORP.                              */
*/*                                                               */
*/*****************************************************************/
CEEUOPT  CSECT
CEEUOPT  AMODE ANY
CEEUOPT  RMODE ANY
         CEEXOPT TEST(ALL,*,PROMPT,MFI%LUOTCP08:INSPPREF)
         END
```

This member must be customized to reflect the actual terminal ID to be used in the debugging session. The name of this member also can be customized to reflect the intention of its use.

When specifying the TEST runtime option on a JCL PARM there are dependencies on what is the language of the program's main entry point:

- For C/C++ and PL/I, the PARM should start with the runtime options and have a slash (/) before the program parameters.
- For COBOL, the PARM should start with the program parameters and have a slash before the runtime options.

You can specify RPTOPTS(ON) in the JCL PARM to have the runtime options report generated. This report lists all runtime options that were in effect when the program was executed.

For CICS, you can use the Language Environment provided CICS transaction CLER to have runtime options displayed on the terminal, as shown in Figure 3-9.

```
CLER                                                          PAGE 5 OF 6

          Language Environment Region Level Runtime Options.

Current Settings

LAST WHERE SET          OPTIONS
--------------------    ------------------------------------------------------
Installation default    TERMTHDACT(TRACE,CESE,96)
Installation default  NOTEST(ALL,'*',"PROMPT","INSPPREF")
Installation default    THREADHEAP(4096,4080,ANYWHERE,KEEP)
Installation default    THREADSTACK(OFF,4096,4080,ANYWHERE,KEEP,4096,4080)
Installation default    TRACE(OFF,4096,DUMP,LE=0)
Installation default    TRAP(ON,SPIE)
Installation default    UPSI(00000000)
Installation default  NOUSRHDLR(,)
Installation default    VCTRSAVE(OFF)
Installation default    VERSION()
Installation default    XPLINK(OFF)
Installation default    XUFLOW(AUTO)



PF:  1=Help    3=Quit    7=Back    8=Forward    10=RPT->CES
```

Figure 3-9 Transaction CLER used to display runtime options

3.1.5 Special files

There are four special files used by Debug Tool in full-screen mode:

- Save file (INSPSAFE)

  This file, if allocated by the user, is used by Debug Tool to save the sizes of panels, colors, PF keys setting, and so forth, between debugging sessions. (Note that CICS does not support this file.)

- Preference file (INSPPREF)

  This file contains Debug Tool commands used to customize the debugging session. The information about the user's preference file is passed to Debug Tool by specifying it in the TEST runtime option.

- Commands file (INSPCMD)

  This file contains Debug Tool commands that control the debugging session. It can be used to set breakpoints or set up monitoring for variables. The information about this file should also be specified in the TEST runtime option.

- Log file (INSPLOG)

  This file is used by Debug Tool to record the progress of the debugging session. The results of the executed commands are saved as comments, which allows you to use the log file as a commands file in later debugging sessions. Since this file is written to by Debug Tool, we recommend that you allocate it as a sequential file, to eliminate any contentions.

There is one additional special type of file used by Debug Tool. A separate debug file SYSDEBUG is produced by the compiler when compilation is performed with the SEPARATE suboption of the TEST compiler option. Currently this option is available only for COBOL for OS/390 and VM and Enterprise COBOL compilers.

3.1.6 Global preferences file enhancement

Debug Tool provides a mechanism where an installation-wide default preferences file can be specified and processed. The purpose is to have:

- A mechanism that is easy to set up at Debug Tool installation and customization time, but that is transparent to Debug Tool users.
- Consistent tailoring of the debug session, such as PF key assignments and window configuration. It provides a set of installation-wide preferences to all users.

For users who want personal customization, the existing preferences file parameter in the TEST runtime option provides a way to add additional preferences or override the global settings.

Restrictions

The mechanism works in the following circumstances:

- Debug Tool supported host subsystems (TSO, Batch, CICS, and IMS)
- Full-screen debug mode, but not in the remote debug mode (like the current INSPPREF)

Global preferences file location

The global preferences file is a sequential file or a PDS member residing on the host. The name of the file is coded in an Assembler CSECT called EQAOPTS. During Debug Tool installation and customization, you have to code the name of the global preferences file in the EQAOPTS CSECT, assemble it, and build the load module, EQAOPTS. The load module is then placed in a private data set concatenated in the *load module search path* before hlq.SEQAMOD.

During initialization time, Debug Tool loads in the EQAOPTS module and retrieves the global preferences file name.

Using EQAOPTS options file

EQAOPTS uses the EXAXOPT macro to define the global preferences. Two options are available as shown in Example 3-2.

Example 3-2 EQAOPTS

```
EQAOPTS  CSECT       ,
EQAOPTS  AMODE      31
EQAOPTS  RMODE      ANY
                    EQAXOPT  GPFDSN,'DEVELP.TEST.GLBLPREF'
                    EQAXOPT  SVCSCREEN,CONFLICT=NOOVERRIDE
                    EQAXOPT  END
                    END,
```

The options are:

- Global preferences file data set name:

  GPFDSN provides the data set name.

  To have a consistent tailoring of the debug session such as PF key assignment, window configuration, or other installation-wide default preferences, the global preferences file enforces that every debug session is initialized with the preferences in the global preferences file.

- SVC screening filter:

  SVCSCREEN(parm1,parm2)

  parm1 Enablement, possible values are ON and OFF

  parm2 Conflict resolution, possible values are CONFLICT=OVERRIDE and CONFLICT=NOOVERRIDE

  The filter allows you to enable SVC screening and to override the SVC screening already put in place by other vendor products. More information is available in the *Customization Guide*.

Global preferences file content

Debug Tool commands allowed in the current preference file (INSPPREF) are eligible in the global preferences file.

Function

Debug Tool processes the global preferences file at initialization time, like the existing user preferences file and the commands file. The order of processing is as follows:

1. Global preferences file
2. User preferences file
3. Commands file

If a command is specified multiple times in a file, or in multiple files, the last instance is used.

If a user adds or modifies a preference by issuing the command directly in the command line, it is valid only in the current session and not persistent across sessions.

3.1.7 Frequently used commands

This section describes several commonly used commands. In these examples, all of the commands are entered on the command line and the results are displayed in the log window.

?

The ? command displays a list of all commands or, if used in combination with a command, displays a list of available options for that command.

AT

The AT command defines a breakpoint. The application program's execution is temporarily suspended when the point defined in this command is reached. You can review the processing that has already taken place or issue other Debug Tool commands.

Example:

```
at line 334 list "about to setup files";
go;
```

Result:

```
 AT LINE 334
   LIST "About to set up files" ;
 GO ;
EQA1140I About to set up files
```

AT CHANGE

The AT CHANGE command instructs Debug Tool to halt execution of the program whenever the contents of the defined variable are changed.

Example:

```
at change any-variable
```

CLEAR

The CLEAR command removes the actions of previously issued Debug Tool commands. This includes breakpoints.

Examples:

```
clear at;

clear log;
```

Note: The CLEAR command cannot clear the contents of a log file directed to SYSOUT in a batch job.

COMPUTE (COBOL)

The COMPUTE command assigns the value of an arithmetic expression to a WORKING-STORAGE variable.

Example:

```
compute holdings = dec-no-shares * 10;
```

In PL/I and C/C++, the same would be done by using the assignment; for example:

```
HOLDINGS = DEC_NO_SHARES * 10*
```

DESCRIBE

The DESCRIBE command displays information about the application program, variables, and the environment.

Example:

```
describe attributes ws-current-date;
```

Result:

```
  DESCRIBE ATTRIBUTES WS-CURRENT-DATE ;
 EQA1102I   ATTRIBUTES for WS-CURRENT-DATE
 EQA1105I     Its length is 8
 EQA1103I     Its address is 089826CD
 EQA1112I     02 TRADERB:>WS-CURRENT-DATE
 EQA1112I       03 TRADERB:>WS-YR   XXXX DISP
 EQA1112I       03 TRADERB:>WS-MM   XX DISP
 EQA1112I       03 TRADERB:>WS-DD   XX DISP
```

DISABLE/ENABLE

The DISABLE command makes the AT breakpoint inoperative, but does not clear it. You can ENABLE it later without typing the entire command again.

Example:

```
disable at statement 334;
```

GO

The GO command instructs Debug Tool to start or resume running the program. The program executes until a breakpoint is reached or a condition occurs.

GOTO

The GOTO command causes Debug Tool to resume program execution at the specified statement ID. The code in between is skipped. The program then continues to execute until a breakpoint is reached or a condition occurs. Note how this differs from JUMPTO and RUNTO.

JUMPTO

The JUMPTO command instructs Debug Tool to transfer control to the specified statement ID, where execution stops. The code in between is skipped. Note how this differs from GOTO and RUNTO.

LIST

The LIST command displays information about a program, such as the values of variables, frequency information, and the like.

Use parenthesis around working storage variables to prevent any confusion with actual LIST operands.

Example:

```
list (ws-current-date);
```

MONITOR LIST

The MONITOR LIST command allows you to observe changes to variables in the Monitor window while the program executes.

Example:

```
monitor list dec-no-shares;
```

MOVE (COBOL)

The MOVE command transfers data from one area of storage to another. This allows you to manipulate the contents of WORKING-STORAGE variables, and possibly alter the flow of the program as it executes.

Example:

```
move 250 to dec-no-shares;
```

In PL/I and C/C++, the same would be achieved by using an assignment; for example:

```
DEC_NO_SHARES = 250;
```

QUERY

The QUERY command displays the values of Debug Tool settings and information about the current program. There are more than 30 forms to this command.

Example:

```
query location;
```

Result:

```
  QUERY LOCATION ;
 EQA1242I You are executing commands in the STATEMENT TRADERB ::> TRADERB :
           334.1 breakpoint.
 EQA1238I The current location is TRADERB ::> TRADERB :> 334.1.
```

RUN

The RUN command is synonymous with the GO command.

RUNTO

The RUNTO command instructs Debug Tool to resume execution until the specified statement is reached, where execution stops. Note how this differs from GOTO and JUMPTO.

SET

The SET command sets various switches that affect the operation of Debug Tool.

Example:

```
set echo off;
```

Result: STEP and GO statements do not appear in the log window, but do go to the log file.

STEP

The STEP command causes Debug Tool to execute one or more statements in the program.

Example:

```
step 5;
```

Result: Debug Tool executes five statements, one statement at a time.

3.1.8 Other useful commands

This section identifies some other commands that can be useful when debugging.

SET AUTOMONITOR (COBOL and PL/I)

Automonitor support automatically displays the variables referenced in the current statement and is available only when Debug Tool Utilities and Advanced Functions is installed. To use this feature, you must enable automonitoring with the following command:

```
SET AUTOMONITOR ON ;
```

Refer to *Debug Tool for z/OS Reference and Messages*, SC18-9304, for more information about specific supported compilers and required PTFs.

GO BYPASS

The GO BYPASS command instructs Debug Tool to resume running the program after a condition occurred, hereby ignoring the condition.

PLAYBACK (COBOL)

Playback support allows you to replay and review the application paths and data values starting at the point where you began recording. You can simulate the backward execution of the application and review application data values using appropriate Debug Tool commands. To do this, playback must be enabled using the following Debug Tool command:

```
PLAYBACK ENABLE ;
```

This command starts playback recording. To view the recorded material, use the following commands:

- PLAYBACK START;

  Replay the statements starting at the current statement.
- STEP; (or PF2)

  Move backward one statement.
- PLAYBACK FORWARD; PLAYBACK BACKWARD;

  Change the replay direction. STEP moves in the direction last set.
- PLAYBACK STOP;

  Return to the point where the playback was started (using PLAYBACK START).
- PLAYBACK DISABLE;

  Stop recording.

This command is available only with Debug Tool Utilities and Advanced Functions.

WINDOW CLOSE/OPEN/SIZE/ZOOM

This command closes, opens, resizes, or toggles between the full screen and the currently defined size of Debug Tool full-screen mode Log, Monitor, and Source windows. The affected window is defined by the cursor position or can be added as a next parameter in the command itself.

3.1.9 Finishing a Debug Tool session

There are several ways to finish working with Debug Tool. It is important to choose an appropriate one because it affects what actions will be performed in relation to the databases used in the program.

QUIT

Soft termination of the program occurs at the current statement, with a prompt message.

QQUIT

Soft termination of the program occurs at the current statement, with no prompt message.

QUIT ABEND

The program is abended (ABENDU4038) at the current statement, with a prompt message.

Note: When using QUIT ABEND, any non-committed database updates are rolled back. We recommend that this setting be used as the default for PF3.

You can associate QUIT ABEND with the PF3 key by issuing the following command:

```
SET PF3 'ABEND' = QUIT ABEND;
```

Note: This PF key setting is saved to the INSPSAFE file, if one was allocated. CICS does not support INSPSAFE. Therefore, this command should be placed in the INSPPREF file when running under CICS. This approach can also be used for other environments.

QUIT DEBUG

The debugging session is terminated, but the program continues to run to completion.

3.1.10 Built-in functions

There are several built-in functions defined in Debug Tool. Two of them are presented here.

- %HEX

  Returns the hexadecimal value of the operand

- %GENERATION (PL/I)

  Returns a specific generation of a controlled variable in the program

3.1.11 Dynamic Debug Facility

The Dynamic Debug facility enables you to debug COBOL programs compiled with the NONE suboption of the TEST compiler option, assembler, and disassembled programs.

You must activate the Dynamic Debug facility (by using the command SET DYNDEBUG ON) to debug programs that run without the Language Environment runtime.

The Dynamic Debug facility can be used to improve the performance of programs with compiled-in hooks (compiled with certain compilers) while debugging them.

Programs written in C/C++ and PL/I must be compiled with the TEST option.

If the Dynamic Debug facility has been installed, the initial setting is ON. If it was not installed, the initial setting is OFF and the facility cannot be activated by the user.

3.2 Debug Tool Utilities and Advanced Functions

Debug Tool Utilities and Advanced Functions provides enhancements to Debug Tool, and the combined strength of these products can help you to examine and debug programs.

3.2.1 Debug Tool Utilities

Debug Tool Setup Utility (DTSU) is provided with Debug Tool to help with creating and managing setup files. Debug Tool Utilities and Advanced Functions provides additional tools to help you with the following tasks:

- Preparing high-level language programs for debugging by helping convert, compile, and link
- Preparing assembler programs for debugging by helping assemble, create debug information, and link
- Conducting analysis on test cases to determine how thoroughly test cases test programs (also called *code coverage*)
- Starting and running a program in foreground or batch by storing and using setup information, which can be the runtime parameters, libraries, and names of input and output data sets
- For IMS Version 8, browsing and editing the Language Environment runtime parameters table
- Creating a batch job for a private IMS message region with customized load libraries and region attributes
- Determining the language translator (compiler or assembler) used to generate the object module for each CSECT in any load module by using the Load Module Analyzer
- Converting old COBOL source code and copybooks to new versions of COBOL

The combination of DTSU and these tools is called Debug Tool Utilities. The first screen of the main panel of Debug Tool Utilities is presented in Figure 3-10.

```
------------------------------ Debug Tool Utilities ------------------------------
Option ===> G_

                                                                        More:     +
0  Manage Job Card
   For Program Preparation and Setup File Management

1  Program Preparation
   Compile old or new COBOL programs with newer compilers, convert old COBOL
   source into new COBOL source, use other compilers, and link edit.

2  Manage and Use Debug Tool Setup Files
   You can manage setup files and use them to run your program interactively
   with Debug Tool in TSO Foreground or submit your program to run in
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 3-10 Debug Tool Utilities - main panel

Tip: Using the 32x80 or even 43x80 screen size for a 3270 display is helpful when working with Debug Tool Utilities.

Enter the option G on this panel (optionally, scroll down by pressing PF8 to see this option). Debug Tool Utilities provides an extensive online introduction (about 20 screens).

Debug Tool Utilities has a convenient cursor-driven help function. To use it, move the cursor to any input field and press PF1.

Creating and managing setup files

To create and manage setup files, select the option 2 on the main panel to start Debug Tool Setup Utility (DTSU), which is part of Debug Tool. You do not require Debug Tool Utilities and Advanced Functions to use this tool. The DTSU starting panel is shown in Figure 3-11.

```
-------------------- Debug Tool Foreground - Edit Setup File ------------------
Command ===> _____________________________________________________________________


Setup File Library:
   Project . . . ABAPHEN
   Group . . . . DEBUG    . . . ________ . . . ________ . . . ________
   Type  . . . . SETUPFIL
   Member  . . . DEBUGTR        (Blank or pattern for member selection list)
                                (or existing or new member name)

Other Data Set Name:
   Data Set Name . . . _____________________________________________________
   Volume Serial . . . ______    (If not cataloged)


_ Initialize New setup file for DB2 (/)




 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 3-11 DTSU - initial panel

Setup files can save time when debugging a program that has to be restarted multiple times. Setup files store information required to run the program and start Debug Tool. You can create several setup files for each program. Each setup file can store information about starting and running the program in different circumstances.

Fill in the setup file library fields on the panel and press Enter. The next panel is shown in Figure 3-12.

```
EDIT - Edit Setup File 'ABAPHEN.DEBUG.SETUPFIL(DEBUGTR)'      Row 1 to 5 of 11
Command ===> _                                                Scroll ===> CSR

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs / OS/VS COBOL - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,'*',PROMPT,'MFI:INSPPREF') '

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ********** Top of Data **********                 _____
___ COMPFILE 1   _ 'CHABERT.TRADER.COMPFILE'                         SHR
___ CUSTFILE 1   _ 'CHABERT.TRADER.CUSTFILE'                         SHR
___ INSPPREF 1   _ 'CHABERT.DEBUG.PREF(TRAD1)'                       SHR
___ REPOUT   1   _ SYSOUT=*                                          _____
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 3-12 Editing a setup file

Select the option to enter or modify parameters and press Enter. The panel shown in Figure 3-13 is displayed. Fill in the values.

```
--------------- Debug Tool Foreground - Modify Parameter String ---------------
Command ===> _
                                                                        More:     +
Modify Test options, other run-time options, and Program arguments

Select Test Options:

Test Option     ==> TEST            Test/Notest
Test Level      ==> ALL             All/Error/None
Commands File                       *, DDname, or Data Set Name
                ==> *
Prompt Level    ==> PROMPT          Prompt/NoPrompt
Preference File                     *, DDname, or Data Set Name
                ==> INSPPREF

Select (/) a session type and provide parameters:

 / Full-screen mode
   Terminal LU     ==> _____________ blank or MFI VTAM Terminal LU

 _ Remote debug mode
   Connection type ==> _________     SINGLE/MULTIPLE socket
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 3-13 Runtime options and session type selection, part 1

Press PF8 to scroll to see the lower part of it and, if required, add more run-time options and program arguments, as shown in Figure 3-14.

```
--------------- Debug Tool Foreground - Modify Parameter String -------------
Command ===> ____________________________________________________________
                                                                More:   -

 / Full-screen mode
   Terminal LU     ==> _____________ blank or MFI VTAM Terminal LU

 _ Remote debug mode
   Connection type ==> ________      SINGLE/MULTIPLE socket
   Address         ==> ____________________________________________________
   Port            ==> 8000________

Other run-time options: ____________________________________________________
                        ____________________________________________________
                        ____________________________________________________

Program arguments: _________________________________________________________
                   _________________________________________________________
                   _________________________________________________________




 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 3-14 Runtime options and session type selection, part 1

Press PF3. The newly constructed PARM string is shown in Figure 3-15.

```
EDIT - Edit Setup File 'ABAPHEN.DEBUG.SETUPFIL(DEBUGTR)'      Row 1 to 5 of 11
Command ===> _____________________________________________ Scroll ===> CSR

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name IRADERB__
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs / OS/VS COBOL - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,'*',PROMPT,'MFI%IRMLU001:INSPPREF')

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ********** Top of Data **********                 _____
___ COMPFILE 1   _ 'CHABERT.TRADER.COMPFILE'                         SHR
___ CUSTFILE 1   _ 'CHABERT.TRADER.CUSTFILE'                         SHR
___ INSPPREF 1   _ 'CHABERT.DEBUG.PREF(TRAD1)'                       SHR
___ REPOUT   1   _ SYSOUT=*                                          _____
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 3-15 Runtime parameter string created

Load Module Analyzer

This program can process all or selected program modules in a concatenation of PDS or PDSE data sets. To access the Load Module Analyzer (LMA), scroll (PF8) to the second screen of the main panel of Debug Tool Utilities, as presented in Figure 3-16.

```
------------------------------ Debug Tool Utilities ------------------------------
Option ===> _

                                                                          More:    -
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs
   You can dynamically update LE Runtime options and create private
   message region.

5  Load Module Analyzer
   Analyze load modules to determine the language translator used to
   compile or assemble each CSECT in the load module.

G  Getting Started

Enter X to Terminate

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 3-16 Debug Tool Utilities - second screen of main panel

Selecting option 5 takes you to Figure 3-17.

```
---------------------- Debug Tool - Load Module Analyzer ----------------------
Command ===> _

                                                                  More:      +
Load Module Library:
   Project . . . CHABERT
   Group . . . . BOOK2005 . . . ________ . . . ________ . . . ________
   Type  . . . . LOAD
   Member  . . . TRADERD        (Asterisk for all members)


Other Data Set Name:
   Data Set Name . . . ____________________________________________________
   Volume Serial . . . ______    (If not cataloged)


Listing Data Set . . . DTU.LIST_____________________________________________


_ OS/VS COBOL only (/)
_ Display prefix and program data (/)
/ Show information for all compiler / system library routines (/)
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 3-17 Debug Tool - Load Module Analyzer

Here we have selected a single load module TRADERD for analysis. This job runs in the TSO foreground. If you select an entire PDS or PDSE to be analyzed, JCL is generated to start LMA in MVS batch. Pressing Enter in the previous screen causes the analysis to run. Figure 3-18 shows a fragment of the generated output.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Iest  Help
VIEW       ABAPHEN.DTU.LIST                                  Columns 00001 00072
Command ===> ____________________________________________________ Scroll ===> CSR
000013            Load Module  CHABERT.BOOK2005.LOAD(TRADERD)
000014
000015  CSECT      Offset Len/Ent  Program-ID  Trn-Date    Program-Description
000016  DFHECI          0      1E   569623400  2003/11/14  High Level Assembler
000017  TRADERD        20    3F20   5655G5300  2005/10/21  Enterprise COBOL for
000018  DSNCLI       3F40      6C   569623400  2003/11/14  High Level Assembler
000019  CEESG005     3FB0      18   569623400  2005/03/22  High Level Assembler
000020                              PL/X390    2005/03/22  PL/X390
000021  CEEGMTO      3FC8      14   569623400  2005/03/21  High Level Assembler
000022  CEELOCT      3FE0      14   569623400  2005/03/21  High Level Assembler
000023  DSNAA        3FF8     108   569623400  2002/12/06  High Level Assembler
000024                              PL/X390    2002/12/06  PL/X390
000025  DSNHADD2     4100      58   569623400  2002/10/15  High Level Assembler
000026  DSNHADDR     4158      58   569623400  2002/10/15  High Level Assembler
000027  DSNHMVHW     41B0      1C   569623400  2002/10/15  High Level Assembler
000028  CEEBETBL     41D0      28   569623400  2005/03/21  High Level Assembler
000029  CEESTART     41F8      B0   569623400  2005/03/21  High Level Assembler
000030  IGZCBSO      42A8     570   569623400  2005/03/22  High Level Assembler
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 3-18 Sample Load Module Analyzer output

Note that you can scroll right with PF11 to get the full description of the compiler or assembler used to compile each CSECT.

Compiling, converting, linking

Debug Tool Utilities can help you to prepare programs for debugging. The option 1 on the Debug Tool Utilities main panel allows you to proceed with the actions presented in Figure 3-19 and Figure 3-20.

```
----------------------- Debug Tool Program Preparation -----------------------
Option ===> _____________________________________________________________________

                                                                        More:     +
1  COBOL Compile
   Using 5655-G53 IBM Enterprise COBOL for z/OS

2  COBOL Convert and Compile
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid
   and   5655-G53 IBM Enterprise COBOL for z/OS

3  PL/I  Compile
   Using 5655-H31  IBM(R) Enterprise PL/I for z/OS

4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 3-19 Debug Tool Utilities program preparation panel, part 1

You can scroll down using PF8 to see more options (Figure 3-20).

```
----------------------- Debug Tool Program Preparation -----------------------
Option ===> _____________________________________________________________________

                                                                        More:   -
4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder

F  Fault Analyzer Side file Create
   Using 5655-G74 IBM Fault Analyzer for z/OS and OS/390

C  Convert old 68/74 Std COBOL to 85 Std COBOL
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid

S  Manage System and User Settings


 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 3-20 Debug Tool Utilities program preparation panel, part 1

By selecting the options, you can specify system and user settings for program preparation, for example, compiler, DB2 precompiler, CICS translator, link-editor and Fault Analyzer product data sets, input and output data sets, and so forth, as shown in Figure 3-21.

```
-------------- Debug Tool Program Preparation - Manage Settings --------------
Command ===> ________________________________________________________________

                                                                    More:     +
System Settings for Program Preparation (compiler product datasets, etc):
These settings are refreshed at the start of every session.
  1  COBOL Compile
  2  COBOL Conversion (CCCA)
  3  PL/I  Compile
  4  C and C++ Compile
  5  HLASM
  6  Link Edit
  7  Fault Analyzer IDILANGX

User Settings for Program Preparation (input and output datasets, etc):
These settings are saved between sessions until RESET.
  8  COBOL Compile
  9  COBOL Convert and Compile
 10  PL/I  Compile
 11  C and C++ Compile
 12  HLASM
 13  Link Edit
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 3-21 Options available for managing program preparation

As an example, some system settings for the PL/I compiler are shown in Figure 3-22.

```
---------------------Manage User Settings - PL/I Compiler---------------------
Command ===> _____________________________________________________________________

                                                                        More:     +
PL/I Compiler Settings:
 Temporary data set allocation parameters:
   CYLINDER SPACE(1 1)_____________________________________________________________

 Listing data set:
   Pattern . . . . '/U./B.PLILIST(/M)'_____________________________________________
   Data set type . PDSE_____ (PDSE PDS SEQ)
   Allocation parameters:
   CYLINDER SPACE(1 1) LRECL(137) RECFM(V B A)_____________________________________

 Object data set:
   Pattern . . . . '/U./B.OBJECT(/M)'______________________________________________
   Data set type . PDSE_____ (PDSE PDS SEQ)
   Allocation parameters:
   CYLINDER SPACE(1 1)_____________________________________________________________

 SYSDEBUG data set:
   Pattern . . . . '/U./B.SYSDEBUG(/M)'____________________________________________
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 3-22 System settings for the PL/I compiler

User settings for the same compiler include several data sets, for example, listing, object, DBRM, allocation parameters, and naming patterns.

Code coverage

Determining code coverage helps to improve test cases so you can test programs more thoroughly. Debug Tool Utilities provides the Coverage Utility, a tool to report which code statements have been run by the used test cases. You can access DTCU by selecting option 3 on the Debug Tool Utilities main panel, providing you with the options shown in Figure 3-23.

```
------------------------ Debug Tool Coverage Utility -------------------------
Option ===> _____________________________________________________________________

0  Defaults      Manipulate defaults
1  CntlFile      Work with the Control File
2  Setup         Create JCL for Setup
3  StartMon      Create JCL to Start the Monitor
4  Reports       Create JCL for Reports
5  Monitor       Control the Monitor
6  FastPath      FastPath

Enter X to Terminate









 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F10=Actions  F12=Cancel
```

Figure 3-23 Coverage Utility main panel

Using the code coverage report, you can enhance test cases to ensure that all valid code is exercised and to identify dead code in your program.

Preparing IMS run-time environment

You can create private IMS message regions that can be used to debug test applications and, therefore, not interfere with other regions. For IMS Version 8, the Language Environment run-time parameters table can be modified without relinking the applications.

3.2.2 Advanced functions

Debug Tool provides an extensive set of commands to debug programs. Debug Tool Utilities and Advanced Functions enhances this set of commands by adding more than 15 new commands, including:

ALLOCATE Allocates a file to an existing or temporary data set or a concatenation of existing data sets.

CALL %FA Starts and instructs Fault Analyzer to provide a formatted dump of the current state.

DESCRIBE Displays attributes of file allocations, references, compile units, and execution environment.

FREE Releases previously allocated file.

LOAD Enables you to load a module for Debug Tool to use in debugging an application. Use CLEAR LOAD to delete the previously LOADed module.

LOADDEBUGDATA Specifies that a compile unit is an assembler compile unit and loads the corresponding debug file.

PLAYBACK * Directs Debug Tool to start or stop recording steps and data history, perform recorded STEP and RUNTO commands forward or backward.

SET ASSEMBLER Turns ON or OFF additional information useful when debugging an assembler compile unit.

SET AUTOMONITOR Controls the automonitoring.

4

Introduction to Fault Analyzer for z/OS

The purpose of Fault Analyzer (FA) is to determine the cause of any abends in an application program. You do not have to read through application or system dumps, because the product has the ability to isolate the exact instruction that caused a particular error.

In this chapter we identify the software levels that are required to use Fault Analyzer, and we provide an overview of the mechanics of Fault Analyzer.

4.1 Validating your software level

To effectively use Fault Analyzer, you must have the appropriate levels of software installed on your system. You should review the Program Temporary Fix (PTF) we have listed to ensure that it is appropriate for your operating environment.

4.1.1 PTF information

Systems programmers responsible for installing and maintaining Fault Analyzer should review the RETAIN® PSP bucket **UPGRADE FAULTANALYZE**.

The research for this book was done with the following maintenance level of Fault Analyzer: Fault Analyzer for z/OS V7R1M0, SYSMOD HAC4710, as shown in Figure 4-1.

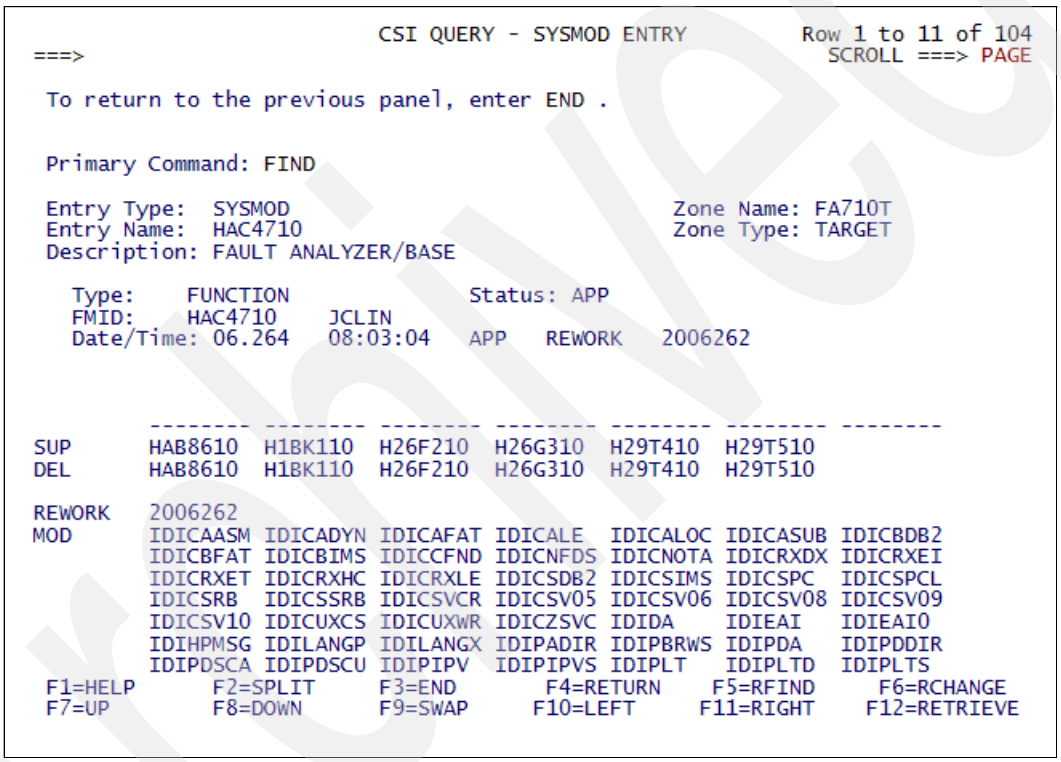


Figure 4-1 CSI query of FA sysmod entry point

You can see the software level by looking at the help About Fault Analyzer in the Fault Analyzer main panel. It is shown in Figure 4-2.

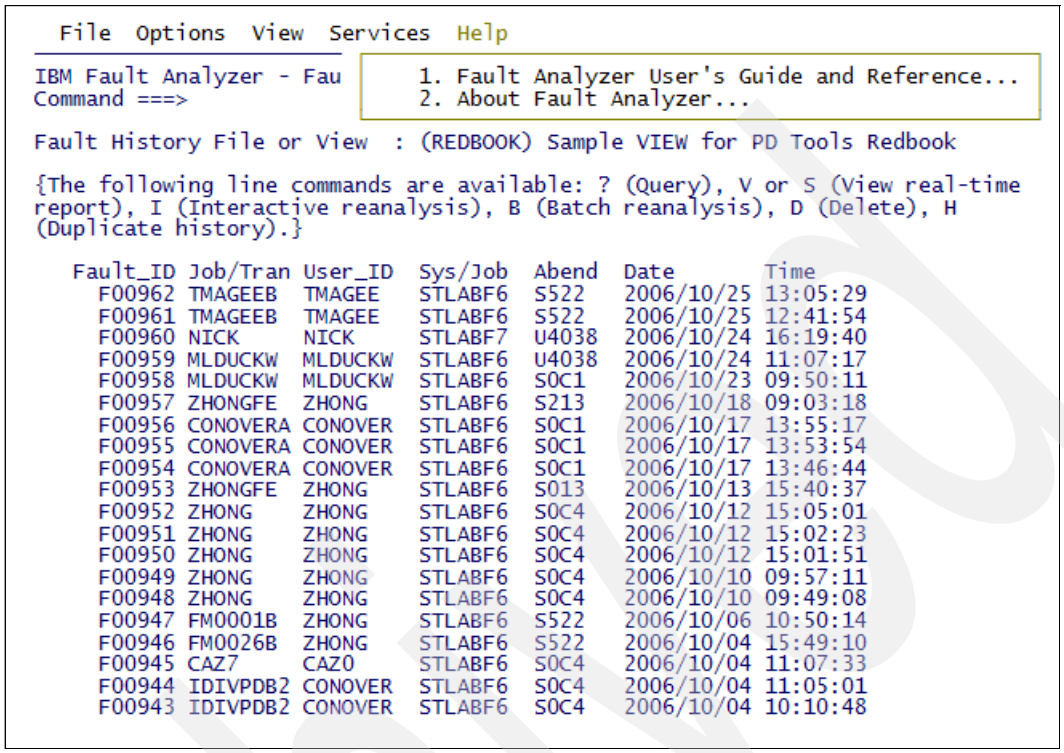


Figure 4-2 Help option from Fault Analyzer Main menu

Selecting option 2 (About Fault Analyzer) from the pull-down menu displays FA general information, as shown in Figure 4-3.

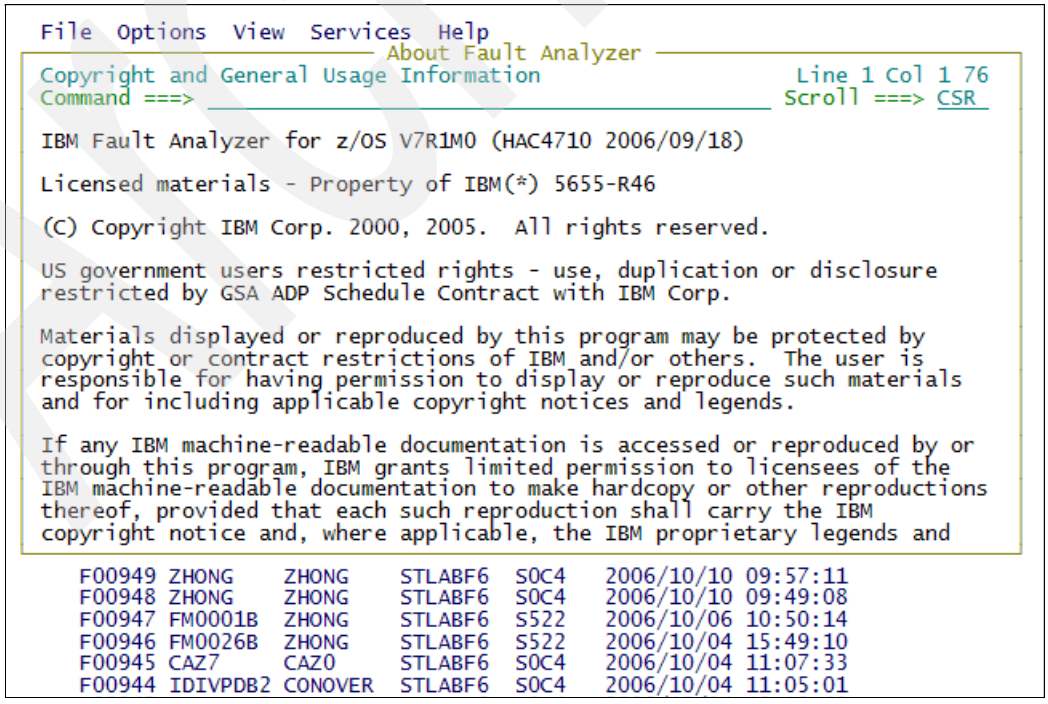


Figure 4-3 Fault Analyzer copyright and general usage information

The Fault Analyzer Main menu is shown in Figure 4-4. The default history file, as per the IDICNFG00 member of system PARMLIB, points to a file named IDI.HIST.

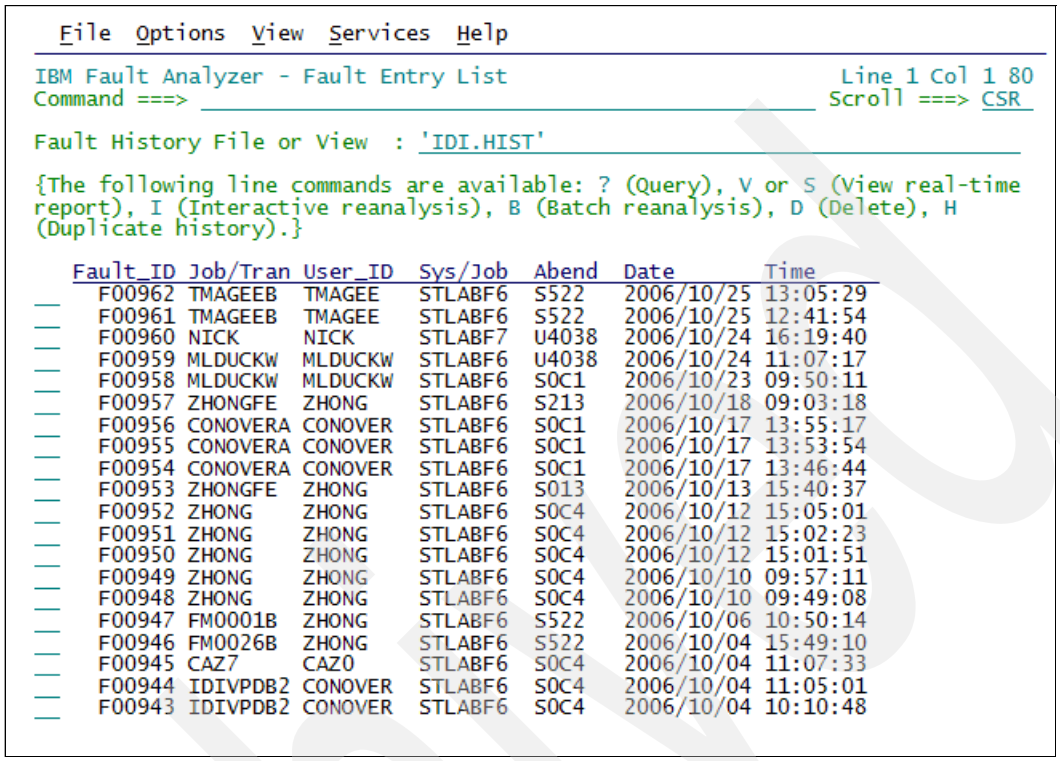


Figure 4-4 Fault Analyzer main menu

4.2 Mechanics of Fault Analyzer

After Fault Analyzer has been installed and customized according to your system environment requirements, it is invoked automatically whenever an application program abends. Fault Analyzer offers maximum flexibility through support for IBM CICS, IBM DB2, IBM IMS, Enterprise PL/I and COBOL,C/C++, ASSEMBLER and IBM WebSphere Application Server for z/OS systems.

Figure 4-5 illustrates how FA works.

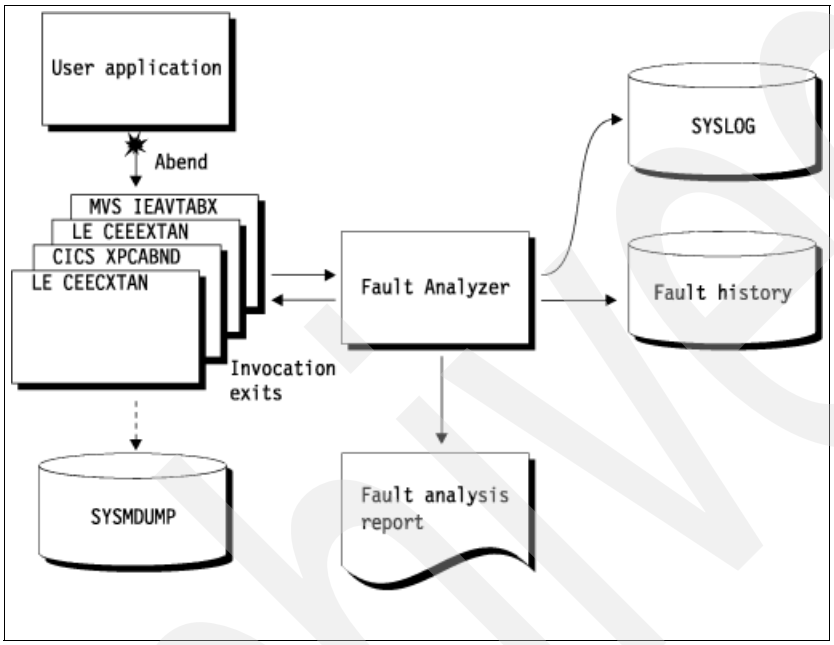


Figure 4-5 Fault Analyzer functional overview

When a program abends, the abend is intercepted by the system and Fault Analyzer is automatically invoked. Fault Analyzer performs fault analysis processing, and then records details about the abend in a fault history file. Fault Analyzer writes the fault analysis report to the job, and a summary to the SYSLOG.

One of FA's powerful features is the ability to use the application program compiler's listing to identify the source statement of the line that caused the abend. Another feature that benefits you as an application programmer is its ability to make use of the IBM library of error messages and abend codes.

4.2.1 Fault history file

Fault Analyzer records a summary of an abend in a fault history data set. The FA panel shown in Figure 4-4 provides access to the fault history file and displays the following information:

- Fault ID
- Job name or transaction ID that experienced the abend
- User ID that submitted the job
- System on which the abend occurred
- Type of abend
- Date and time of abend

Note: You can change the column configuration of this panel by selecting **8** from the **VIEW** menu. Figure 4-6 shows the column configuration from the View pull-down menu item.

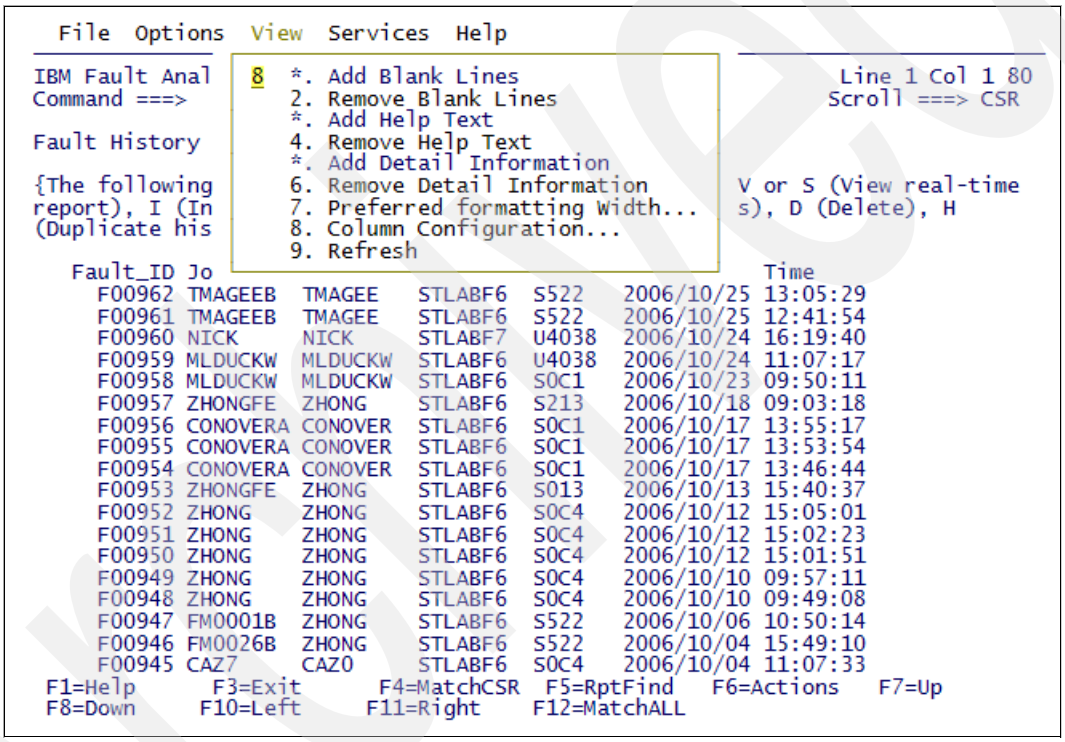


Figure 4-6 Fault Analyzer column configuration list selection

This panel also shows you the line commands that are available to process each entry in the list: ? (query), V (view real-time report), I (interactive re-analysis), B (batch re-analysis), D (delete) and H (Duplicate History Display), as shown in Figure 4-6.

Note: You can select another history list by selecting an entry from the File pull-down menu selection **1** (Last Accessed Fault History Files or Views) or by typing over the history file name if you know the name of the file you wish to view.

4.2.2 Supported application environments

Fault Analyzer supports applications running under z/OS and OS/390 in the following applications environments:

- COBOL

- PL/I

> **Important:** The minimum level of Enterprise PL/I required for complete Fault Analyzer source level support is Version 3 Release 2 with PTFs UQ71704 and UQ71690 installed.

- Assembler
- C/C++
- Language Environment
- UNIX System Services
- CICS
- IMS
- DB2
- MQSeries
- WebSphere
- Java

In the z/OS environment, Fault Analyzer executes in 31-bit addressing mode and performs analysis on 24-bit or 31-bit addressing mode applications. Multithread and DLL applications are supported. Fault Analyzer does not yet perform analysis on applications using 64-bit addressing mode. C++ support is basic and does not provide any class information.

Binder-related dependencies

In order to map CSECTs in load modules, Fault Analyzer utilizes the Application Programming Interface of the IBM Binder program. This generally occurs during real-time analysis, but might also be done during interactive analysis of CICS system dumps. Since the Binder supports load modules residing in PDS(E) data sets only, Fault Analyzer is not able to identify CSECTs in load modules that have been loaded from any other type of storage.

Setting up existing programs for fault analysis

You do not have to make any changes at all to existing programs to allow Fault Analyzer to produce an analysis of any fault. Nor do you have to recompile programs. However, if you store compiler listings or side files in the appropriate repository, then Fault Analyzer is able to identify the source statement of the abending program. (If you choose to not store listings or side files, you can still provide one after an abend has occurred. This makes it possible for Fault Analyzer to extract more information when you perform reanalysis.)

To provide a side file, you might have to recompile your programs, since appropriate side files are only produced when certain compiler options are requested. If you already have compiler listings that were produced with the correct compiler options, you can create side files without having to compile again. The advantage of the side file is that it is more compact than a listing.

Additional region size required

Fault Analyzer runs in the same region as your abending program at the time of the abend. Therefore, there must be spare GETMAIN storage that is not used by the application in order for Fault Analyzer to run and analyze the program storage in its abend state. Initially, up to 16 megabytes of storage might be required, depending on the execution environment. This additional region size increases as the size and complexity of the abending program increases. In situations where Fault Analyzer is unable to obtain sufficient storage for the real-time analysis of a fault, a SYSMDUMP can be taken for subsequent batch or interactive reanalysis.

Compiler listing or side file selection criteria

Fault Analyzer basically performs two types of check when selecting a compiler listing or side file to be used for source-level analysis:

- A size check is performed, which varies from language to language, where an attempt is made to match the size and contents of the load module with the compiler listing or side file. For example, when the COBOL compiler LIST option is used, the size checks include matching the offset and contents of the last 12 assembler instructions in the CSECT. Also for the current COBOL compilers, the working storage size and TGT size are also checked.
- A date and time check is performed between the load module and the compiler listing or side file. Provision is made for compiler listings being created after the date and time associated with the load module. To obtain detailed information about why a particular compiler listing or side file was selected or rejected, the IDITRACE facility can be used.

4.2.3 Summary of real-time analysis

Real-time analysis occurs when an application abends and Fault Analyzer is invoked through one of the supplied invocation exits, or a call to the program SNAP interface is made, and analysis has not been excluded via one of the EXCLUDE options.

Generally, real-time analysis produces two results:

- A report, which is written to JES. This is the default for all environments apart from CICS, where by default, no report is written to JES.
- A fault entry in a history file, which provides the ability to perform reanalysis of the fault.

  A copy of the report written to JES is also included in the fault entry, and can be viewed from the ISPF interface. You cannot change the report by setting options to different values at the time you view it. If you want to look at more (or less) detail, you must reanalyze the fault with adjusted options or a supplied listing or side file.

This is the first step in the fault analysis process. In most cases, the analysis is deemed satisfactory, and you do not have to reanalyze the fault.

All virtual storage pages that were referenced during the analysis in the abending task's address space is written to the history file as a minidump, unless the MaxMinidumpPages option in effect specifies a lower limit.

Restriction: LOADER

Fault Analyzer does not work correctly if using the LOADER (IEWBLDGO), since the load-and-go technique of link-editing modules does not write them to a data set. The data set copy of the load module is required in order to determine CSECT names, lengths, and starting offsets.

Dump suppression

The types of dumps that can be suppressed are:

- SYSABEND, SYSUDUMP, or SYSMDUMP (if Fault Analyzer was invoked using the IEAVTABX MVS change options/suppress dump exit, IDIXDCAP)
- CICS transaction dumps

Note: Suppression of dumps is the default, except for CICS transaction dumps when using EXEC CICS DUMP TRANSACTION DUMPCODE(xxxx).

To override the default dump suppression, use the RetainDump(ALL) option. Dump suppression should not be confused with suppression of analysis or suppression of fault entry creation.

Fault history file selection

When a fault is being analyzed in real-time by Fault Analyzer, a fault history file must be available in which details of the analysis can be recorded.

There are a number of ways in which the name of the fault history file can be provided to Fault Analyzer. The following is a list of these in the order of their override significance (each entry in the list overrides all previous entries):

1. The product default name, IDI.HIST.
2. The IDIHIST suboption of a DataSets option specified in the PARMLIB config member, IDICNF00. This includes either the logical PARMLIB concatenation or the installation-wide alternate PARMLIB data set name provided via the IDISCNF USERMOD supplied with Fault Analyzer.
3. The IDIHIST suboption of a DataSets option specified in a config member identified via the IDICNFUM user options module.

   Note: If a user options module is used, it replaces the default IDICNF00 PARMLIB config member. Thus, even if the user options module designated config member did not include an IDIHIST suboption of a DataSets option, any specification of IDIHIST in the default IDICNF00 PARMLIB config member would not be recognized.

4. The IDIHIST suboption of a DataSets option provided via the IDIOPTS DDname in the abending job step.
5. An explicitly coded IDIHIST DD statement in the abending job step.
6. The data set name provided by an Analysis Control or End Processing user exit in the ENV data area IDIHIST field.

Controlling the real-time analysis with options

Set options globally so that they control the output for all jobs. However, you can also set an option just for one job. In this case, you should set the option in the user options file.

Options that you are more likely to use for real-time analysis are:

- RetainDump(ALL)

  Specify this option if you want to want to retain the SYSABEND, SYSUDUMP, or SYSMDUMP unconditionally. Fault Analyzer permits the writing of the dump and records the name of the dump data set in the fault history file if a SYSMDUMP DD statement was specified. Without this option, many dumps are suppressed when Fault Analyzer deems that the analysis it has performed is adequate. This option does not affect the writing of the minidump to the history file. The dump disposition part of this option is applicable to the use of the MVS IEAVTABX change options/suppress dump exit only.

- Detail

  Specify this option if you want to adjust the level of detail given in the real-time analysis report.

  Note: If a dump is produced, you can change this option when you perform a reanalysis.

- Exclude

  Specify this option if you want to exclude this job from analysis.
- NoDup

  Specify this option if you want to change the way that duplicate faults are handled by default.
- CICSDumpTableExclude

  Specify this option if you want to exclude CICS transaction fault analysis using the CICS transaction dump code table.
- You can also use the DataSets option to point to listings and side files.

Pointing to listings with JCL DD statements

No DD statements are required to run Fault Analyzer in either batch or real time, although a SYSMDUMP DD statement is required for normal SYSMDUMP processing in real time when using the MVS IEAVTABX change options/suppress dump exit, unless the IDITABD USERMOD is applied.

You can specify the DD statements listed in Table 4-1 in the JCL if appropriate. If they are not specified, the definitions from the PARMLIB configuration member IDICNF00, the IDIOPTS user options file, or an Analysis Control user exit are used to identify these data sets.

Table 4-1 DDNAMES specifications

DDNAME	DSN requirement	Description
IDILC	PDS(E)	Data set containing C compiler listings
IDILCOB	PDS(E)	Data set containing COBOL compiler listings (other than OS?VS COBOL)
IDILCOBO	PDS(E)	Data set containing OS/VS COBOL compiler listings
IDISYSDB	PDS(E)	Data set containing COBOL SYSDEBUG side files
IDILPLI	PDS(E)	Data set containing PL/I compiler listings (other than Enterprise PL/I)
IDILPLIE	PDS(E)	Data set containing Enterprise PL/I compiler listings
IDIADATA	PDS(E)	Data set containing SYSADATA from Assembler compilations
IDILANGX	PDS(E)	Data set containing IDILANGX from the Assembler compilations

Note: Do not specify a member name on any of the foregoing DD statements.

The real-time analysis report

The real-time analysis report is produced whenever Fault Analyzer analyzes an abend or is invoked by IDISNAP. It is written to the IDIREPRT DDname, which is dynamically allocated to SYSOUT=class, if no prior allocation exists, and thus is included as part of the normal job output on the JES spool.

The SYSOUT class used (class) is the default job output class (SYSOUT=*), or if a SYSUDUMP DD statement in the abending job step specifies a JES SYSOUT class, then the same output class is used for the Fault Analyzer real-time report.

If you wish to divert the real-time analysis report to another file, then adjust the DD card as required, as shown in Example 4-1.

Example 4-1 DD card requirement

```
//IDIREPRT DD DISP=(,CATLG),DSN=MY.REPORT.DS,
// DCB=(RECFM=VB,LRECL=137),SPACE=(CYL,(1,1))
```

Alternatively, a user exit can be used to allocate IDIREPRT to a different output class. The IDIREPRT DDname is opened with LRECL=137. Any existing data set attributes must be compatible with this logical record length.

The IDIREPRT allocation for CICS transaction abends is the same as for any other type of abend.

Combining Fault Analyzer real-time reports

By default, all real-time reports are written to separate JES spool files. This is generally considered advantageous for subsystems, such as CICS, IMS message-processing regions, or WLM-managed DB2, where multiple reports can be expected written before the subsystem is restarted.

If, for any reason, the reports are preferred written to a single spool file, then this can be accomplished by adding an IDIREPRT DD statement to the job or startup procedure, as shown in Example 4-2.

Example 4-2 DD statement IDIREPRT sample

```
//IDIREPRT DD SYSOUT=*
```

Controlling the SYSOUT class of real-time reports

If no IDIREPRT allocation already exists, then Fault Analyzer dynamically allocates IDIREPRT to SYSOUT=*, or to the same SYSOUT class as the SYSUDUMP DDname. This can be changed to a different SYSOUT class by adding a DD statement to the job or startup procedure, as shown in Example 4-3.

Example 4-3 DD statement IDIREPRT sample

```
//IDIREPRT DD SYSOUT=sysout-class
```

Alternatively, an Analysis Control user exit can be used to allocate IDIREPRT to a required class, as shown in Example 4-4.

Example 4-4 REXX sample

```
/* REXX */

/*Sample Fault Analyzer Control user */
/* exit to allocate IDIREPRT to SYSOUT class F */
/* */
"IDIALLOC DD(IDIREPRT) SYSOUT(F)"
EXIT 0
```

If the foregoing sample exit existed as member ABC in data set X.Y.Z, then providing the following options in either the IDICNF00 configuration member or the IDIOPTS user options file would cause it to be invoked, as shown in Example 4-5.

Example 4-5 Options sample

```
DATASETS(IDIEXEC(X.Y.Z))
EXITS(CONTROL(REXX(ABC)))
```

Suppressing real-time reports

To suppress the writing of any Fault Analyzer reports to the JES spool, you can add the following DD statement to the job or startup procedure:

```
//IDIREPRT DD DUMMY
```

Alternatively, you can use an Analysis Control user exit to allocate IDIREPRT to DUMMY, as shown in the REXX sample in Example 4-6.

Example 4-6 REXX sample

```
/* REXX */
/* */
/* Sample Fault Analyzer Analysis Control user */
/* exit to suppress the analysis report. */
/* */
"IDIALLOC DD(IDIREPRT) DUMMY"
exit 0
```

See the foregoing discussion for an explanation of how this sample exit could be executed.

Note: The real-time report can be written to the fault history file, regardless of the suppression of the JES spool report, and can be viewed from there by using the Fault Analyzer ISPF interface.

Suppressing real-time reports can be useful, for example, in a CICS environment, where the CICS job step might otherwise end up with hundreds of reports in the IDIREPRT output.

The SYSLOG summary

During real-time analysis, a message is written to the operator console, providing a one-line summary of the fault reason.

Example 4-7 is an example of such a message.

Example 4-7 Message sample

```
IDI0002I There was an unsuccessful REWRITE of file MYFILE01 (file status 44) in program
COBFERRD at line #21
```

If IDI0002I has been explicitly coded in the QUIET option, then this message is not written to the SYSLOG.

Using the program SNAP interface (IDISNAP)

A program SNAP interface is provided to assist users in debugging problems with applications that do not abend, or that for any other reason cannot be analyzed by Fault Analyzer using one of the normal abend invocation exits. This permits a call to Fault Analyzer from anywhere within an application program to request an analysis of the current environment. The program SNAP interface module name is IDISNAP.

An example of where a call to IDISNAP might be used is in a DB2 application after execution of an SQL statement that results in a negative SQLCODE.

Apart from the way in which Fault Analyzer is invoked, there is no difference between this type of analysis and any other real-time analysis caused by an abend.

We recommend that you invoke IDISNAP dynamically to ensure that you are always using the most current version.

For programs written in C, IDISNAP can only be invoked dynamically.

Dump registration processing

Unlike the SYSABEND, SYSMDUMP, and SYSUDUMP processes, which run in the user address space, the SVC dump process in MVS runs from the DUMPSRV address space. This difference means that the MVS change options/suppress dump exit, which is one of the normal means of invoking Fault Analyzer, does not work for SVC dumps. For SVC dumps, Fault Analyzer provides the IDIXTSEL exit module. SVC dumps occur for system abends, and are also used by CICS for its system dumps.

If the IEAVTSEL post-dump exit, IDIXTSEL, has been installed, then a skeleton fault entry is created whenever an SVC dump is written. This differs from normal real-time processing in that no analysis is performed, and therefore no report or minidump is produced. This Fault Analyzer process is known as dump registration.

The dump registration processing permits the use of two user exits that effectively are the equivalent of the normal analysis control and notification user exits. These are specified using the DumpRegistrationExits option.

The dump registration fault entry contains only limited information, such as the time of its creation, the system name, and the name of the job that caused the SVC dump to be written. If available, the abend code and abending program name is also provided. However, the first reanalysis of the dump registration fault entry refreshes the fault entry and saves a report and minidump with it.

Excluding an abend from real-time analysis

There are a number of ways to exclude various elements of the Fault Analyzer processing. This is described fully in Real-time exclusion processing in the The *Fault Analyzer User Guide and Reference,* SC19-1088-02 (Real-time exclusion processing).

4.3 Preparing your program for Fault Analyzer

FA always provides the analysis of an abend. However, your application program must be compiled with specific compiler options for FA to display the source statement that caused the error.

Fault Analyzer uses the compiler listing to analyze the cause of abend, list the statement that caused the abend, and list the data values in the working-storage section.

4.3.1 Compiler options

In order for generate listings or side-files suitable for use by Fault Analyzer certain compiler options must be used. For the complete list of required options, see the *Fault Analyzer User Guide and Reference,* SC19-1088-02 (Required compiler options for IDILANGX).

4.3.2 Locating listing, IDILANGX and SYSDEBUG files

Fault analyzer performs a series of searches in order to find a suitable side file. These searches are described fully, in sequence, in the *Fault Analyzer User Guide and Reference,* SC19-1088-02 (Locating compiler listings or side files).

5

Introduction to File Export

In this chapter we describe the main concepts and functions of File Export for z/OS with the four key components of the product. These File Export components are:

- User interface
- Repository
- Parser
- Extract Engine

5.1 File Export overview

File Export is a related data extract tool that lets you create test environments using production data. Developers and testers require test data and a test environment that mimics production data to adequately test new or modified applications. File Export is a tool that can simplify test data generation using a related subset of production data.

5.2 User interface

The File Export ISPF-based UI is used to create the objects stored in the Repository that are used to create the Extract and Load Requests, as well as allowing you to specify the parameters that dictate how the Extracts and Loads perform. In addition, the UI generates the JCL statements that specify for the extract engine what is to be done.

Navigation through the user interface is similar to other PD Tools. There are Primary Commands and Line Commands.

Figure 5-1 displays a panel within File Export that contains both Primary Commands, which are options entered on the Command line, and Line Commands, which are options entered next to objects in a list.

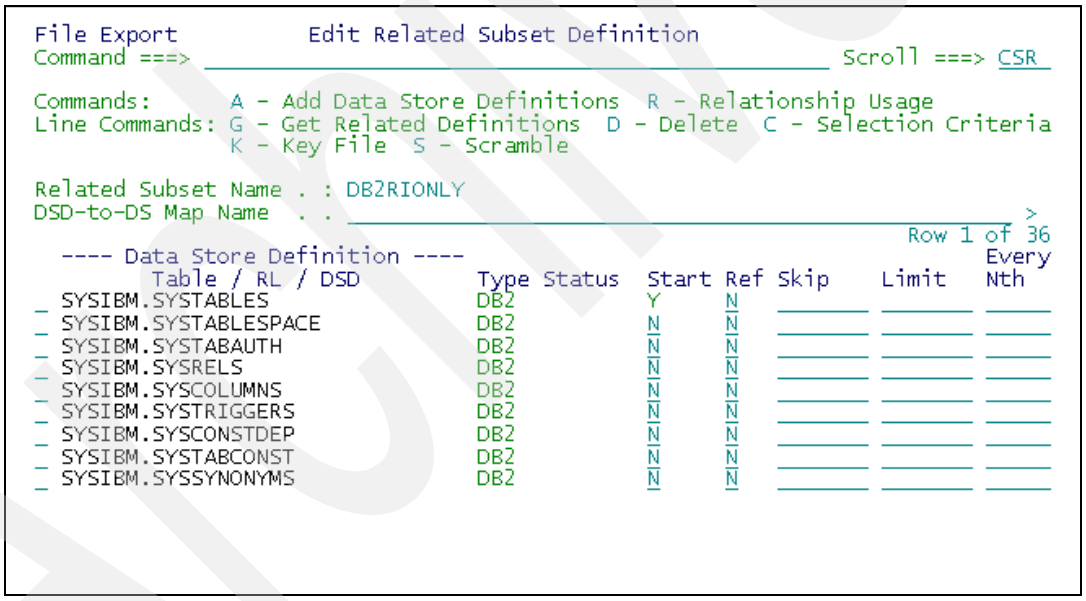


Figure 5-1 An example panel with both Option Commands and Line Commands

The Primary Commands A and R are specified on the Command line to navigate to other panels which are used to manipulate other portions of the Related Subset, while the Line Commands G, D, C, K and S are specified next to a specific object (in this case a DB2 table) within the Related Subset to perform some action on that specific object.

5.3 Repository

The Repository is used to store all the FIle Export-specific objects (such as Application Relationships) which are then utilized in constructing Extract and Load Requests. These objects can also be referred to as definitions, or metadata. A Repository can be managed privately or, shared among users.

A Repository contains several kinds of metadata, such as Record Layouts, Data Store Definitions, Application Relationships, Related Subsets, and Data Store Definitions-to-Data Store Maps.

Here we define the various Repository components:

Record Layout (RL)

Describes the structure of a record, that is, the sequence and format of the fields that comprise the record. This object is typically utilized for non-relational data.

Data Store Definition (DSD)

Used to group all related Record Layouts for a given source and enables them to be treated as a single entity. For simple data records that require only a single Record Layout, a DSD is not required and the Record Layout can be used anywhere a DSD is used.

Application Relationships (AR)

A correlated set of one or more fields from records in two different Data Store Definitions that are used as the basis for finding related data records.

Related Subset (RSS)

Defines the set of DSDs (including RLs and DB2 tables) from which data is extracted, including the rules that control the extract such as, the starting objects, selection criteria, relationship usage, scrambling, and so on.

Data Store Definition-to-Data Store Map (DSD-to-DS Map or DDM)

Used to map “logical” DSDs to their “physical” counterparts (called Data Stores), that is, the underlying files. DSD-to-DS Maps are used for both the Extract process and the Load process.

Repository data tends to be static and does not have to change unless the definitions of the application data change, at which point the Repository must be updated by a user. Record Layouts can be created manually by a user defining each field position, length and data format, or, Record Layouts can be created by parsing a COBOL or PL/I copybook using the File Export parser.

A Repository is necessary to define source and target objects to File Export and must be created before an extract request can be created. Use the following instructions to create a Repository.

Step 1: Select option **1** Metadata Management from the File Export Primary Option menu, as shown in Figure 5-2. Press Enter.

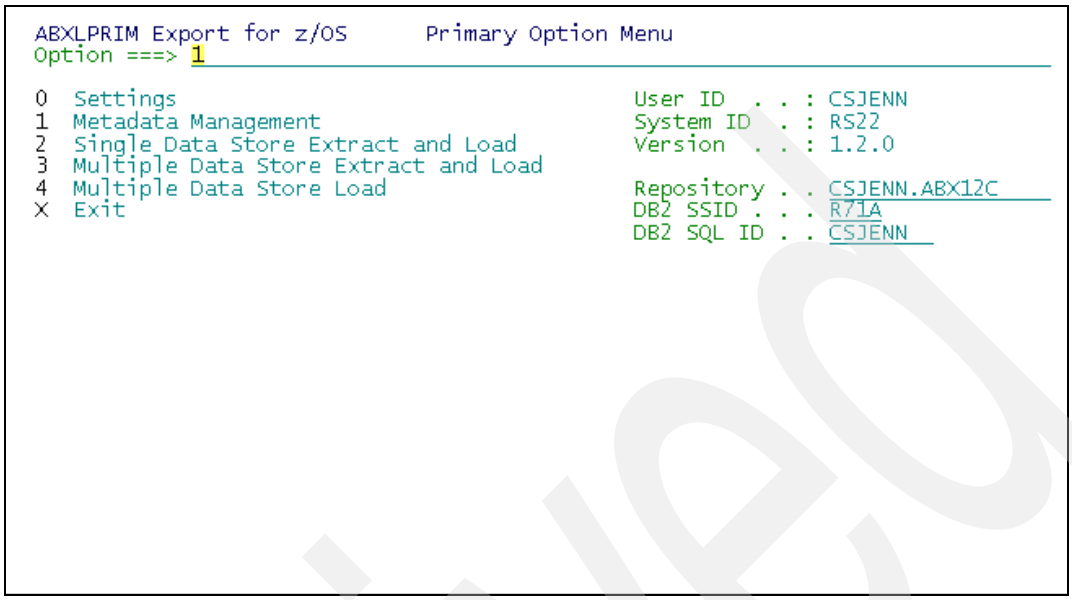


Figure 5-2 Select option 1 Metadata Management

Step 2: S elect option **0** Manage Repository from the Metadata Management panel to get to the Repository management panels. Press Enter, and the panel in Figure 5-3 is displayed.

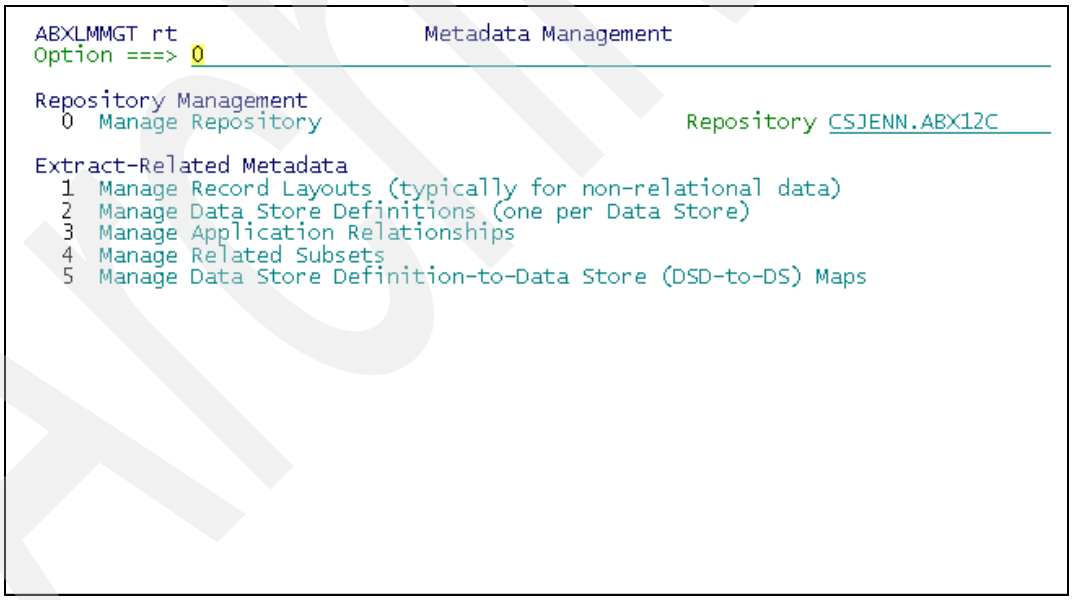


Figure 5-3 Select option 0 to get to the Repository Management panel

Step 3: From this panel, you can select to create a new Repository, delete an existing Repository, or import or export an existing Repository's data. Select option **1** to Create a Repository. Press Enter. See Figure 5-4.

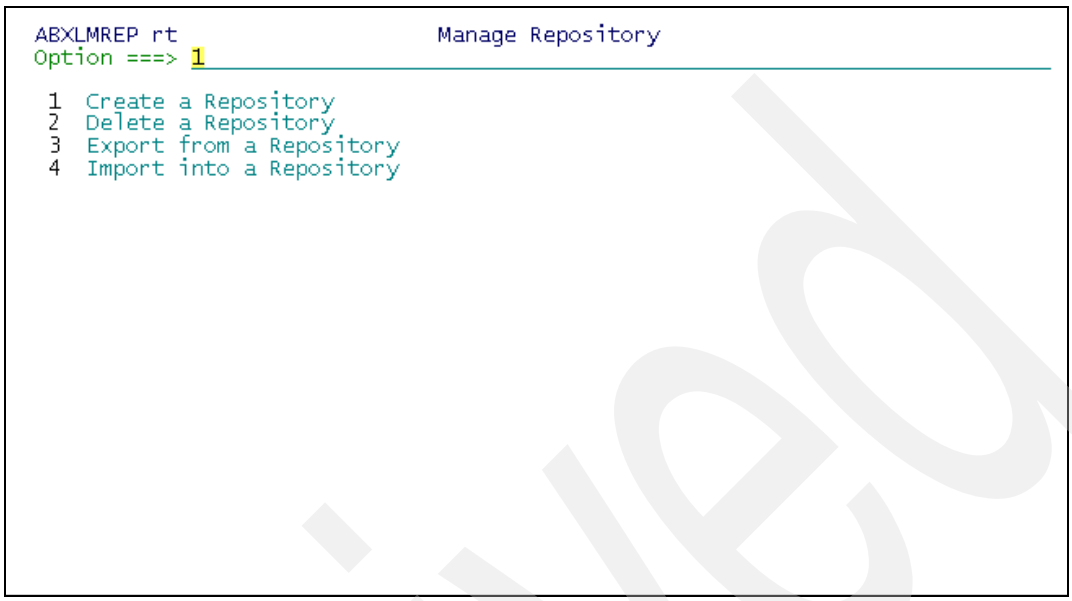


Figure 5-4 Select option 1 to create a Repository

Step 4: On the Create Repository panel, provide a high level qualifier of up to 17 characters to name the new Repository. Provide an optional description that describes what this Repository is intended to be used for.

The fields, *Number of Record Layouts* and *Average Number of Fields per Record*, help to determine the size of the Repository's key lengths that are used for the alternate indexes, not the size of the Repository's VSAM datasets.

For SMS-managed datasets, you must also provide values for Management Class, Storage Class and Data Class. Optionally provide a Volume Serial if one is required.

When all values have been filled in, press PF3 to process the parameters on the panel as shown in Figure 5-5.

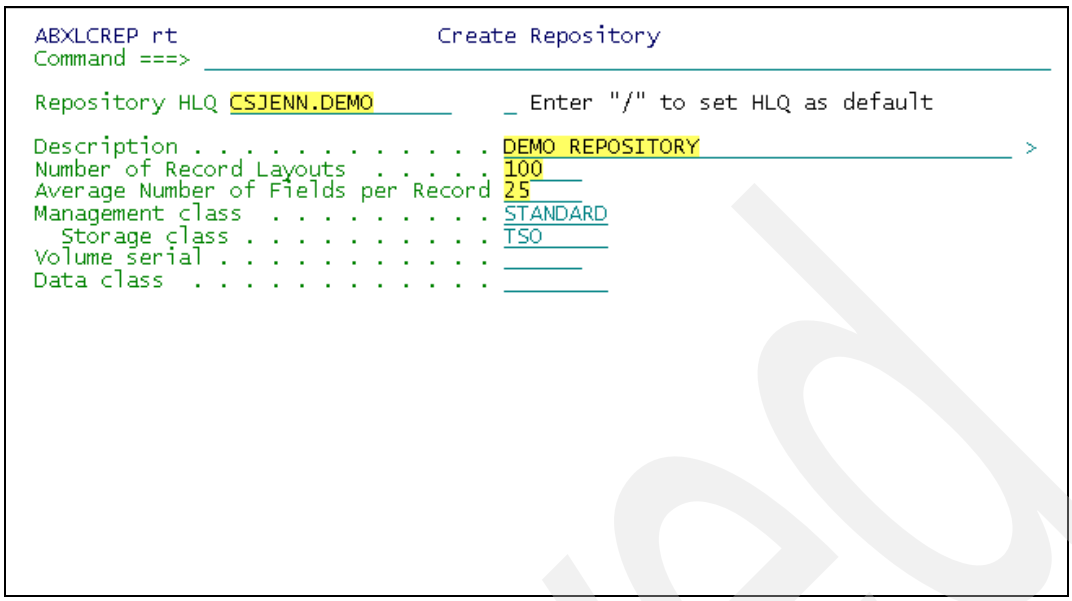


Figure 5-5 Specify the Repository's parameters

A complete Repository consists of approximately 150 data sets and contains metadata about data sources that File Export uses when processing an extract request. When the Repository creation has been completed, a message is displayed indicating the process has completed, as shown in Figure 5-6.

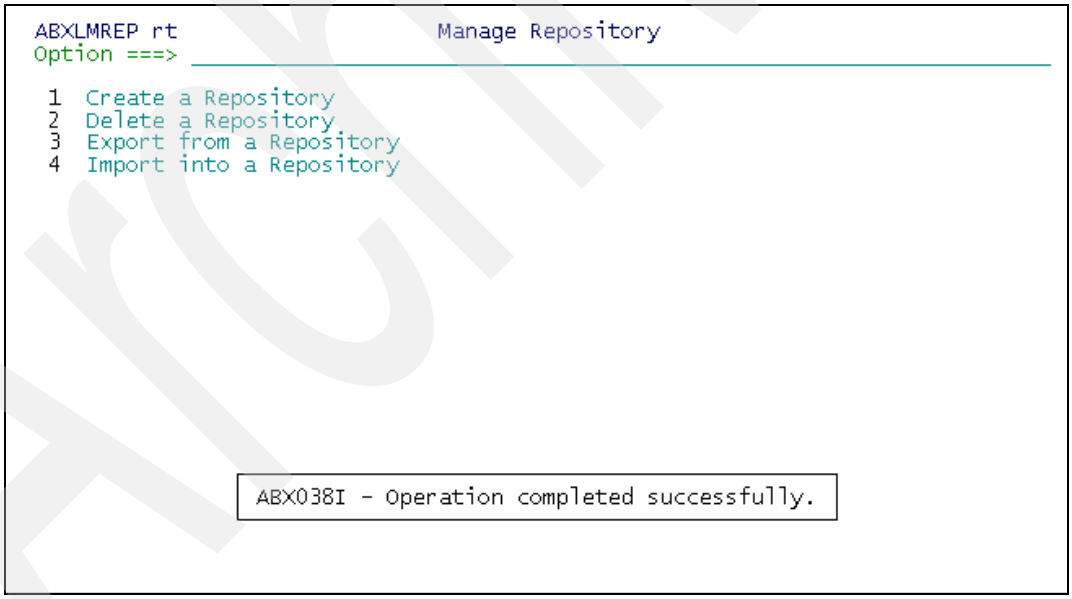


Figure 5-6 The Repository has been created successfully

5.4 Parser

File Export contains a parser that can parse COBOL and PL/I source copybooks into File Export-specific data definitions. The copybooks are parsed into a Record Layout that can then be used in Data Store Definitions and ultimately extract requests.

From the Metadata Management Panel, option **1** Manage Record Layouts allows you to parse a copybook into a File Export-format data definition.

Example 5-1 shows a COBOL copybook with the following fields defined:

Example 5-1 An example of a COBOL copybook

```
****************************** Top of Data ******************************
-  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  7 Line(s) not Displayed
      ******************************************************************
      * COBOL DECLARATION FOR ABX.TRAINING.ABXSEQ03.INPUT               *
      ******************************************************************
       01  ABXSEQ03.
           05 RT-ACCOUNT           PIC X(8).
           05 RT-RECORD-TYPE       PIC X(2).
           05 RT02-ADDR-LINE-1     PIC X(20).
           05 RT02-ADDR-LINE-2     PIC X(20).
           05 RT02-ADDR-CITY       PIC X(18).
           05 RT02-ADDR-STATE      PIC XX.
           05 RT02-ADDR-ZIP        PIC X(10).
      ******************************************************************
      * END OF DECLATION FOR ABX.TRAINING.ABXSEQ03.INPUT
      ******************************************************************
```

When File Export parses the COBOL copybook, the field definitions look like those in Figure 5-7.

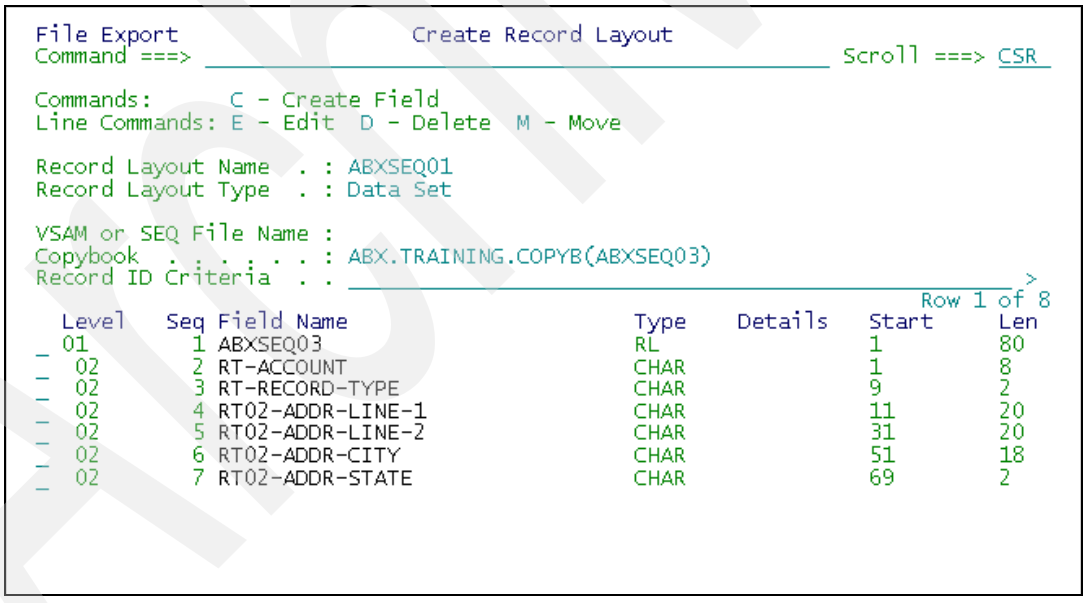


Figure 5-7 The copybook has been parsed into File Export-format field definitions

Use the following steps as a suggestion to parse a COBOL or PL/I copybook:

Step 1: Select option 1 off of the Metadata Management panel to Manage Record Layouts, and press Enter.

Step 2: On the panel titled Manage Record Layouts, provide the Record Layout a name and specify DS as the underlying source type.

Step 3: Tab down to the section titled 'If Using a Copybook' and fill in the appropriate options for either a COBOL or PL/I copybook.

The completed panel looks like the sample in Figure 5-8:

```
File Export                    Manage Record Layouts
Command ===> C                                                  More:     +
Commands: E - Edit  C - Create  V - View  D - Delete  L - List Record Layouts
          S - Specify Copybook SYSLIB Data Sets

Record Layout Name  . . . . ABXSEQ01                                        >
Underlying Data Store Type  DS  I-IMS, DS-VSAM or Sequential, D-DB2
Save Data Store Association Y   Y-Yes, N-No

If using a Copybook
  Compiler to Use . . . . . C   C-COBOL, P-PL/I
  Compiler Options  . . . .                                                 >
  Data Set Name . . . . . . ABX.TRAINING.COPYB
  Member Name . . . . . . . ABXSEQ03  Blank or pattern for member list

If IMS Database
  SSID  . . . . . . . . . . ____
  DBD . . . . . . . . . . . ________
  Segment . . . . . . . . . ________
If VSAM or Sequential Data Set
  Data Set Name . . . . . . ____________________________________________
  Member Name . . . . . . . ________  Blank or pattern for member list
If DB2 Table
```

Figure 5-8 The Manage Record Layouts panel completely filled in.

Step 4: Type C on the Command Line to create the Record Layout and press Enter.

Step 5: A temporary copy of the copybook is presented in an ISPF Edit session. Any last-minute edits to the copybook definitions can be made here. When all edits have been completed, press PF3 to save the edits and begin parsing the copybook.

Figure 5-9 shows the ISPF Edit session of a copybook.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CSJENN.ABX.TEMP.D061026.T201719.ORG            Columns 00001 00072
Command ===>                                               Scroll ===> CSR
****** ***************************** Top of Data ********************************
000001       ************************************************************************
000002       *                                                                      *
000003       * COBOL COPYBOOK DEFINITION FOR                                        *
000004       *       ABX.TRAINING.ABXSEQ03.INPUT                                    *
000005       *       LANGUAGE(COBOL)                                                *
000006       *                                                                      *
000007       ************************************************************************
000008       ************************************************************************
000009       * COBOL DECLARATION FOR ABX.TRAINING.ABXSE03.INPUT                     *
000010       ************************************************************************
000011        01  ABXSEQ03.
000012            05 RT-ACCOUNT           PIC X(8).
000013            05 RT-RECORD-TYPE       PIC X(2).
000014            05 RT02-ADDR-LINE-1     PIC X(20).
000015            05 RT02-ADDR-LINE-2     PIC X(20).
000016            05 RT02-ADDR-CITY       PIC X(18).
000017            05 RT02-ADDR-STATE      PIC XX.
000018            05 RT02-ADDR-ZIP        PIC X(10).
000019       ************************************************************************
```

Figure 5-9 ISPF Edit session containing a temporary copy of a COBOL copybook.

Step 6: Once the parse is complete, File Export-format field definitions are displayed in the panel titled Create Record Layout, see Figure 5-7 on page 101.

5.5 Extract Engine

The extract engine is a batch facility called ABXUTIL. It is the program which reads the Control Syntax in the generated JCL, and performs the instructions for the extract and/or load. File Export Control Syntax is part of the generated JCL and resembles SQL syntax. As such, users familiar with SQL can understand the instructions the Control Syntax passes to the ABXUTIL program. However, as a reminder, the Control Syntax is generated by File Export and no editing is required by the user.

Example 5-2 shows the generated JCL containing Control Syntax.

Example 5-2 Generated Extract JCL with Control Syntax In-stream

```
******************************** Top of Data*********************************
//JOBCARD JOB CSJENN,CLASS=A,NOTIFY=&SYSUID
//*
//STEP1    EXEC PGM=ABXUTIL
//STEPLIB  DD   DISP=SHR,DSN=ABX.WRK0120.LOADLIB
//         DD   DISP=SHR,DSN=DSN.V710.SDSNLOAD
//*
//ABXCTL   DD   *
OPTIONS(DB2, PLAN(JNABX120));
SOURCEDB2(R71A);
TARGETDB2(R71A);
DB2RESULTSET(RESULTSET1) SOURCE(CSJENN.ORG)
TARGET(CSJENNA.ORG) DD:TABL0001 LOAD(RESUME);
SELECT * FROM CSJENN.ORG;
DB2RESULTSET(RESULTSET2) SOURCE(CSJENN.STAFF)
TARGET(CSJENNA.STAFF) DD:TABL0002 CHILD OF RESULTSET1 WHERE
RESULTSET2.DEPT = RESULTSET1.DEPTNUMB LOAD(RESUME);
SELECT * FROM CSJENN.STAFF;
DB2RESULTSET(RESULTSET3) SOURCE(CSJENN.IT_ASSETS)
TARGET(CSJENNA.IT_ASSETS) DD:TABL0003 CHILD OF RESULTSET2 WHERE
RESULTSET3.ID, DEPT = RESULTSET2.ID, DEPT
SELECT * FROM CSJENN.IT_ASSETS;
//*
//ABXLOADC DD   DSN=CSJENN.DB2LAB1.EXTRACT.LOADCNTL,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=SYSDA,
//            SPACE=(TRK,(1,1))
//ABXPRINT DD  SYSOUT=*
//SYSABEND DD  SYSOUT=*
//TABL0001 DD  DSN=CSJENN.DB2LAB1.EXTRACT.TABL0001,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=SYSDA,
//            SPACE=(CYL,(10,10),RLSE)
//TABL0002 DD  DSN=CSJENN.DB2LAB1.EXTRACT.TABL0002,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=SYSDA,
//            SPACE=(CYL,(10,10),RLSE)
//TABL0003 DD  DSN=CSJENN.DB2LAB1.EXTRACT.TABL0003,
//            DISP=(NEW,CATLG,DELETE),
```

```
//             UNIT=SYSDA,
//             SPACE=(CYL,(10,10),RLSE)
***************************** Bottom of Data*****************************
```

The Control Syntax in this example begins with the keyword OPTIONS and ends with the last SELECT statement. In this example, the Control Syntax provides information to the ABXUTIL batch processor program about the name of the SOURCE to find the data to extract, the TARGET object to load the extracted data to, the relationships between the three DB2 tables, and the SELECT criteria to find specific data.

The Control Syntax keywords are defined here for clarity:

SOURCEDB2(<SSID>);

The Source DB2 SSID on which the source DB2 tables reside.

TARGETDB2(<SSID>);

The Target DB2 SSID on which the target DB2 tables reside.

DB2RESULTSET(RESULTSETn)

Indicates that the extracted data from the named SOURCE is to be held in virtual memory identified by the specified result set name.

SOURCE(<tbowner.tbname>)

The source table from which data is to be extracted.

TARGET(<tbowner.tbname>)

The target table into which related extracted data is to be loaded.

DD:<dd name>

The intermediate dataset where the extracted data is to be held before loading into the target table.

CHILD OF RESULTSETb WHERE RESULTSETa.<column> = RESULTSETb.<column>

Syntax that indicates an object is a child of the object referenced in the result set name. The WHERE clause behaves much like SQL syntax in that the column from the child extract must match the column from the parent extract.

Using the syntax in Example 5-2 above, table STAFF is a child of table ORG, and related data from STAFF is only selected when data in column STAFF.DEPT matches data in column ORG.DEPTNUMB.

6

Introduction to File Manager for z/OS

In a volatile economic climate, organizations are faced with constant pressure to create new applications that help reduce costs, engender customer loyalty, and earn competitive advantage. The requirement to tighten budgets and do more with less only increases the challenge. To succeed in a fiercely competitive marketplace, companies must be able to make the most of existing investments and manage application development effectively.

Through its comprehensive, user-friendly tools, IBM File Manager for z/OS Version 7.1, introduces a new component to run and permit the usage of CICS data and other data directly from the CICS environment. IBM intends to continue and improve the functions in order to help and work more efficiently with IBM z/OS data sets, IBM DB2 data, and IBM IMS data.

Extending the standard browse, edit, copy, and print utilities of IBM Interactive System Productivity Facility (ISPF) and Custom Information Control System (CICS), File Manager delivers enhancements that are designed to address the requirements of application developers working with structured data files. And because the CICS and basic features of the File Manager editor and the ISPF and portable document format (PDF) editor are almost identical, you can take advantage of extra features without having to relearn fundamental skills. In the following section we show the initial CICS and ISPF selection panels.

6.1 Overview

File Manager is composed of four components as shown in Figure 6-1:

- A base component to manage z/OS data sets, such as queued sequential access method (QSAM), virtual storage access method (VSAM), partitioned data sets (PDS), and IBM z/OS UNIX Systems Services (USS) Hierarchical File System (HFS) files
- A DB2 component to manage DB2 data
- An IMS component to manage IMS data

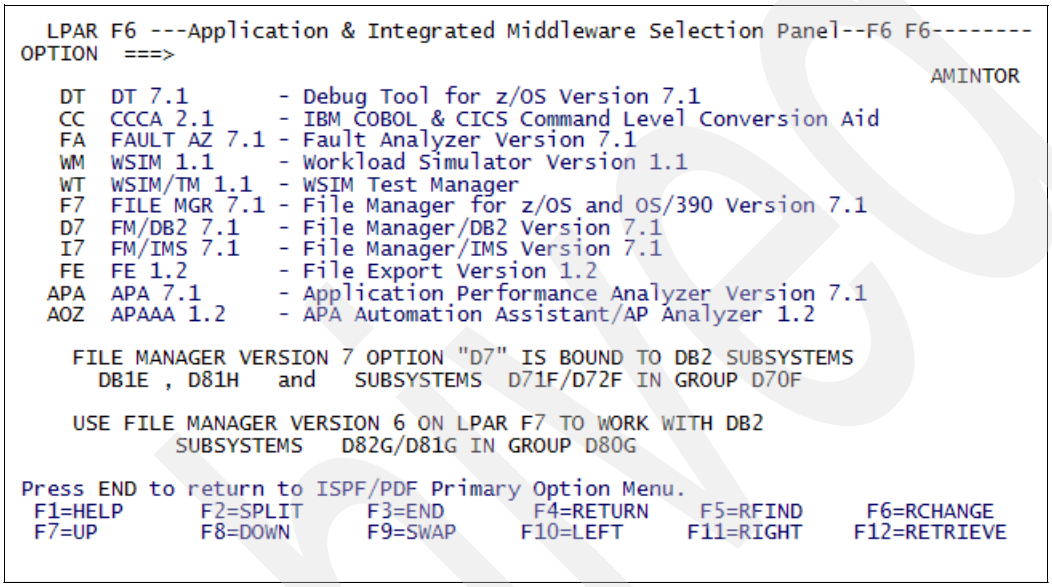


Figure 6-1 ISPF File manager options

- A CICS component to include familiar Base and IMS utilities found in ISPF and to manage CICS resources as shown in Figure 6-2.

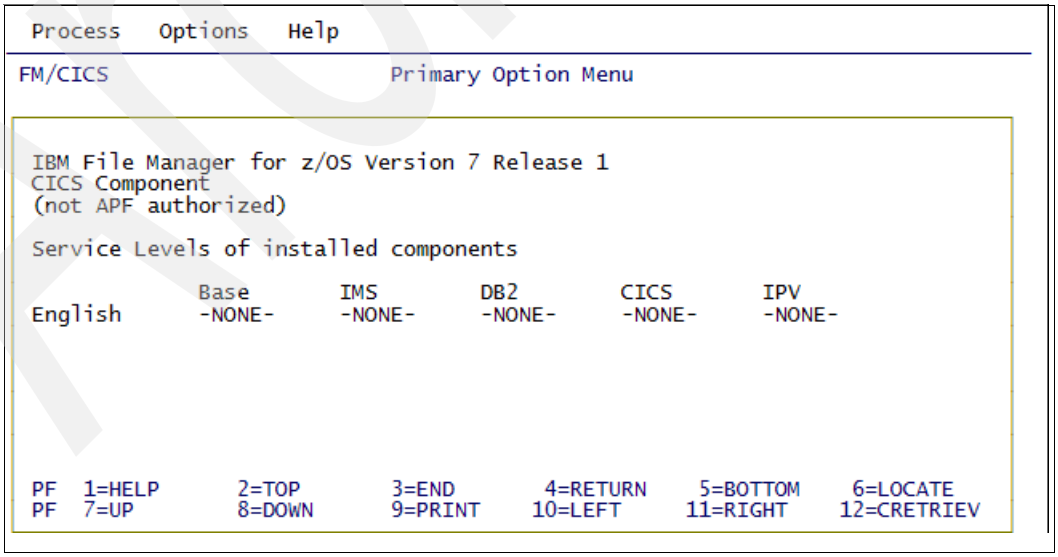


Figure 6-2 CICS File Manager Component

File Manager Version 7.1 highlights

File Manager Version 7.1.0 introduces a new CICS component; here are the highlights:

- You can invoke File Manager functions from a CLIST in batch mode (or REXX procedure) or via the IBM Productivity Tool (IPT) Program.
- You can edit sharing options control when VSAM files are potentially being shared by other users.
- You can copy and generate data in external format - XML representation.
- There is enhanced usability with *Templates* by improving and facilitating existing utilities. A new utility, the Batch Template Update utility, allows you to update multiple templates from a copybook PDS.
- The Template Update utility (both interactive and batch) allows you to update one or more templates and to easily identify templates that are affected by copybook changes.
- You can use Batch Template Build and Update (BTB and BTU) to create new or update existing templates based upon existing templates.
- Support is also provided for:
  - Segmented records
  - Large Block Interface (LBI) for tape devices
  - Large DASD volumes and extended datasets to manage SMF records

6.1.1 File Manager for working with MVS data sets

The File Manager base component helps speed the application-development process by identifying the structure of your records and displaying each field according to its data type. This component supports QSAM, VSAM, PDS, HFS, and indexed access method (IAM) data sets, including support for double-byte character set (DBCS) data in these data sets. A new feature of the Copy utility includes support to generate data in external format, XML representation, from an input data set (file) and a template. You can edit entire files (regardless of size) and sort data during an edit or browse session. You can use the IBM CICS VSAM Copy tool (separately available) in conjunction with File Manager capabilities to produce ad hoc copies of VSAM data sets for testing purposes without affecting CICS activity.

Using templates, you can select the records you want based on field values, find and change data in individual fields, display fields with headings showing name, data type, and length — and then print or copy only the fields you desire. You can quickly and easily work with files containing multiple record structures and many segment types, called segmented records, each of which will display as a separate shadow line, and copy data between fields of different data types and lengths.

In the base component, you can generate templates from a copybook that contains either COBOL data-description entries or PL/I DECLARE statements, or you can define your own fields in a dynamic template. You can then add record-selection criteria and other formatting information, and save the templates for reuse with different data sets that have the same record structure, or for reuse with different File Manager utilities. For example, while browsing through data sets, you can create, refine, and save a template that displays records meeting certain criteria, such as records with particular field values. Later, you can reuse that template in the File Manager copy utility to extract the records that meet those criteria and copy them to another data set.

The base component processes most record identification and selection criteria internally, while providing fast access to Restructured Extended Executor (REXX) for complex criteria statements. The copy and print utilities use the IBM DFSORT utility for supported data sets, for significant performance improvements. Support is also available for DFSORT statements.

The copy utility also offers various PDS-member copying capabilities, including the ability to copy members based on finding a particular condition to be true from records in the member. The find/change utility offers flexible options to increase efficiency for batch and online modes. With the data-set compare utility, you can use a number of different synchronization types. You can also enhance File Manager with your own custom REXX procedures (such as tallying field values) and automate tasks in batch jobs, REXX procedures, or command lists (CLISTs). A high-performance subset of REXX and File Manager REXX functions, called FASTREXX, is also available. The FASTREXX subset is broad enough to handle most normal processing tasks.

The File Manager base component also includes other utilities, such as these:

- Tape utilities enable you to perform actions such as copy, update, and compare with tape files.
- Catalog services utilities provide a user interface for working with catalog entries, and for listing or printing catalog information. System Catalog Services (SCS) enhancements have been done in this FM release to improve the usability.
- OAM utilities enable you to list, browse, update, and erase OAM objects, as well as copy, back up, and restore them.
- Disk utilities allow you to list the data sets on a disk, as well as their disk extents, and search for data within a disk extent.
- Capabilities are provided to:
  - Show the total number of records in a file, and the number of records currently selected
  - Edit load modules with in-place editing capabilities
  - Generate IDCAMS cards for batch file creation
  - Display data set information for IAM data sets
- The Data Set Copy utility has an option to copy job control language (JCL) record-format data sets. This option enables you to process multiple lines of continued JCL as a single logical JCL record.
- The Data Set Compare (DSM) utility allows for optional printing of template, template criteria, and template mapping information, to enable auditing of the effect of templates on the compare operation.
- The Find/Change utility can now find members that do not contain a given string.
- The View/Print Template and Copybook utility can optionally display record-identification and record-selection criteria in a template.

6.1.2 Base component enhancements in this release

A number of new functions and enhancements are included with the base component in File Manager for z/OS to provide quicker processing or to reduce processing requirements. The following functions are affected by these enhancements:

- Usability improvements make it possible to:
  - Generate data in XML format.
  - Change the length of the record for those files that support variable-length records.
  - Display the alphanumeric fields in long hexadecimal.
  - Edit the related ID criteria in the template when working with “segmented records”.
  - Work with files that are used in a shared environment, controlling the behavior when a VSAM file is being shared by other users.
  - Benefit from the improved Compare Utility: Options panel.

- Display details about the current COBOL compiler during the template's generation.
- FIND and CHANGE primary command support (ISPF like) for **F** or **C X ALL**.

- Other base component enhancements:
  - You can now choose whether to display the Relative Block Address (RBA) and length of VSAM file records when using the File Manager browse function.
  - You can specify the default view mode (such as TABL, CHAR, or Previous) upon entering File Manager edit or browse mode.
  - The File Manager editor and browser have improved the processing of a subset of a data file by providing new options to sample records from a data file, a new option to drop unselected records from memory, and a faster editor and browser startup near the end of a data file.
  - The File Manager base display VTOC utility provides new statistics, a new report format, and more sorting options.
  - You can display:
    - Current Edit/Browse options when using the File Manager Browse or Edit functions
    - VTOC improvement
    - Volume information for a VSAM or non-VSAM data set,
    - Extent details for non-VSAM data sets
- Performance:

  Many File Manager functions have been enhanced to provide quicker processing or to reduce processing requirements. Functions affected by these enhancements include the FIND and CHANGE commands, FASTREXX, and a range of utilities, such as Find/Change, Display Volume Table of Contents (VTOC), Data Set Copy and DataSet Update.
- Mixed-workload enhancements:
  - The ability to invoke the File Manager functions from the Product IBM Productivity Tool (IPT or SPIFFY) and whenever you select a VSAM Cluster
  - Invoke File Manager functions from a REXX procedures or CLIST in /BATCH mode.
  - The ability to invoke UNIX Systems Services (USS) Hierarchical File System (HFS) utilities from a File Manager base panel.

File Manager supports large-scale file management tasks and will be possible to generate tagged XML text for all input data types. Data creation and data copying process can be performed by treating your data sets as entire units, rather than working with individual records within data sets. You can request that the output of the copy operation be well-formed XML data. The output format derived from the input template also conforms to XML rules.

File Manager manages *record integrity checking*. This means that when File Manager is operating in a shared environment, it enables record integrity checking before rewriting the contents of a record to a file. File Manager first checks if the record has been updated (by another user) since the record was retrieved. File Manager performs this check by comparing the contents of the record at the time that it retrieved it with the contents of the current record as it exists on the file.

The File Manager editor and browser have improved the processing of a subset of data file by providing new options to drop unselected records from memory, and a faster editor and browser startup near the end of a data file.

The Display VTOC panel is the entry panel for the Work with VTOC utility. The File Manager base display VTOC utility provides new statistics such as a count of the number of volumes selected, a new report format, and more sorting options. In this panel, you can specify a data set name or name pattern, and the disk volume or volumes with which you want to work.

6.2 Major functions

This section describes four major functions of File Manager, as follows:

- DFSORT support
- Templates
- REXX functions
- batch processing

6.2.1 Support for DFSORT

If you plan to enable File Manager to use the DFSORT COPY function to improve File Manager performance, you might want to add the DFSORT libraries to the STEPLIB DD statement. If you do this, you must add them in the order SICELINK, followed by SORTLPA. If you choose to do this and DFSORT is not your primary sort product, you must also add your sort products libraries in front of the DFSORT libraries.

Customizing to use DFSORT to improve File Manager performance

You can optionally enable File Manager to use the DFSORT COPY function to improve the performance of the File Manager Data Set Copy and Data Set Print functions, both under ISPF and in batch.

In addition, if DFSORT is enabled, its COPY function will also be used to copy data to Auxiliary Storage (VSAM RRDS) when editing large data sets under ISPF.

Requirements

The requirements for DFSORT are as follows:

- You do not require a DFSORT license to enable File Manager to use DFSORT.
- Minimum DFSORT R14 with PTFs UQ90054 and UQ84945 applied.

Note: DFSORT R14 is available with all releases of z/OS, and with all releases of OS/390 starting with 2.7.

Overview

File Manager will use DFSORT to improve the performance of the Data Set Copy and Print functions, provided that it can find DFSORT and verify that DFSORT is at the correct maintenance level for File Manager use. DFSORT is an optional, priced feature of z/OS and OS/390; you must have a DFSORT license to use DFSORT outside File Manager.

However, the DFSORT code is always shipped with z/OS. As long as you did not delete the DFSORT libraries when you installed z/OS or OS/390, you can enable File Manager to use the DFSORT code. File Manager accesses DFSORT using the aliases ICEDFSRT and ICEDFSRB. These aliases are associated with DFSORT when PTF UQ90054 is applied to DFSORT.

Important setup for DFSORT as the primary sort product

The setup is as done follows:

- Resident:
  - The DFSORT library, SORTLPA, is in LPALST.
  - The DFSORT library, SICELINK, is in LINKLIST.
- Non-resident:

  The DFSORT libraries SICELINK and SORTLPA are private libraries.

  - If you have installed the DFSORT resident, no further action is required to make DFSORT available to File Manager.
  - If DFSORT is installed non-resident, you must take different actions to enable File Manager to use DFSORT under ISPF and in batch.
- Under ISPF (interactively):

  Add the DFSORT libraries, SICELINK and SORTLPA, in that order, to the STEPLIB DD statement in your TSO logon procedure.

  Important: If you do not want to add these libraries to your TSO logon procedure, you can add them via the TSOLIB command, before invoking ISPF.

- In batch:
  - To enable File Manager to use DFSORT in batch, add the DFSORT libraries, SICELINK and SORTLPA, in that order, to the JOBLIB or STEPLIB DD statement for the job or step that uses File Manager.
  - You might want to create a File Manager JCL procedure for use at your site.
  - You could also add the DFSORT libraries to the STEPLIB DD statement in the batch JCL skeleton. If you do this, you must add them in the order SICELINK, SORTLPA.

DFSORT customization as the non primary sort product

If you use a non-IBM sort product as your primary sort product, and you want to enable File Manager to use DFSORT, you must place the DFSORT libraries after your primary sort product libraries in the system search order. Ensure that you have DFSORT R14 with PTFs UQ90054 and UQ84945 applied.

Attention: DFSORT and non-IBM sort products typically have entry points with the same names, for example, SORT and ICEMAN. Therefore, if the DFSORT libraries are placed before your primary sort products libraries, DFSORT will become the primary sort product. In this situation, if you do not have a license for DFSORT, all sort jobs that use your primary sort product will fail.

The use of a non-IBM sort product as your primary sort product has to be installed in one of the following ways:

- Resident: Your primary sort products libraries are in LPALST or the LINKLIST (or both).
- Non-resident: Your primary sort products libraries are private libraries.
  - If you have installed your sort product resident, place DFSORT's SORTLPA and SICELINK libraries in the LINKLIST (after your primary sort product libraries in the LINKLIST, if any) to enable File Manager to use DFSORT.
  - If you have installed your sort product non-resident, you must take different actions to enable File Manager to use DFSORT under ISPF and in batch.

Customizing to use DFSORT to improve File Manager performance

Here we describe how to enable File Manager to use DFSORT:

- Under ISPF (interactively):

  To enable File Manager to use DFSORT under ISPF, add the DFSORT libraries, SICELINK and SORTLPA, in that order, after your primary sort product libraries, to the STEPLIB DD statement in your TSO logon procedure. (If you do not want to add these libraries to your TSO logon procedure, you can add them via the TSOLIB command, before invoking ISPF.)

- In batch:

  To enable File Manager to use DFSORT in batch, add the DFSORT libraries, SICELINK and SORTLPA, in that order, to the JOBLIB or STEPLIB DD statement for the job or step that uses File Manager, after your primary sort products libraries.

  You might want to create a File Manager JCL procedure for use at your site.You could also add the DFSORT libraries to the STEPLIB DD statement in the batch JCL skeleton. If you do this, you must also add your primary sort products libraries.

  You must add them in this order: your sort product libraries, followed by SICELINK, then SORTLPA/.

DFSORT uses an SVC

SVC resides in the SORTLPA library.

By default, this is SVC 109.

- This SVC is used by DFSORT to record SMF type-16 records or to use IBM cached DASD devices (such as the 3990 model 3).
- If your non-IBM sort product uses its own version of SVC 109, you will have to make an alternative SVC number the default SVC for DFSORT.

Important: File Manager does not require the DFSORT SVC for its own operations. Therefore, if DFSORT if not your primary sort product, it is not necessary to install the DFSORT SVC.

Determining whether DFSORT is being used

There is a way to verify whether DFSORT is being used to improve the performance of File Manager. A verification can be performed both in batch and under ISPF. To check whether DFSORT is being used in batch, submit a File Manager batch job with the following control statement.

The JOB JCL is displayed in Figure 6-3.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       DENARDI.BNL.JCL(DFSORT01) - 01.00               Columns 00001 00072
Command ===> ____________________________________________________ Scroll ===> CSR
000011 //*************************************************************
000012 //*MNVERIF EXEC PGM=FILEMGR
000013 //*FMNVER   EXEC PGM=FMNMAIN
000014 //FILEMGR  EXEC PGM=FMNMAIN
000015 //STEPLIB  DD DSN=ADT7.FMN.SFMNMOD1,DISP=SHR
000016 //*        DD DSN=IGY.SIGYCOMP,DISP=SHR
000017 //*FMNCOB  DD DUMMY     Uncomment to force use of FM COBOL Compiler
000018 //SYSPRINT DD SYSOUT=*
000019 //FMNTSPRT DD SYSOUT=*
000020 //SYSTERM  DD SYSOUT=*
000021 //SYSIN    DD *
000022 $$FILEM VER
000023 $$FILEM SHOWSORT
000024 $$FILEM DSP FORMAT=CHAR,
000025 $$FILEM MEMBER=AAA4FILE,
000026 $$FILEM PACK=UNPACK,
000027 $$FILEM POSITION=0,
000028 $$FILEM REF=YES,
000029 $$FILEM RDF=YES,
000030 $$FILEM PIC=YES,
000031 $$FILEM SLOC=YES,
000032 $$FILEM STR=YES,
000033 $$FILEM TYPE=YES,
000034 $$FILEM TCIN=DENARDI.FMN.DATA(AAA4COPY),
000035 $$FILEM LANG=COBOL,
000036 $$FILEM COMPMAXRC=04,
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 6-3 Sample JCL with control card $$FILEM SHOWSORT

The following result is shown in Figure 6-4, where DFSORT is NOT being used as the sort utility, and Figure 6-5, where DFSORT is active.

```
  Display  Filter  View  Print  Options  Help
 SDSF OUTPUT DISPLAY DENARDIB JOB01168  DSID   102 LINE 6       COLS 02- 81
 COMMAND INPUT ===>                                          SCROLL ===> CSR
PRINTOUT=SYSPRINT  HEADERPG=YES  ASCII=BOTH      RECLIMIT=(1,*)
PRINTLEN=132       PAGESKIP=NO   PAD=OFF         EOD=/*
PAGESIZE=60        DATAHDR=YES   DBCSPRT=OFF     LANGUAGE=ENGLISH
PRTTRANS=ON        DUMP=UPDOWN   TAPELBL=SL      CYLHD=ABSOLUTE
SMFNO=253          PRTDISP=MOD   USEIOX=DISABLE  IOX=
TEMPHLQ=
ABENDCC=NONE

IBM File Manager for z/OS
$$FILEM VER
IBM File Manager for z/OS Version 7 Release 1
APF authorized
Service Levels of installed components
            Base        IMS         DB2         CICS
English     -NONE-      -NONE-      -NONE-      -NONE-
$$FILEM SHOWSORT
DFSORT not available
$$FILEM DSP FORMAT=CHAR,
$$FILEM MEMBER=AAA4FILE,
$$FILEM PACK=UNPACK,
$$FILEM POSITION=0,
$$FILEM REF=YES,
$$FILEM RDF=YES,
$$FILEM PIC=YES,
$$FILEM SLOC=YES,
 PF 1=HELP      2=SPLIT     3=END       4=RETURN    5=IFIND     6=BOOK
 PF 7=UP        8=DOWN      9=SWAP     10=LEFT     11=RIGHT    12=RETRIEVE
```

Figure 6-4 DFSORT not available

```
 Display  Filter  View  Print  Options  Help
--------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY AMINTORB JOB02993  DSID   102 LINE 9       COLS 02- 81
 COMMAND INPUT ===>                                           SCROLL ===> PAGE
PRTTRANS=ON        DUMP=UPDOWN   TAPELBL=SL     CYLHD=ABSOLUTE
SMFNO=000          PRTDISP=MOD   USEIOX=DISABLE IOX=
TEMPHLQ=
ABENDCC=NONE

IBM File Manager for z/OS
$$FILEM VER
IBM File Manager for z/OS Version 7 Release 1
APF authorized
Service Levels of installed components
             Base       IMS        DB2        CICS
English      -NONE-     -NONE-     -NONE-     -NONE-
$$FILEM SHOWSORT
SORT debugging is on
$$FILEM DSP FORMAT=CHAR,
$$FILEM MEMBER=*,
$$FILEM PACK=UNPACK,
  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=IFIND     F6=BOOK
  F7=UP        F8=DOWN      F9=SWAP      F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 6-5 Sample result where DFSORT shows to be active

The same result can be achieved interactively. Figure 6-6 shows using the SHOWSORT command on any command line of File Manager.

```
 Process   Options   Help
--------------------------------------------------------------------------------
FM/IMS                          Primary Option Menu          SORT debugging is on
Command ===> showsort

0  Settings      Set processing options                 User ID . : AMINTOR
1  Browse        Browse data                            System ID : STLABF6
2  Edit          Edit data                              Appl ID . : FMN1
3  Utilities     Perform utility functions              Version . : 7.1.0
4  Templates     Template/view/criteria set utilities   Terminal. : 3278A
X  Exit          Terminate FM/IMS                       Screen. . : 1
                                                        Date. . . : 2006/11/01
                                                        Time. . . : 12:23

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 6-6 SHOWSORT command sample in interactive mode

DFSORT is ON shown in the top right corner of panel. If DFSORT is available to File Manager under ISPF, the message `SORT debugging is on` is displayed at the top right corner, as shown above. If DFSORT is not available, the message is `DFSORT not available Customizing to use DFSORT to improve File Manager performance.` DFSORT is available in FM Base, IMS, and DB2 Components (all versions).

Summary

Once you have determined, from the foregoing information, the correct way to implement DFSORT for use by File Manager at your site, follow the instructions in the *DFSORT Installation and Customization Guide*, R14.

You do not have to customize DFSORT for use by File Manager, since File Manager automatically customizes its use of DFSORT's COPY function. File Manager does not use DFSORT's SORT or MERGE functions.

Note: When File Manager uses DFSORT, it presumes that DFSORT will produce message output and will open data set FMNSRTP for DFSORT messages. If the DFSORT installation option MSGPRT is set to NONE (MSGPRT=NONE), then an OPEN abend might occur when the File Manager DFSORT interface module (FMNDFSRT) tries to open message data set FMNSRTP. To avoid this, set the DFSORT installation option MSGPRT to ALL (MSGPRT=ALL).

6.2.2 Templates

File Manager uses templates to provide a logical view of your data. To enable File Manager to determine the record structure of a file, supply a copybook containing COBOL data description entries, or a PL/I include. File Manager interprets each level-01 group item in the copybook as a record structure, and each elementary item as a field.

After File Manager creates a template, you can add selection criteria and other formatting information. Use templates to map the data in your application files for a concise view of the contents. This includes the ability to view multi-record files.

You can save templates, eliminating the necessity of recreating them each time you browse or edit a file, and making them available for use with various File Manager utilities.

Templates can also be generated from DB2 table and IMS copybook (PSB) definitions. New FM utilities improve the Template maintenance, update and generations. For instance, you can create multiple templates from a PDS of copybooks by using the Template Build Utility. The utility builds one template from each selected copybook. The template name is derived from the input member name and any output member mask that you have specified. You can run the Template Build Utility in batch or foreground.

6.2.3 Segmented records

File Manager offers template support for segmented records. A *segmented record* is a record that contains one or more logical segments, each of which is defined and identified by a copybook layout. When you apply a *segmented data template* to these records, each segment is treated as a logical record within the supported functions.

To define each record segment, we have to:

- Obtain the copybook definitions in order to map the record structure.
- Apply or generate a segmented data template from the copybook.

There are many FM utilities/functions that can support the use of a segmented record template:

- Browse and Edit, DSU, and DSEB functions
- Copy utility and DSC function
- Compare utility and DSM function
- Print utility and DSP function

The following restrictions apply to template editing during the usage of a segmented data template:

- You cannot update the Template for segmented data field in the Record Type Selection panel (accessed with the TEDIT command).
- You cannot change the Record Identification Criteria, Related Identification Criteria, or Selection Criteria set in the Field Selection/Edit panel (accessed with the TEDIT command and then the E command on the Record Type).
- You cannot use the TVIEW command (which would otherwise display the Template Workbench, allowing you to select or create a different template).
- The length of a segment is calculated as the length of the matching structure, unless you provide a length field. Any binary or alphanumeric field less than 4 bytes in length can be set as the length field by editing the field attributes. If a length field is provided, then File Manager uses the binary value provided as the segment length and adjusts the structure length accordingly.
- Segmented data templates cannot be used for ISPF Packed data sets.

6.2.4 REXX functions

REXX is an extremely versatile programming language, providing powerful functions and extensive mathematical capabilities. When used with File Manager, the possibilities for data analysis and manipulation, coupled with complex business logic, are extensive.

File Manager's external REXX functions allow you to manipulate data in the foreground, even while using templates. This gives you the opportunity to selectively work with only the records you are interested in. In addition to all of the functions available in REXX, File Manager has several product-specific functions, which include:

- VSAM support: VSAM data can be copied in and out of REXX stem variables.
- FLD: Lets you refer to a field from the current input record.
- NCONTAIN: Lets you check for the existence of numeric values in a field.
- TALLY: Lets you total a field and report the value.

You can develop REXX procedures to take the place of repetitive, manual functions, and then save these routines to a common data set. You can use File Manager with your own custom REXX procedures (such as tallying field values) and automate tasks in batch jobs, REXX procedures, or command lists (CLISTs). A high-performance subset of REXX and File Manager REXX functions, called FASTREXX, is also available. The FASTREXX subset is broad enough to handle most normal processing tasks.

6.2.5 Batch processing

All of the File Manager functions are available as primary commands in batch mode. You can easily enhance File Manager with your own procedures, built using either DFSORT or REXX. Compiled REXX runs considerably faster than interpreted REXX. By supplying a DD card for the REXX compiler library (if it is installed), you instruct File Manager to compile your REXX before running it. This approach reduces CPU usage and runtime.

In addition, FM/IMS provides two functions that you can use in batch jobs. FM/IMS Edit Batch (IEB) runs a REXX procedure that can insert, update, retrieve, delete, or print segments and create views. FM/IMS Batch Print (IPR) can print the entire database in one of several available display formats, or a selected sub-set of the database, based on a view addition.

If you do not require user interaction (such as DSC, DSP, DVT, FCH, and SCS) and are otherwise available for JCL batch processing can also be invoked from a REXX program under TSO/ISPF in a *pseudo-batch* mode. This can be achieved by specifying a /BATCH parameter on File Manager invocation. When specified, it forces batch-type processing while executing under TSO/ISPF. Prior to program invocation, you must allocate all the required data sets, including SYSIN and SYSPRINT. All program output is written to the data set defined by SYSPRINT.

7

Introduction to Workload Simulator

In this chapter we describe Workload Simulator (WSim), which is a terminal and network simulation tool that can be used to evaluate network design, to perform and automate testing, and to determine system performance and response time.

WSim allows you to evaluate and test systems without having real terminals and terminal operators present. WSim can be used to simulate actions of several types of applications and terminals. The simulated resources communicate among themselves and with the real teleprocessing system (called system under test) as though they physically existed. The system under test does not have to be modified.

7.1 Overview

Workload Simulator is an automated testing tool that can simulate terminals and other network devices and associated network traffic, and report the status and results of tests. It can be used to perform several types of tests. Workload Simulator has several components:

- Batch utilities:
  - Capture data from live sessions
  - Prepare scripts
  - Run simulation tests
  - Generate reports
- WSim ISPF panels:
  - Run utilities online
  - Generate JCL
  - Run simulation tests interactively
  - Review test results
- WSim Test Manager ISPF application:
  - Simplifies and automates test process
  - Organizes tests by maintaining projects, test cases, network definitions, documentation, reports, and logs

Figure 7-1 provides a general overview of the WSim use context.

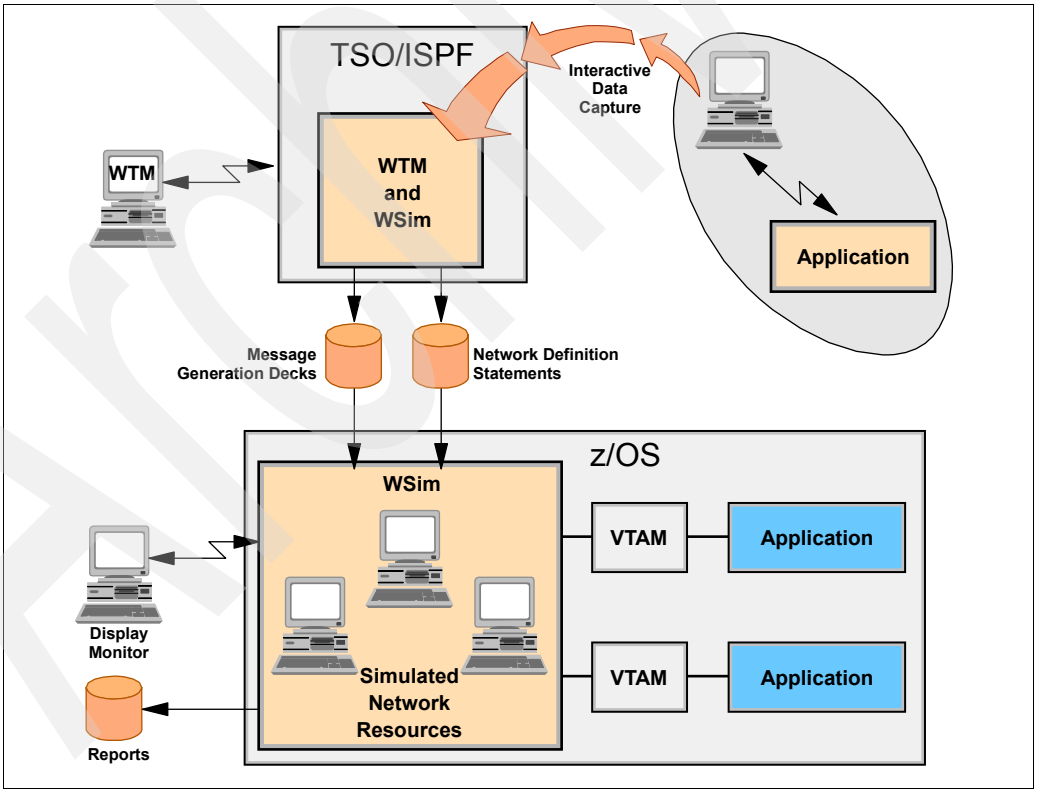


Figure 7-1 WSim - How it works

7.1.1 Use of WSim

To use WSim, you must prepare two types of information:

- Network definition statements describing the configuration of the network being simulated
- Message generation decks that send and receive messages

Network definition statements and message generation decks form a script, which WSim uses to send messages to the system under test. WSim collects and records the information received from the system under test. It also uses this information to determine what messages to send back to the system under test.

WSim enables the system under test to operate, to a certain degree, as it would under actual conditions.

The general sequence of tasks to perform when conducting a test is as follows:

1. Plan the test:
   a. Define the objectives.
   b. Prepare the test plan.
2. Configure the system:
   a. The actual system used to run WSim (physical configuration)
   b. The simulated network (logical configuration)
3. Prepare testing scripts:
   a. Prepare network definitions for the network to be simulated.
   b. Prepare message generation decks.
4. Run the test.
5. Analyze the results from WSim.

Planning is a very important task in this process. You should view it as an ongoing task and be prepared to refine the plan until desired results are achieved. Every test, especially when you begin using WSim, should start with a small sample network definition and a small simple message generation deck. After successful runs, more refined system configurations and more complicated scripts should be prepared until the simulation is done for the complete network to be tested.

7.1.2 Resources WSim can simulate

WSim can simulate the following three types of resources:

- System Network Architecture (SNA) logical units (LU) running as Virtual Telecommunications Access Method (VTAM) applications
- Common Programming Interface for Communications (CPI-C) transaction programs (TP)
- Transmission Control Protocol/Internet Protocol (TCP/IP) clients using Telnet 3270, 3270E, 5250, Network Virtual Terminal (NVT), File Transfer Protocol (FTP), or Simple TCP and UDP protocols attached to the TCP/IP network via the IBM TCP/IP for Multiple Virtual Storage (MVS) product

7.1.3 Testing with WSim

WSim can be used to conduct several types of tests:

- Function
- Regression
- Performance
- Stress
- Capacity planning

Function testing is usually used to test a particular function of the system and answers the question, “Is it working correctly?” WSim can be used to test functions such as new application transactions, logon and logoff procedures, error transactions, new hardware additions, and new software products. The scripts used in functional tests can be saved and reused later for regression or stress tests.

Regression testing verifies that old functions operate correctly after the addition of new ones, or after any other changes to the existing system, and answers the question, “Is it *still* working correctly?” Here are some advantages of using WSim for regression tests:

- Scripts are repeatable. Once created, a script can be reused many times until the tested functions are changed.
- WSim can be run automatically. Execution parameters and operator commands that control WSim operation, including ending the simulation after completing all of the test cases, can be included in scripts.

Performance testing includes taking measurements, changing parameters, and then taking measurements again. It answers the question, “How well does it perform?” WSim can be used to report terminal response times and it also provides the possibility to create a controlled, repeatable transaction load for the system under test.

Stress testing is performed when you must find problems in interactions and resource contentions. By loading the system under test with extremely high transaction rates you can answer the question, “What will break first?” This type of test is almost impossible to conduct without a special tool. WSim can generate controlled message traffic at controlled rates.

Capacity planning helps to predict how the system under test behaves when new resources are brought online or when one or more of the existing resources are overused. This type of test helps to determine if the system under test still performs adequately under predicted increased load, and answers the question, “What happens if this many resources are added?” WSim can drive the system under test with a higher than normal transaction rate and simulate additional terminals or different types of terminals.

When conducting performance, stress, and capacity planning tests, WSim should be run on a separate host from the system under test to avoid an impact on the results.

7.2 System configuration

The following terms are used in this chapter:

- **Logical unit (LU):**

  This is a port through which an end user accesses an SNA network to communicate with another end user or the system services control point (SSCP).

- **Transaction program (TP):**

  In WSim, this is any program using LU6.2 communications protocols to communicate with another program. WSim implements TPs using CPI-C.

- **Session:**

  A session is a logical connection enabling two network addressable units to communicate with each other, such as an LU-LU, or an SSCP-LU session. Each half of a session is a half-session.

Both the network to be simulated and the system to be used to run WSim must be configured before testing. Configuration of the network containing resources to be simulated by WSim and the real system to be tested (the system under test) is known as a *logical configuration*. For each logical configuration, a specific *physical configuration* must be used, which is the configuration of the real system used to run WSim. Resources of a physical configuration include a host processor, system software, application software, WSim, and so forth.

7.2.1 Physical configurations

WSim can operate in either one of two basic physical configurations:

- VTAM and CPI-C application configuration:

  This configuration is used to simulate LUs in the same subarea as VTAM. An LU can have a session with any other LU that VTAM allows it to start with. It is also used to simulate client and server CPI-C TPs in the same subarea as VTAM. TPs can have a conversation with any other TP on any LU to which VTAM allows a conversation to be started.

  This physical configuration contains WSim, VTAM, and VTAM applications, or TPs under test. WSim runs as a VTAM application program.

- TCP/IP application configuration:

  This configuration is used to simulate Telnet 3270, 3270E, 5250, NVT, and FTP clients. These clients can have a session with any Telnet 3270, 3270E, 5250, NVT, or FTP server that TCP/IP allows. This configuration can also be used to simulate Simple TCP or UDP clients in session with various servers.

  This physical configuration contains WSim, TCP/IP, and TCP/IP applications under test. WSim runs as a TCP/IP application program.

7.2.2 Logical configurations

WSim can operate in one of three basic logical configurations:

- VTAM application configuration:

  Used to simulate SNA LUs accessing VTAM applications. LUs could be terminals or other VTAM applications.

  This logical configuration contains VTAM, VTAM applications, and VTAM applications and LUs simulated by WSim.

- CPI-C application configuration:

  Used to simulate CPI-C client (allocates outbound conversations but does not accept inbound ones), TPs to test server (accepts inbound conversations), CPI-C TPs and network resources, or to simulate server CPI-C TPs to test client prototypes.

  This logical configuration contains VTAM, VTAM applications, and VTAM application CPI-C TPs and LUs simulated by WSim.

- TCP/IP application configuration:

  Used to simulate TCP/IP clients in a TCP/IP network, or simple TCP or UDP clients accessing an application through a TCP/IP server.

  This logical configuration contains a TCP/IP server and any Telnet 3270, 3270E, 5250, NVT, FTP, simple TCP, and simple UDP clients simulated by WSim.

7.3 Script preparation

After system configuration is defined, the definition of the network to be simulated is next. This is done by creating a script. Scripts contain two parts:

- Network definition statements to describe the devices to be simulated by WSim
- Message generation decks to define messages to be sent by the simulated resources to the system under test

7.3.1 Network definition statements

Network definition statements specify the following information:

- Types of the simulated resources in the network
- Attributes of the simulated resources
- Connections between the simulated resources and the system under test
- Special information about delays, logic tests, the order in which message generation decks are used, logging, or tracing of the messages, and so forth

NTWRK is always the first statement used to define a network. It names the network and specifies characteristics that apply to the network as a whole. It also can specify operands that establish defaults for lower-level statements. All other statements in the network definition follow the NTWRK statement in a prescribed order. The statements from the general simulation statements group immediately follow the NTWRK statement.

Different statements are used depending on what type of network is being simulated. For example, when simulating LUs accessing VTAM applications, the VTAMAPPL and LU statements must be used. When simulating CPI-C TPs, APPCLU and TP statements must be used. When simulating TCP-IP clients, TCP/IP and DEV statements must be used.

Not all statements are mandatory, and some could be coded more than once, but all the statements in each and every group, including the optional ones, should follow the prescribed order.

7.3.2 Message generation decks

Message generation is the process by which terminals send and receive messages. Message generation decks are used to control messages being sent out and actions taken when messages are received by a simulated terminal.

A message generation deck contains one or more statements used to generate messages, set delays, define logic tests, define and control event actions, save data for future use, and so forth.

Any terminal can use one or more message generation decks in any order.

Preparation of message generation decks involves the following steps:

1. Decide what transactions to test.
2. Decide which application files and what data to use.
3. Create message generation decks using one of the available methods.
4. Combine created message generation decks with network definition statements to form a script.
5. Test the script and modify and revise if required.

In WSim, a *transaction* is an exchange of data between a simulated resource and the system under test. The choice of transactions depends on the objectives of the test. Usually, it is not necessary to test all possible transactions in the application. Criteria for the inclusion can be:

- Transactions taking the most processor time
- Transactions generating the most messages
- Transactions being the most important in the application

The following items also should be considered:

- The content of the messages to be sent
- The messages expected to be received
- The mix of transactions, such as the order in which WSim executes the message generation decks and which terminals use which decks
- The transaction rate

Use the PATH statement to specify the order in which the decks are executed and the PATH operand on the DEV, LU, and TP statements to specify which paths a specified simulated resource executes.

Example 7-1 represents a fragment of the script for the RESNET1 network. The path SMALL specifies that the deck LOGON is executed before the deck LOGOFF by the LU TERM1. The path LONG specifies that the decks LOGON, ALLOC, BROWSE, and LOGOFF are executed in this order by the LU TERM2.

Example 7-1 Script fragments for the RESNET1 network

```
RESNET1 NTWRK
   .
   .
SMALL PATH LOGON,LOGOFF
LONG PATH LOGON,ALLOC,BROWSE,LOGOFF
   .
   .
TERM1 LU PATH=(SMALL)
TERM2 LU PATH=(LONG)
   .
   .
```

WSim executes the paths repeatedly, that is, when the terminal has executed the last deck in the path that is defined for it, it starts again with the first deck in its path. Terminals maintain their positions in the paths and are not affected by other terminals. BRANCH, CALL, and IF statements can be used to alter linear sequences of paths. The order in which WSim executes decks in any path can be certain, random, or based on probability distribution.

WSim can generate messages with controlled intermessage delays. This can be used to simulate the delays of real operators as they view the screen, think about the information, or enter more data. Intermessage delays can be defined for the entire network, a specific resource, or even on a message-by-message basis.

Before starting with the creation of message generation decks, the transactions to be tested should be thoroughly analyzed. All steps should be listed.

7.3.3 Methods for creating message decks

WSim provides several methods for creating message generation decks:

- Directly written message generation statements
- Programs in Structured Translator Language (STL)
- Use of one of the script-generating utilities provided with WSim to convert captured data traces

WSim provides the following script-generating utilities:

- Interactive Data Capture (can produce STL programs)
- Script Generator utility
- SNA 3270 Reformatter Utility

The method used depends on what is being tested and on the following factors:

- Familiarity with WSim
- What kinds of messages are sent to the system under test by WSim

It makes sense to trace actual system activity and use the Script Generator utility to convert the trace records if the test involves simulating a number of real users using an application.

Some or all of the methods might have to be used when preparing real tests.

Writing message generation statements

Knowledge of message generation statements is very important when interpreting the output from the STL translator, and when using the script generation utilities and debugging the scripts.

The message generation statements have to be coded by hand in situations such as these:

- When modifying the output from one of the script generation utilities
- When adding additional message generation decks in a script produced by the STL Translator or one of the script generation utilities
- When adding some special types of messages or special conditions in an SNA network
- When modifying already existing message generation decks

The syntax for message generation statements is similar to that for network definition statements.

Use the preprocessor to check the syntax and store message generation statements in data sets for use in simulations.

Using STL and the STL Translator

STL is a high-level structured programming language that can be used to create message generation decks and define terminals and devices to be simulated by WSim. STL utilizes constants, variables, expressions, and structured control statements.

An STL program is usually divided into one or more procedures. The STL Translator translates STL programs into message generation decks, each message generation deck corresponding to one STL procedure. Network definitions can be included in STL programs.

The STL Translator invokes the preprocessor to validate and store the network definition statements.

Example 7-2 shows two very simple STL procedures. Procedures begin with a MSGTXT statement and end with an ENDTXT statement.

Example 7-2 Message generation decks written in STL

```
/* STL procedure logging terminal on to RESAPPL */
Logon: Msgtxt
Initself('RESAPPL')
Endtxt
/* STL procedure testing message generation */
Tstmsg: Msgtxt
Do i = 1 to 5
Type "Hello, I expect you to respond Hi"
Transmit using PF4,
and Wait until on substr(screen,40,2) = "Hi"
End
Endtxt
```

The first STL procedure, named LOGON, defines the text that a terminal uses to log on to an application named RESAPPL. When the second STL procedure, named TSTMSG, is executed, WSim simulates a user typing `Hello, I expect you to respond Hi` and then pressing PF4 to send the message to the application. WSim waits for the application response `Hi` to appear at position 40 on the screen. These messages are sent five times.

The STL Translator can be invoked using JCL, a TSO CLIST, or the WSim/ISPF interface.

Using the Interactive Data Capture Utility

The Interactive Data Capture Utility (IDC) ITPIDC is a host application that can capture 3270 device session data and generate scripts. A user logs on the same way as for any other VTAM application, and through it can log on to the VTAM application to be tested and perform all the actions to be simulated by WSim. IDC capturing the session traffic is transparent to the VTAM application.

IDC can directly generate, from the captured session data, an STL program, WSim message generation decks, or both.

Using the script generator utility

The script generator utility creates message generation decks based on traces of real users using real applications. The captured trace must be put in a specified format and sorted by resource name, date, and time. The sorted trace is used as an input for ITPSGEN, which actually generates the message generation decks.

The methods that can be used to obtain a system activity trace are:

- The NetView® Performance Monitor (NPM) capturing path information units for selected LUs
- The Generalized Trace Facility (GTF) capturing the VTAM Buffer Trace
- User-written capture routines

WSim provides a special program ITPVTBRF to help with reformatting traces, which are not in the format required by ITPSGEN.

ITPSGEN also requires complete, syntactically correct network definitions as input. It uses the network definition statements to determine the terminal names for which to generate the decks. The names in the DEV and LU statements must correspond to the resource names used in the trace.

Using the SNA 3270 Reformatter Utility

The SNA 3270 Reformatter Utility (ITPLU2RF) is a batch utility for reformatting NPM log records (FNMVLOG) from LU2 sessions into log records. ITPLSGEN can be used to create STL programs or message generation decks based on ITPLU2RF output.

7.3.4 Testing scripts

The scripts must be tested to ensure that they are coded correctly, and that they function as intended. Statement syntax can be checked by using the Preprocessor or the STL Translator. To ensure that the message generation decks function as intended, the following methods can be used:

- Message trace records tracing the message generation process
- STL trace records tracing the message generation process for STL programs
- Self-checking scripts

To ensure that unexpected situations encountered during simulations are handled properly, use self-checking scripts. They do not have to be used for all simulations (for example, they can be skipped for short and simple ones). On the other hand, they definitely should be considered for a long-running test, which could be wasted if terminals were to go out of synchronization.

IF statement logic tests are added to the scripts to check for the expected response and to take action if an unexpected one is received. This action could be simple (such as stopping the device) or complex, or include several possible courses of action based on the actual response. The logic tests can be written in decks created by STL, one of the script generation utilities, or manually. They can also be coded in network definitions.

7.4 WSim output

WSim provides several online and printed reports to analyze test results. Some reports are produced by default, while some must be requested by issuing specific operator commands or running one of the WSim utilities. The types of reports are as follows:

- Operator reports indicating what is happening during operation
- The complete message log
- Reports generated by the following utilities based on the message log:
  - Formatted reports produced by the Loglist Utility
  - Reports on differences between 3270 display records in two message logs produced by the log compare utility
  - Detailed statistical analysis of response times produced by the response time utility
- Online response time statistics

Most of the reports are intended to represent how WSim is interacting with the system under test and not the effectiveness of the network or the application.

Interval reports monitor the current activity and status of each simulated resource in the network. The statistics are accumulated until the network is canceled or reset. End of run reports provide summary data from the simulated network. They are produced automatically and have the same format as interval reports. The inactivity report contains information about each inactive resource in the network.

The log data set is the single most valuable tool for debugging the scripts. This data set contains all data that has been transmitted or received by the WSim simulated resources. The message logging facility is active for the entire network, but it can be deactivated completely or just partially for a VTAMAPPL in the network. A separate log data set can be used for a particular network. This is convenient when running multiple networks since the results are logged separately.

The loglist utility uses the log data set. The control commands can be contained in a file or, as an alternative, entered at the operator console. Use the WSim/ISPF interface, JCL, or TSO CLIST to start the utility, to name the input files, and to specify where the formatted log is printed.

The loglist utility uses different formats for each type of log records. One particularly useful feature is the printing of screen image records. These images are updated each time a message is sent or received by the device. The output from the Loglist Utility for this type of log records looks the same as the screen images a user would see at the real device.

7.5 Operating WSim

WSim can be run using JCL, TSO CLIST, or by using the WSim/ISPF Interface.

The sample JCL can be found in the WSIMPRC6 member of the data set HLQ.SITPSAMP. The sample TSO CLIST can be found in the member WSIMRUN of the data set HLQ.SITPCLS. The value of HLQ and the method to invoke the WSim/ISPF Interface depend on how WSim and this interface are installed on the site. A typical WSim/ISPF main panel is presented on Figure 7-2.

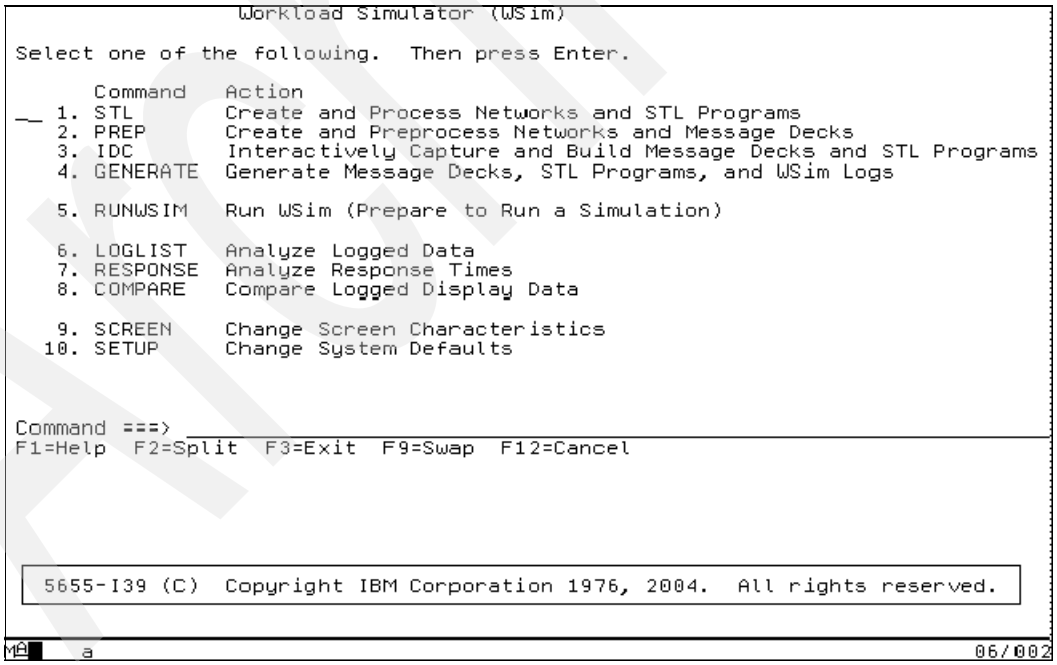


Figure 7-2 WSim/ISPF main panel

7.6 WSim Test Manager

The WSim Test Manager (WTM) is a usability enhancement that provides guidance through the test process. WTM offers selectable modes of operation, test management services, automatic script generation, and task automation.

The primary concept of the WTM testing structure is a project, which is a set of libraries containing schedules and test scenarios. Projects can be archived and reused. A project must be created before any schedules or test scenarios can be created using WTM.

A WTM schedule is a WSim network definition and the associated test scenario definition.

Test scenarios are organized into three levels: test case, test group, and test cycle. A test group is an ordered list of test cases. Test cases can be reused within multiple test groups. A test cycle is an ordered group of test groups and test cases.

WTM offers various ways to automate the development of test cases, which are WSim scripts written in STL. For 3270 environments, WTM can automate the script generation process from 3270 screen/keyboard captures (IDC), SNA traces, WSim or IDC logs, or from STL models and skeletons. Automated CPI-C test case generation uses SNA traces. The STL source is automatically translated into WSim MSGTXTs.

Generated test cases are paired with network resource definitions as part of developing WTM schedules. The WTM schedule is used by WTM to define and control the WSim simulation run (test). WTM schedules can be archived and reused.

The typical WTM main panel is shown in Figure 7-3.

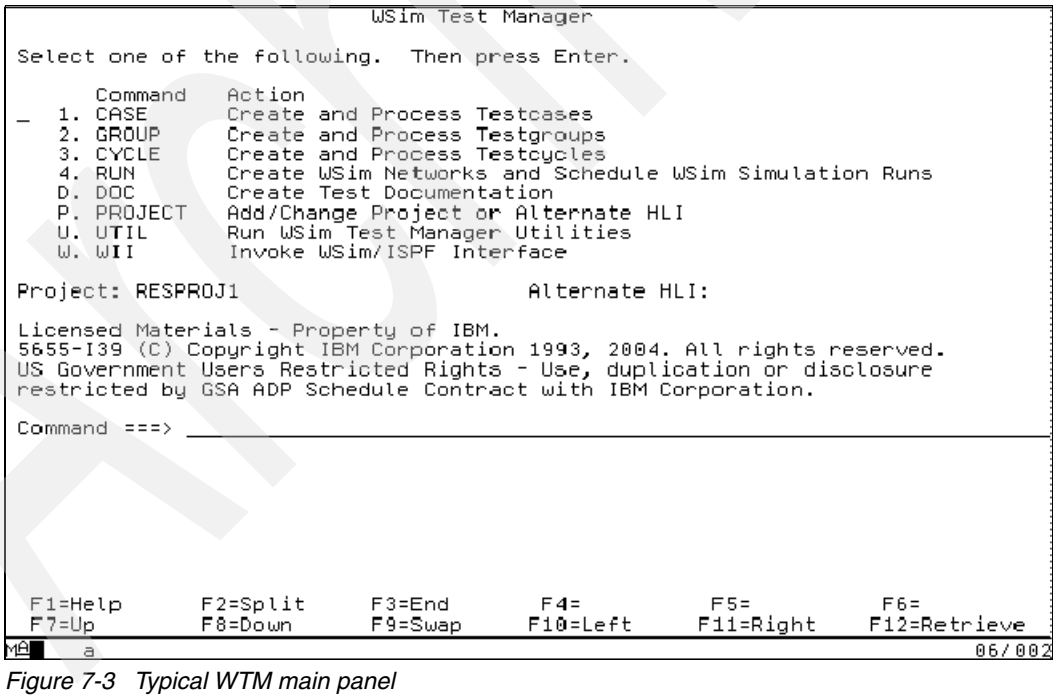


Figure 7-3 Typical WTM main panel

7.7 Latest enhancements

Applying the PTF, which fixes the APAR PQ94132 for the Workload Simulator, provides several general enhancements to this tool, two of them of a high significance:

- Password masking on formatted 3270 screens
- WSim Adapters for Rational TestManager

Passwords are usually maintained on the 3270 screens in unprotected non-display fields. While not visible, the passwords are sent to host applications in the clear and so are captured by the Interactive Data Capture utility or generated by script generation utilities.

The enhancement masks passwords by encrypting or hiding them using asterisks in test scripts and logs. The utility ITPGNKYZ is supplied to generate required USERMODE.

The Workload Simulator Adapters for Rational TestManager allow WSim Test Manager schedules and JCL scripts to be launched form the IBM Rational TestManager running on a remote workstation.

To run WTM schedules from Rational TestManager, the schedules must already exist in WTM on the host system. Some migration steps must be performed first. Also, a user ID and a password for a TSO user and the user ID of the WSim user (who created WTM projects and schedules) are required to run the WTM schedules from the Rational TestManager.

The white paper *IBM Workload Simulator Adapters for Rational TestManager Version 1, Release 1.0.1* and the install program are included with the PTF.

Part 3

Application Performance Analyzer for z/OS

In this part of the book, we describe how to customize and use Application Performance Analyzer (APA). It is divided into four chapters:

- How to set defaults for Application Performance Analyzer
- How to select a job to monitor
- How to monitor the job
- How to monitor applications accessing subsystems

How to navigate in and use Application Performance Analyzer

Application Performance Analyzer/ISPF is the main interface to Application Performance Analyzer. It is used for submitting new observation requests, and for navigating the Performance Analysis Reports generated from observation requests. Almost all panels in Application Performance Analyzer/ISPF are implemented as interactive reports.

Retaining open reports

When a report is opened, it can be retained for later viewing. Multiple reports (from multiple observation requests) can be readily available for viewing without regenerating the reports. We can navigate through all the open reports using the primary commands.

Report headings

Many reports present information in a tabular format with tows and columns. The table begins with heading lines, which contain title fields for each of the columns. Only data lines are affected by vertical scrolling commands. For help about title fields, position the cursor anywhere on the body of the report (not on an input field, almost all input field are underlined) and press PF1.

Commands

Two types of commands are available to navigate and control the Application Performance Analyzer reports:

- Primary commands, which we type on the command line
- Line commands, which we type in input fields in the body of the report

Primary commands

Report navigation primary commands can be displayed by:

- Positioning the cursor on the command line and pressing PF1
- Selecting Navigate on the action bar menu

> **Note:** The action bar menu can be removed from the display by entering the PREF command line, and de-selecting the Action Bar Visible option.

Here is a partial list of commands used to navigate reports and explanations for using them:

WIN	To display a selection list of open reports, and make a selection to jump to a selected report
PF4 - JUMP	To jump to the next open report, on a rotating basis
REPORT CODE	To open any valid three character report code
PREF	To set preferences for general display
CONNECT	In a multiple APA instances installation, to change the one we are connected with
VERSION	To list all the APA instances available and their versions
IMPORT	To import a sample file previously exported by the EXP command line

Line commands

Report navigation line commands can be displayed by:

- Positioning the cursor on the command line and pressing PF1, then PF8
- Positioning the cursor on any input field in the body of a report (not column heading), typing /, and pressing Enter.

The line command is entered in an input field in the body of a report such as a column heading or a data field in the report. Input fields are always underlined.

The allowable line commands vary depending on the report. The generally available line commands are summarized here:

/	Displays context menu
?	Displays context help information (or PF1 on an input field as a shortcut)
+	Expands
++	Shows additional details (or just press Enter as a shortcut)
-	Collapses
SV	Sorts by value
SN	Sorts by name

8

Application Performance Analyzer: Settings and customization

System programmers and system administrators have to customize Application Performance Analyzer to be used by application developers for monitoring performances of a job, started task, or TSO/E session.

In this chapter we discuss how you can configure Application Performance Analyzer with parameters that offer the greatest advantage to your installation.

Each application developer or group of application developers can have their own listings data sets, their own DB2, and their own rules to use Application Performance Analyzer for z/OS.

In the first part of this chapter we explain how system programmers can customize Application Performance Analyzer:

- To define the Application Performance Analyzer started task configuration settings
- To define the Application Performance Analyzer security rules
- To define the Application Performance Analyzer Descriptive Program Attribution
- To customize DB2 Universal Database to work with Application Performance Analyzer

In the second part of this chapter we show how application developers can customize Application Performance Analyzer to work with their own preferences and their own environment.

The defaults we discuss here are stored in our ISPF profile, in member *tso-userid*.ISPPROF(CAZ1PROF).

8.1 APA started task configuration settings

The Application Performance Analyzer started task CAZ0 (by default) obtains these settings from a file allocated to DDname CONFIG. The file is a SYSIN-type member, which we edit in order to change the settings. The member CAZCNFG0 in *hlq*.SCAZSAMP contains the initial settings. A group of configuration settings is specified by a CONFIG statement. Settings are organized into categories or classes:

BASIC To specify values for basic characteristics of the APA installation
SAMPLE To set values pertaining to the sampling (measurement) process

We do not explain all of the settings, but rather, only those that have a direct impact on the performance reports. For a description of each of them, refer to *IBM Application Performance Analyzer for z/OS: Customization Guide*, SC18-9599.

8.1.1 CONFIG BASIC statement

Here is the relevant information about the main parameters:

ExpiryDays Default number of days a sample file should be retained.

Security Specify *External* to use an external security product.
Specify *Internal* to use Application Performance Analyzer's internal security scheme.

DeleteOnJCLError Specify whether multiple-step requests that get a JCL error in one of the steps should delete all the sample files and entries for the remaining steps.

8.1.2 CONFIG SAMPLE statement

The relevant information about the main parameters is as follows:

SampleDur The default length of time (in seconds) that a measurement session collects sampling data.

NSamples The default number of samples that a measurement session collects during the measurement session.

MaxSampleRate The maximum rate (samples per second) at which a measurement session is permitted to sample (default 1,000).

MaxSampleSize The maximum number of samples that a measurement session can record. The maximum value is 175,000.

DB2I Specify that the DB2 Intercept should be turned on. Turning this feature on makes the DB2+ data extractor available, which allows additional DB2 data to be collected.

> **Note:** We might want to restrict its use in production environments, using security rules to restrict the DB2+ feature to specific users.

DB2PlanName The plan name must be specified if we want to report plan and package BIND timestamps or to use the DB2 EXPLAIN feature.

DB2EXPLAIN DB2EXPLAIN and DB2+ parameters must be specified to use the DB2 EXPLAIN feature.
The syntax for the DB2EXPLAIN keyword is as follows:
`DB2EXPLAIN=(ssid,D|Q|A,Yes|No,Yes|No,dbname,tsname)`

The first parameter specifies a DB2 subsystem name. The next five positional parameters are positional and optional. For a description of each of them, refer to *IBM Application Performance Analyzer for z/OS: Customization Guide,* SC18-9599.

IMSI Specify that the IMS Intercept should be turned on. Turning this feature on makes the IMS+ data extractor available, which allows additional IMS data to be collected.

Note: We might want to restrict its use in production environments, using security rules to restrict the IMS+ feature to specific users.

8.2 APA security settings

In this topic we describe how to define internal (if Secutiry=Internal has been specified in the CONFIG BASIC statement) or to define external (if Security=External has been specified in the CONFIG BASIC statement) security rules.

8.2.1 Internal security rules

These rules are generated into the load module CAZCNFG1. We specify these in assembly macro statements, then edit and submit sample job CAZASMCF in *hlq*.SCAZSAMP to generate the CAZCNFG1 module. Source member CAZCNFG1 in *hlq*.SCAZSAMP contains model source code. Example 8-1, Figure 8-1, and Table 8-1 illustrate the syntax of the statements and the access rules.

Example 8-1 Internal security rules

```
*-------------------------------------------------------------------*
 CAZCNFG1          CAZCVSET START
                   CAZCVSET END
 01  CAZRULE *,allow,ViewRequestsOwnedBy,========
 20  CAZRULE *,allow,DeleteRequestsOwnedBy,========
 21  CAZRULE CHABERT,allow,DeleteRequestsOwnedBy,*
 30  CAZRULE *,allow,UpdateRequestsOwnedBy,========
 31  CAZRULE CHABERT,allow,UpdateRequestsOwnedBy,*
 40  CAZRULE *,allow,MeasureJOB,*
 50  CAZRULE *,allow,MeasureSTC,*
 60  CAZRULE *,allow,MeasureTSU,========
 70  CAZRULE *,allow,UseDB2Plus,*
 80  CAZRULE *,allow,UseIMSPlus,*
*-------------------------------------------------------------------*
         END
```

Figure 8-1 shows a partial syntax of these statements.

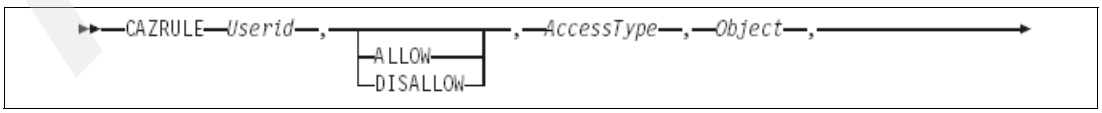


Figure 8-1 CAZRULE syntax

Here is the relevant information about the main parameters:

Userid User ID to which the rule applies. Either a full user ID or a user ID pattern.

- *** (asterisk)** To specify a wildcard value. An asterisk must be the last character in the value.
- **? (question mark)** To specify any character match. A question mark specifies that any character in the same position is to be matched.

ALLOW/DISALLOW Specifies whether access type is permitted (ALLOW) or denied (DISALLOW).

AccessType Specifies the type of access. Refer to Table 8-1.

Object Specifies the object to which the rule applies. Refer to Table 8-1. The types of objects are:

- User ID name or pattern
  - When we use a pattern, an equal sign (=) is permitted. Specifying = has the effect of substituting the character from the actual user ID (the user ID for which the rule is being checked) before the comparison is made.
- Job name or pattern
- Application Performance Analyzer ID or asterisk (*).

Table 8-1 contains the access rules description.

Table 8-1 Access rules description

Code this	For this type of access	Object type
ViewRequestsOwnedBy	Controls which measurement requests are visible to the user	User ID of user that created the request
DeleteRequestsOwnedBy	Controls which measurement requests the user is allowed to delete	User ID of user that created the request
UpdateRequestsOwnedBy	Controls which measurement requests the user is allowed to modify	User ID of user that created the request
MeasureJOB	Controls what job names the user is allowed to measure	Job name of the job to be measured
MeasureSTC	Controls what STCs the user is allowed to measure	Name of the started task procedure to be measured
MeasureTSU	Controls what TSUs (TSO regions) the user is allowed to measure	User ID of user creating the request
UseDB2Plus	Controls whether the user is allowed to use the DB2+ data extractor	User ID of user creating the request
UseIMSPlus	Controls whether the user is allowed to use the IMS+ data extractor	User ID of user creating the request

Precedence of rules

Whenever separate statements specify conflicting effects for the same user ID, the statement specified later takes precedence over the earlier one. For example:

```
CAZRULE *,allow,UpdateRequestsOwnedBy,========
CAZRULE CHABERT,allow,UpdateRequestsOwnedBy,*
```

The first statement specifies that all users are allowed to update their own requests. The second statement specifies that CHABERT is allowed to update all requests.

8.2.2 External security rules

When Security=External is coded in the CONFIG BASIC section in CAZCNFGO we must use RACF or a RACF-compatible external security product. This section describes how to do this. We have to define resource profiles using the SAF FACILITY class. The types of access are:

VIEW	To be able to view observations
UPDATE	To be able to update a scheduled observation
DELETE	To be able to delete an observation
MEASURE.JOB	For measuring a batch job
MEASURE.STC	For measuring a started task
MEASURE.TSU	For measuring TSO users
USE.DB2PLUS	To be able to request a DB2 PLUS measurement
USE.IMSPLUS	To be able to request an IMS PLUS measurement

By default, all users have access to VIEW, UPDATE, and DELETE any observation they create and monitor from their own TSO user ID. The equivalent RACF profiles for the rules supplied in Example 8-1 on page 137 are shown here in Example 8-2.

Example 8-2 Internal security rules and their equivalent RACF profiles

```
/* 01  CAZRULE *,allow,ViewRequestsOwnedBy,======== */
/* This is a default, no RACF profile required */
/* 20  CAZRULE *,allow,DeleteRequestsOwnedBy,======== */
/* This is a default, no RACF profile required */
/* 21  CAZRULE CHABERT,allow,DeleteRequestsOwnedBy,* */
RDEFINE FACILITY CAZ0.DELETE.* UACC(NONE)
PERMIT CAZ0.DELETE.* CLASS(FACILITY) ID(CHABERT) ACCESS(READ)
/* 30  CAZRULE *,allow,UpdateRequestsOwnedBy,======== */
/* This is a default, no RACF profile required */
/* 31  CAZRULE CHABERT,allow,UpdateRequestsOwnedBy,* */
RDEFINE FACILITY CAZ0.UPDATE.* UACC(NONE)
PERMIT CAZ0.UPDATE.* CLASS(FACILITY) ID(CHABERT) ACCESS(READ)
/* 40  CAZRULE *,allow,MeasureJOB,*
RDEFINE FACILITY CAZ0.MEASURE.JOB.* UACC(READ)
/* 50  CAZRULE *,allow,MeasureSTC,* */
RDEFINE FACILITY CAZ0.MEASURE.SCT.* UACC(READ)
/* 60  CAZRULE *,allow,MeasureTSU,======== */
/* This is a default, no RACF profile required */
/* 70  CAZRULE *,allow,UseDB2Plus,* */
RDEFINE FACILITY CAZ0.USE.DB2PLUS UACC(READ)
/* 80  CAZRULE *,allow,UseIMSPlus,* */
RDEFINE FACILITY CAZ0.USE.DB2PLUS UACC(READ)
```

In the foregoing example, CAZ0 is the STCid of the Application Performance Analyzer task to which these profiles applied.

8.3 APA Descriptive Program Attribution table

Application Performance Analyzer comes with the Descriptive Program Attribution table, hereafter referred to as DPA, containing descriptions of known IBM modules.

A sample DPT table is supplied in the *hlq*.SCAZSAMP library as member CAZUSR1 and the JCL to compile and link this member is provided in the same library as member CAZASMDP.

For a complete explanation, refer to Chapter 2, “Customizing Application Performance Analyzer,” in *IBM Application Performance Analyzer for z/OS: Customization Guide*, SC18-9599.

We have updated the default table in order to add our own description of load modules and CICS transactions. Example 8-3 shows the updated member CAZUSR1.

Example 8-3 DPA table

```
CAZUSR01 TITLE 'DPA Table for Customer Modules'
*-------------------------------------------------------------------*
*   System:   CAZ - IBM Application Performance Analyzer
*   Module:   CAZUSR01
*-------------------------------------------------------------------*

CAZUSR01 CAZDPAGN PROLOG

* GROUP is TRADER
TRADER CAZDPAGN GROUP,DESCRP='Trader Application'

* SUBGROUP is TRADER
TRADER CAZDPAGN SUBGROUP,GRP=TRADER,DESCRP='Trader Application'
*
*  Specific definitions
         CAZDPAGN NAME=TRADERB,SUBGRP=TRADER,TYPE=USER,               X
               DESCRP='Trader Batch VSAM'
         CAZDPAGN NAME=TRADERD,SUBGRP=TRADER,TYPE=USER,               X
               DESCRP='Trader Batch DB2'
         CAZDPAGN NAME=TRADERI,SUBGRP=TRADER,TYPE=USER,               X
               DESCRP='Trader Batch IMS'
*
*  WildCard definition
         CAZDPAGN NAME=MYTRAD*,SUBGRP=TRADER,TYPE=USER,               X
               DESCRP='Trader CICS'
*
*  CICS Transactions
         CAZDPAGN NAME=TDB2,SUBGRP=TRADER,TYPE=CICSTXN,               X
               DESCRP='Trader Application'
*
         CAZDPAGN NAME=TDB3,SUBGRP=TRADER,TYPE=CICSTXN,               X
               DESCRP='Trader Application'
*
         CAZDPAGN END
*
         END   CAZUSR01
```

Application Performance Analyzer is now able to identify any program or CICS transaction belonging to the TRADER application. The information displayed by Application Performance Analyzer is determined as follows:

- If the program name is TRADERB: Trader Batch VSAM
- If the program name is TRADERD: Trader Batch DB2
- If the program name is TRADERI: Trader Batch IMS
- If the program name is MYTRAD*: Trader CICS
- If the transaction name is TDB2 or TDB3: Trader Application

A sample is shown in Figure 8-2.

```
Name      Description         Percent of CPU Time * 10.00%  ±0.6%
                                  *....1....2....3....4....5....6....7....8....9
TRADERB   Trader Batch VSAM   0.10
```

Figure 8-2 Customized DPA: illustration

8.4 APA and DB2 Universal Database

The following tasks are only required if we want Application Performance Analyzer to gather the necessary data to report plan and package BIND timestamps or if we want to use the Dynamic DB2 EXPLAIN facility.

8.4.1 Binding a DB2 package and plan

This is how to bind the DB2 UDB package and plan:

1. Edit and customize member CAZBIND in our *hlq*.SCAZSAMP library.
2. The default package name used CAZPACK1. We can change this name to match our site requirements.
3. The default plan name used is CAZPLAN1. We can change this name to match our site requirements.

We repeat these BIND operations if we are binding this plan to multiple DB2 subsystems.

8.4.2 Enabling DB2 EXPLAIN: creating a PLAN_TABLE

To use dynamic EXPLAIN with the Application Performance Analyzer, the APA started tasks must have access to a PLAN_TABLE. You can create it as follows:

1. Edit and submit job CAZPLANT in *hlq*.SCAZSAMP to create a PLAN_TABLE.
2. The qualifier for this PLAN_TABLE defaults to the authorization ID of the started task.

8.4.3 Providing DB2 authorization

The authorization ID of the started task must have SYSADM authority if we want to use the DB2 EXPLAIN feature.

Note: Use the following SQL statement to grant SYSADM privilege:

```
GRANT SYSADM TO authid
```

Otherwise, grant EXECUTE authority on the plan CAZPLAN1 to the authorization ID of the APA started task. We can use the following SQL statement to grant this privilege:

```
GRANT EXECUTE ON PLAN CAZPLAN1 TO authid
```

8.4.4 Updating the Application Performance Analyzer started task

Add a DSN for the DB2 load library to the STEPLIB in the Application Performance Analyzer started task JCL. This load library must contain the following load modules: DSNALI, DSNTIAR, and alias DSNHLI2.

8.5 Application developer's preferences

Next we cover the application developer's preferences.

8.5.1 PREFerence command line

By entering the PREF command, we can set our preferences for general display settings. Put a slash (/) beside an option to select it. The available options are:

- Have the action bar visible on panels.
- Use 3270 graphic characters.
- Show long descriptions on multiple lines.
- Suppress use of special +/- character.

8.5.2 SETUP command line

Use the SETUP command to specify our own options for specific reports. The options available using the SETUP command vary by report. For each of them, we can get more details from the report descriptions.

8.6 Application developer's environment

When we specify source program mapping files, many reports allow us to enter a P line command to view the program source associated with that entry in the report. The P line command is available on many object types: CSECTs, DB2 SQL statements, CICS commands, and so on. The data is displayed in the Source Program Attribution report.

The source program mapping is only available if report A01 has been filled.

8.6.1 Overview of the Source Program Mapping feature

Panel A01 allows us to specify and manage associations between Source Program Mapping (SPM) files and observation sessions. To map measured addresses to their corresponding source program statements, we must identify SPM files for each of the observation sessions that use this feature.

An SPM file can be sequential or a member in a partitioned data set. It can be one of the following file types:

- A listing produced by the compiler (COBOL)
- An ADATA (Associated Data) file produced by High Level Assembler
- A side file member produced by the IDILANGX utility (COBOL and PL/I)
- A SYSDEBUG file produced by the compiler (COBOL)

The A01 panel consists of two sections:

- An input area in which we can specify an SPM file name and type.
- A report area in which existing SPM file associations are listed.

A sample Source Program Mapping panel is shown in Figure 8-3.

```
  File  View  Navigate  Help
 ──────────────────────────────────────────────────────────────────────────
A01 - Source Program Mapping (4807)                      Row 00001 of 00004
Command ===> _____________________________________________ Scroll ===> CSR

Enter the following information to specify a source mapping file to be
used in the analysis of this measurement information.

 File type . . . .   _       (L=listing, A=ADATA, S=LANGX SideFile, D=SYSDEBUG)
 Data set name . .   ____________________________________________
 Member name . . .   ________

Seqn  ID-ReqNum  Load   Type/Status  Lang   Member     DSN
0001  CAZ0-4807  Auto    L-Loaded    COB    MYTRADS    CHABERT.BOOK2005.COBLIST
0002  CAZ0-4807  Auto    L-Loaded    COB    MYTRADMV   CHABERT.BOOK2005.COBLIST
0003  CAZ0-4790  Auto    L-Inact     COB    TRADERB    CHABERT.BOOK2005.COBLIST
0004  CAZ0-4800  Auto    D-Inact     COB    TRADERI    CHABERT.BOOK2005.SYSDEBU
```

Figure 8-3 Source program mapping panel

8.6.2 File specification input area

File type Specify L for a compiler listing file, A for an assembler ADATA file, S for a LANGX SideFile, or D for a SYSDEBUG file.

> **Note:** For COBOL, the listing files must be created with the options SOURCE, NONUMBER, combined with either LIST or OFFSET.
>
> For OS/VS COBOL, use LIST,NOPMAP or CLIST,NOPMAP or CLIST,PMAP.

Repository T for a third party repository, or O for any other type such as a PDS or sequential data set.

Data set name Specify the name of the sequential or partitioned data set containing the SPM file.

Member name Include the member name if the data set is partitioned.

8.6.3 List of existing SPM file associations

Any entries for file associations applicable to the current observation session appear at the top of the list. The value under the ID-ReqNum field is displayed in red to indicate this.

The fields displayed in the File Association List are described here:

Seqn Line number of the entry. Line commands can be entered to this field.

ID-ReqNum The observation session request number with which the SPM file is associated.

Load Either Auto or NO is displayed here to indicate whether the SPM file is to be loaded automatically when a reporting session is started.

> **Note:** The A line command toggles AutoLoad on and off.

Type/Status Type of SPM file and whether the mapping information has been loaded (available for use). L for compiler listing file, A for ADATA file, S for sidefile, and D for sysdebug file.

Lang The source program language is shown here (ASM, COB, or PLI).

Member Member within a partitioned data set is shown here.

DSN Data set name of the SPM file is shown here.

8.6.4 MAP ALL primary command

Once we have selected an observation request, we can enter a MAP ALL primary command from any panel. It searches our entire list of existing SPM entries, and when a match is found for any module, it loads the source mapping data.

9

Application Performance Analyzer: Creating a monitoring request

In this chapter we explain how to create an observation request intended to select a job that has to be monitored.

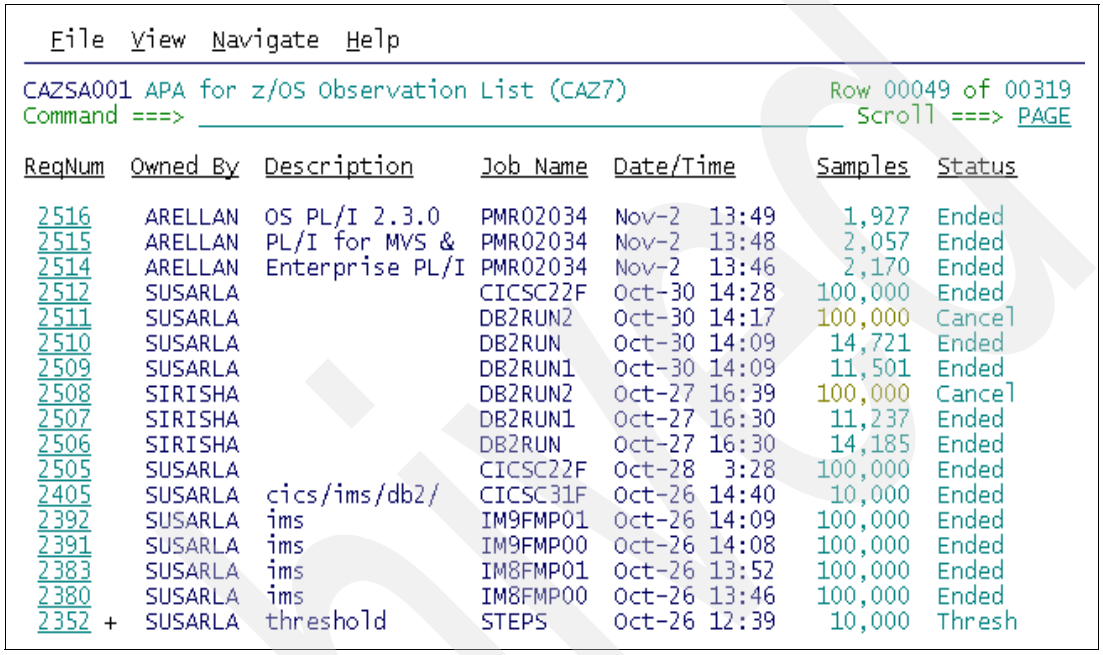
Jobs selected can be active jobs or they can be jobs that will execute in the future. This chapter first introduces the Observation Session List (the first panel displayed when we start Application Performance Analyzer) and then describes how to select active jobs and how to create scheduled observations for jobs that will execute later.

For both active jobs and scheduled observations, Application Performance Analyzer stores the request in the checkpoint data set, and the data that it captures during the observation in a sample file. While active jobs can be analyzed in real time, this data set is used to analyze scheduled observations. It also can be used or reused in the future to review active job or scheduled job information.

9.1 Observation Session List panel (R02)

This panel displays a scrollable list of all the observation session requests, whether they are complete, active, or pending. The list can be filtered by the owner ID or job name. The SETUP command is used to specify how the list is to be filtered.

A sample Observation List panel is shown in Figure 9-1.



File View Navigate Help

CAZSA001 APA for z/OS Observation List (CAZ7) Row 00049 of 00319
Command ===> Scroll ===> PAGE

ReqNum		Owned By	Description	Job Name	Date/Time	Samples	Status
2516		ARELLAN	OS PL/I 2.3.0	PMR02034	Nov-2 13:49	1,927	Ended
2515		ARELLAN	PL/I for MVS &	PMR02034	Nov-2 13:48	2,057	Ended
2514		ARELLAN	Enterprise PL/I	PMR02034	Nov-2 13:46	2,170	Ended
2512		SUSARLA		CICSC22F	Oct-30 14:28	100,000	Ended
2511		SUSARLA		DB2RUN2	Oct-30 14:17	100,000	Cancel
2510		SUSARLA		DB2RUN	Oct-30 14:09	14,721	Ended
2509		SUSARLA		DB2RUN1	Oct-30 14:09	11,501	Ended
2508		SIRISHA		DB2RUN2	Oct-27 16:39	100,000	Cancel
2507		SIRISHA		DB2RUN1	Oct-27 16:30	11,237	Ended
2506		SIRISHA		DB2RUN	Oct-27 16:30	14,185	Ended
2505		SUSARLA		CICSC22F	Oct-28 3:28	100,000	Ended
2405		SUSARLA	cics/ims/db2/	CICSC31F	Oct-26 14:40	10,000	Ended
2392		SUSARLA	ims	IM9FMP01	Oct-26 14:09	100,000	Ended
2391		SUSARLA	ims	IM9FMP00	Oct-26 14:08	100,000	Ended
2383		SUSARLA	ims	IM8FMP01	Oct-26 13:52	100,000	Ended
2380		SUSARLA	ims	IM8FMP00	Oct-26 13:46	100,000	Ended
2352	+	SUSARLA	threshold	STEPS	Oct-26 12:39	10,000	Thresh

Figure 9-1 Observation session list

Here is the relevant information about the main columns:

Request Number A unique four-digit request number assigned to identify the observation session. It is an input field that accepts line commands.

> **Note:** Type the / line command directly on top of the request number to display the list of available line commands.

Owned By The TSO ID of the creator/owner of the request.

Job Name The name of the job (or started task or TSO ID) that was measured.

Date/Time The date and time of the completion of the measurement. If the measurement is not yet complete, the date and time when the request was made is shown.

Samples If the session has a status of ended or active, this is the number of observation samples done. If the observation session has not yet started (a status of Sched or Future) then this shows the number of observation samples requested.

Status Status of the observation session:

- **Sched** The session has been scheduled but measurement has not yet been started.
- **Active** The session is currently taking place.

Ended	The session has completed.
Ended+	The session has completed, but this request repeats if the target job runs again.
Steps	Multi-step request.
Repeat	Repeating schedule request.

> **Note:** The requests under STEPS and REPEAT can be displayed by using the + line command to expand to the next level.

Cancel	The request was cancelled using the **CAN** line command.
Stoppd	The request was stopped.
Failed	The request failed to complete normally, and did not create a valid sample file.

If you want to view the Reports for one of the observation items, simply enter the **R** line command.

You can initiate a new observation request by using the **NEW** primary command or the **NEW** line command (to display an empty Schedule New Measurement panel).

When using the **NEW** primary command on the command line, the Schedule New Measurement panel with blank input fields is displayed.

A sample Schedule New Measurement panel is shown in Figure 9-2.

```
 File  View  Navigate  Help
──────────────────────────────────────────────────────────────────────────────
CAZSA001 dule New Measurement                                 Row 00001 of 00012
Command ===> ____________________________________________________ Scroll ===> PAGE

  1. Job Information   3. Multi Steps     5. CICS Options   7. Schedule
  2. Options           4. Active Jobs     6. Sysplex        8. Sched Options

──────────────────────────────────────────────────────────────────────────────
Panel 1. Job Information

  Job Name/Pattern . . _________   System Name . . . *________

  Step Specification
    Step No. . . . . . ____        Specify step number, program name,
    Program Name . . . _________   step name or step name + Proc step
    Step Name  . . . . _________   name. Use panel 3 to specify more
    ProcStepName . . . _________   than one step.

  Description  . . . . ____________________________________________
  Number of Samples  . ________     Measure to step end . . . N
  Duration (min:sec) . ________     Delay by (secs) . . . . . _______
  Notify TSO User  . . MLDUCKW      Retain file for (days)  . ____
```

Figure 9-2 Schedule New Measurement (report R03)

There are two distinct areas on this screen. The first, at the top of the screen, shows us a list of measurement options. One of the options is always highlighted as the selected one. The area at the lower half of the panel lists the available input fields that belong to the selected option.

We can change the measurement options, and hence change the list of available input fields, by selecting an option by either entering its number code on the command line or moving the cursor onto its description and pressing Enter.

The next topics describe the available settings.

9.2 Job Information panel (R03 - Panel 1)

Begin the R03 Schedule New Measurement dialog by selecting ***1 Job Information***, as shown in Figure 9-3.

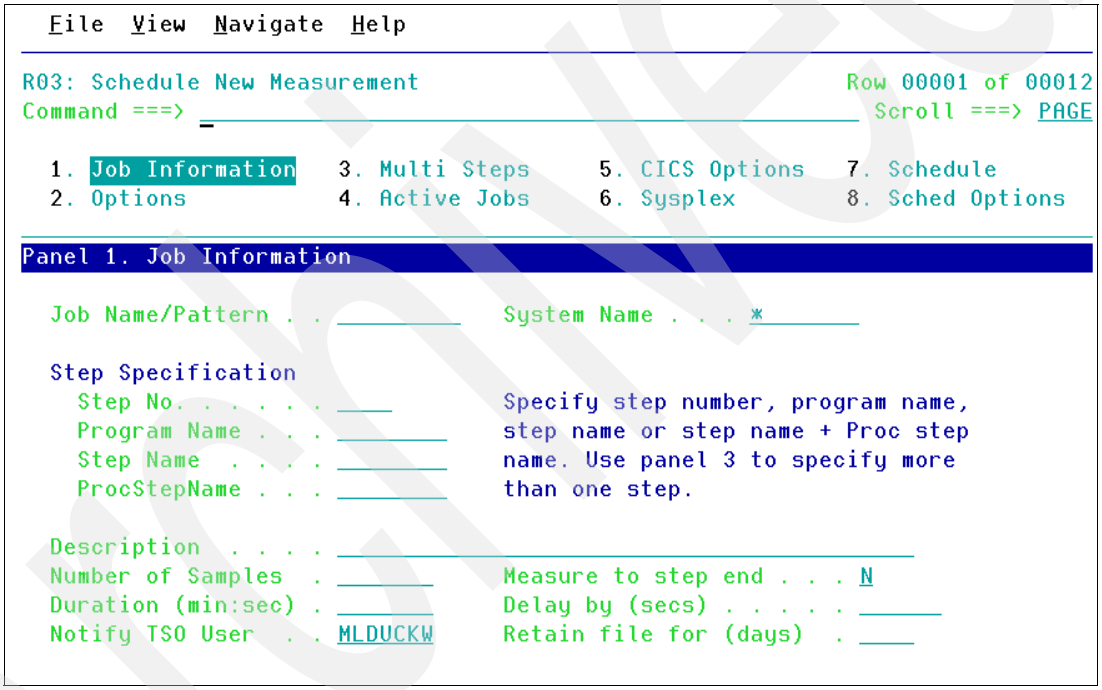


Figure 9-3 R03: Job Information

Here is the relevant information about the main input fields:

Job Name/Pattern This is a mandatory field. Specify the name of the job, started task, or TSO region to be measured. A partial name terminated by an asterisk (*) to indicate a wildcard pattern is possible, and a list of active jobs whose names match the wildcard pattern are displayed in the Active Jobs panel.

System name Use only if the APA is configured as a member in a SYSPLEX group. Specify the name of the system on which the measured job is to run. Specify an asterisk (*) in this field to indicate that the job could run on any of the systems in the group.

Step Specification To identify the step, we can specify one of the following items: Step number only, Program name only, Step name only, or Step name and Proc name.

> **Note:** If we leave all of these fields blank, the first job step is assumed. If we want to measure all the steps, specify an asterisk (*****) as Step Number.

Number of Samples Number of times execution of the measured jobstep is to be sampled.

> **Note:** Samples are taken in equal intervals. The sampling frequency is determined by dividing the number of samples by the specified measurement duration. If the number of samples is too large for the duration, a message is shown indicating you should increase duration or reduce the number of samples.

Measure to step end Having a **Y** in this field indicates that the measurement is to continue to the end of the step even if the specified number of samples has been recorded.

> **Note:** Measurement continues at the sampling rate calculated based on the specified duration and number of samples.

Duration (min:sec) Duration of the measurement. To specify the duration in minutes and seconds, separate the minutes value from the seconds value using a colon. Without a colon, the number is assumed to be in seconds.

Delay by (secs) Delay time in seconds, before initiation of the measurement.

Notify TSO User TSO user ID to be notified upon completion of the measurement.

Retain file for (days) Specify the number of days after completion of the measurement for which the measurement file is to be retained.

> **Note:** If the Retain file for (days) is blank, an automatic deletion occurs based on the ExpiryDays value coming from CAZCFNG0. If ExpiryDays is blank, then no automatic deletion occurs.
>
> If Retain file for (days) is zero, no automatic deletion occurs.

9.3 Options panel (R03 - Panel 2)

This panel is used to enter extended measurement options (data extractors).

Enter a slash (/) beside each of the data extractors required for the measurement. The data extractors are used to measure additional information about CICS, DB2 (and DB2+), IMS (and IMS+), and MQSERIES.

Figure 9-4 shows the Options panel.

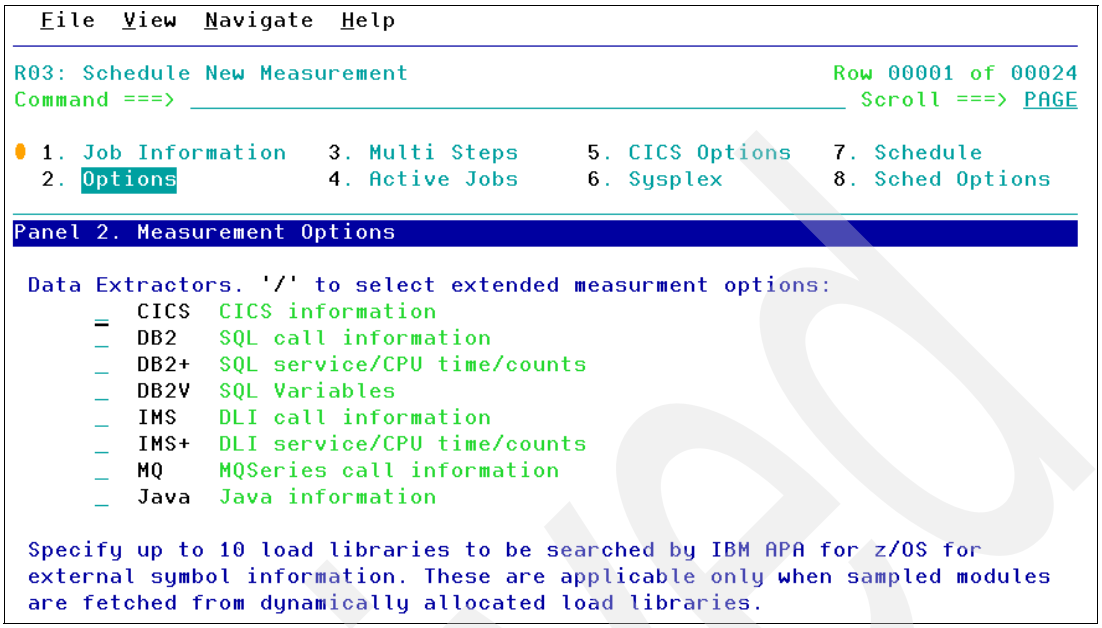


Figure 9-4 R03: Options (screen 1 of 2)

Note: DB2+ and IMS+ are used to collect additional data about DB2 and IMS, respectively, allowing exact call counts, service times, and CPU times to be measured and reported.

When you have selected the appropriate options, pressing Enter displays the second panel to allow you to specify up to 10 load libraries to be search (Figure 9-5).

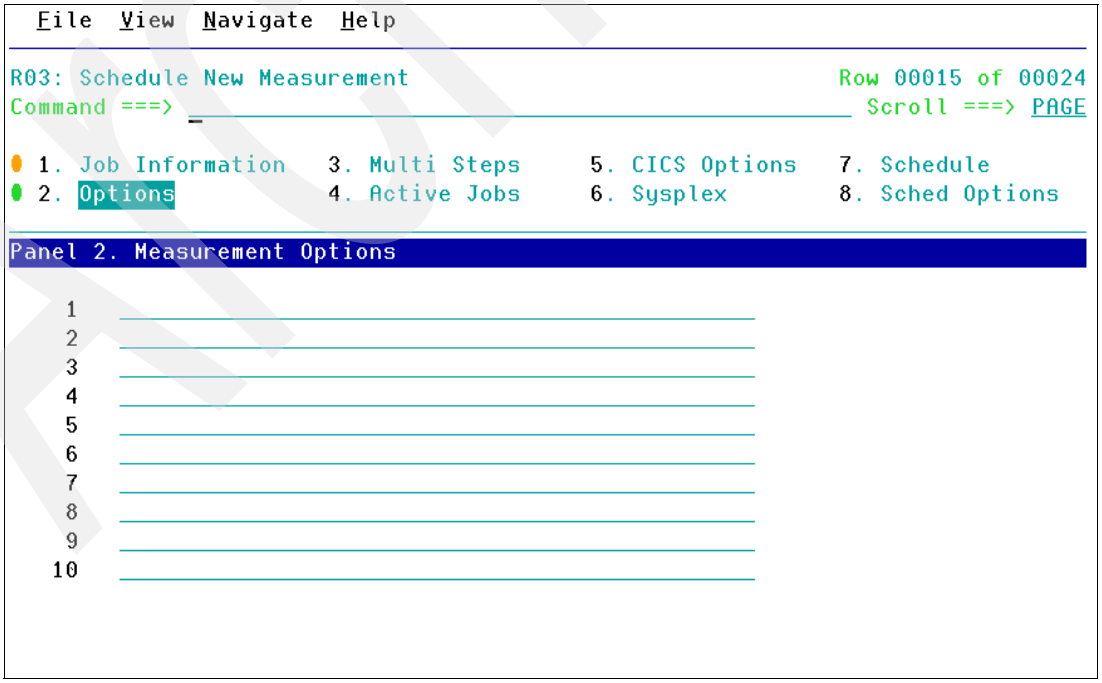


Figure 9-5 Options (screen 2 of 2)

Tip: When a panel has been filled and validated, a colored flag appears on the left of the field (just before the field number, as shown in Figure 10-4).

- Green color indicates that all the fields have been filled.
- Orange color indicates that all the mandatory fields have been filled, but some optional fields are blank (a default value is used).
- Red color indicates that at least one mandatory field has not been filled.

9.4 Multi-Steps panel (R03 - Panel 3)

Panel 3 is used to specify that multiple job steps are to be measured.

Figure 9-6 shows the panel.

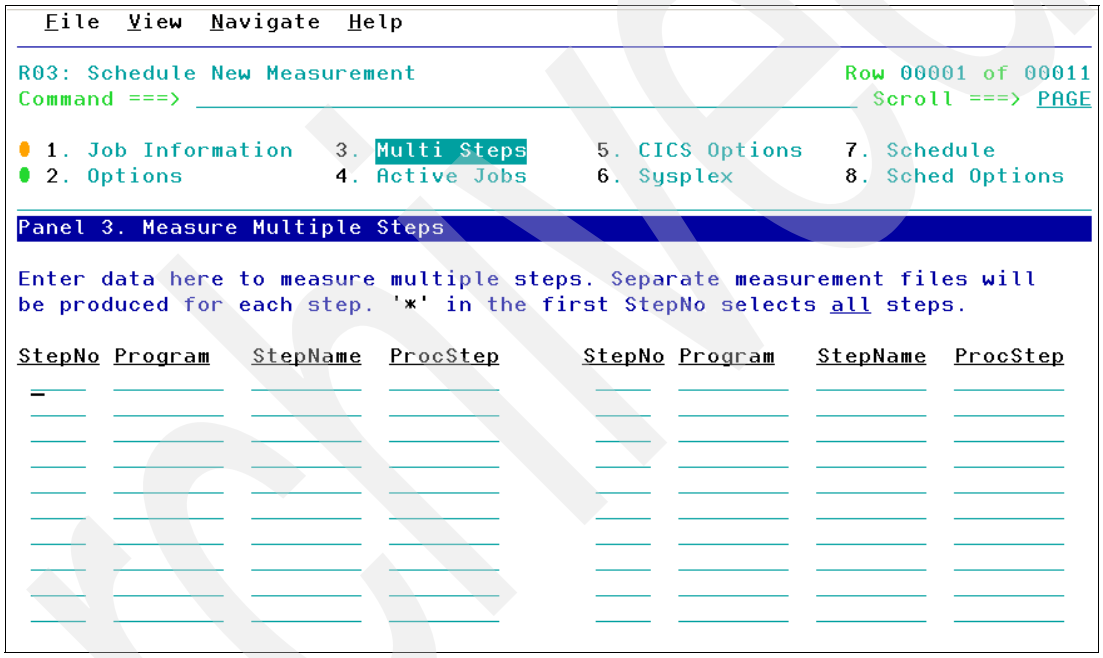


Figure 9-6 R03: Multiple steps

The relevant information about the main input fields is as follows:

Step No. This specifies the numeric step number.

Step Program This specifies the name of the program coded in the PGM = parameter of the EXEC statement for the step we want to measure.

Step Name This specifies the symbol coded in the name field of an EXEC PGM = statement or an EXEC PROC = statement.

ProcStepName This specifies the symbol coded in the name field of an EXEC PGM = statement that is part of a PROC. If ProcStepName is specified, then input must also be supplied in the Step Name field.

9.5 Active Jobs panel (R03 - Panel 4)

Figure 9-7 is used to select active jobs from a list. We can enter a prefix to limit the jobs listed. If we had entered a pattern on the Job Information panel, this would be appended as the prefix to that pattern.

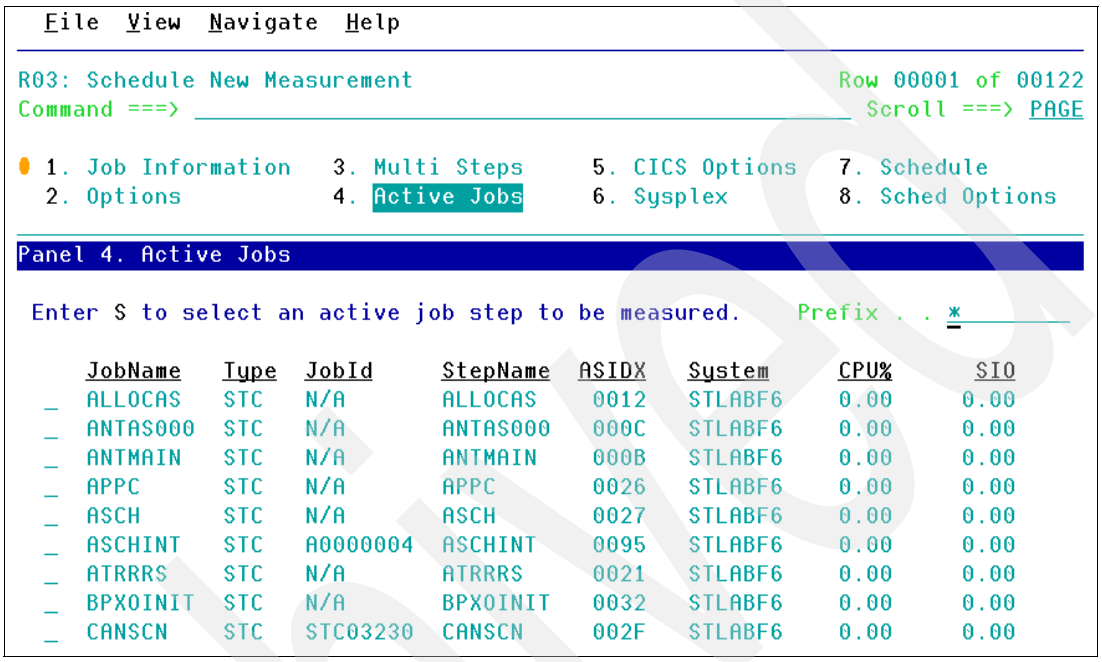


Figure 9-7 R03: Active Jobs

Enter an S beside the active job you want to measure.

9.6 CICS Options panel (R03 - Panel 5)

If the job we want to measure is a CICS address space, we can use this panel to specify a set of CICS transactions for which CICS measurement information is to be recorded.

Note: Limiting the CICS transactions can have a significant impact on the resources consumed by the measurement process.

A sample panel is shown in Figure 9-8.

```
 File  View  Navigate  Help
─────────────────────────────────────────────────────────────────────────────
R03: Schedule New Measurement                          Row 00001 of 00015
Command ===> _____________________________________________ Scroll ===> PAGE

  1. Job Information   3. Multi Steps     5. CICS Options   7. Schedule
  2. Options           4. Active Jobs     6. Sysplex        8. Sched Options
─────────────────────────────────────────────────────────────────────────────
Panel 5. CICS Transactions and Terminals

Specify up to 16 CICS trancodes for which measurement data is to be recorded.

   01 ____  02 ____  03 ____  04 ____  05 ____  06 ____  07 ____  08 ____
   09 ____  10 ____  11 ____  12 ____  13 ____  14 ____  15 ____  16 ____

Include CICS system transactions in measurement(Y/N): N

Wildcard character '*' can be specified at the end of a partial name.
'*' by itself specifies all transactions or terminals.

Specify up to 8 CICS terminal ids for which measurement data is to be recorded
```

Figure 9-8 R03: CICS Transaction and terminals

By specifying **N** for *Include CICS system transactions in measurement*, Application Performance Analyzer does not measure system transactions.

9.7 Sysplex panel (R03 - Panel 6)

This panel is used to select a target sysplex system from a list. We can also choose *all* systems (the same as entering an asterisk (*) in the System Name field on Panel 1) to measure the job on the first system to run it.

A sample panel is shown in Figure 9-9.

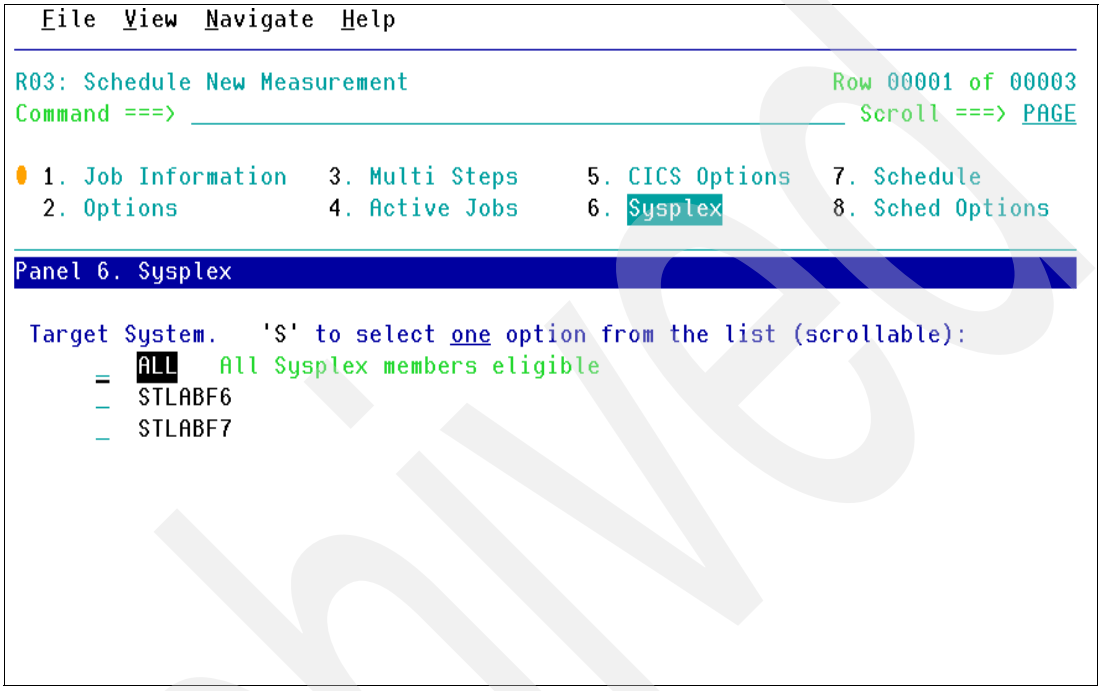


Figure 9-9 R03: Sysplex selection

Enter an S beside the system you want to select.

9.8 Schedule panel (R03 - Panel 7)

Use this panel to specify that the measurement is to be repeated at different dates and times. The panel is divided vertically into two areas. The upper portion of the screen is the date and time input area. The lower portion of the screen shows the measurement schedule.

A sample panel (before screen validation) is shown in Figure 9-10.

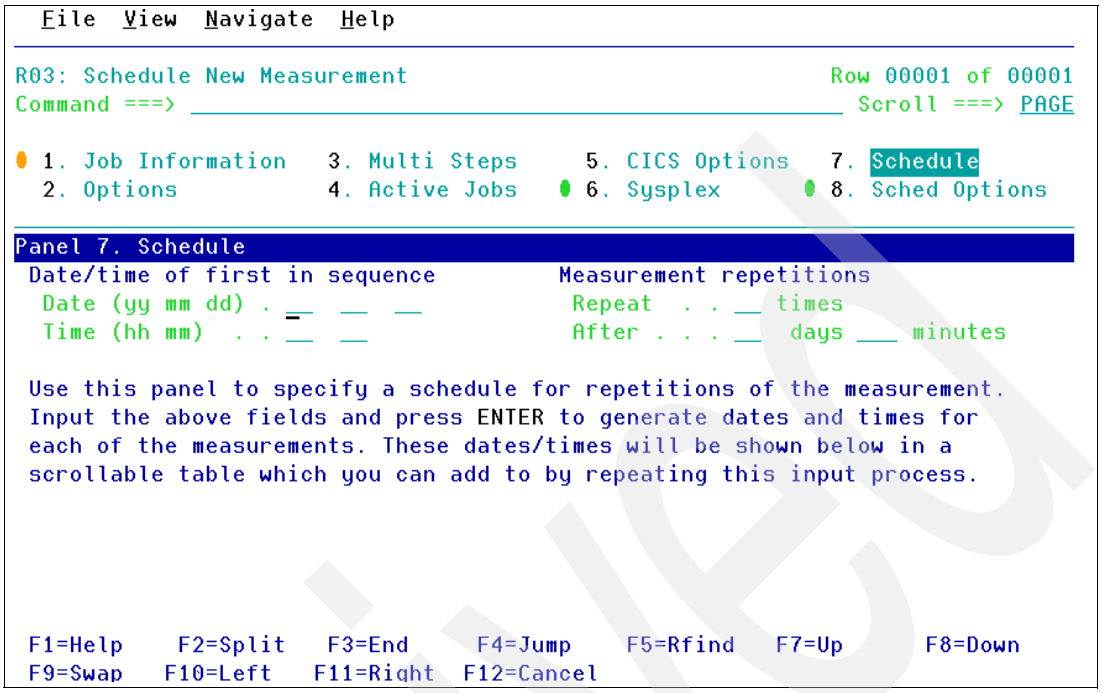


Figure 9-10 R03: Schedule before validation

Now we discuss the relevant information about the main input fields.

Date/time input area

The measurement schedule is shown in the upper portion of the panel:

- Date/time of first request:

  Date (yy mm dd) and time (hh mm) of the first measurement we would like added to the schedule.

- Measurement repetitions:

  Number of occurrences of the measurement and the interval between each occurrence.

A sample panel (after screen validation) is shown in Figure 9-11.

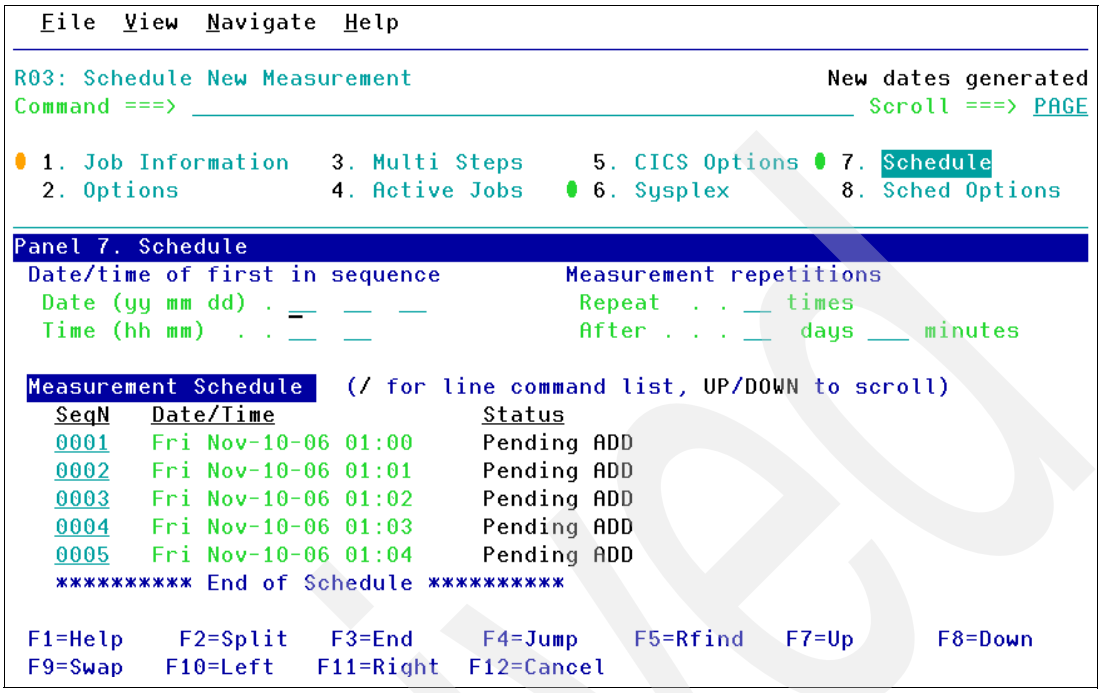


Figure 9-11 R03: Schedule after validation

Below is the relevant information about the main new fields.

Measurement schedule area

The measurement schedule is shown in the lower portion of the panel.

In addition to the schedule date and time, each line displays a status. The statuses are as follows:

- Pending ADD:

  Date/time entries have been added during this dialog. When we exit from the dialog, it is added to the schedule record.

The other status values apply when the dialog was entered using a **MOD**ify request:

- Measurement Scheduled:

  This applies when the dialog was entered using a **MOD**ify request. This indicates a schedule item already in the schedule record.

- Measurement Completed:

  The measurement has already been done. We are not allowed to make changes to this item.

- Measurement Expired:

  Schedule item already in the schedule record that has expired.

- Pending DELETE:

  Item in the schedule record for which we have issued a **D** (Delete) line command. When we exit from the dialog, it is deleted from the schedule record.

Line commands

Enter **D** to delete an entry from the schedule.

9.9 Schedule Options panel (R03 - Panel 8)

The available fields on this panel vary depending on whether **Y** or **N** is entered in the *Job will be active (Y/N)* field, and whether a future schedule has been entered on the Schedule panel as shown in Figure 9-11.

- No future schedule and active YES:

  No additional fields appear on panel 8.

- No future schedule and active NO:

  The fields Times to Repeat and Within interval (minutes) appear.

- Future schedule and active YES:

  The fields Number of times to retry and Retry interval (minutes) appear.

- Future schedule and active NO:

  The fields Expire after (minutes), Times to repeat measurement, and Within interval (minutes) appear.

A sample panel is shown in Figure 9-12.

```
 File  View  Navigate  Help
R03: Schedule New Measurement                          Row 00001 of 00011
Command ===> _____________________________________________ Scroll ===> PAGE

 1. Job Information   3. Multi Steps     5. CICS Options  7. Schedule
 2. Options           4. Active Jobs      6. Sysplex       8. Sched Options

Panel 8. Schedule Options

 Specify if the job to be measured will be active when the scheduling occurs
 (e.g. a CICS region) or pending (a batch job).

 Job will be active (Y/N) . . . . N

   Expire after (minutes) . . . . ____  Number of minutes from schedule time
                                        to wait for job to be submitted.

   Times to repeat measurement  . __    Repeat the measurement if the job
   Within interval (minutes)  . . ___   runs more than once within the
                                        specified interval.

 F1=Help    F2=Split   F3=End     F4=Jump    F5=Rfind   F7=Up      F8=Down
 F9=Swap    F10=Left   F11=Right  F12=Cancel
```

Figure 9-12 R03: Schedule Options

Here is the relevant information about the main input fields:

- Job active (N):

  N to indicate that the job is pending (a batch job) and Application Performance Analyzer is to wait for its execution.

  - Times to Repeat:

    Number of times the measurement is to be repeated if the job is cancelled or abends and then rerun during the specified interval.

  - Within Interval:

    Interval during which the Application Performance Analyzer started task is to check for reruns of the job.

- Job Active (Y):

  Y to indicate the job is active and the measurement is to begin immediately.

  - Times to Retry:

    Number of times Application Performance Analyzer is to check again for the job in the event that it was not active.

  - Retry Interval:

    Interval between each check for the job being active.

9.10 Creating a request using the batch interface commands

Application Performance Analyzer has a command language that allows us to submit requests using JCL. The command language can be used to create the same types of requests available in Application Performance Analyzer/ISPF.

9.10.1 Available commands and syntax

Here we review available commands and syntax.

NEW

The NEW command allows us to create a new monitoring request having the same characteristics as a definition done through the ISPF interface. Its syntax is shown in Figure 9-13.

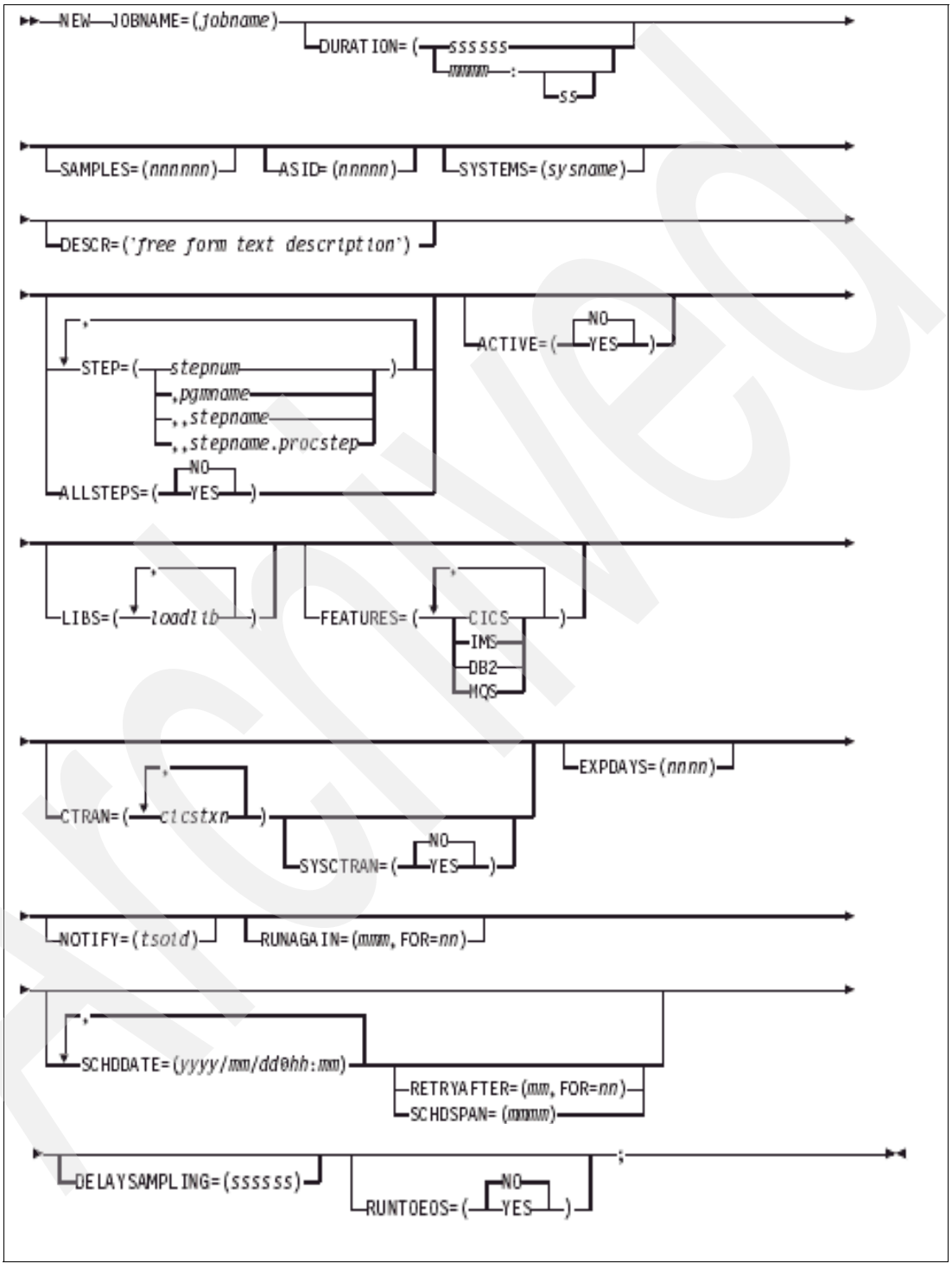


Figure 9-13 Batch NEW command syntax

DELETE

The DELETE command allows us to delete an observation identified by it number. Its syntax is as follows:

```
DELETE REQNUM=reqnum-number;
```

KEEP

The KEEP command allows us to keep the request until it is manually deleted. No expiration date applies. Its syntax is as follows:

```
KEEP REQNUM=reqnum-number;
```

9.10.2 Example

Example 9-1 illustrates how to set up the Batch Interface JCL and create a measurement request with the following main parameters:

- Jobname: APADB2.
- Duration: 1 minute.
- Sample: 10 000 per minute.
- Must run until it ends with the same sampling rate.
- Expiration time: 30 days.
- Steps: All.
- Data Extractors: DB2 and DB2+.
- Each Friday at 8 p.m. for 5 weeks starting on Friday November 11th.
- At 10 p.m. on Friday November 11th, Saturday November 12th. and Sunday November 13th.
- Request number 2306 has to be deleted.
- Request number 2362 has to be kept.

Example 9-1 NEW, DELETE, and KEEP batch commands

```
//CAZBATCH EXEC PGM=CAZBATCH,PARM='STCID=CAZ1'
//STEPLIB  DD   DISP=SHR,DSN=APAMON.V1R1.SCAZAUTH
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
NEW
   JOBNAME=(APADB2)
   DURATION=(1:00)
   SAMPLES=(10000)
   DESCR="APA Lab 3: Run APADB2"
   RUNTOEOS=(Y)
   NOTIFY=(ERIC)
   ACTIVE=(N)
   EXPDAYS=(30)
   ALLSTEPS=(YES)
   SYSTEMS=(ZT01)
   FEATURES=(DB2,DB2+)
   SCHDDATE=(2007/11/11@20:00)
   SCHDDATE=(2007/11/11@22:00)
   SCHDDATE=(2007/11/12@22:00)
   SCHDDATE=(2007/11/13@22:00)
   SCHDDATE=(2007/11/18@20:00)
   SCHDDATE=(2007/11/25@20:00)
   SCHDDATE=(2007/12/02@20:00)
```

```
   SCHDDATE=(2005/12/09à20:00)
  ;
DELETE REQNUM=2306
   ;

KEEP REQNUM=2362
   ;
 /*
```

9.10.3 How to create JCL that can be reused

Application Performance Analyzer provides an easy way to create a set of sample JCLs that can be used as skeletons. First we create a request using the Application Performance Analyzer. To illustrate our purpose, we monitor the TDB2 CICS transaction, which contains DB2 calls. Figure 9-14, Figure 9-15, and Figure 9-16 show the definition.

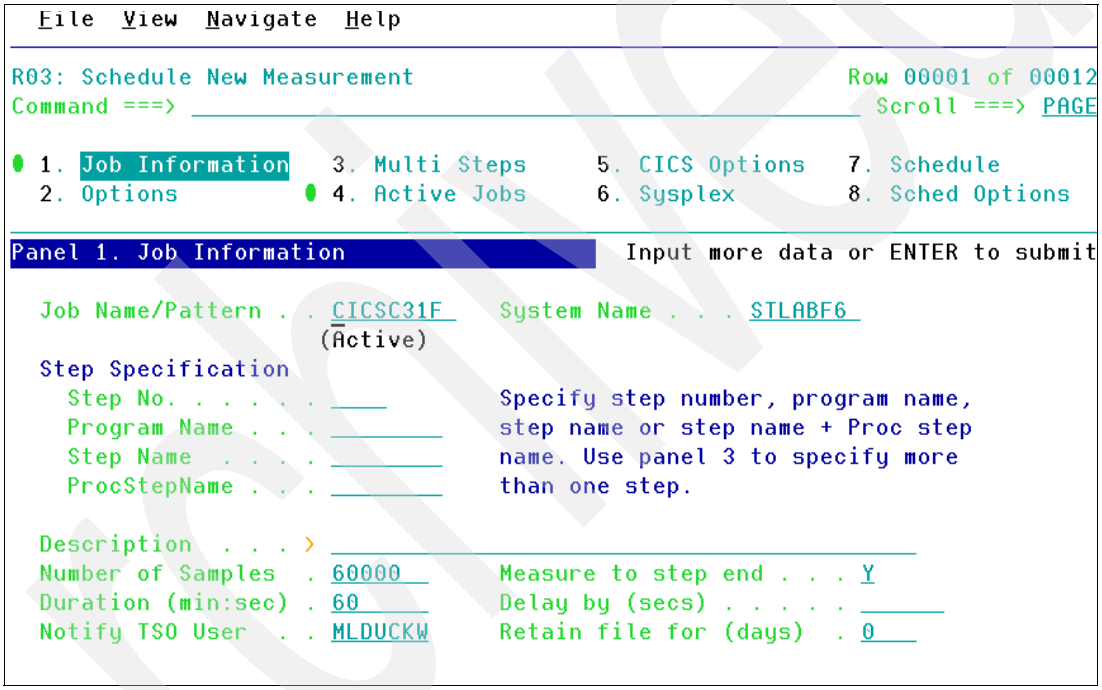


Figure 9-14 R03: Schedule New Measurement (Job Information)

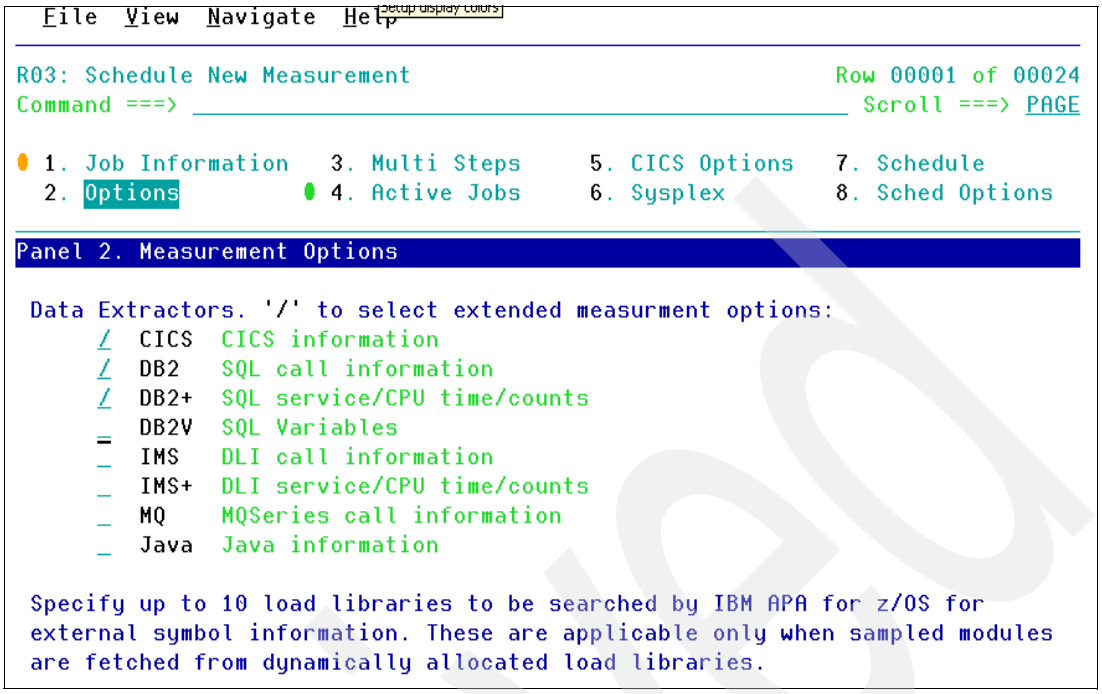


Figure 9-15 R03: Schedule New Measurement, Options panel

Select any transactions beginning with **MYT**.

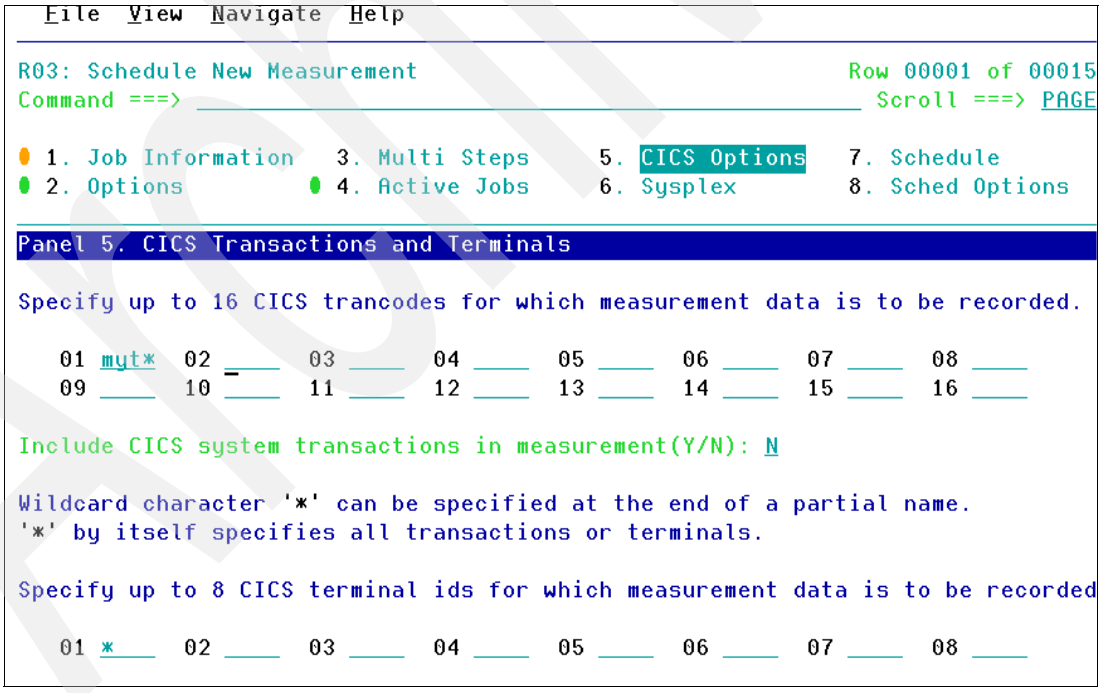


Figure 9-16 R03: Schedule New Measurement (CICS Transaction)

Because we have completed the request definition, we enter **JCL** on the command line to get the corresponding JCL, as shown in Figure 9-17.

```
 File  View  Navigate  Help
 ──────────────────────────────────────────────────────────────────────
 R03: Schedule New Measurement                          Row 00001 of 00015
 Command ===> jcl_____________________________________ Scroll ===> PAGE

 1. Job Information   3. Multi Steps    5. CICS Options   7. Schedule
 2. Options           4. Active Jobs     6. Sysplex       8. Sched Options

 Panel 5. CICS Transactions and Terminals

 Specify up to 16 CICS trancodes for which measurement data is to be recorded.

    01 MYT*  02 ____  03 ____  04 ____  05 ____  06 ____  07 ____  08 ____
    09 ____  10 ____  11 ____  12 ____  13 ____  14 ____  15 ____  16 ____

 Include CICS system transactions in measurement(Y/N): N

 Wildcard character '*' can be specified at the end of a partial name.
 '*' by itself specifies all transactions or terminals.

 Specify up to 8 CICS terminal ids for which measurement data is to be recorded

    01 *___  02 ____  03 ____  04 ____  05 ____  06 ____  07 ____  08 ____
```

Figure 9-17 R03: Command to generate batch JCL

Using the **SAVE** command, we are able to save it into our JCL library as shown in Figure 9-18 and Figure 9-19.

```
 File  View  Navigate  Help
 ──────────────────────────────────────────────────────────────────────
 R03: Schedule New Measurement                          Row 00001 of 00020
 Command ===> save____________________________________ Scroll ===> PAGE

 1. Job Information   3. Multi Steps    5. CICS Options   7. Schedule
 2. Options           4. Active Jobs     6. Sysplex       8. Sched Options

 JCL and Batch Command Syntax

  The following control statement can be used for CAZBATCH

  //CAZBATCH EXEC PGM=CAZBATCH,PARM='STCID=CAZ7'
  //STEPLIB  DD   DISP=SHR,DSN=ADTOOLS.APA710.SCAZAUTH
  //SYSPRINT DD   SYSOUT=*
  //SYSIN    DD   *
   NEW
    JOBNAME=(CICS*)
    DURATION=(60)
    SAMPLES=(60000)
    SYSCTRAN=(N)
    NONCTERM=(Y)
    RUNTOEOS=(Y)
    NOTIFY=(MLDUCKW)
```

*Figure 9-18 Entering **SAVE** from the JCL display panel*

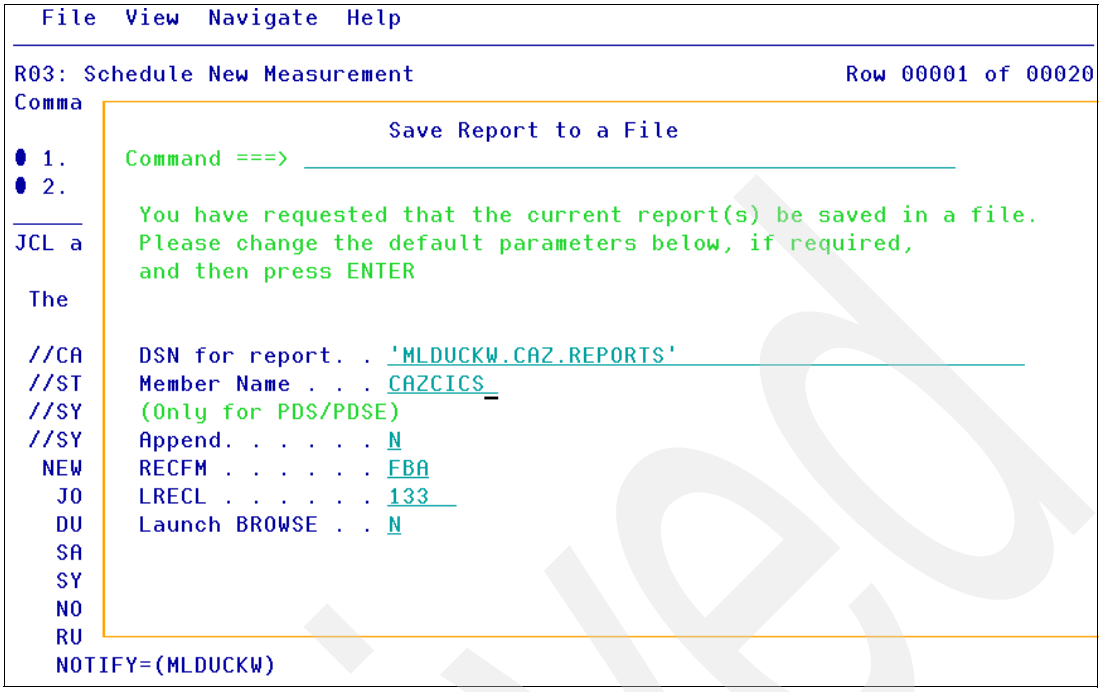


Figure 9-19 Save Report to a File

We can easily create several JCL members that can be used to create other requests.

We can also add a JCL request definition as the first step of a batch job we have to monitor. Using the Multi-Step panel described in 9.4, “Multi-Steps panel (R03 - Panel 3)” on page 151, you exclude it from the monitoring request, as shown in Example 9-2.

Example 9-2 Batch job with its own APA monitoring definition

```
//APABATCH JOB ,CLASS=A,NOTIFY=&SYSUID,MSGCLASS=H,MSGLEVEL=(1,1)
//CAZBATCH EXEC PGM=CAZBATCH,PARM='STCID=CAZ1'
//STEPLIB  DD   DISP=SHR,DSN=APAMON.V1R1.SCAZAUTH
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 NEW
  JOBNAME=(APABATCH)
  DURATION=(60)
  SAMPLES=(150000)
  DESCR="APA Lab 2: Run APABATCH"
  RUNTOEOS=(Y)
  STEP=(,,DELETE)
  STEP=(,,DEFINE)
  STEP=(,,LOADCUST)
  STEP=(,,LOADCOMP)
  STEP=(,,APABATCH)
  NOTIFY=(ERIC)
  ACTIVE=(N)
  EXPDAYS=(30)
  ALLSTEPS=(YES)
  SYSTEMS=(ZT01)
 ;
//*********************************************************************
//*  Used to start Trader as batch application                        *
//*********************************************************************
//*********************************************************************
```

```
//* Delete the VSAM datasets for Batch
//*****************************************************************
//DELETE   EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *
 DELETE ERIC.CAZ.BATCH.CUSTFILE
 DELETE ERIC.CAZ.BATCH.COMPFILE
 SET MAXCC=0
/*
//*****************************************************************
//* Define the VSAM datasets for Batch
//*****************************************************************
//DEFINE   EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *
 /*                              */
   DEFINE CLUSTER -
             (NAME(ERIC.CAZ.BATCH.CUSTFILE) -
              VOLUMES(CFGOOE) -
              INDEXED -
              TRACKS(1) -
              SHAREOPTIONS(2 3)) -
          DATA -
             (NAME(ERIC.CAZ.BATCH.CUSTFILE.DATA) -
              KEYS(81 0) -
              RECORDSIZE(136 136) -
              CONTROLINTERVALSIZE(4096)) -
          INDEX -
             (NAME(ERIC.CAZ.BATCH.CUSTFILE.INDEX))
 /*                              */
   DEFINE CLUSTER -
             (NAME(ERIC.CAZ.BATCH.COMPFILE) -
              VOLUMES(CFGOOE) -
              INDEXED -
              TRACKS(1) -
              SHAREOPTIONS(2 3)) -
          DATA -
             (NAME(ERIC.CAZ.BATCH.COMPFILE.DATA) -
              KEYS(20 0) -
              RECORDSIZE(90 90) -
              CONTROLINTERVALSIZE(4096)) -
          INDEX -
             (NAME(ERIC.CAZ.BATCH.COMPFILE.INDEX))
/*
//*****************************************************************
//* Load VSAM files for Batch
//*****************************************************************
//LOADCUST EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *
     REPRO                                    -
         INFILE(INPUT2)                       -
         OUTDATASET(ERIC.CAZ.BATCH.CUSTFILE)
/*
//INPUT2   DD  DISP=SHR,DSN=ADTCFG.TRADER.CUSTMAST
//*
//LOADCOMP EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *
     REPRO                                    -
```

```
         INFILE(INPUT2)                          -
         OUTDATASET(ERIC.CAZ.BATCH.COMPFILE)
/*
//INPUT2   DD  DISP=SHR,DSN=ADTCFG.TRADER.COMPMAST
//*
//APABATCH EXEC PGM=TRADERB,PARM=RPTOPTS(YES)
//STEPLIB  DD DISP=SHR,DSN=ADTCFG.PDTOOLS.LOAD
//SYSPRINT DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//COMPFILE DD DISP=SHR,DSN=ERIC.CAZ.BATCH.COMPFILE
//CUSTFILE DD DISP=SHR,DSN=ERIC.CAZ.BATCH.CUSTFILE
//TRANSACT DD DISP=SHR,DSN=ERIC.CAZ.BATCH.TRANFILE
//REPOUT   DD SYSOUT=*
//TRANREP  DD SYSOUT=*
//*
/*
```

9.11 How to select a report to analyze

The Observation Session List panel R02 is the first panel we have when we start Application Performance Analyzer. A sample Observation List panel is shown in Figure 9-20.

```
  File  View  Navigate  Help
R02: IBM APA for z/OS Observation List (CAZ7)              Row 00018 of 00329
Command ===>                                               Scroll ===> CSR

ReqNum  Owned By  Description     Job Name  Date/Time      Samples  Status

 5975    APEDU01  VSAM batch job  APEDU01B  Feb-8  14:17     1,000  Ended
 5906 +  APEDU01                  APEDU01B  Jan-30 18:58      5,000  STEP$
 5688    APEDU01                  APEDU01V  Jan-25 13:22      1,000  REPEAT
 5683    APEDU01                  APEDU01V  Jan-25  9:55      1,000  Ended
 5677 +  APEDU01                  APEDU01V  Jan-24 18:23     10,000  STEP$
 5666    APEDU01                  APEDU01B  Jan-21 18:15      1,000  Ended
 5660    LEAKE2                   LEAKE2    Jan-18 21:14          1  Ended
 5659    LEAKE2                   LEAKE2    Jan-18 21:13          1  Ended
 5655    APEDU01                  APEDU01B  Jan-16 18:53      1,000  Ended
 5654    APEDU01                  APEDU01B  Jan-16 18:45      1,000  Ended
 5650    APEDU01  Batch job       APEDU01X  Jan-15 21:53     60,000  Ended
 5640    LEAKE2   CRF TEST #2     LEAKE2    Jan-13 15:40          1  Ended
 5466    LEAKE    BBH SOURCE MAPP LEAKE     Jan-6  13:05          1  Ended
 5430    MOTO     IMS             MOTO      Jan-5  23:14          1  Ended
 5404    LEAKE                    LEAKE     Jan-4  10:27          1  Ended
 5401    LEAKE    CRF DB2 JOB     LEAKE     Dec-30 15:02          1  Ended
 5399    MOTO     R5840           MOTO      Dec-30  0:08          1  Ended
```

Figure 9-20 R02: Observation List (unexpanded)

Request number 5906 is followed by a plus sign (+) and it has STEPS at the end of its line. This means that several steps have been recorded for the same job.

Request number 5688 has REPEAT at the end of its line. This means that the same job has been recorded several times.

In order to analyze one of the entries, we use the **+** line command to expand to the next level, as shown in Figure 9-21.

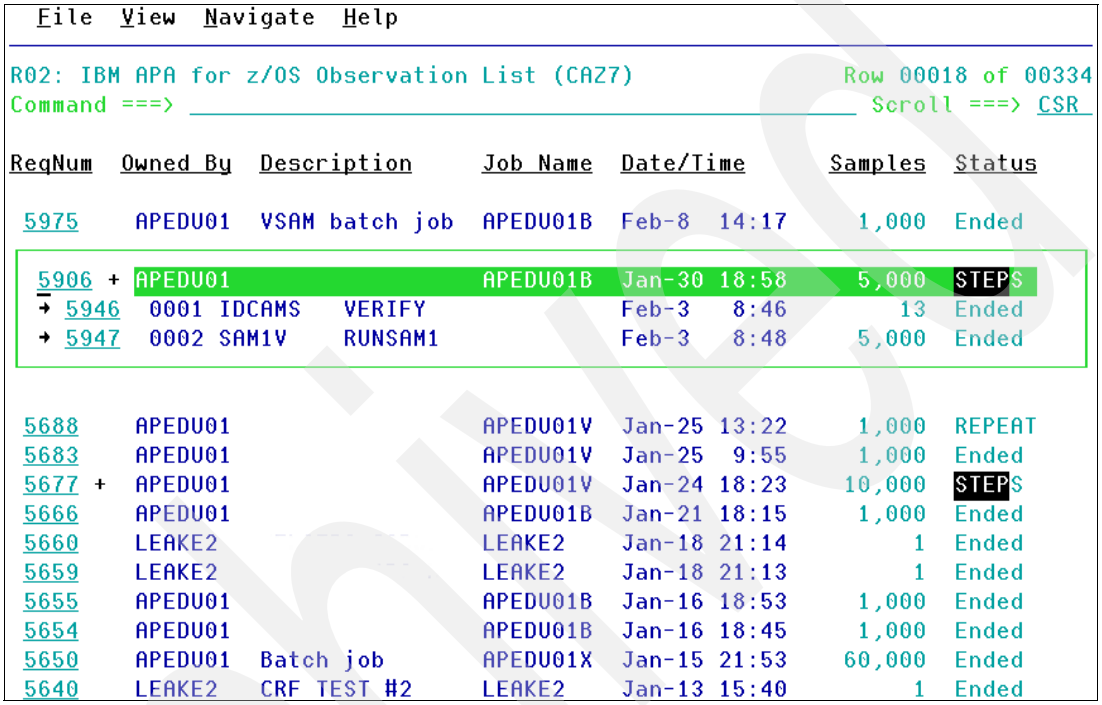


Figure 9-21 R02: Observation List (expanded)

9.12 How to compare several reports (Win command line)

The Observation Session List panel R02 displayed in Figure 9-22 shows that two request numbers (2542 and 2541) are for the same jobname, CICSC31F.

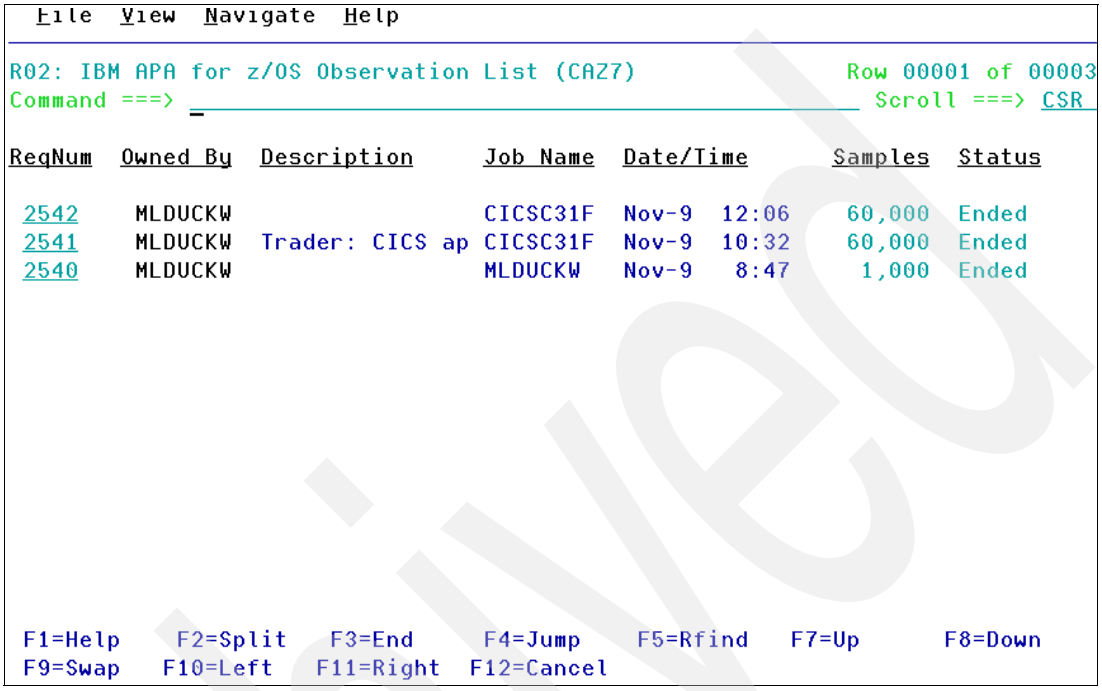


Figure 9-22 Observation list

It could be useful to compare the same reports coming from each one to check what the trends are.

We first select, request number 2542 (using the **S** or **R** line command), then with the **S05** command, we view the report, as shown in Figure 9-23.

Note: You can select any report(s) here, we are just using S05 as an example.

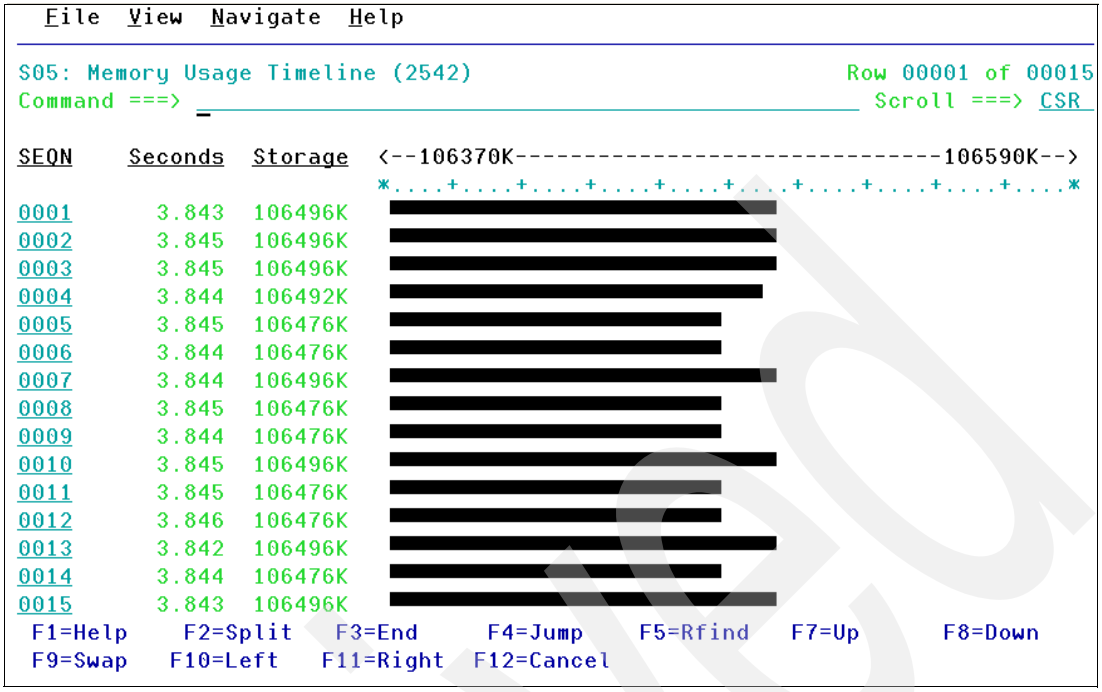


Figure 9-23 S05 Memory usage timeline report for job 2542

We now have to start another report analysis for the request number 2541 without closing the one we have for request number 4811. Type **WIN** on the command line as shown in Figure 9-24.

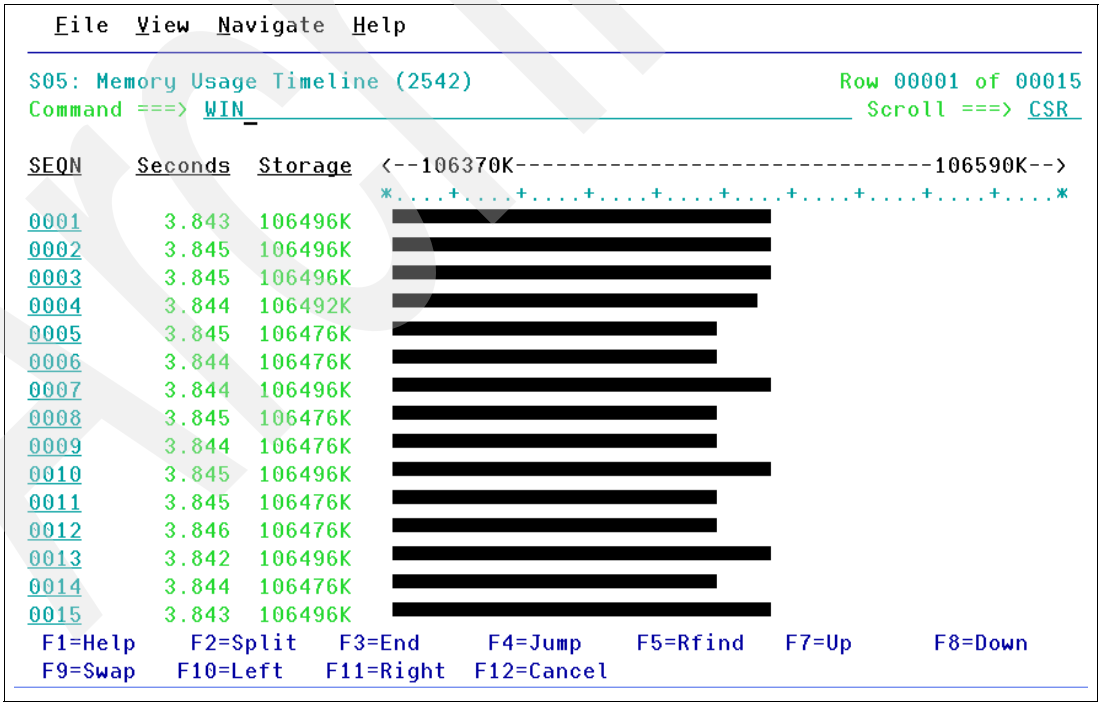


*Figure 9-24 Entering the **WIN** command*

After pressing Enter, we see the *Jump to* window were we can set up our Jump to point by typing an **S** next to the R02 report as shown in Figure 9-25.

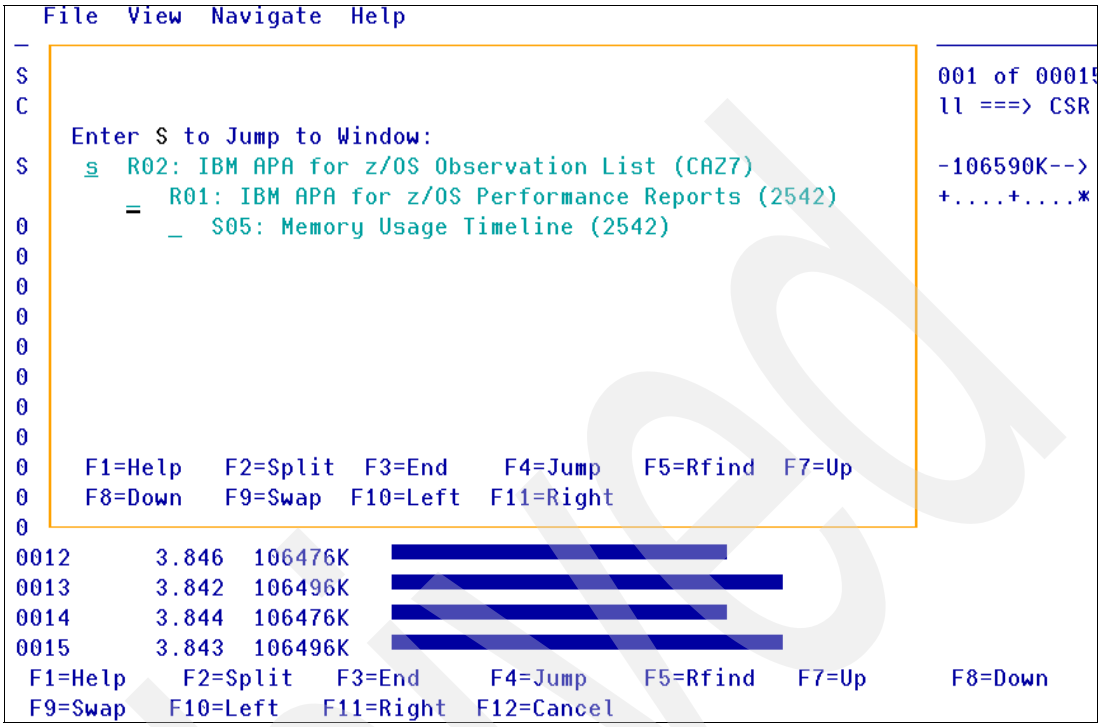


Figure 9-25 Jump to setup window

After you press Enter, Application Performance Analyzer displays a new R02 report from which we select, using the **R** or **S** line command, the request number 2541 as in Figure 9-26.

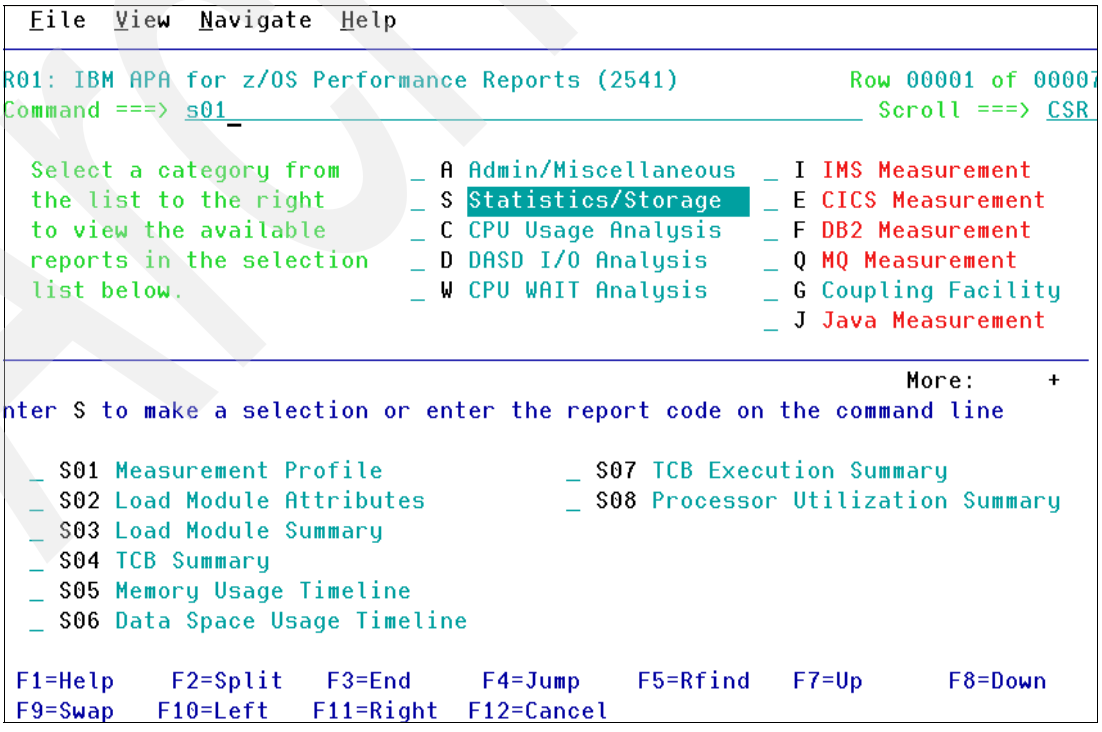


Figure 9-26 For the new job, selecting the appropriate report

When you press Enter, Application Performance Analyzer displays the panel in Figure 9-27.

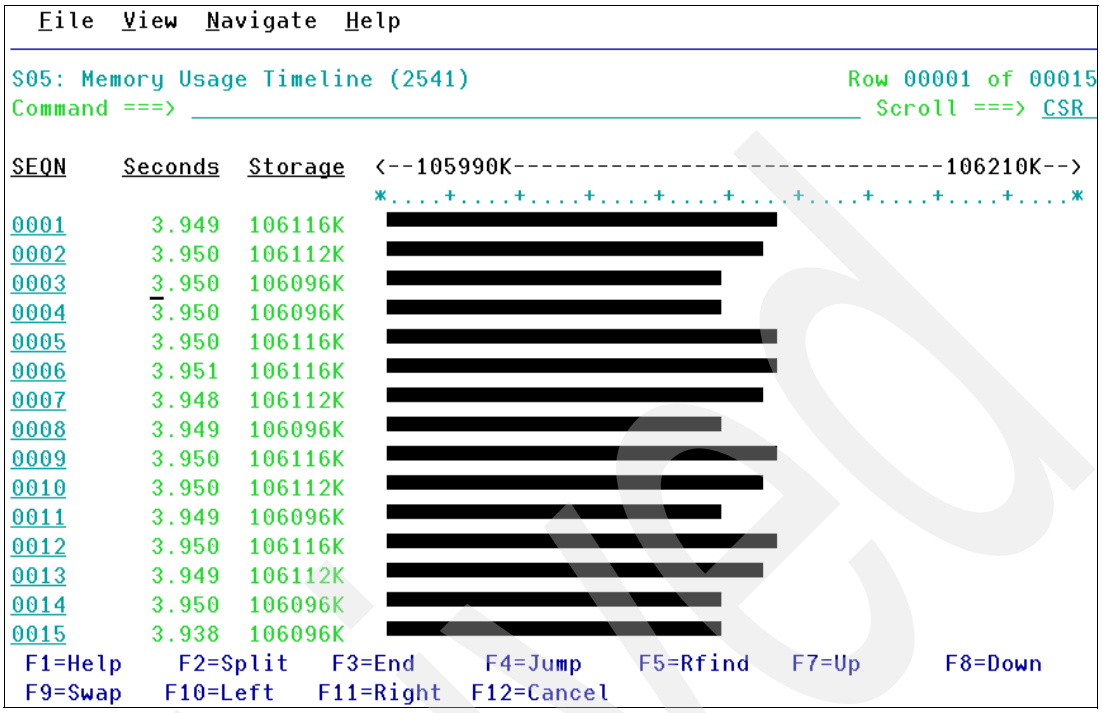


Figure 9-27 Memory Usage timeline report for job 5241

We can now either use the F4 (Jump) command or the Win command line to navigate between reports. This makes it easy to compare several measurement files made for the same job at different periods and to see if an evolution appears.

10

Application Performance Analyzer: Performance Analysis Reports

In the first part of this chapter we provide images of the Application Performance Analyzer request used to generate reports.

In the remainder of the chapter we describe reports provided by Application Performance Analyzer to analyze a batch job without any subsystem (CICS, DB2, IMS, or MQSeries).

The following Application Performance Analyzer report categories are explained:

- Statistics and Storage
- CPU Usage Analysis
- CPU WAIT Analysis

To illustrate these categories, we used the TRADER application in batch mode only using VSAM data sets. This application is described in Chapter 33, “Guided tour of the Trader application” on page 1107.

Attention: The purpose of this book is not to explain all the fields of each report, but to illustrate some of the reports and their uses.

For a full description, refer to *IBM Application Performance Analyzer for z/OS: User’s Guide*, SC19-1066, or use the online Help on Report Interpretation.

10.1 Observation session request definition

We have to monitor all steps of a batch job with the following characteristics:

- A sampling rate of 1000 samples per second is used for all its duration.
- The job has a single step.
- The measurement file has to be retained for 30 days.
- User **MLDUCKW** is to be notified at the end of the measurement.

When you enter the NEW command line on the Observation List (R02) panel, Application Performance Analyzer displays the Schedule New Measurement (R03) panel. The settings required to monitor the TRADERB job are shown in Figure 10-1.

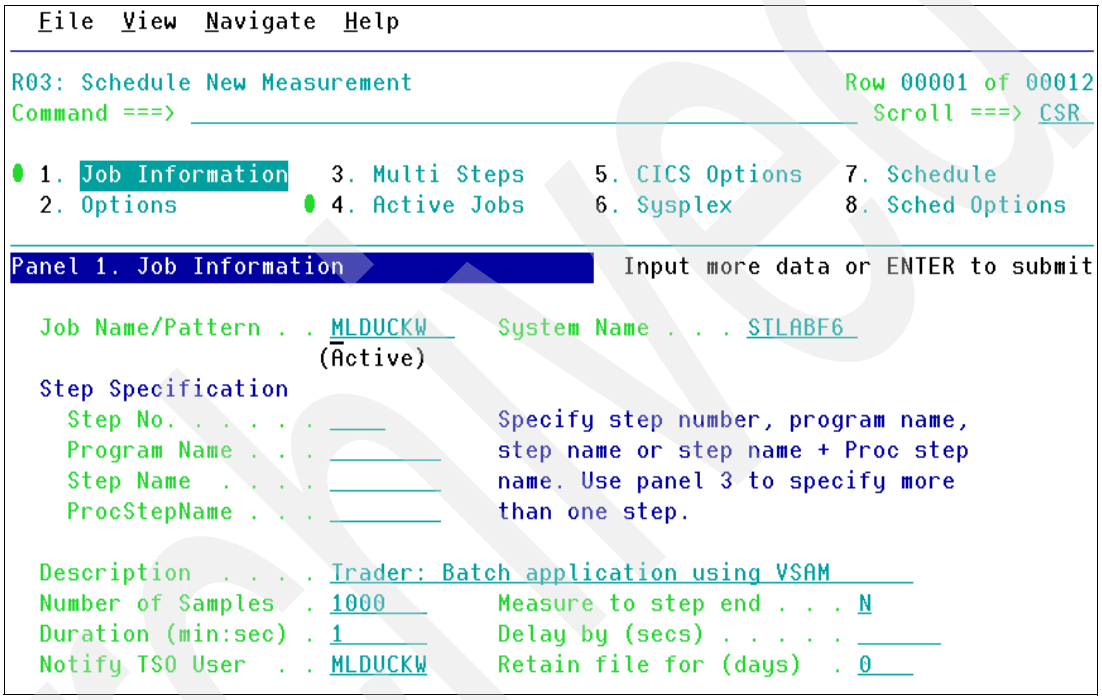


Figure 10-1 Schedule New Measurement (R03) for TRADERB

When you press Enter to submit the previous definition, Application Performance Analyzer displays the Observation List (R02) panel with our measurement definition at the top of the list, as shown in Figure 10-2.

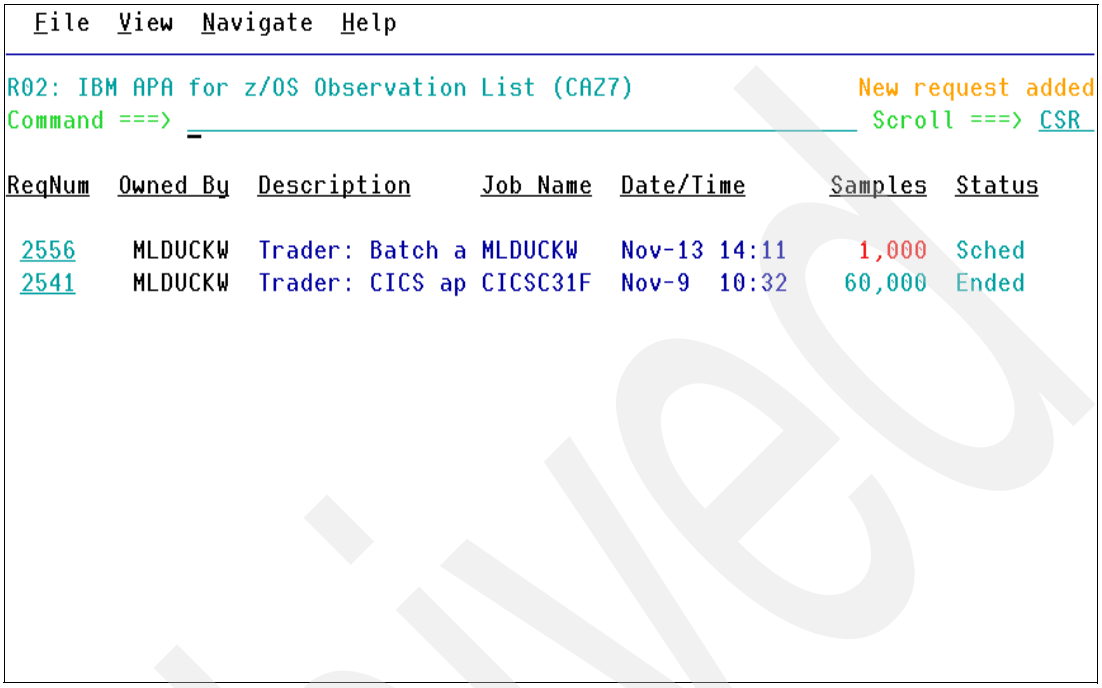


Figure 10-2 Observation List (R02)

We now start the batch job and check which status has the measurement request number 2556.

Figure 10-3 shows that the request's status is Ended when the job is ended.

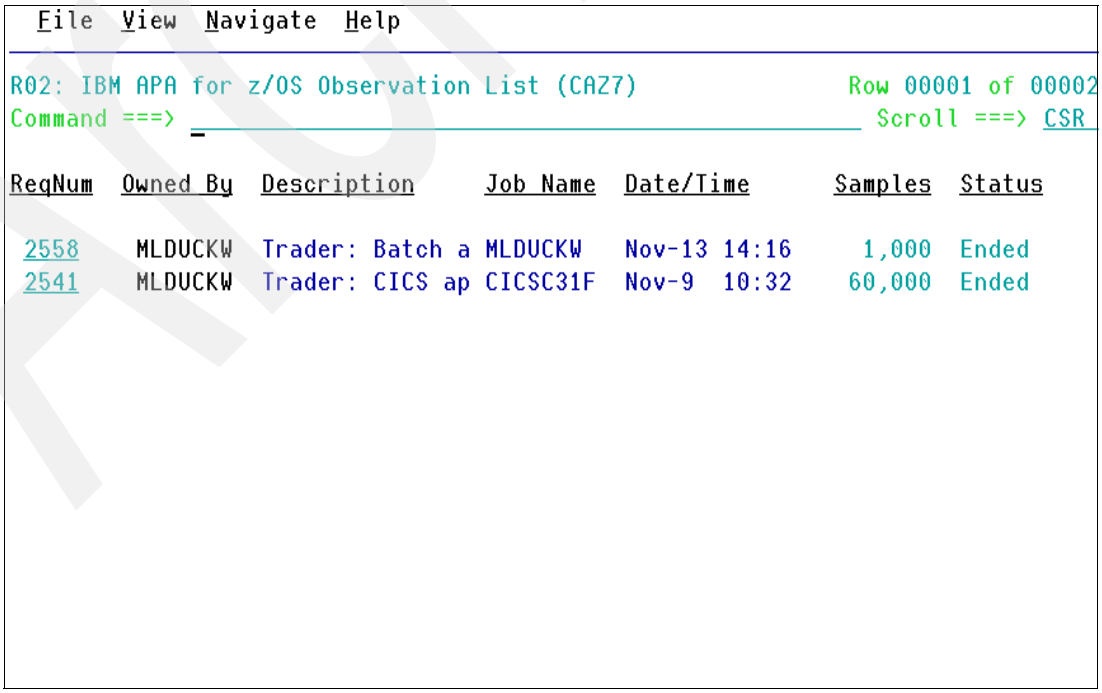


Figure 10-3 Observation List (R02): Status of an ended job

Tip: We have 1,000 samples. Because our sampling rate has been defined as 1000 samples per second, we can easily deduce that TRADERB ran approximately 1 second.

To see the details on the measurement request and on the sample file created just before, we can use the "++" line command on request number 2558. Application Performance Analyzer displays the pop-up menu shown in Figure 10-4 and Figure 10-5.

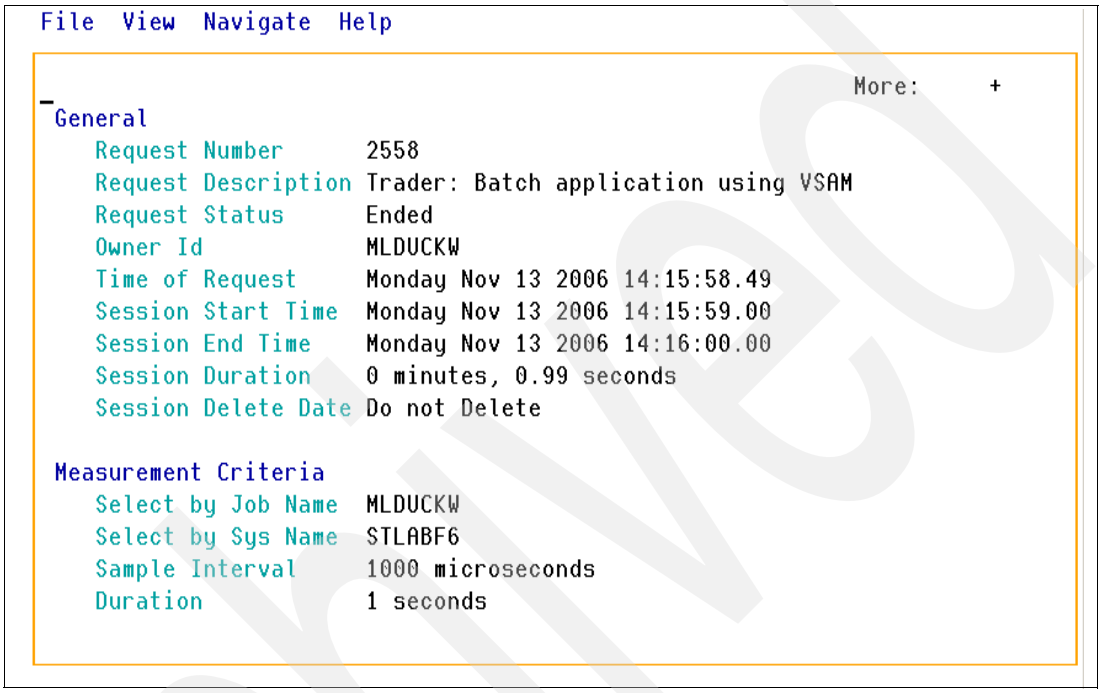


Figure 10-4 Observation List (R02): Detailed pop-up window (1 of 2)

When you press PF8 to scroll down, Application Performance Analyzer displays the last details, as shown in Figure 10-5.

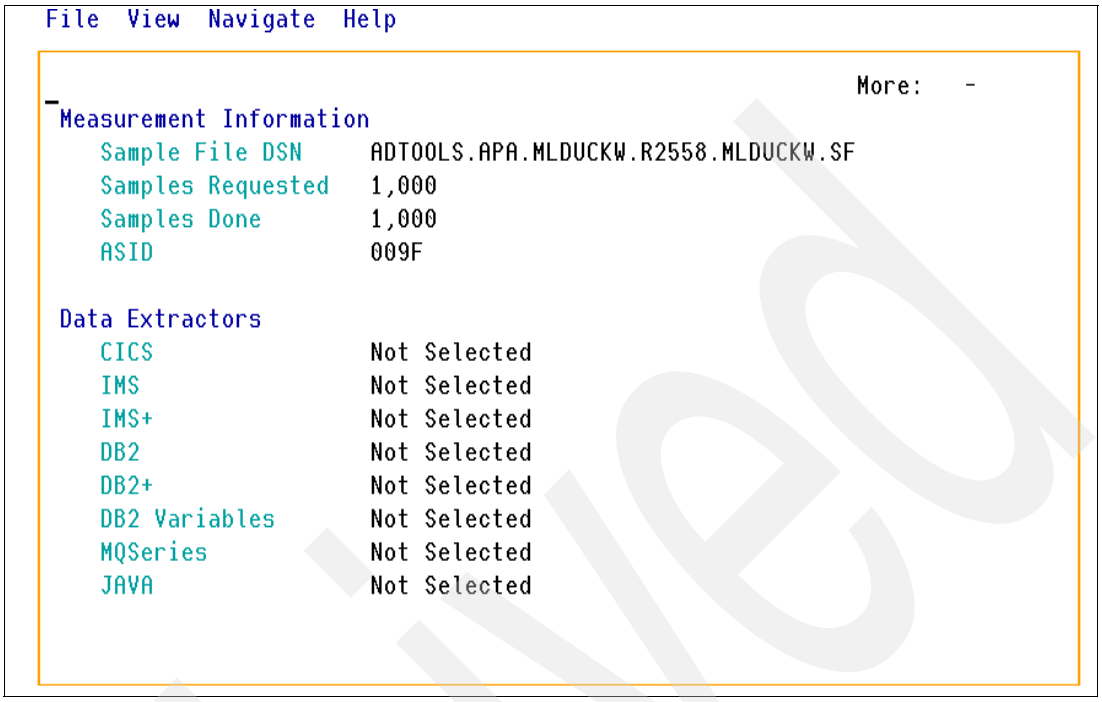


Figure 10-5 Observation List (R02): Detailed pop-up window (2 of 2)

Details include, but are not limited to, the following parameters:

- Session start time and end time
- Session duration
- Session delete date and time
- Sample interval
- Samples requested and samples done
- Data extractors status.

We use the **R** or **S** line commands on the request to view the reports.

The Performance Reports panel is displayed. The unavailable reports categories (either because the data extractor has not been selected, or because no data has been recorded for this specific extractor) are displayed in red as shown in Figure 10-6.

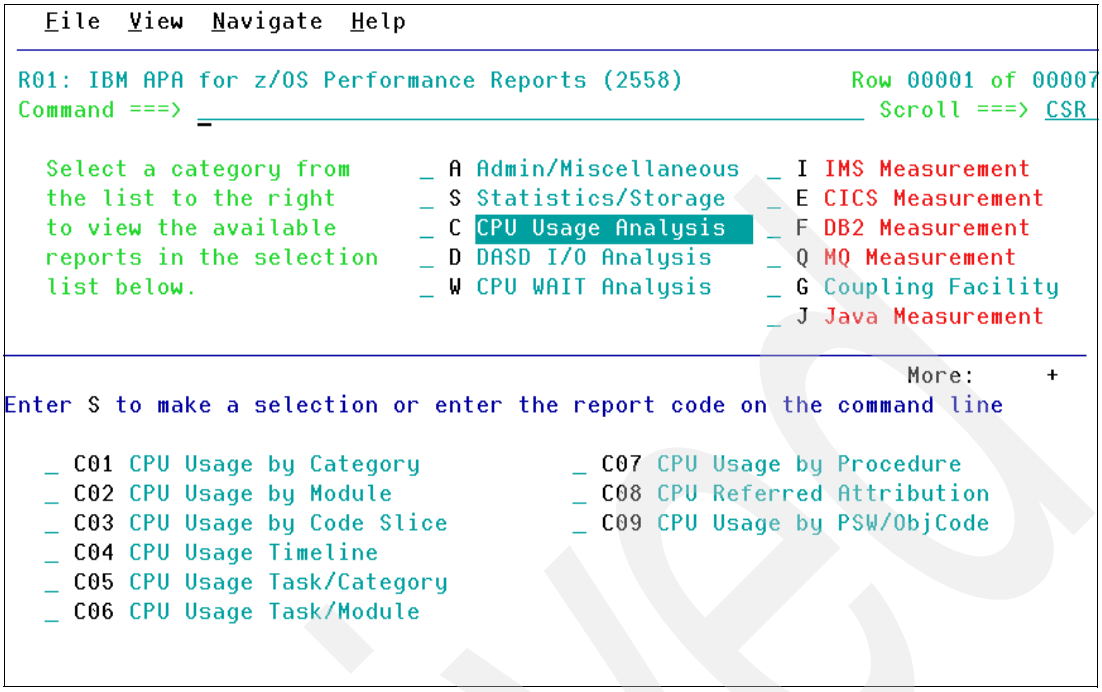


Figure 10-6 Performance Reports (R01)

Note: The most recently selected category persists from one session to the next.

In the remainder of this chapter we describe reports provided by Application Performance Analyzer to analyze a batch job without any subsystem (CICS, DB2, IMS, or MQSeries).

10.2 Statistics/Storage category

Nine reports belong to the Statistics and Storage category. Each of them are described in the following sections. The displays might be from different measurement reports.

10.2.1 S01 - Measurement profile

In this section is a general overview of the measurement.

Note: This is a good report to examine first when analyzing a measurement. It provides an at-a-glance summary of various aspects of the measurement data and helps us choose which other reports to concentrate on.

The first section of this report consists of a series of mini-performance graphs illustrating various types of activity that were measured. This is followed by a section that reports measurement values.

Note: If the DB2 data extractor has been selected, then the most active DB2 plans, the most active packages/DBRMs, and the most active SQL statements are also displayed.

The S01 - Measurement Profile can be divided into several sections seen by scrolling down through the report:

- CPU
- DB2 (if DB2 extractor has been turned on)
- Measurement information
- CPU consumption

CPU

Figure 10-7 illustrates the CPU information given by Application Performance Analyzer. This is a set of histograms quantifying measurement data. To the right of each graph, report codes of reports that show related and more detailed information are displayed. We can display the report by skipping the cursor to one of these fields and by pressing Enter.

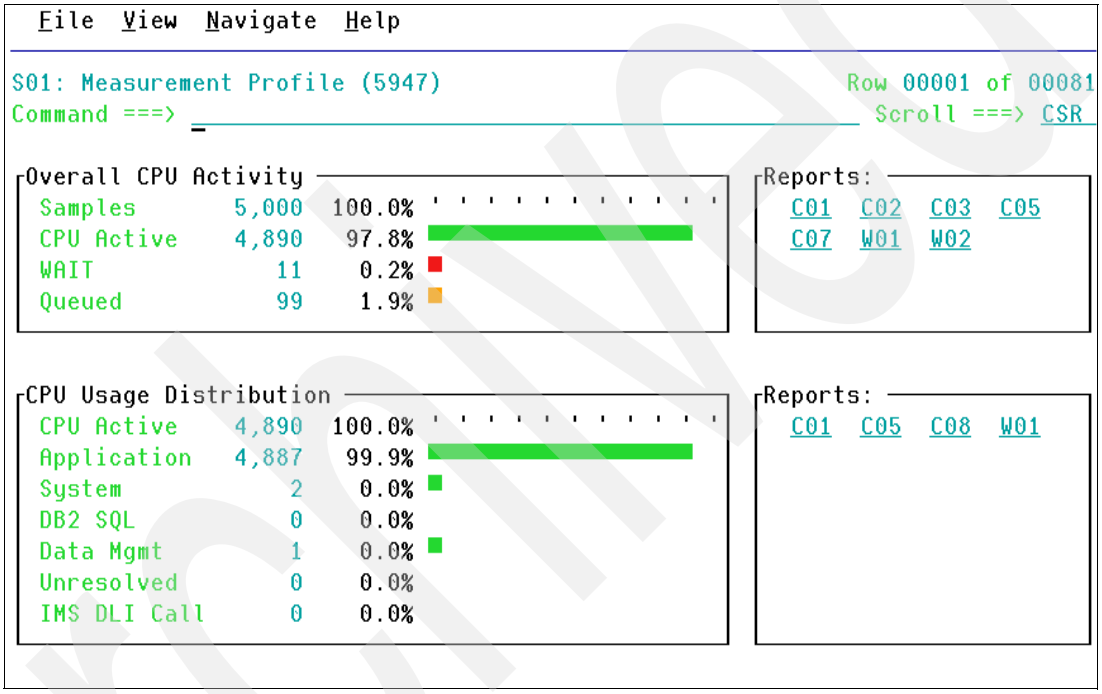


Figure 10-7 S01 - Measurement Profile: CPU information

The report section Overall CPU Activity shows that a total of 5,000 (or 100%) samples were taken during this observation session. Those CPU samples were:

- Active for 4,890 samples (or 97.8%)
- WAIT for 11 samples (or 0.2%)
- Queued for 99 samples (or 1.9%)

The next section, CPU Usage Distribution, shows how the 4,890 CPU active samples were distributed between:

- Application (Application Code)
- System (System/OS services)
- DB2 SQL (DB2 modules)
- Data management (DASD Request)
- Unresolved
- IMS DLI calls (IMS modules)

Notes:

- The CPU Active value can be different from the value reported in the Overall CPU Activity because multiple CPUs executing concurrently are counted separately here.
- This number represents 100% of the data upon which this section is based and it is used as the divisor to compute the percentages.

DB2

The TRADERB batch application uses only VSAM files. Therefore no DB2 or IMS data are provided.

Measurement values

This section of the report shows values relating to the measurement, and can be classified into several categories:

- Request parameters
- Measurement environment
- Measurement statistics
- CPU consumption

The request parameters contain information such as:

- Sample file DSN
- Retention
- Date and time of the request
- Step information

This is shown in Figure 10-8.

```
 File  View  Navigate  Help

S01: Measurement Profile (5947)                          Row 00030 of 00081
Command ===> _                                            Scroll ===> CSR

Request parameters
 Request number      5947
 Description         Unspecified
 Sample file DSN     ADTOOLS.APA.APEDU01.R5947.APEDU01B.SF
 Retention           Retain until deleted by user
 Data extractors     IMS,DB2,DB2+,IMS+

 Requesting user     APEDU01          Nbr of samples      5,000
 Time of request     08:46:44         Duration            120 sec
 Date of request     Fri Feb-03-2006  Active/pending      Pending
 Job name            APEDU01B         Proc step name      n/a
 Step name/number    n/a              Delay time          none
 Step program        n/a
```

Figure 10-8 S01 - Measurement values: request parameters

The next two sections, Measurement environment and Measurement statistics, provide details relating to the environment on which the measurement took place.

This includes the following values for the Measurement environment:

- Job name
- Region type and size
- DB2 attachment (if relevant)
- O/S level
- Application Performance Analyzer version
- CPU model
- Number of CPUs and MIPS per CPU

Measurement statistics gives the following values:

- Total number of samples
- Sampling rate
- Sample file size
- CPU/WAIT samples
- CPU queued samples
- Meas significance: Ratio of number of CPU/WAIT samples to number of TCB samples

This is shown in Figure 10-9.

```
 File  View  Navigate  Help

S01: Measurement Profile (5947)                          Row 00045 of 00081
Command ===> _                                            Scroll ===> CSR

Measurement environment
 Job name           APEDU01B        Region size <16MB  4,160K
 Job number         JOB02327        Region size >16MB  32,768K
 Step name          RUNSAM1         Step program       SAM1V
 ASID               51              Region type        Batch
 DB2 Attach type    n/a

 System ID          STLABF6         IBM APA Version    1.110F
 SMFID              SYE1
 O/S level          z/OS 01.04.00

 Nbr of CPUs        4               CPU model          2064
 CPU rate factor    1,484           CPU version        00
 MIPS per CPU       222             SUs per second     10781.6

Measurement statistics
 Start time         08:46:44        Start date         Fri Feb-03-2006
 End time           08:48:44        End date           Fri Feb-03-2006

 Total samples      5,000           Duration           2 min 0.01 sec
 Sampling rate      41.66 per sec   Sample file size   1.10MB
 CPU/WAIT samples   4,901           Meas significance  98.02%
 TCB samples        5,000           CPU queued samples 99
 Overall CPU        26.30%

 Pages in           0               EXCPs              240
 Pages out          0
```

Figure 10-9 S01 - Measurement values: Measurement environment and statistics

Note: Meas significance (the field of Measurement statistics) gives the quality of the measurement data. A low value indicates that CPU resources were unavailable to service the job.

The last section CPU consumption, shown in Figure 10-10, contains CPU information.

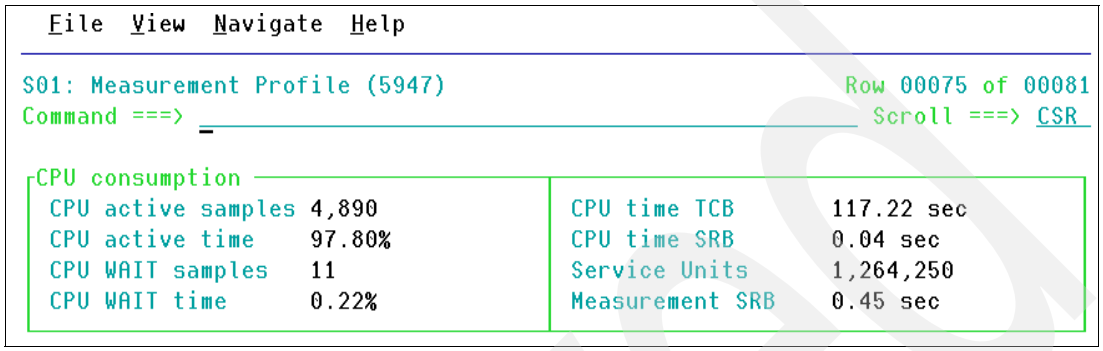


Figure 10-10 S01 - Measurement values: CPU consumption

Note: Measurement SRB is the number of CPU seconds in SRB mode consumed by the Application Performance Analyzer measurement. It gives an idea of its overhead.

10.2.2 S02 - Load module attributes

This report lists information about each load module for which activity was measured during the observation session. Figure 10-11 shows that various attributes of the modules are reported.

Tip: The SETUP command line allows us to exclude (or not) the following information:

- External Symbol Dictionary information
- Modules loaded in PLPA
- Modules loaded in the NUCLEUS

You can overtype the word *SORT* in the upper left of the report, to sort the report by:

- Name using **SN**
- Address using **SA**
- Size using **SS**
- Loadlib using **SL**

Note: You cannot enter these commands from the command line. You must overtype the word SORT with the desired 2-character command.

```
  File  View  Navigate  Help
S02: Load Module Attributes (2558)                        Row 00001 of 01266
Command ===> _                                            Scroll ===> CSR

SORT by name enter: SN, by load address: SA, by size: SS, by loadlib: SL
Information reported for 24 load modules.

Module Information for IKJEFG00
   Load Address      00068000 to 0006AD0F
   Module Size       11,536
   Attributes        REUS,RENT,APFLIB
   Module Location   JPA
   Loadlib DDNAME    -VLF-
   Load Library      SYS1.CMDLIB
   Program Group     MVS System
   Subgroup          MVS Services
   Function          TSO Services

ESD Information for IKJEFG00
   External  Offset  Length  Start Addr  End Addr
   IKJEFG00  000000    7072   00068000   00069B9F
   IKJEFT06  001BA0    4176   00069BA0   0006ABEF
   IKJEFG01  002BF0     132   0006ABF0   0006AC73
```

Figure 10-11 S02 Load Module Attributes

10.3 CPU Usage Analysis category

Nine reports belong to the CPU Usage Analysis category. Each of them is described below using request number 4790 as support for illustration.

10.3.1 C01 - CPU Usage by Category

C01 analyzes measured CPU consumption and attributes it to six general categories:

- APPLCN - Application Code
- SYSTEM - System/OS Services
- DB2SQL - SQL Processing
- DATAMG - Data Management (DASD) Requests
- IMSDLI - IMS DL/I call processing
- NOSYMB - No Module Name Found

Tip: Use the SETUP command line to specify whether DB2SQL, DATAMG, and IMSDLI have to be displayed.

When C01 is first displayed, as shown in Figure 10-12, only the relevant categories are visible. To expand any of these categories to show the next n hierarchical levels, we can type the **+** line command on any detail line.

```
 File  View  Navigate  Help
C01: CPU Usage by Category (0017)                          Row 00001 of 0000
Command ===>                                                Scroll ===> CSR

Name      Description              Percent of CPU Time * 10.00%  +1.9%
                                       *....1....2....3....4....5....6....7...
SYSTEM    System/OS Services       57.99
APPLCN    Application Code         41.50
NOSYMB    No Module Name            0.49
```

Figure 10-12 C01 - CPU Usage by Category initial view

Figure 10-13 shows C01 with up to the next two hierarchical levels of the Data Management category and the next hierarchical level of the Application category.

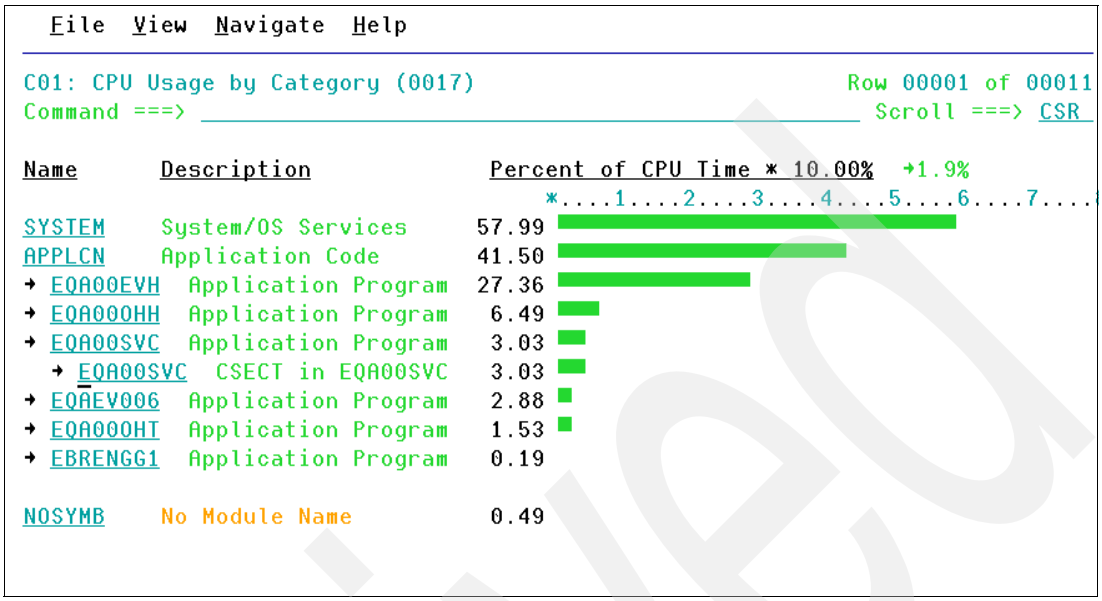


Figure 10-13 C01 - CPU Usage by Category - Expanded

10.4 CPU Wait Analysis category

Three reports belong to the CPU Wait Analysis category. Viewing these reports can assist in reducing the elapsed time of the job/task, but have little or no impact on CPU usage.

10.5 W01 - WAIT Time by Task/Category

W01 for each task (TCB) shows the percentage of elapsed time during which the task was in WAIT. When the report is first displayed, only the top level of the hierarchy (tasks) is visible. To expand a task to show the next hierarchical level, use the + line command. Figure 10-14 shows an expanded entry.

```
File  View  Navigate  Help
W01: WAIT Time by Task/Category (9267)                    Row 00003 of 00596
Command ===> _____________________________________________ Scroll ===> CSR

Name          Description          Percent of Time in WAIT * 10.00%  +0.3%
                                      *....1....2....3....4....5....6....7....8.
IKJEFT02-006  TCB=008CBB00       100.00
EXEC-008      TCB=008CB518        87.20
+ SYSTEM        System/OS         85.93
                Services
  + SVC           SVC Routines    85.29
    + SVC018        BLDL/FIND     58.58
    + SVC027        OBTAIN        11.14
    + SVC026        CATALOG/INDE   6.57
                    X/LOCATE
    + SVC006        LINK           3.21
    + SVC020        CLOSE          2.80
    + SVC019        OPEN           1.83
    + SVC022        OPEN TYPE=J    0.58
    + SVC122        LOAD/LINK/XC   0.38
                    TL/EVENTS
    + SVC048        DEQ            0.08
    + SVC056        ENQ RESERVE    0.08
```

Figure 10-14 W01 - WAIT Time by Task/Category (expanded)

Note: Line commands + and - have been used to respectively increase the Description field size and zoom out the scale of Percent of Time in Wait.

The SETUP command line has been used to eliminate inactive tasks from the report.

11

Application Performance Analyzer: Analyzing a job with subsystem

In this chapter we describe reports provided by Application Performance Analyzer to specifically analyze a subsystem environment.

This chapter contains a section for each of the following subsystems:

- IMS
- CICS
- DB2
- MQSeries

Each section begins with the screen shots used to create the measurement request followed by all the reports belonging to a specific subsystem.

11.1 IMS measurement

To illustrate IMS reports, we use the TRADER application in batch mode using IMS databases. This application is described in detail in Guided Tour of the Trader application.

We do not spend much effort on the things that are common to Application Performance Analyzer and that have been described in the previous chapter, but rather focus on the unique aspects of using Application Performance Analyzer with batch job accessing IMS databases.

11.1.1 Observation session request definition

We have to monitor the TRADERI batch job with the following characteristics:

- We use a sampling rate of 1000 samples per second for all its duration.
- TRADERI has a single step.
- The measurement file has to be retained for 30 days.
- All available IMS data must be recorded.
- CHABERT has to be notified at the end of the measurement.

By entering the NEW command line on the Observation List (R02) panel, Application Performance Analyzer displays the Schedule New Measurement (R03) panel. Both IMS and IMS+ data extractors have to be turned on to get all IMS data.

When the IMS data extractor is active all DL/I calls in the target address space are recorded each time a sample is taken. For each DL/I call-in flight, all of the call parameters, SSAs, and so on are recorded, as well as the module and offset from which the call was made. Additionally, several IMS environment parameters are recorded each time a sample is taken. These parameters include the region type and its status, the PSB, transaction code, IMS version and release, message sequence number, and time stamp from the IOPCB.

IMS+ data extractor is an IMS measurement option in which the precise number of DL/I calls is counted as well as the exact DL/I service time and CPU time by DL/I call.

The settings required to monitor the TRADERI job are shown in Figure 11-1 for panel 1 (Job Information) and Figure 11-2 for panel 2 (Options).

```
  File  View  Navigate  Help
R03: Schedule New Measurement                                  Row 00001 of 00012
Command ===> _                                                 Scroll ===> CSR

● 1. Job Information     3. Multi Steps      5. CICS Txns      7. Schedule
  2. Options             4. Active Jobs      6. Sysplex        8. Sched Options

Panel 1. Job Information                    Input more data or ENTER to submit

  Job Name/Pattern . . TRADERI    System Name . . . *

  Step Specification
    Step No. . . . . ____         Specify step number, program name,
    Program Name . . . ________   step name or step name + Proc step
    Step Name  . . . . ________   name. Use panel 3 to specify more
    ProcStepName . . . ________   than one step.

  Description  . . . . Trader: Batch application using IMS DB
  Number of Samples  . 1000       Measure to step end . . . Y
  Duration (min:sec) . 1          Delay by (secs) . . . . . _____
  Notify TSO User  . . CHABERT    Retain file for (days)  . 30

MA   a                                                                  04/015
```

Figure 11-1 R03: Schedule New Measurement (panel 1: Job Information)

With panel 1 we specify:

- Job name
- Sample rate
- Retention period

```
  File  View  Navigate  Help
R03: Schedule New Measurement                                  Row 00001 of 00022
Command ===> _                                                 Scroll ===> CSR

● 1. Job Information     3. Multi Steps      5. CICS Txns      7. Schedule
  2. Options             4. Active Jobs      6. Sysplex        8. Sched Options

Panel 2. Measurement Options

 Data Extractors. '/' to select extended measurement options:
      _  CICS  CICS information
      _  DB2   SQL call information
      /  IMS   DLI call information
      _  MQ    MQSeries call information
      _  DB2+  SQL service/CPU time/counts
      /  IMS+  DLI service/CPU time/counts

 Specify up to 10 load libraries to be searched by IBM APA for z/OS for
 external symbol information. These are applicable only when sampled modules
 are fetched from dynamically allocated load libraries.

    1  _____________________________________________
    2  _____________________________________________
    3  _____________________________________________
    4  _____________________________________________
    5  _____________________________________________
    6  _____________________________________________
    7  _____________________________________________
    8  _____________________________________________
    9  _____________________________________________
   10  _____________________________________________

MA   a                                                                  23/010
```

Figure 11-2 R03: Schedule New Measurement (panel 2: Options)

With panel 2 we turn on both IMS and IMS+.

When we press Enter to submit the previous definition, Application Performance Analyzer displays the Observation List (R02) panel, with our new measurement definition at the top of the list. To check whether we do not forget any parameter, we can use the ++ line command (or only press Enter) on the request number 4800 (which is the done by Application Performance Analyzer for our new measurement request).

Figure 11-3 and Figure 11-4 give the full definition of the request made to monitor our TRADERI batch job.

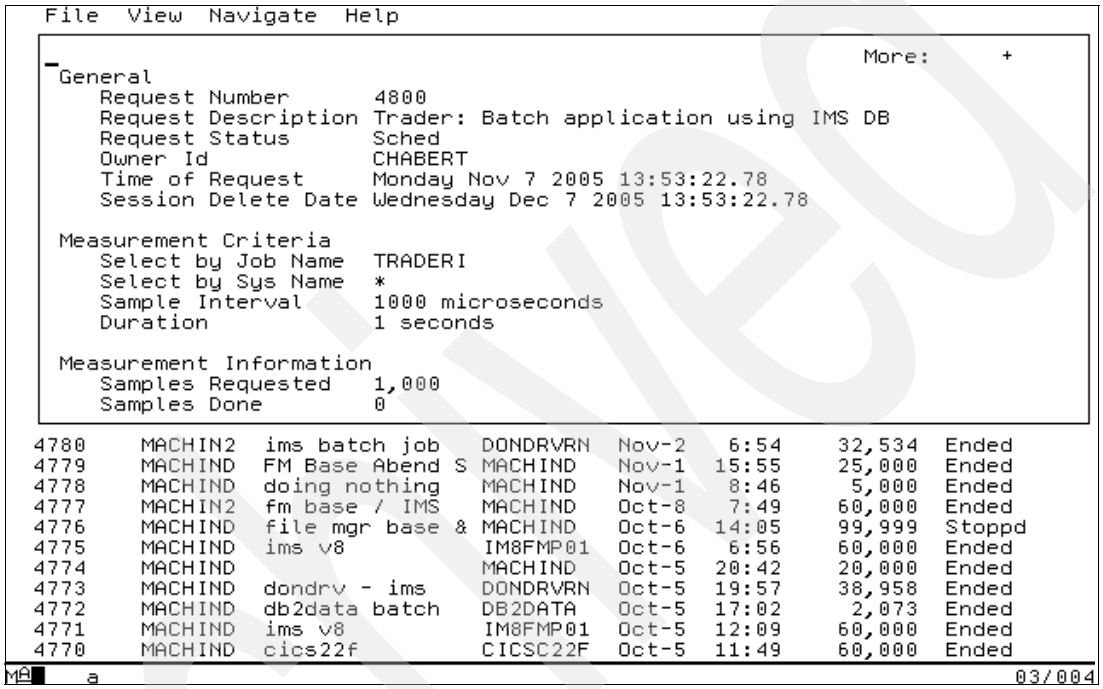


Figure 11-3 Detail about ReqNum 4800 (1 of 2)

The first part shows that the job name, sampling rate, and retention period are what we want. Using the PF8 key to scroll towards the bottom of the report, we have the status of each data extractor.

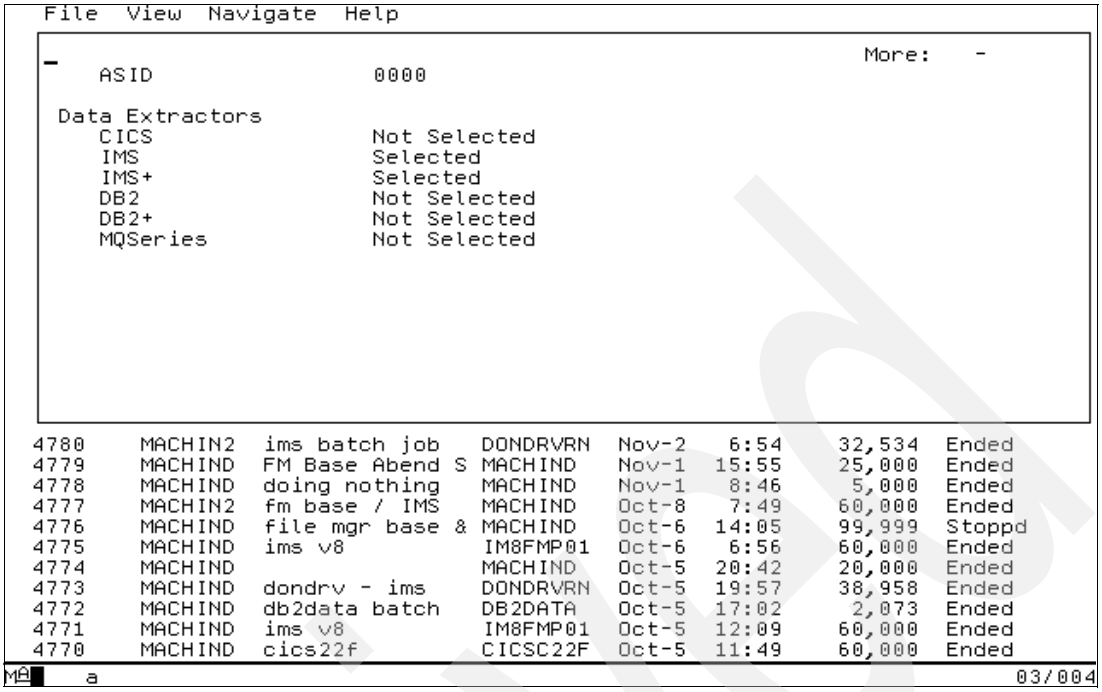


Figure 11-4 Detail about ReqNum 4800 (2 of 2)

Because both IMS and IMS+ are selected, we can start the TRADERI batch job and let Application Performance Analyzer record samples.

When the job ends, because CHABERT has been specified as the TSO user ID to be notified upon completion of the measurement, he receives the Application Performance Analyzer message shown in Figure 11-5.

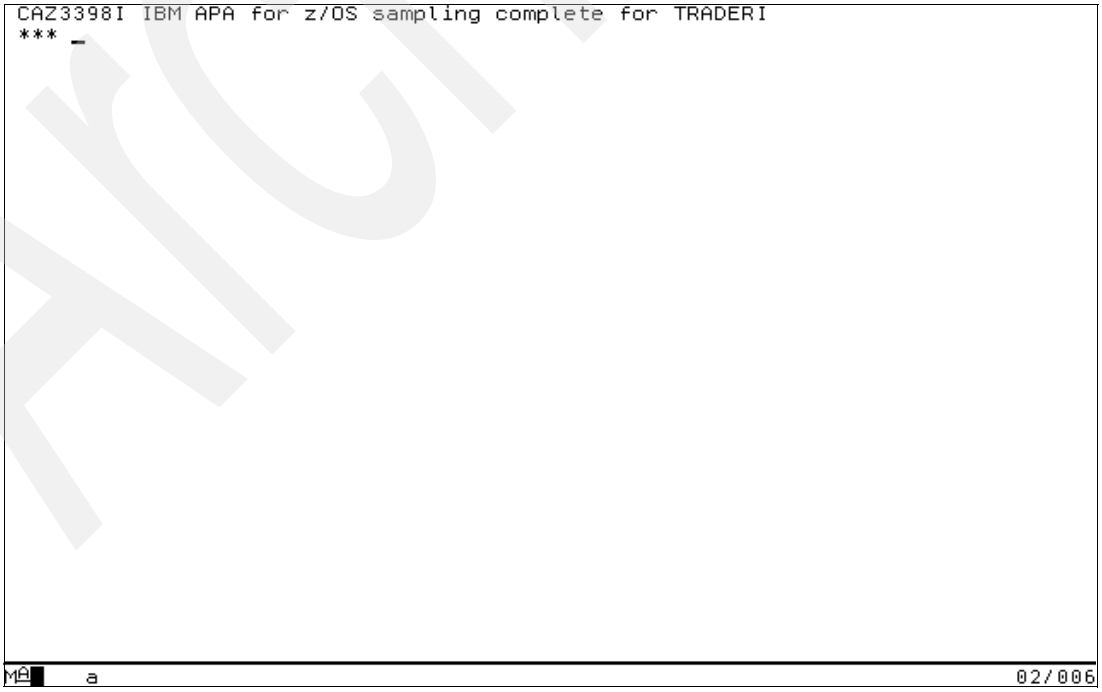


Figure 11-5 APA: Completion message

We use the **R** line command on request number 4800 to launch the associated Performance Reports, as shown in Figure 11-6.

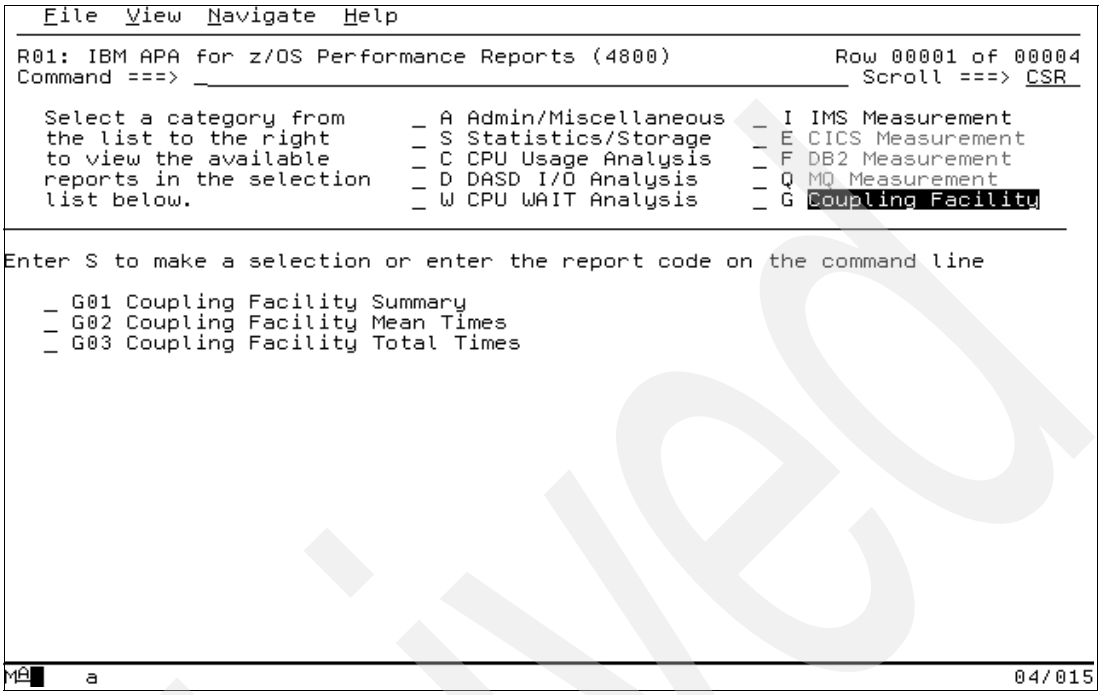


Figure 11-6 R01: Performance Reports (last selected category)

Note: The Coupling Facility category is highlighted because it was the last selected category.

Because `I IMS Measurement` is displayed in red (in gray in our figure), IMS reports are available. To get the list of available IMS reports, shown in Figure 11-7, we just type `I` on the command line and press Enter.

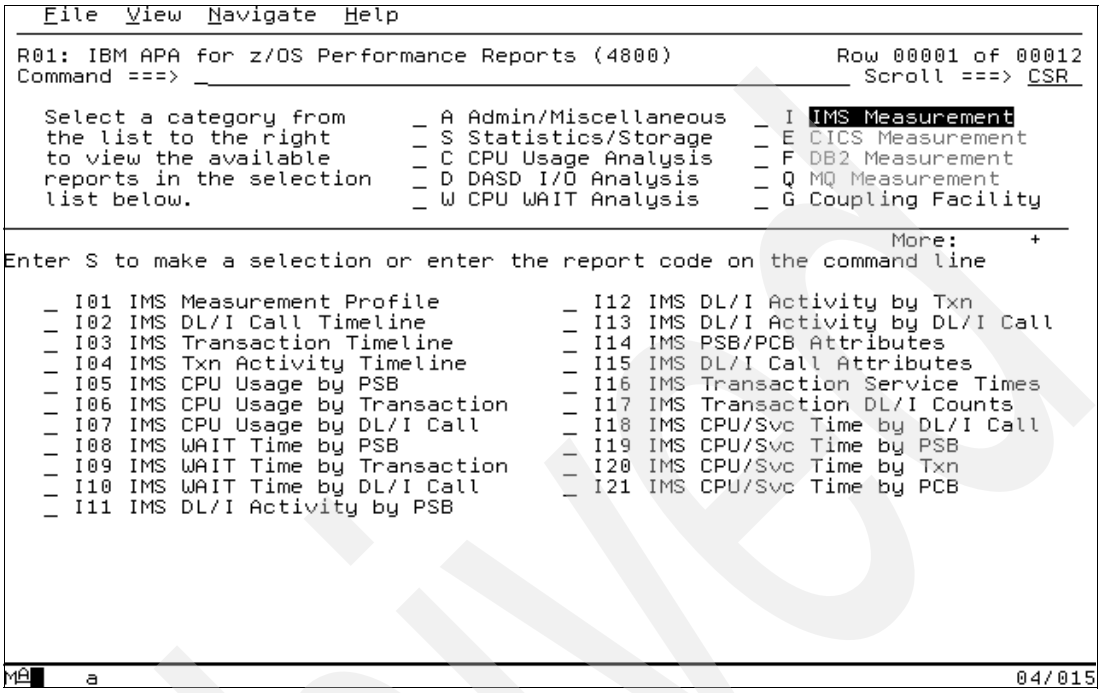


Figure 11-7 R01: Performance Reports (IMS category)

Twenty-one reports belong to the IMS Measurement category.

11.1.2 Remark

Because the TRADER application has no IMS transaction, some reports are irrelevant or are a duplication of others.

The first part of the IMS section describes available reports for a DL/I batch application. The reports are:

- I01: IMS Measurement profile
- I02: IMS DL/I Call timeline
- I05: IMS DL/I CPU usage by PSB
- I07: IMS DL/I CPU usage by DL/I call
- I08: IMS DL/I WAIT time by PSB
- I10: IMS DL/I WAIT time by DL/I call
- I11: IMS DL/I activity by PSB
- I13: IMS DL/I activity by DL/I call
- I14: IMS PSB/PCB attributes
- I15: IMS DL/I call attributes
- I18: IMS CPU/Svc time by DL/I calls
- I19: IMS CPU/Svc time by PSB
- I21: IMS CPU/Svc time by PCB

Reports I06, I09, and I12 are available but contain the same information as reports I05, I08, and I11, respectively, when there is no IMS transaction involved.

If report I03, I04, I16, I17, or I20 is selected, Application Performance Analyzer displays the message shown in Figure 11-8.

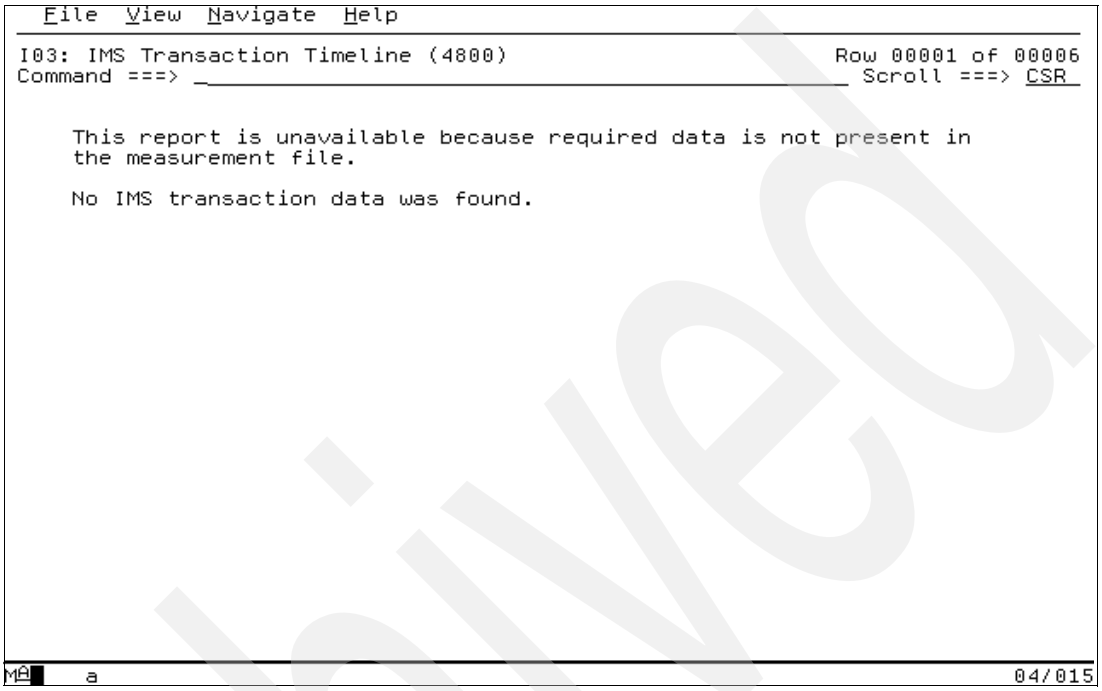


Figure 11-8 APA message when no IMS transaction available

For the second part of the IMS section, an IMS transaction is used to describe the other IMS reports, which are:

- I03: IMS transaction timeline
- I04: IMS transaction activity timeline
- I06: IMS DL/I CPU usage by transaction
- I09: IMS DL/I WAIT time by transaction
- I12: IMS DL/I activity by transaction
- I16: IMS transaction service times
- I17: IMS transaction DL/I call counts
- I20: IMS CPU/Svc time by transaction.

11.1.3 I01 - IMS measurement profile

This report gives a general overview of the IMS measurement data.

Note: This is a good report to examine first when analyzing IMS information. It provides an at-a-glance summary of various aspects of the measurement data and helps us choose which other reports to concentrate on.

I01 - IMS Measurement Profile can be divided into several sections:

- IMS environment
- Most active
  - IMS PSBs
  - IMS transactions
  - DL/I calls
- Most CPU consumptive DL/I
- Most frequent:
  - Transactions
  - DL/I calls

  Note: This requires that the IMS+ measurement option is active.

- Transaction statistics

  Note: This requires that the IMS+ measurement option is active.

Note: The TRADER application has no IMS online transaction. Therefore the Most active IMS transactions, Most Frequent Transactions, and the Transaction statistics sections are shown, later in this chapter, using another application.

IMS environment

Figure 11-9 illustrates the IMS environment information given by Application Performance Analyzer. It includes the PARM data that was passed to DFSRRC00, the IMS version, and the IMS region name and type.

```
I01: IMS Measurement Profile (4800)                    Row 00001 of 00043
Command ===> _____________________________________________ Scroll ===> CSR

IMS Environment
 DFSRRC00 parms     DLI,TRADERI,PTRDI,,,,,,,,,,,N,,

 IMS system id      IMS8              IMS region name    TRADERI
 IMS version        8.1.0             IMS region type    DL/I Batch
```

Figure 11-9 I01: IMS Measurement Profile (IMS Environment)

Note: All sections above are sets of histograms quantifying measurement data. To the right of the graphs, report codes of reports that show related and more detailed information are displayed. We can display the report by skipping the cursor to one of these fields and pressing Enter.

Most active: IMS PSBs or IMS transactions or DL/I calls

Figure 11-10 illustrates the *Most Active* information given by Application Performance Analyzer.

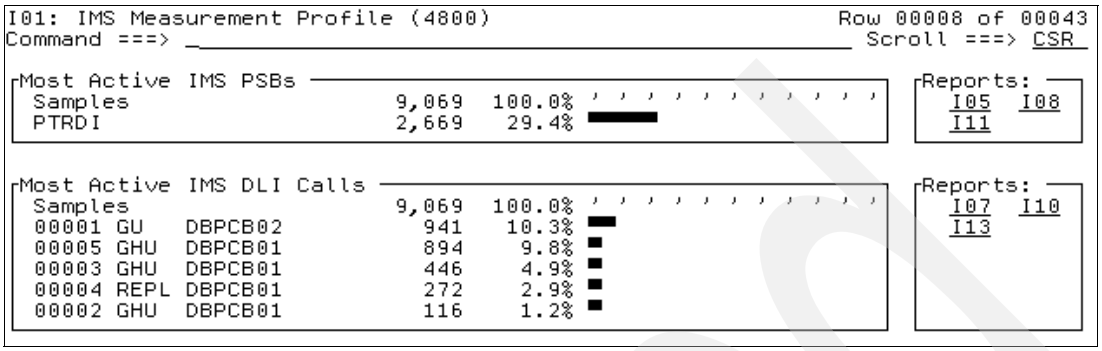


Figure 11-10 I01: IMS Measurement Profile (Most Active)

A DL/I call is identified by three fields:

- A unique sequence number assigned to the DL/I call
- Its DL/I function
- Its PCB name

Most CPU consumptive DL/I

Figure 11-11 illustrates the *Most CPU consumptive DL/I* information given by Application Performance Analyzer.

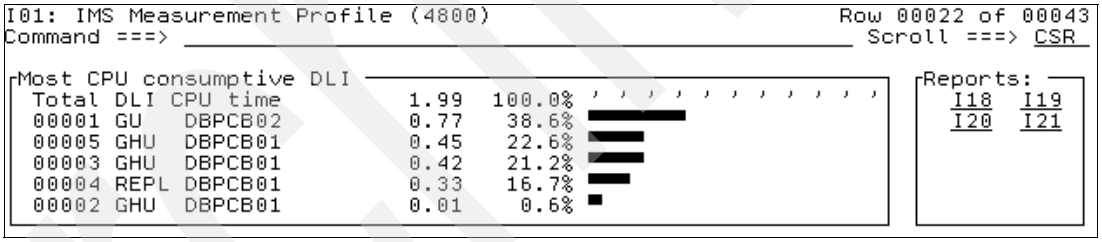


Figure 11-11 I01: IMS Measurement Profile (Most CPU consumptive DLI)

Most frequent: Transactions or DL/I calls

Figure 11-12 illustrates the *Most Frequent DL/I calls* information given by Application Performance Analyzer.

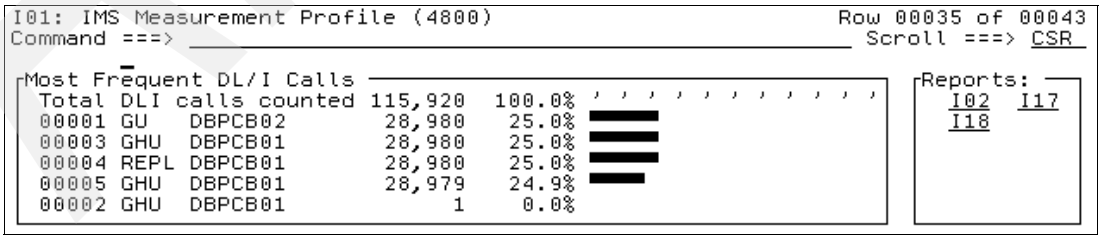


Figure 11-12 I01: IMS Measurement Profile (Most Frequent DL/I Calls)

11.1.4 I05 - IMS DL/I CPU usage by PSB

I05 shows the distribution of CPU usage aggregated by PSB. IMS CPU usage measured during the execution of DL/I calls is reported under detail lines, which identify each of the DL/I calls. Non-DL/I call CPU usage is reported as application code or system routines. A non expanded report shows a line for each IMS PSB in which CPU usage was observed. It reports CPU usage for which there was no IMS PSB on a separate line named NONIMS. We can expand each line to reveal additional hierarchical levels of detail, as shown in Figure 11-13.

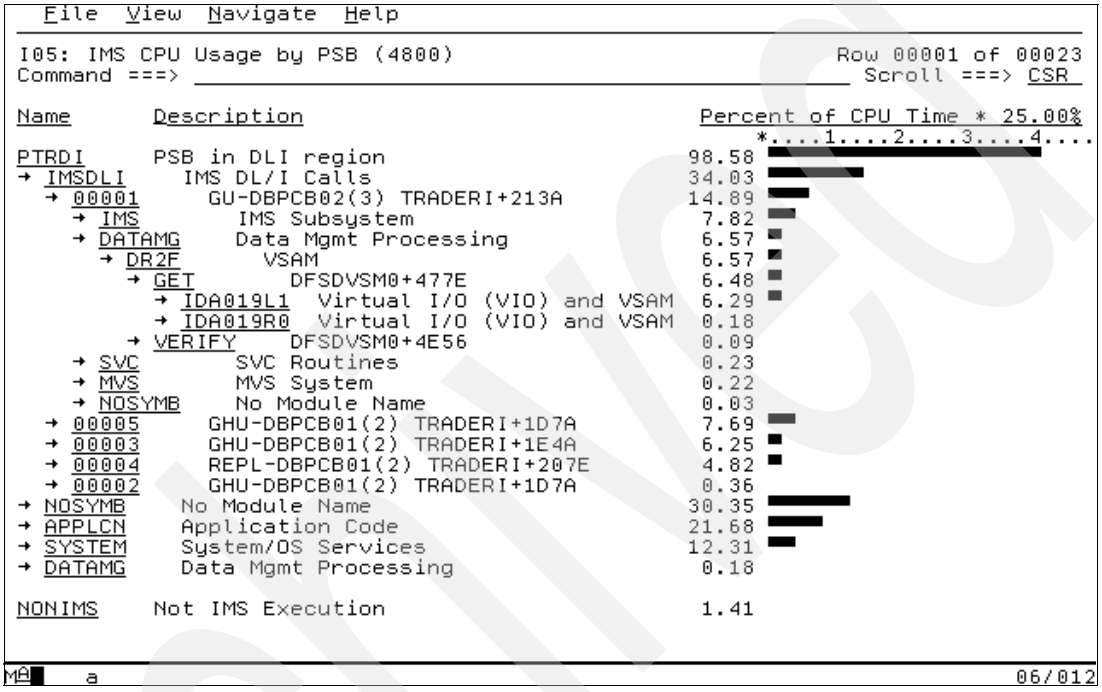


Figure 11-13 I05: IMS CPU Usage by PSB (expanded)

This report is a full expansion of the GU DL/I call (or GET function) done by the PSB PTRDI against the IMS database DR2F.

Note: The size of the Description column has been expanded using the + line command on the heading and the scale of the Percent of CPU Time * column has been reduced (zoomed out) by using the - line command on the heading.

We can enter the ++ line command (or press Enter) on any line to list a pop-up window containing additional information about PSB, DL/I Call, category, load module, SVC, or CSECT, as shown in Figure 11-14, for the entry named GU-DBPCB02(3), where:

- GU is the DLI function code.
- DBPCB02 is the PCB name.
- (3) is the relative PCB number.

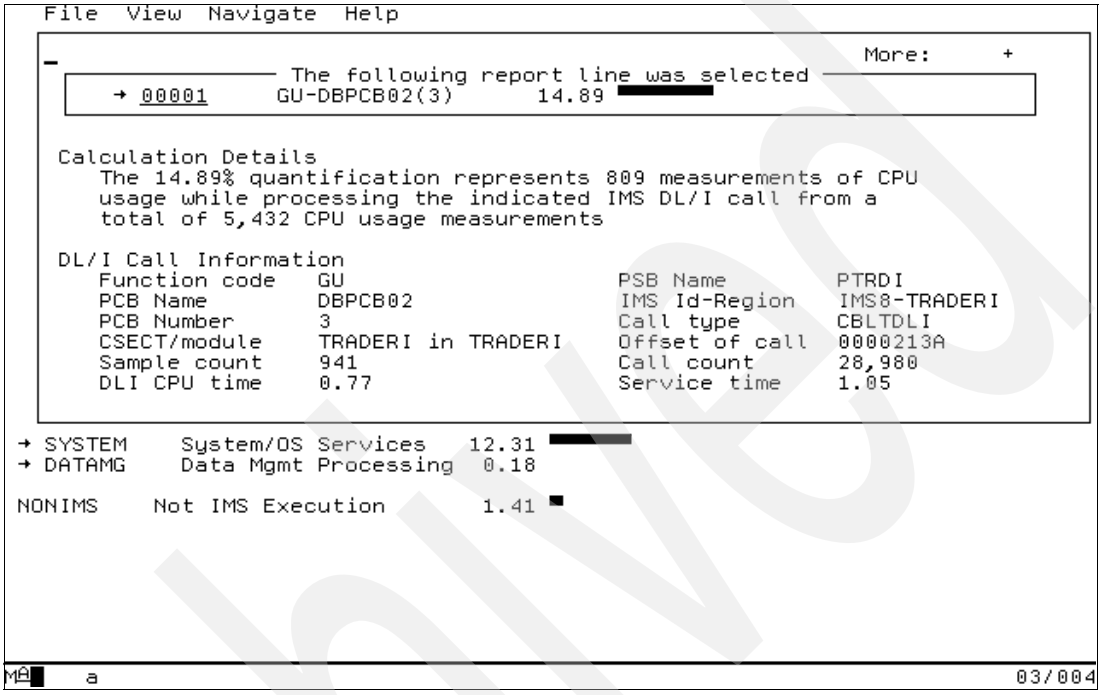


Figure 11-14 I05: Detail window for DL/I call

Note: Use F8 to display the associated SSA/FSA.

11.1.5 I07 - IMS DL/I CPU usage by DL/I call

I07 shows the distribution of CPU usage across IMS DL/I calls. It reports only CPU activity that occurred during processing of DL/I calls.

Each line can be expanded to show CPU distribution between each category (IMS, DATAMG, SVC, MVS, and so on). Figure 11-15 shows an expanded version of I07.

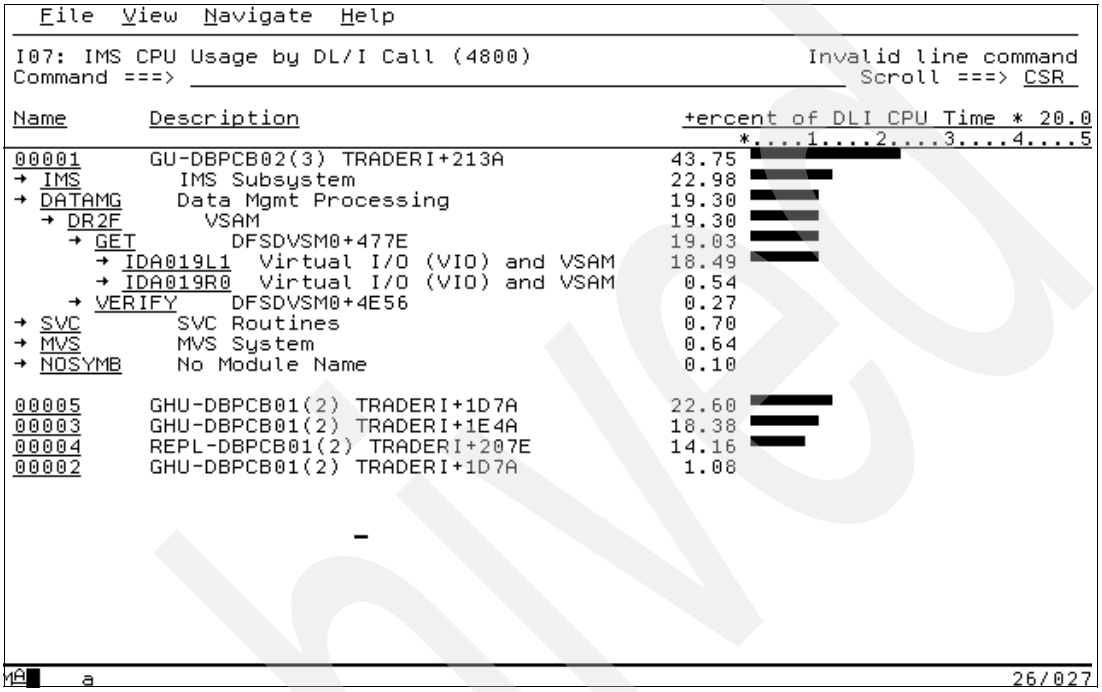


Figure 11-15 I07: IMS CPU Usage by DL/I Call (expanded)

11.1.6 I08 - IMS DL/I WAIT time by PSB

I08 identifies any delays caused by wait conditions in IMS regions. This report shows wait time by IMS PSB and is meaningful when measuring a region in which multiple IMS PSBs are scheduled. Only wait time observed when an IMS PSB is active is reported. Wait time is identified both within the processing of DL/I calls and outside of DL/I call processing.

Each report line quantifies wait time as a percentage of the overall time during which IMS PSBs were active, as shown in Figure 11-16.

Note: Any time when no IMS programs were active is excluded. This ensures that quantifications are not distorted by inactive intervals such as those that occur between scheduled transactions.

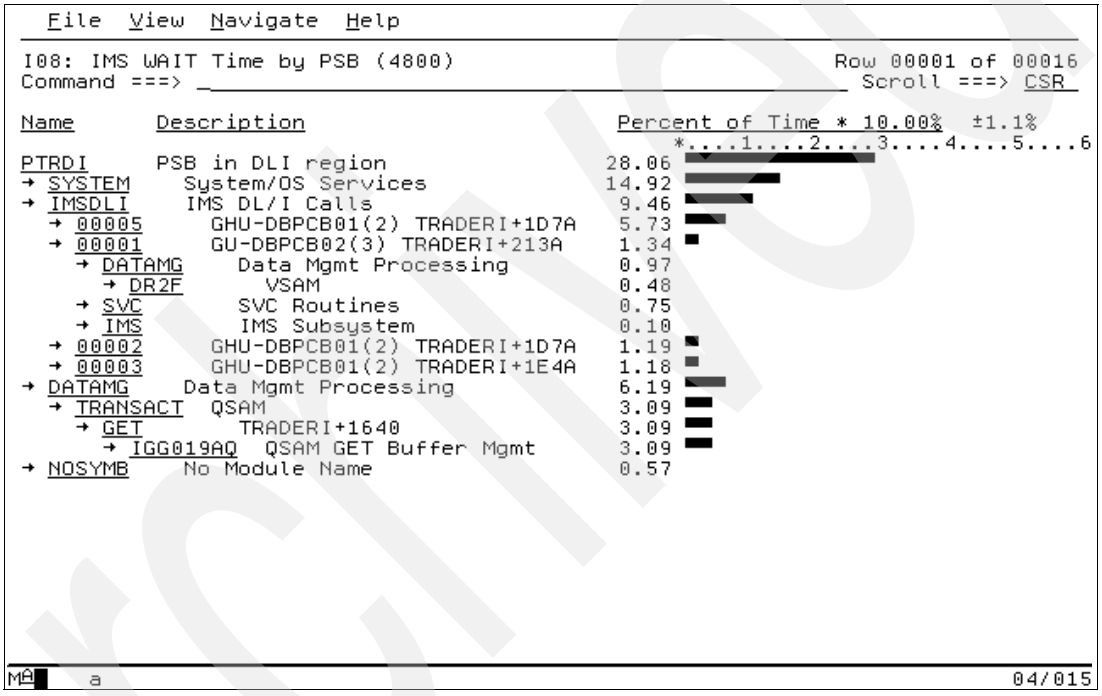


Figure 11-16 I08: IMS WAIT Time by PSB (expanded)

Because source program mapping data is available, by entering the **P** line command on any line where csect + offset is displayed, Application Performance Analyzer displays the source code for the statement, as shown in Figure 11-17, for DLI call number 00001.

```
  File  View  Navigate  Help
 ─────────────────────────────────────────────────────────────────────────────
 P01: Source Program Attribution (4800)                      Row 00001 of 00022
 Command ===> ________________________________________________ Scroll ===> CSR

 LineNo Offset Count Source Statement

 000877 0020DE                           DISPLAY 'WAS REPLACING BY ' CUSTCOMP-SEG
 000878 0020EC                           MOVE BAD-CUST-REWRITE TO RETURN-VALUE
 000879                             END-EVALUATE
 000880                             .
 000881 0020F2               UPDATE-CUSTFILE-EXIT.
 000882 0020F2                   EXIT.
 000883                     ************************************************************
 000884 0020F8               READ-COMPFILE SECTION.
 000885 0020F8                   MOVE 'READING RECORD FROM COMPANY DB   ' TO COMME
 000886
 000887 002104   105             CALL 'CBLTDLI' USING GU
 000888                                                CO-PCB
 000889                                                COMPANY-SEG-IN
 000890                                                SSA-COMPANY
 000891                               .
 000892                     *
 000893 002146                   EVALUATE CO-PCB-STATUS
 000894                             WHEN OK
 000895 002154                        MOVE CLEAN-RETURN TO RETURN-VALUE
 000896                             WHEN NOT-FOUND
 000897 00216C                        DISPLAY 'COMPANY NOT FOUND' CO-PCB-STATUS


 MA     a                                                                04/015
```

Figure 11-17 P01: Source Program Attribution (from I08)

It is also possible (by entering the **+** line command on any DDname) to show additional details (the same information as D04: Dataset Attributes for non-VSAM file or D06: DASD VSAM Statistics for VSAM file), as shown in Figure 11-18, for DR2F.

```
  File  View  Navigate  Help
  ┌──────────────────────────────────────────────────────────────────────────┐
  │ _                                                      More:     +       │
  │ ┌──────────── The following report line was selected ─────────────┐     │
  │ │     → DR2F      VSAM                          0.48               │     │
  │ └──────────────────────────────────────────────────────────────────┘     │
  │                                                                          │
  │  Calculation Details                                                     │
  │     The 0.48% quantification represents 38 measurements                  │
  │     of wait time observed in data management routines                    │
  │     while servicing I/O requests for the file allocated                  │
  │     to DDNAME DR2F from a total of 7,778 overall                         │
  │     measurements.                                                        │
  │                                                                          │
  │  VSAM file DR2F OPENed at 13:55:58.89 Monday Nov 7 2005                  │
  │                                                                          │
  │     DDNAME               DR2F                                            │
  │     Open Intent          CNV,KEY,DFR,DIR,SEQ,SKP,IN,LSR SHRPOOL=0        │
  │     Dataset Name         CHABERT.IMS.TRADER.DB.DR2F.DATA                 │
  └──────────────────────────────────────────────────────────────────────────┘
      → IGG019AQ  QSAM GET Buffer Mgmt      3.09
 → NOSYMB    No Module Name                0.57





 MA     a                                                                03/004
```

Figure 11-18 Additional details (D06) for VSAM file from I09

11.1.7 I10 - IMS DL/I WAIT time by DL/I call

I10 identifies delay caused by wait conditions in specific DL/I calls. Each line can be expanded to the next level to show CPU distribution between each category (IMS, DATAMG, SVC, MVS, and so on). Figure 11-19 shows an expanded version of I10.

```
  File  View  Navigate  Help
I10: IMS WAIT Time by DL/I Call (4800)                    Row 00001 of 00018
Command ===> _____________________________________________ Scroll ===> CSR

Name       Description                                Percent of DLI Time * 10.00%
                                                        *....1....2....3....4....
00005      GHU-DBPCB01(2) TRADERI+1D7A                16.71
→ IMS        IMS Subsystem                            16.71
  → DFSKBDP0  DFSKBP00-IMS Batch Dispatching Routines 16.71

00001      GU-DBPCB02(3) TRADERI+213A                  3.93
→ DATAMG     Data Mgmt Processing                      2.84
  → DR2F       VSAM                                    1.42
→ SVC        SVC Routines                              2.21
→ IMS        IMS Subsystem                             0.29

00002      GHU-DBPCB01(2) TRADERI+1D7A                 3.48
→ DATAMG     Data Mgmt Processing                      2.69
  → DR1F       VSAM                                    1.34
→ SVC        SVC Routines                              1.83
→ IMS        IMS Subsystem                             0.29

00003      GHU-DBPCB01(2) TRADERI+1E4A                 3.44

MA  a                                                          20/004
```

Figure 11-19 I10: IMS WAIT Time by DL/I Call (expanded)

11.1.8 I11 - IMS DL/I activity by PSB

I11 shows how time is consumed in IMS-dependent regions. All time is reported regardless of whether the time is CPU usage or wait. This report shows activity by IMS PSB, as shown in Figure 11-20.

Only time observed while an IMS PSB was active is reported. Time is identified both within the processing of DL/I calls and outside of DL/I call processing.

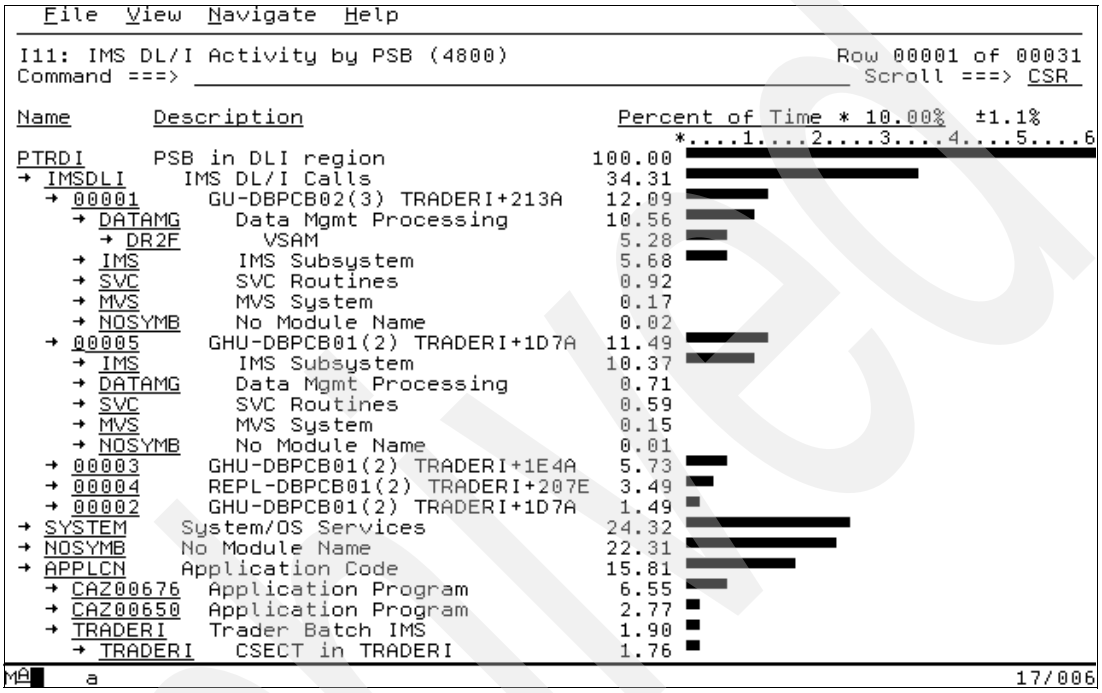


Figure 11-20 I11: IMS DL/I Activity by PSB (expanded)

Note: Because the Descriptive Program Attribution table has been updated, the TRADERI load module has *Trader Batch IMS* as a description. DPA is described in detail in the APA Descriptive Program Attribution table.

11.1.9 I13 - IMS DL/I activity by DL/I call

I13 shows how time is consumed in IMS-dependent regions by specific DL/I calls. All time is reported regardless of whether the time is CPU usage or wait (time observed outside of DL/I call processing is excluded), as shown in Figure 11-21.

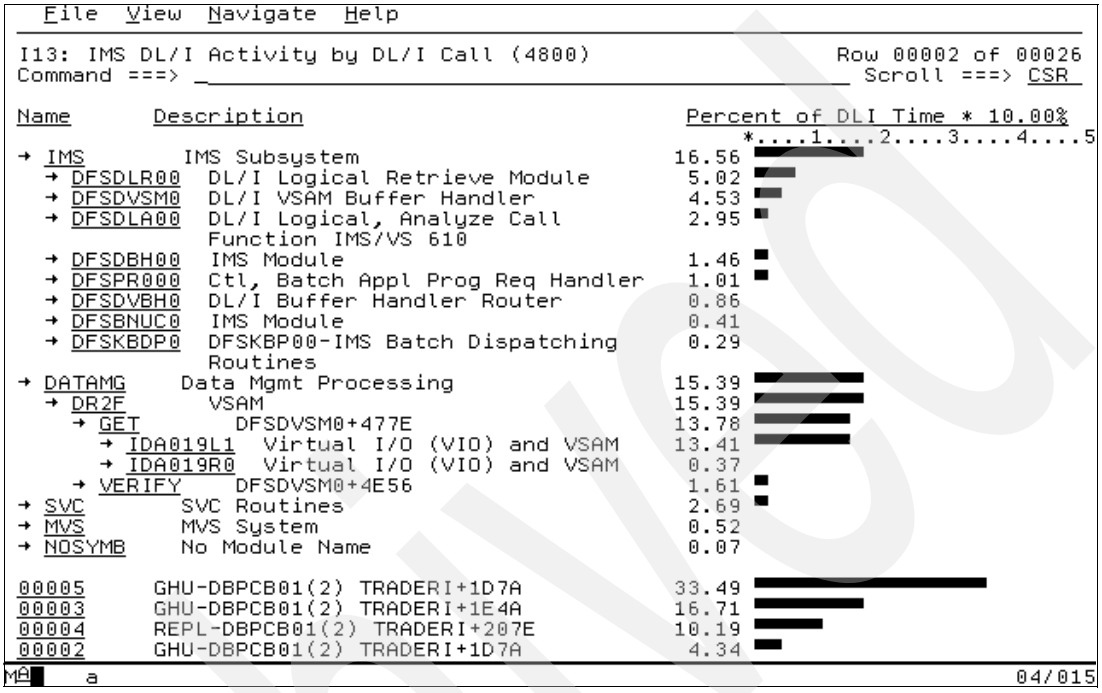


Figure 11-21 I13: IMS DL/I Activity by DL/I Call (expanded)

11.1.10 I14 - IMS PSB/PCB attributes

I14 is primarily used when analyzing measurements using printed reports. Use it to look up detailed information about PSBs (and their PCBs) referred to by detail lines in other reports.

Note: When analyzing data in interactive mode, we can look up this information by pressing Enter (or entering the ++ line command) on any detail line that refers to the PSB.

A sample report is shown in Figure 11-22.

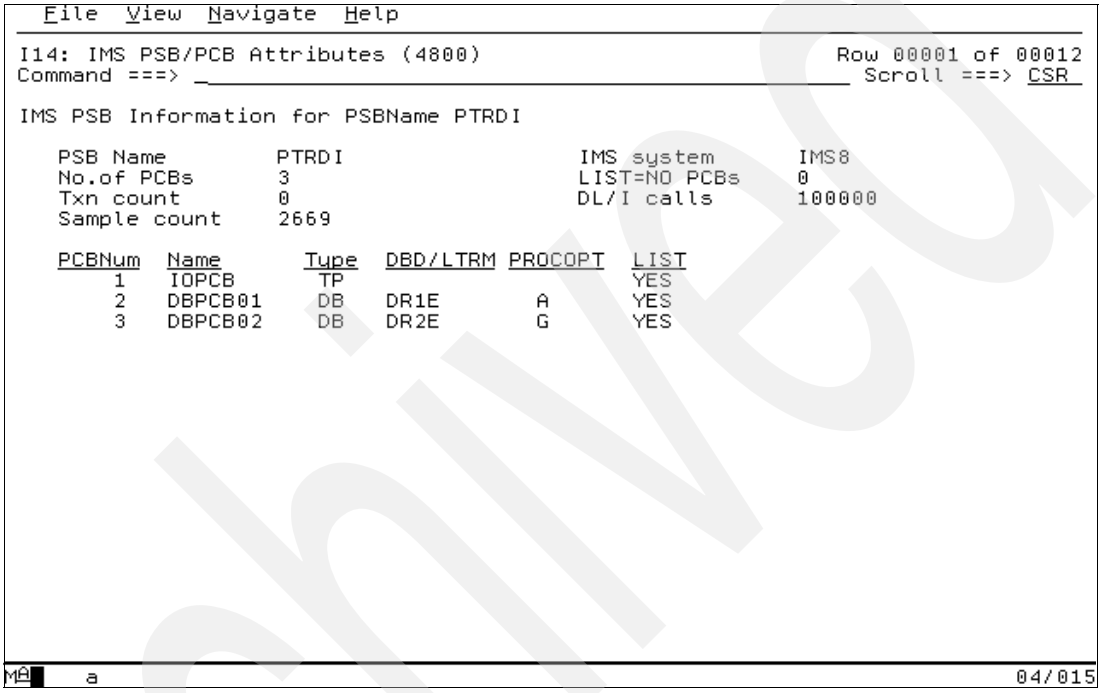


Figure 11-22 I14: IMS PSB/PCB Attributes

11.1.11 I15 - IMS DL/I call attributes

I15 is primarily used when analyzing measurements using printed reports. Use it to look up detailed information about DL/I calls referred to by detail lines in other reports.

Note: When analyzing data in interactive mode, we can look up this information by pressing Enter (or entering the ++ line command) on any detail line that refers to the DL/I call.

A sample report is shown in Figure 11-23.

```
  File  View  Navigate  Help
I15: IMS DL/I Call Attributes (4800)                    Row 00001 of 00058
Command ===> _____________________________________________ Scroll ===> CSR

DL/I Call Id 00001

   Function code   GU                    PSB Name        PTRDI
   PCB Name        DBPCB02               IMS Id-Region   IMS8-TRADERI
   PCB Number      3                     Call type       CBLTDLI
   CSECT/module    TRADERI in TRADERI    Offset of call  0000213A
   Sample count    941                   Call count      28,980
   DLI CPU time    0.77                  Service time    1.05

   SSA/FSA         01 COMPANY *(COMPKEY = ...)

DL/I Call Id 00002

   Function code   GHU                   PSB Name        PTRDI
   PCB Name        DBPCB01               IMS Id-Region   IMS8-TRADERI
   PCB Number      2                     Call type       CBLTDLI
   CSECT/module    TRADERI in TRADERI    Offset of call  00001D7A
   Sample count    116                   Call count      1
   DLI CPU time    0.01                  Service time    0.11

   SSA/FSA         01 CUSTOMER*(CUSTOKEY= ...)
                   02 COMPANY *(COMPAKEY= ...)

DL/I Call Id 00003

   Function code   GHU                   PSB Name        PTRDI
   PCB Name        DBPCB01               IMS Id-Region   IMS8-TRADERI
MA    a                                                          04/015
```

Figure 11-23 I15: IMS DL/I Calls Attributes

11.1.12 I18 - IMS CPU/Service time by DL/I calls

I18 shows information about exact CPU times and service times for DL/I calls. Each report line shows information pertaining to one DL/I call, as shown in Figure 11-24.

Note: The IMS+ feature must have been enabled when the measurement was performed.

```
  File  View  Navigate  Help
I18: IMS CPU/Service Time by DL/I Call (4800)                Row 00001 of 00005
Command ===> _____________________________________________ Scroll ===> CSR

                                                  ---- DL/I Processing Time ----
Call   Func  PCB Name  Location          Count  Svc time/Prcnt  CPU Time/Prcnt
0005   GHU   DBPCB01   TRADERI+1D7A      28979   1.124  33.5%    0.451  22.6%
0001   GU    DBPCB02   TRADERI+213A      28980   1.055  31.4%    0.770  38.6%
0003   GHU   DBPCB01   TRADERI+1E4A      28980   0.637  18.9%    0.423  21.2%
0004   REPL  DBPCB01   TRADERI+207E      28980   0.418  12.4%    0.334  16.7%
0002   GHU   DBPCB01   TRADERI+1D7A          1   0.115   3.4%    0.012   0.6%
MA  a                                                                  04/015
```

Figure 11-24 I18: IMS CPU/Service Time by DL/I Call

11.1.13 I19 - IMS CPU/Service time by PSB

I19 shows information about exact CPU times and service times for DL/I calls by PSB. Each report line shows information pertaining to one DL/I call, as shown in Figure 11-25.

Note: The IMS+ feature must have been enabled when the measurement was performed.

```
  File  View  Navigate  Help
I19: IMS CPU/Service Time by PSB (4800)                    Row 00001 of 00001
Command ===> _                                              Scroll ===> CSR

                 Txn      DL/I       ----- DL/I Processing Time ------
PSB Name        Count     Count      Svc time/Percent CPU Time/Percent

PTRDI               0    115920         3.353  100.0%     1.992 100.0%

MA  a                                                              04/015
```

Figure 11-25 I19: IMS CPU/Service Time by PSB

11.1.14 I21 - IMS CPU/Service time by PCB

I21 shows information about exact CPU times and service times for DL/I calls by individual PCB. Each report line shows information pertaining to one DL/I call, as shown in Figure 11-26.

Note: The IMS+ feature must have been enabled when the measurement was performed.

```
  File  View  Navigate  Help
I21: IMS CPU/Service Time by PCB (4800)                    Invalid line command
Command ===> _                                              Scroll ===> CSR

                                        -------- DL/I Call Counts --------
PSB Name   PCB Name   PCBNum   Count    Svc time/Percent CPU Time/Percent

PTRDI      DBPCB01        2   86940        2.297  68.5%     1.221  61.2%
PTRDI      DBPCB02        3   28980        1.055  31.4%     0.770  38.6%

MA  a                                                                04/015
```

Figure 11-26 I21: IMS CPU/Service Time by PCB

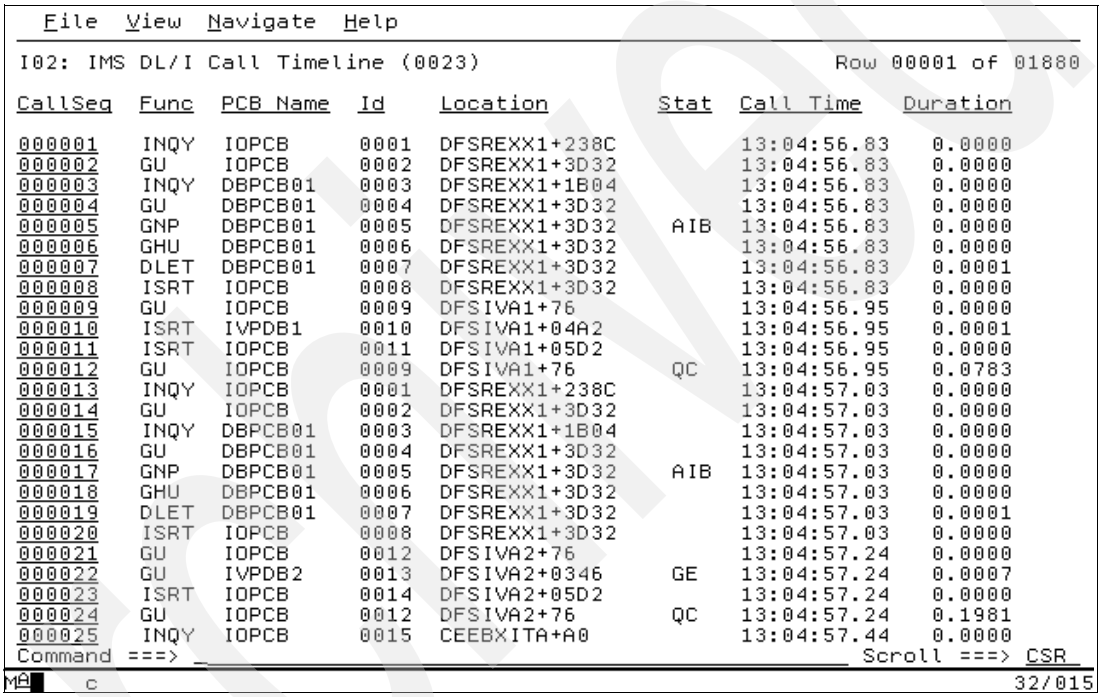
11.1.15 Remarks on reports using a non-TRADER IMS application

Important: We now explain the remaining reports (I02, I03, I04, I06, I09, I12, I16, I17, and I20) by using an online IMS application that is not TRADER.

11.1.16 I02: IMS DL/I call timeline

I02 shows the chronology of DL/I calls observed during the measurement interval. Each line shows information about one executed DL/I call, as shown in Figure 11-27.

Note: The IMS+ feature must have been enabled when the measurement was performed.



File View Navigate Help

I02: IMS DL/I Call Timeline (0023) Row 00001 of 01880

CallSeq	Func	PCB Name	Id	Location	Stat	Call Time	Duration
000001	INQY	IOPCB	0001	DFSREXX1+238C		13:04:56.83	0.0000
000002	GU	IOPCB	0002	DFSREXX1+3D32		13:04:56.83	0.0000
000003	INQY	DBPCB01	0003	DFSREXX1+1B04		13:04:56.83	0.0000
000004	GU	DBPCB01	0004	DFSREXX1+3D32		13:04:56.83	0.0000
000005	GNP	DBPCB01	0005	DFSREXX1+3D32	AIB	13:04:56.83	0.0000
000006	GHU	DBPCB01	0006	DFSREXX1+3D32		13:04:56.83	0.0000
000007	DLET	DBPCB01	0007	DFSREXX1+3D32		13:04:56.83	0.0001
000008	ISRT	IOPCB	0008	DFSREXX1+3D32		13:04:56.83	0.0000
000009	GU	IOPCB	0009	DFSIVA1+76		13:04:56.95	0.0000
000010	ISRT	IVPDB1	0010	DFSIVA1+04A2		13:04:56.95	0.0001
000011	ISRT	IOPCB	0011	DFSIVA1+05D2		13:04:56.95	0.0000
000012	GU	IOPCB	0009	DFSIVA1+76	QC	13:04:56.95	0.0783
000013	INQY	IOPCB	0001	DFSREXX1+238C		13:04:57.03	0.0000
000014	GU	IOPCB	0002	DFSREXX1+3D32		13:04:57.03	0.0000
000015	INQY	DBPCB01	0003	DFSREXX1+1B04		13:04:57.03	0.0000
000016	GU	DBPCB01	0004	DFSREXX1+3D32		13:04:57.03	0.0000
000017	GNP	DBPCB01	0005	DFSREXX1+3D32	AIB	13:04:57.03	0.0000
000018	GHU	DBPCB01	0006	DFSREXX1+3D32		13:04:57.03	0.0000
000019	DLET	DBPCB01	0007	DFSREXX1+3D32		13:04:57.03	0.0001
000020	ISRT	IOPCB	0008	DFSREXX1+3D32		13:04:57.03	0.0000
000021	GU	IOPCB	0012	DFSIVA2+76		13:04:57.24	0.0000
000022	GU	IVPDB2	0013	DFSIVA2+0346	GE	13:04:57.24	0.0007
000023	ISRT	IOPCB	0014	DFSIVA2+05D2		13:04:57.24	0.0000
000024	GU	IOPCB	0012	DFSIVA2+76	QC	13:04:57.24	0.1981
000025	INQY	IOPCB	0015	CEEBXITA+A0		13:04:57.44	0.0000

Command ===> Scroll ===> CSR

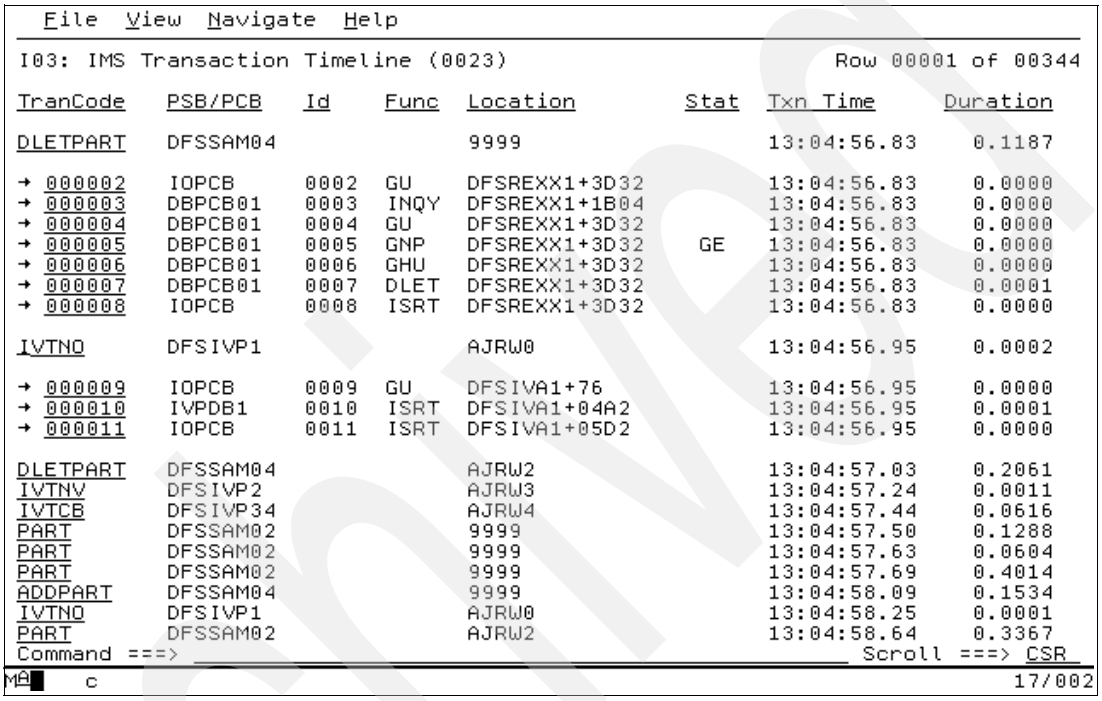
MA c 32/015

Figure 11-27 I02: IMS DL/I Call Timeline

11.1.17 I03: IMS transaction timeline

I03 shows the chronology of IMS transactions observed during the measurement interval. Each line shows information about one executed IMS transaction and can be expanded to show the sequence of DL/I calls executed by the transaction (as shown in Figure 11-28) for IMS transactions DLETPART and IVTNO.

Note: The IMS+ feature must have been enabled when the measurement was performed.



```
  File  View  Navigate  Help
I03: IMS Transaction Timeline (0023)                    Row 00001 of 00344

TranCode   PSB/PCB   Id    Func  Location          Stat  Txn Time      Duration

DLETPART   DFSSAM04                9999                  13:04:56.83    0.1187

→ 000002   IOPCB     0002  GU    DFSREXX1+3D32         13:04:56.83    0.0000
→ 000003   DBPCB01   0003  INQY  DFSREXX1+1B04         13:04:56.83    0.0000
→ 000004   DBPCB01   0004  GU    DFSREXX1+3D32         13:04:56.83    0.0000
→ 000005   DBPCB01   0005  GNP   DFSREXX1+3D32   GE    13:04:56.83    0.0000
→ 000006   DBPCB01   0006  GHU   DFSREXX1+3D32         13:04:56.83    0.0000
→ 000007   DBPCB01   0007  DLET  DFSREXX1+3D32         13:04:56.83    0.0001
→ 000008   IOPCB     0008  ISRT  DFSREXX1+3D32         13:04:56.83    0.0000

IVTNO      DFSIVP1                AJRW0                 13:04:56.95    0.0002

→ 000009   IOPCB     0009  GU    DFSIVA1+76            13:04:56.95    0.0000
→ 000010   IVPDB1    0010  ISRT  DFSIVA1+04A2          13:04:56.95    0.0001
→ 000011   IOPCB     0011  ISRT  DFSIVA1+05D2          13:04:56.95    0.0000

DLETPART   DFSSAM04               AJRW2                 13:04:57.03    0.2061
IVTNV      DFSIVP2                AJRW3                 13:04:57.24    0.0011
IVTCB      DFSIVP34               AJRW4                 13:04:57.44    0.0616
PART       DFSSAM02               9999                  13:04:57.50    0.1288
PART       DFSSAM02               9999                  13:04:57.63    0.0604
PART       DFSSAM02               9999                  13:04:57.69    0.4014
ADDPART    DFSSAM04               9999                  13:04:58.09    0.1534
IVTNO      DFSIVP1                AJRW0                 13:04:58.25    0.0001
PART       DFSSAM02               AJRW2                 13:04:58.64    0.3367
Command ===>                                                Scroll ===> CSR
MA  c                                                                  17/002
```

Figure 11-28 I03: IMS Transaction Timeline (expanded)

11.1.18 I04: IMS transaction activity timeline

I04 shows, for each IMS transaction, how execution of that transaction was distributed over the measurement interval.

A graph, in bar chart format, is displayed for each observed IMS transaction code. The horizontal axis represents the measurement interval, which spans 50 columns (subinterval). The vertical graph shows the approximate percentage of time during the subinterval that execution of the IMS transaction took place, as shown in Figure 11-29.

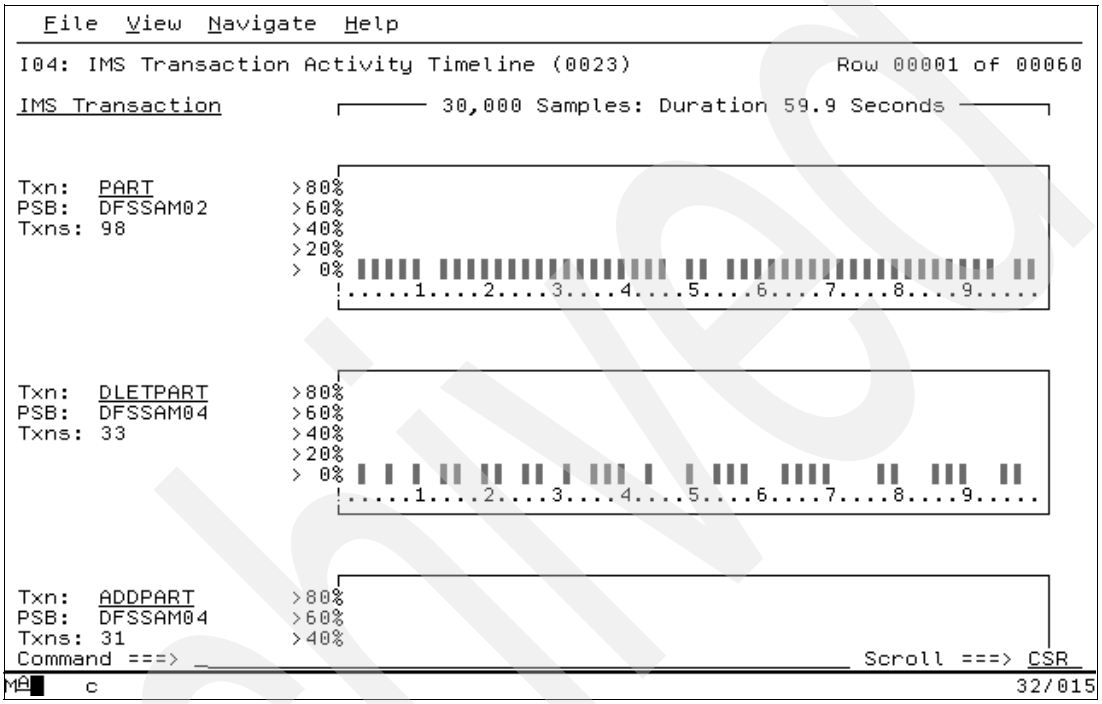


Figure 11-29 I04: IMS Transaction Activity Timeline

When analyzing data in interactive mode, we can look up this information by pressing Enter (or entering the ++ line command) on any transaction name to get PSB detail information (same as report I14) (as shown in Figure 11-30) for PART transaction.

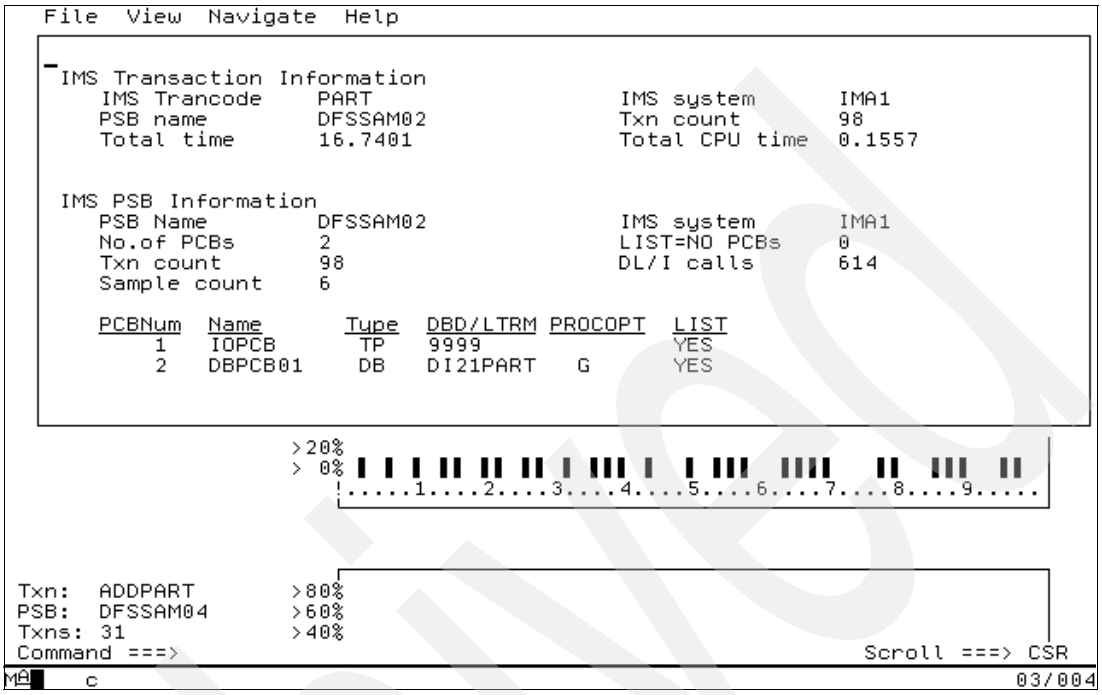


Figure 11-30 Detail screen from I04

11.1.19 I06: IMS DL/I CPU usage by transaction

I06 shows the distribution of CPU usage aggregated by IMS transaction. Both IMS and non-IMS CPU usage is reported. IMS CPU usage measured during the execution of DL/I calls is reported under detail lines, which identify each of the DL/I calls. Non-DL/I call CPU usage is reported as application code or system routines, as shown in Figure 11-31.

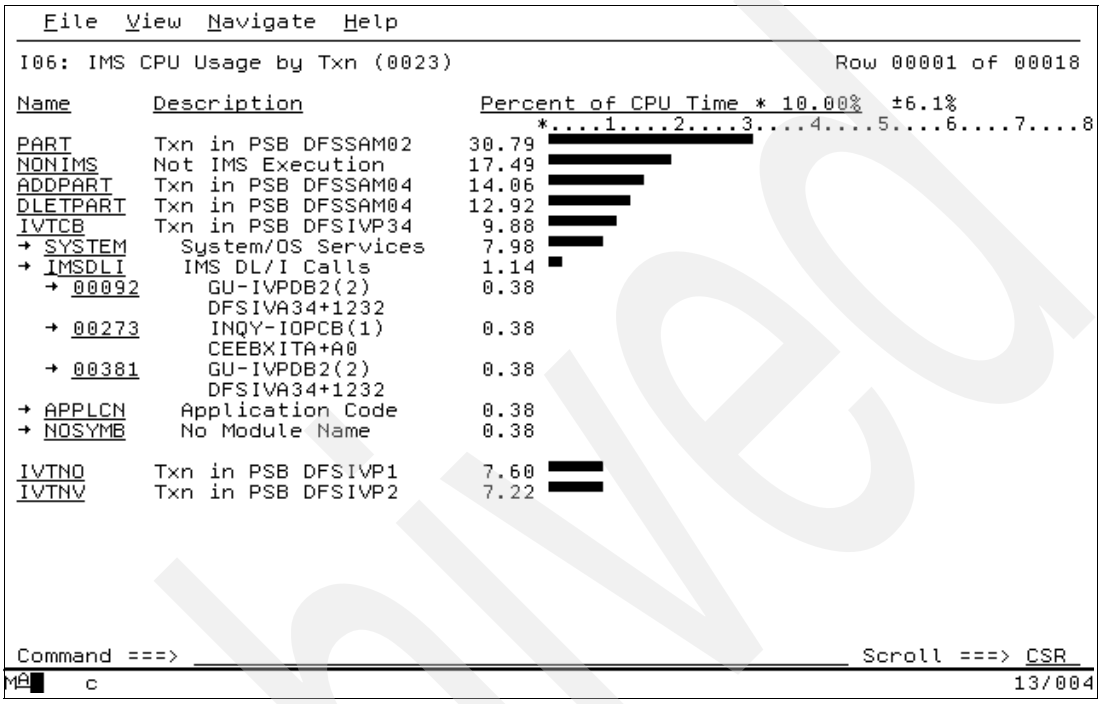


Figure 11-31 I06: IMS CPU Usage by Transaction (expanded)

11.1.20 I09: IMS DL/I WAIT time by transaction

I09 identifies any delays caused by wait conditions in IMS regions. It shows wait time by IMS transaction. Only wait time observed when an IMS PSB is active is reported. Wait time is identified both within the processing of DL/I calls and outside of DL/I call processing, as shown in Figure 11-32.

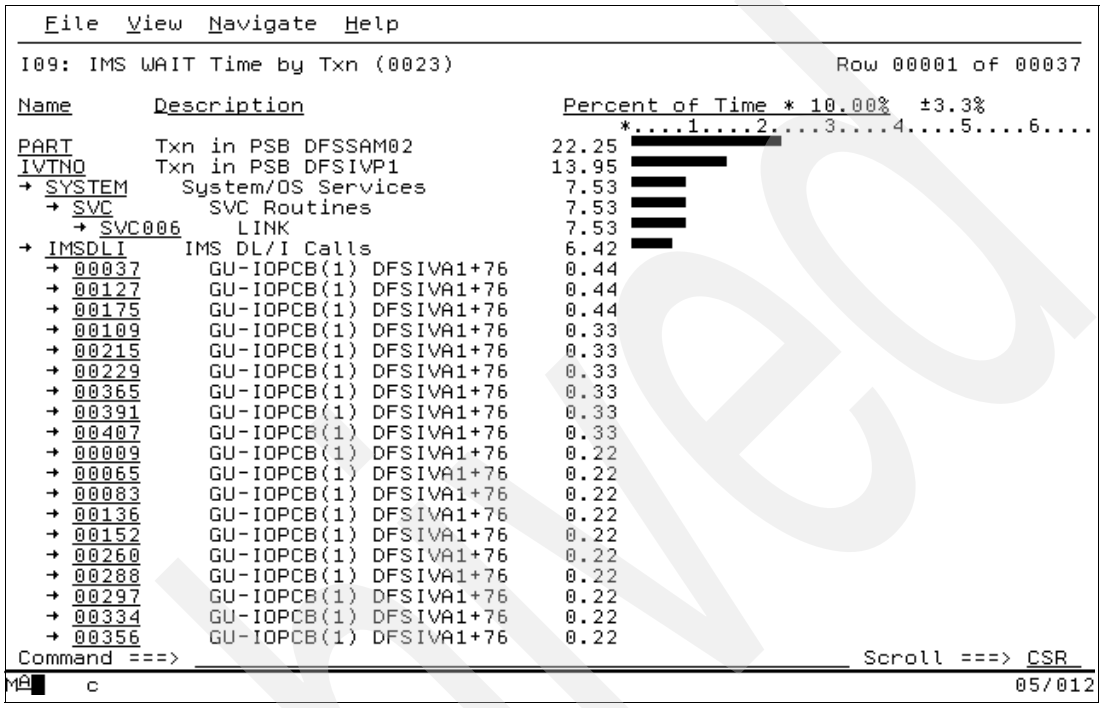


Figure 11-32 I09: IMS WAIT Time by Transaction (expanded)

11.1.21 I12: IMS DL/I activity by transaction

In Figure 11-33, I12 shows how time is consumed by IMS transaction. All time is reported regardless of whether the time is CPU usage or wait. Time is identified both within the processing of DL/I calls and outside of DL/I call processing, but only when an IMS PSB is active.

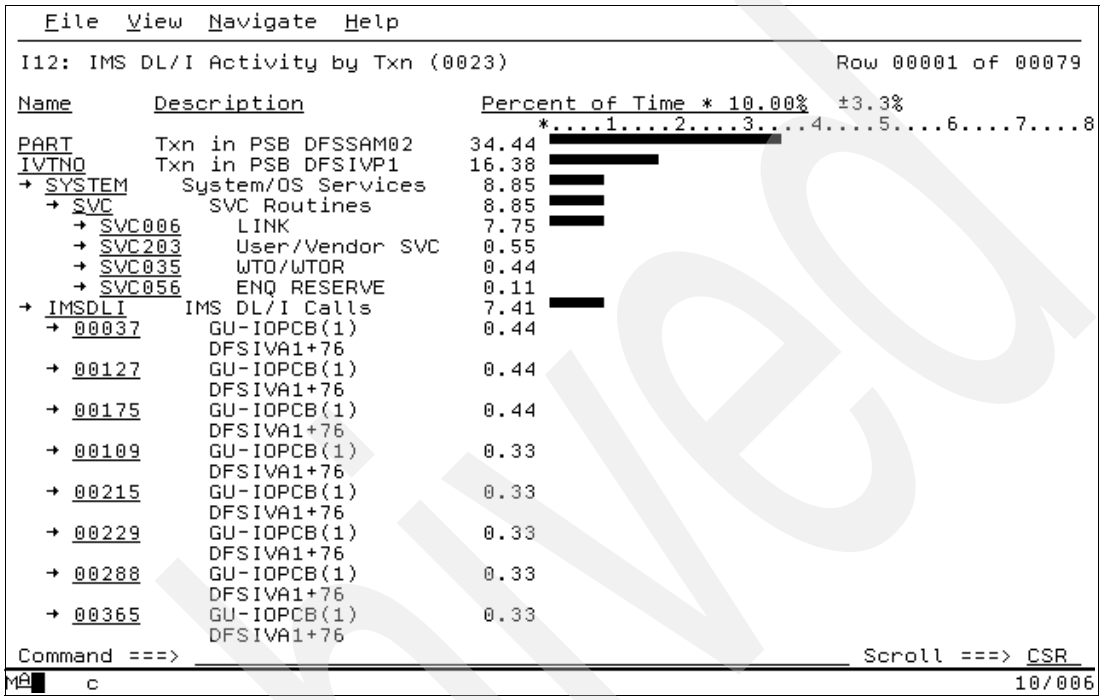


Figure 11-33 I12: IMS DL/I Activity by Transaction (expanded)

11.1.22 I16: IMS transaction service time

I16 shows information about IMS transaction service times, as shown in Figure 11-34.

Note: This report is meaningful only when measuring an IMS-dependent region in which transactions are executed.

The IMS+ feature must have been enabled when the measurement was performed.

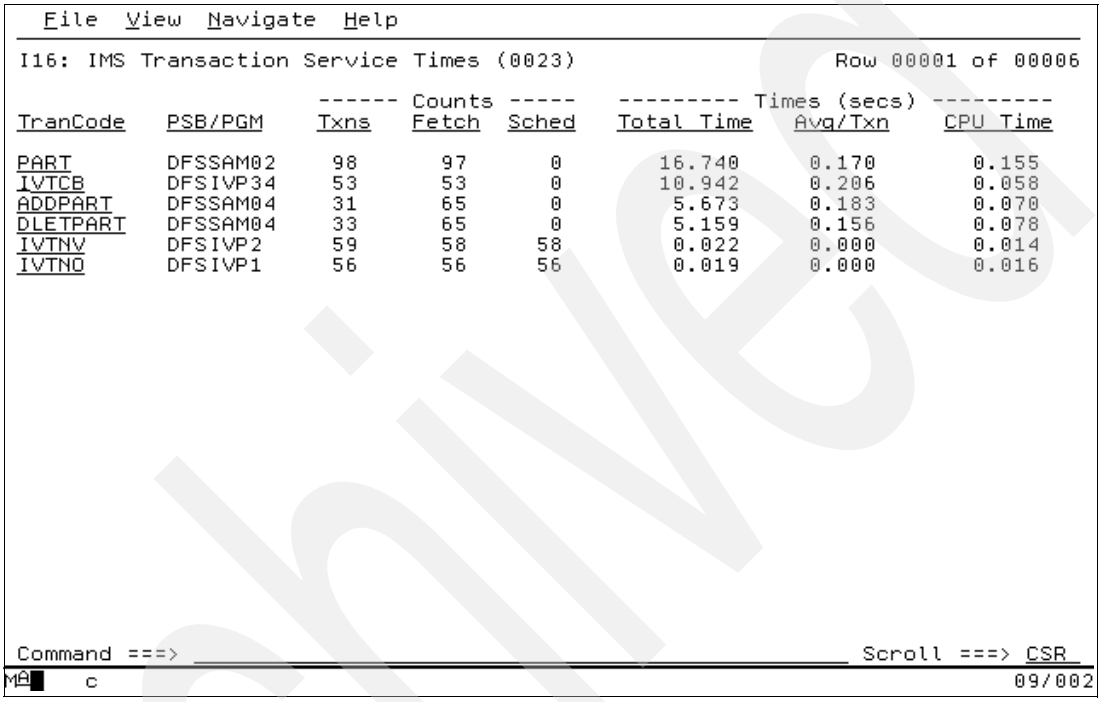


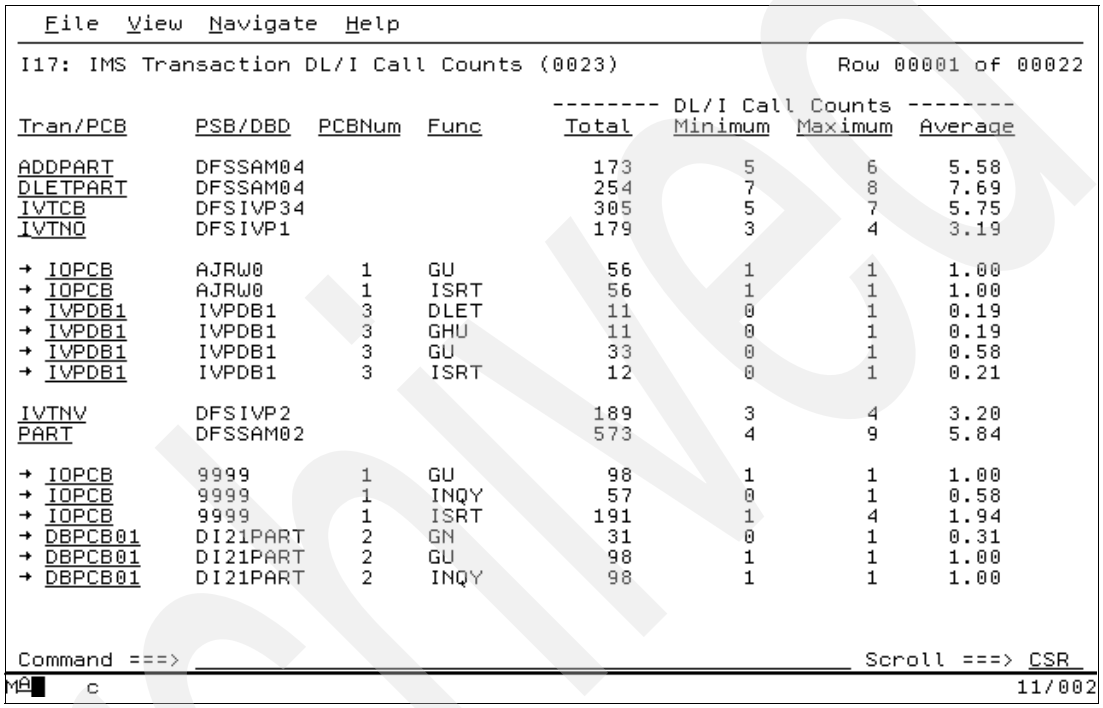
Figure 11-34 I16: IMS Transaction Service Time

11.1.23 I17: IMS transaction DL/I Call counts

I17 shows information about the number of DL/I calls issued by each of the measured IMS transactions, as shown in Figure 11-35.

Note: This report is meaningful only when measuring an IMS-dependent region in which transactions are executed.

The IMS+ feature must have been enabled when the measurement was performed.



File View Navigate Help

I17: IMS Transaction DL/I Call Counts (0023) Row 00001 of 00022

Tran/PCB	PSB/DBD	PCBNum	Func	DL/I Call Counts Total	Minimum	Maximum	Average
ADDPART	DFSSAM04			173	5	6	5.58
DLETPART	DFSSAM04			254	7	8	7.69
IVTCB	DFSIVP34			305	5	7	5.75
IVTNO	DFSIVP1			179	3	4	3.19
→ IOPCB	AJRW0	1	GU	56	1	1	1.00
→ IOPCB	AJRW0	1	ISRT	56	1	1	1.00
→ IVPDB1	IVPDB1	3	DLET	11	0	1	0.19
→ IVPDB1	IVPDB1	3	GHU	11	0	1	0.19
→ IVPDB1	IVPDB1	3	GU	33	0	1	0.58
→ IVPDB1	IVPDB1	3	ISRT	12	0	1	0.21
IVTNV	DFSIVP2			189	3	4	3.20
PART	DFSSAM02			573	4	9	5.84
→ IOPCB	9999	1	GU	98	1	1	1.00
→ IOPCB	9999	1	INQY	57	0	1	0.58
→ IOPCB	9999	1	ISRT	191	1	4	1.94
→ DBPCB01	DI21PART	2	GN	31	0	1	0.31
→ DBPCB01	DI21PART	2	GU	98	1	1	1.00
→ DBPCB01	DI21PART	2	INQY	98	1	1	1.00

Command ===> Scroll ===> CSR

MA c 11/002

Figure 11-35 I17: IMS Transaction DL/I Call counts (expanded)

11.2 CICS measurement

To illustrate CICS reports, we use the online version of the TRADER application using CICS and VSAM files. This application is described in detail in Guided Tour of the Trader application.

We do not spend much effort on the things that are common to Application Performance Analyzer and that have been described in the previous chapter, but rather focus on the unique aspects of using Application Performance Analyzer with CICS.

11.2.1 Observation session request definition

We have to monitor the CICS region named CICSC23G with the following characteristics:

- We use a sampling rate of 1000 samples per second.
- We use a duration of 1 minute.
- The measurement file has to be retained for 30 days.
- Only MYT* transactions have to be recorded.
- CHABERT has to be notified at the end of the measurement.

By entering the NEW command line on the Observation List (R02) panel, Application Performance Analyzer displays the Schedule New Measurement (R03) panel. CICS data extractor has to be turned on to get CICS data.

When the CICS data extractor is active two distinct types of data are gathered by Application Performance Analyzer: session statistics and transaction measurement data.

The settings required to monitor the MYT* CICS transactions are shown in Figure 11-36 for panel 1 (Job Information), Figure 11-37 for panel 2 (Options), and Figure 11-38 for panel 5 (CICS Txns).

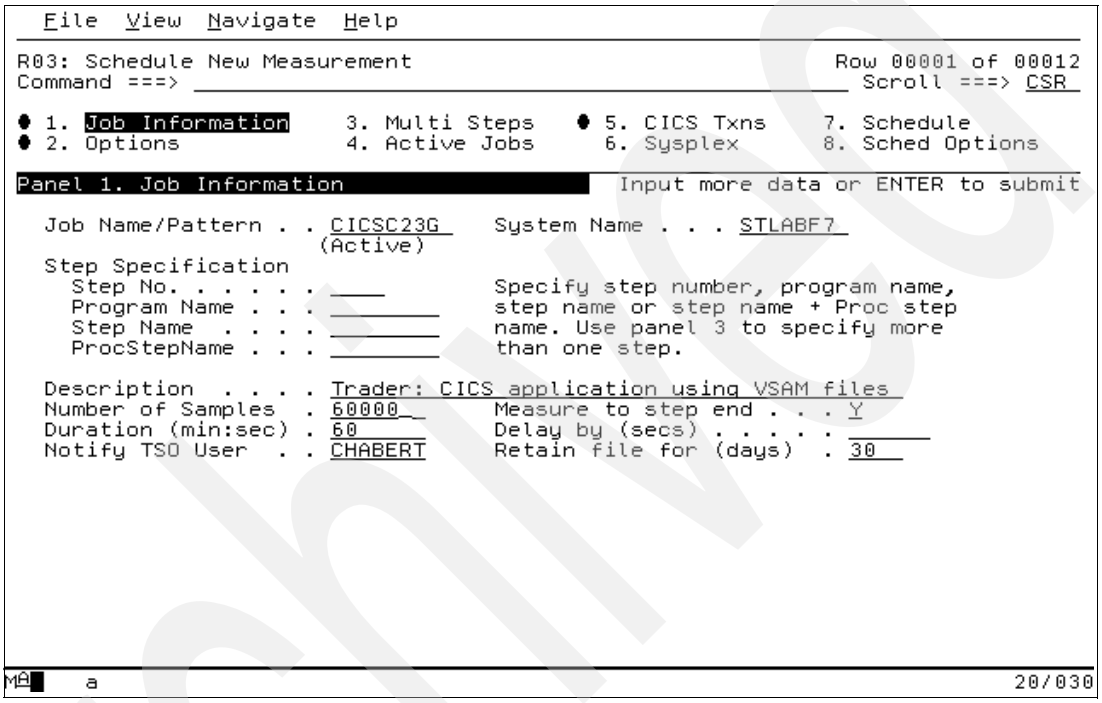


Figure 11-36 R03: Schedule New Measurement (panel 1: Job Information)

With panel 1 we specify:

- Job name
- Sample rate
- Retention period

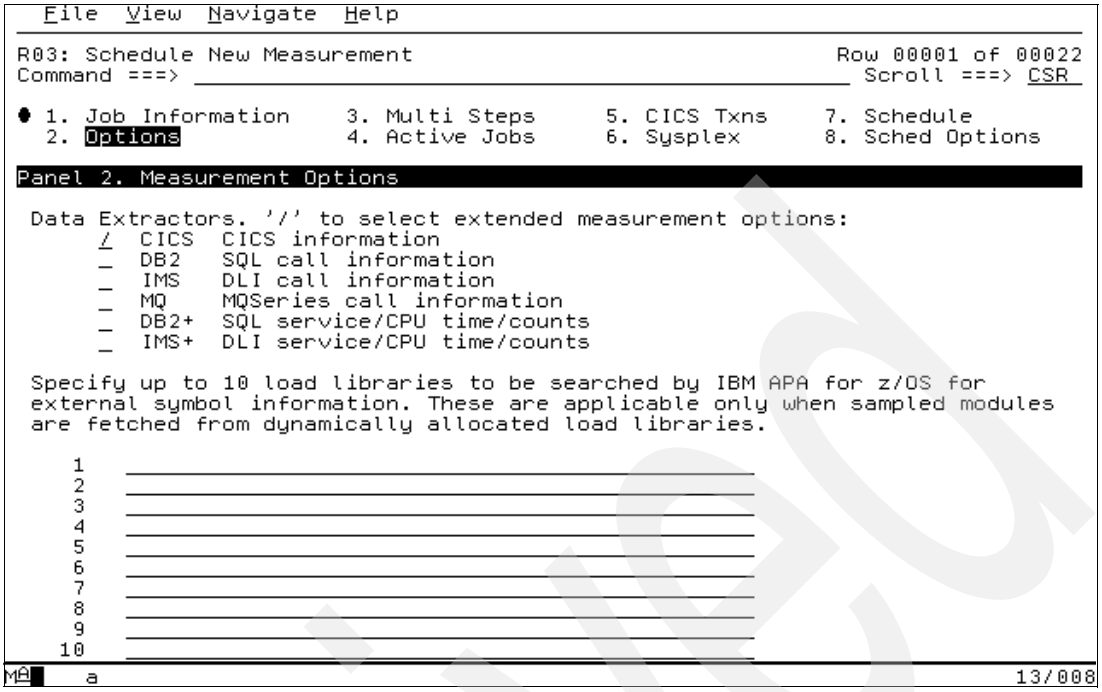


Figure 11-37 R03: Schedule New Measurement (panel 2: Measurement Options)

With panel 2 we turn on the CICS data extractor.

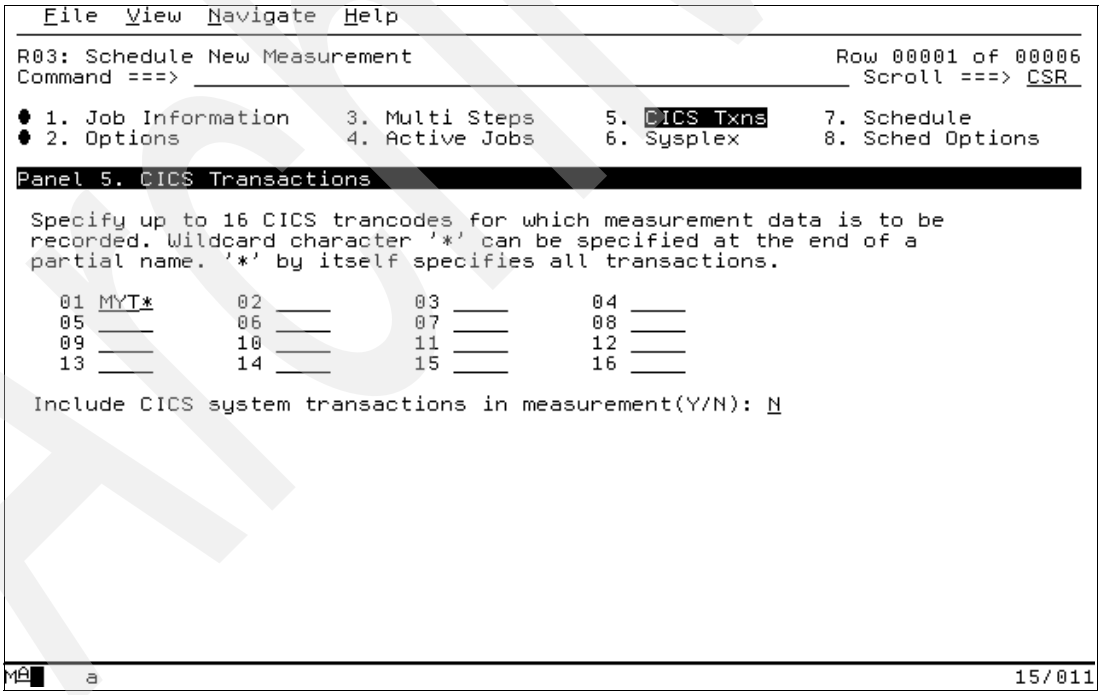


Figure 11-38 R03: Schedule New Measurement (panel 5: CICS Transaction)

And with panel 5 we select which transactions have to be recorded.

When we press Enter to submit the previous definition, Application Performance Analyzer displays the Observation List (R02) panel, with our new measurement definition at the top of the list. To check that we do not forget any parameter, we can use the ++ line command

(or just press Enter) on the request number 4807 (which is the done by Application Performance Analyzer for our new measurement request). Figure 11-39 and Figure 11-40 give the full definition of the request made to monitor our TRADER-CICS application.

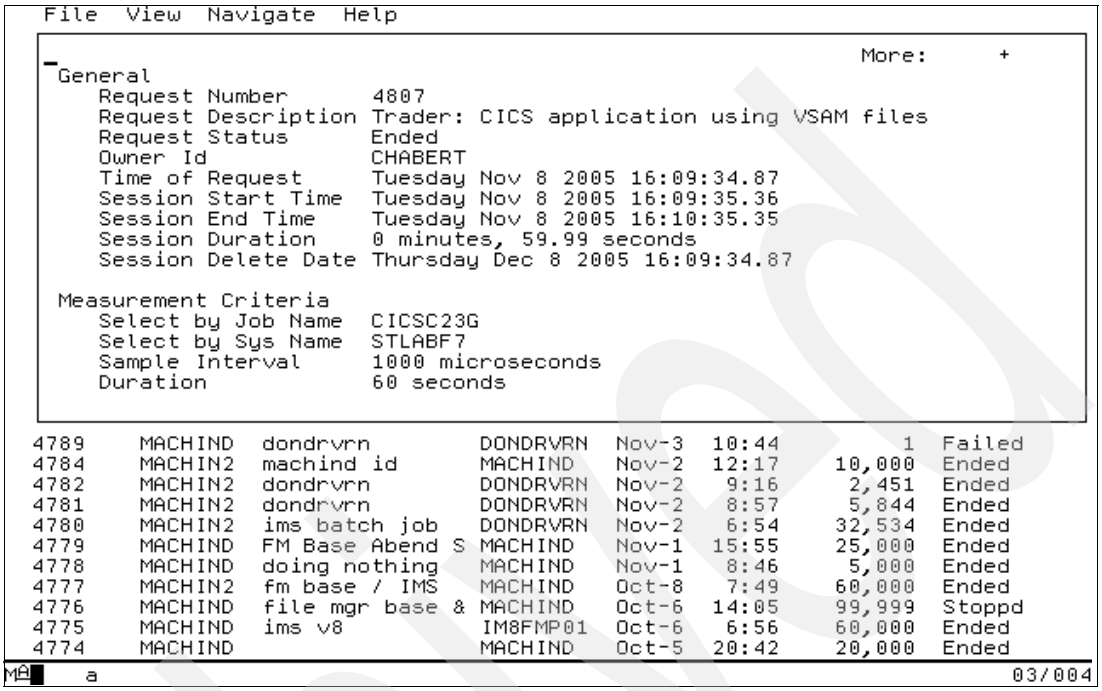


Figure 11-39 Detail about ReqNum 4807 (1 of 2)

The first part shows that the job name, sampling rate, and retention period are what we want. Using the PF8 key to scroll towards the bottom of the report, we can see the status of each data extractor.

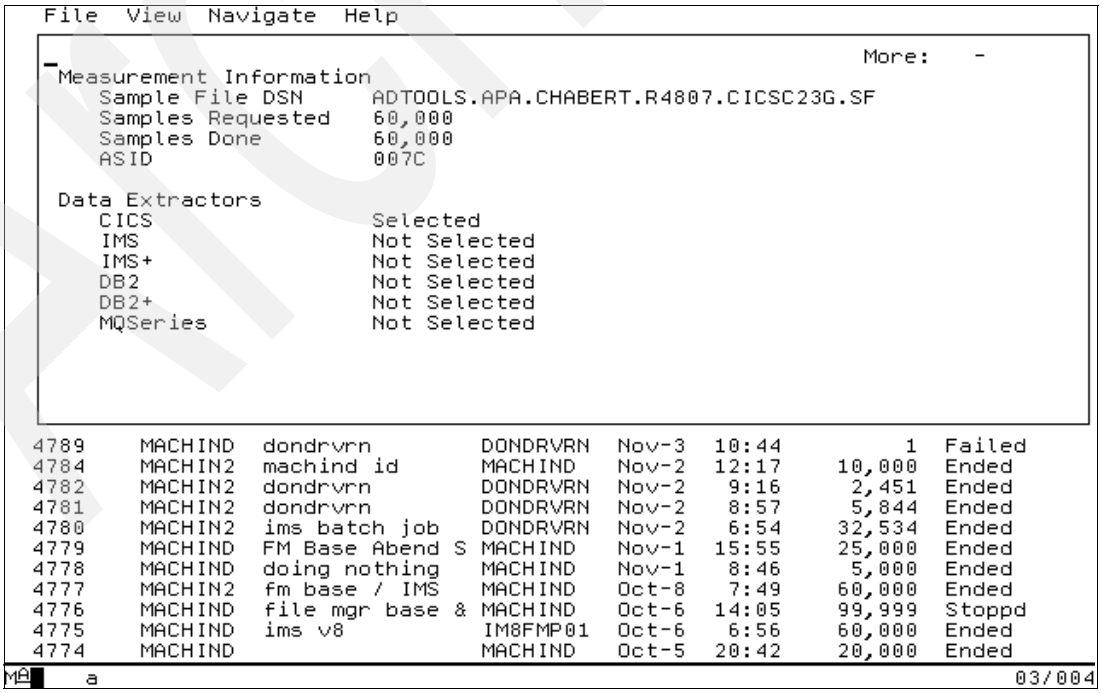


Figure 11-40 Detail about ReqNum 4807 (2 of 2)

We can see that CICS data extractor has been selected. From a CICS screen we start several MYTD transactions to run the application. After 1 minute (and 60 000 samples), Application Performance Analyzer has completed the sampling for CICS. Using the **R** line command on request number 4807 Application Performance Analyzer launches the associated performance reports.

Because `E CICS Measurement` is displayed in red (in gray on our figure), CICS reports are available. To get the list of available CICS reports, shown in Figure 11-41, just type `E` on the command line and press Enter.

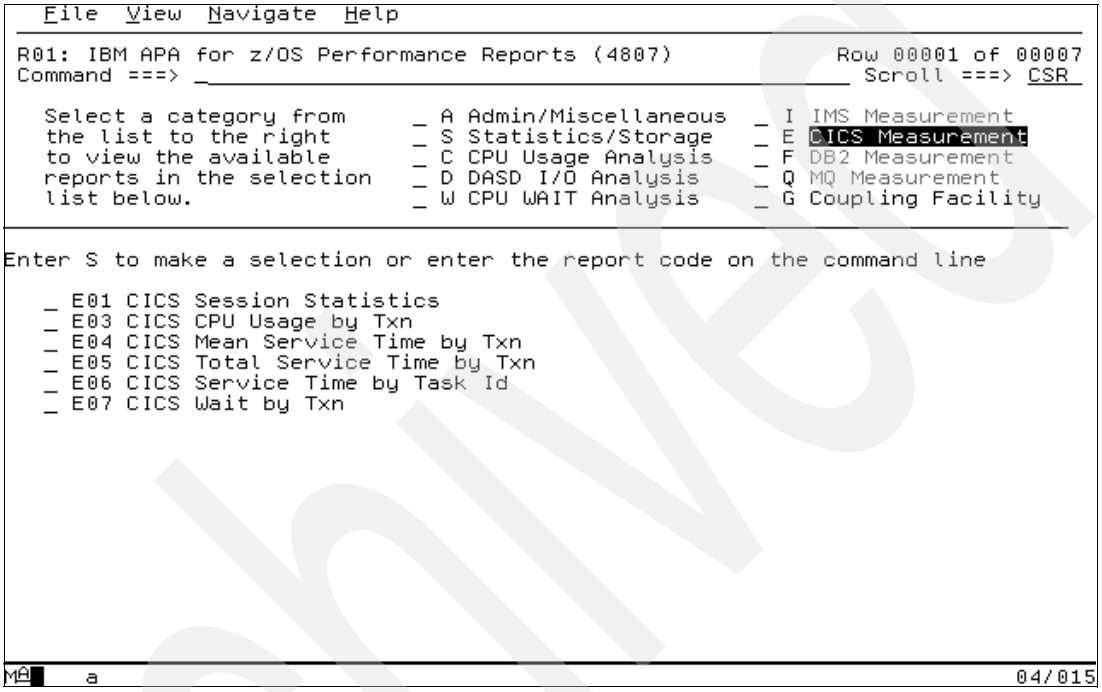


Figure 11-41 R01: Performance Report (CICS reports selected)

Six reports belong to the CICS Measurement category.

11.2.2 E01: CICS session statistics

E01 shows a summary of the CICS measurement data collected during the observation session.

> **Note:** If a reset of CICS statistics is done during the measurement interval, this can invalidate some of the values reported here.

E01 - CICS Session Statistics can be divided into several sections:

- Environmental information (CICS version and release)
- Transaction statistics:
  - First, Last Taskid of the observation session
  - Transaction rate
  - Peak active transactions
  - Mask task
- Mean transaction time

- Service statistics
  - Programs requests
  - Terminal messages
  - Getmains/Freemains
  - File, TSQueue, TDQueue, and Journal requests
- Exception or critical conditions
  - System and Transaction dumps
  - Storage violations
  - SOS conditions
  - MaxTask and ClassMaxTask conditions.
- Transaction count

This is shown in Figure 11-42 and Figure 11-43.

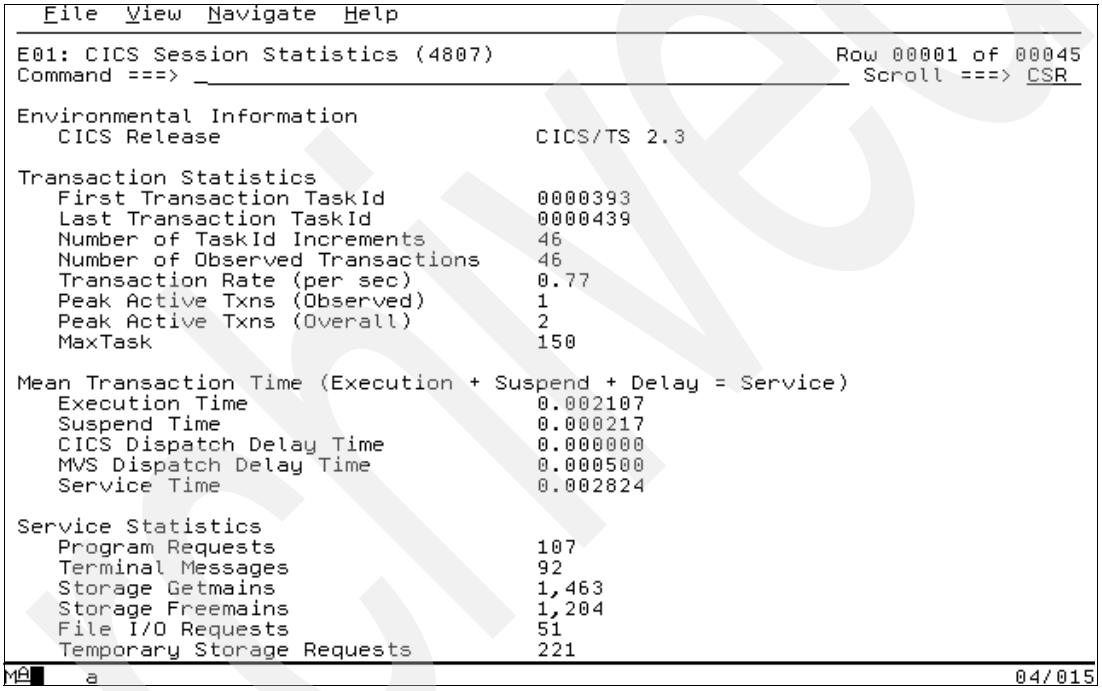


Figure 11-42 E01: CICS Session Statistics (1 of 2)

```
  File  View  Navigate  Help
 ------------------------------------------------------------------------------
 E01: CICS Session Statistics (4807)                          Row 00029 of 00045
 Command ===> _                                                 Scroll ===> CSR
    Transient Data Requests            0
    Journal Write Requests             0

 Exception or Critical Conditions
    System Dumps                       0
    System Dumps Suppressed            0
    Transaction Dumps                  0
    Transaction Dumps Suppressed       0
    Storage Violations                 0
    Short on Storage occurrences       0
    Times at MaxTask                   0
    Times at Class MaxTask             0

 Transaction Counts

        TranId    Count
         MYTD       46

MA    a                                                                  04/015
```

Figure 11-43 E01: CICS Session Statistics (2 of 2)

11.2.3 E03: CICS CPU usage by transaction

E03 shows how CPU consumption was distributed across the CICS transaction that was executing during the observation session. Expand a CICS transaction report line using the ++ line command (or press Enter) to see a further breakdown by programs, CICS commands, SQL requests, DL/I requests, and MQSeries commands, as shown in Figure 11-44.

```
  File  View  Navigate  Help
 ------------------------------------------------------------------------------
 E03: CICS CPU Usage by Transaction (4807)                    Row 00001 of 00019
 Command ===> ____________________________________________     Scroll ===> CSR

 Name    NTxns/Description                 Percent of CPU Time * 20.00%  ±9.6%
                                                *....1....2....3....4....5....6.
 MYTD       46                             83.48 ████████████████████
 → MYTRADMV  CICS Program                  22.01 ██████
 → MYTRADS   EXEC CICS                     13.76 ███
   → +223E     WRITEQ TS QUEUE(CEBR0000)    4.58 ▀
   → +1E6C     ASKTIME ABSTIME              3.66 ▀
   → +1B94     READ FILE(COMPFILE)          1.83
   → +1F6E     FORMATTIME                   1.83
   → +1AAE     REWRITE FILE(CUSTFILE)       0.91
   → +1862     READ FILE(CUSTFILE)          0.91

 → CICS      System Services               12.84 ███
 → MYTRADS   CICS Program                  10.09 ███
   → MYTRADS   Trader CICS                  9.17 ██
   → DFHSIP    CICS Services                0.91

 → MYTRADMV  EXEC CICS                      9.17 ██
 → EQADCXXT  EXEC CICS                      9.17 ██
 → EQADCCXR  EXEC CICS                      3.66 ▀
 → CEECCICS  EXEC CICS                      2.75 ▀

MA    a                                                                  06/045
```

Figure 11-44 E03: CICS CPU Usage by Transaction (expanded)

Note: The second level is the module name of the CICS program. If lines grouped under this line are subsystem (CICS, DB2, IMS, MQSeries) command lines, this field is displayed in red.

Note: Because the Descriptive Program Attribution table has been updated, the MYTRA* CSECTs have *Trader CICS* as a description. DPA is described in APA Descriptive Program Attribution table.

The report above has been expanded for the MYTRADS entries to show both the EXEC CICS command and the CSECT.

Because source program mapping data is available, by entering the **P** line command on any line where *csect name* or where an *EXEC CICS* is displayed, Application Performance Analyzer displays the source code for the statement, as shown in Figure 11-45, for the hexadecimal offset +223E in MYTRADS.

Note: The hexadecimal offset in the identified CICS program is the return address of the EXEC CICS command.

```
  File  View  Navigate  Help
P01: Source Program Attribution (4807)                  Row 00001 of 00022
Command ===> _____________________________________________ Scroll ===> CSR

LineNo Offset Count Source Statement

000889                      * To use:  MOVE 'text' TO COMMENT-FIELD  (max of 50 c
000890 0021A2                    PERFORM TIMESTAMP-ROUTINE
000891 0021BE                    PERFORM REPLACE-FIELDS
000892 0021DA                    PERFORM REMOVE-SPACES
000893                      *EXEC CICS WRITEQ TS
000894                      *      QUEUE('CEBR0000')
000895                      *      FROM(WRITEQ-WORDS)
000896                      *      LENGTH(LENGTH OF WRITEQ-WORDS)
000897                      *      END-EXEC.
000898 0021F6                    Move LENGTH OF WRITEQ-WORDS to dfhb0020
000899 0021FC     5 ■■■■■■      Call 'DFHEI1' using by content x'0a02e0000700004
000900                      -     '404040' by content 'CEBR0000' by reference WRIT
000901                            reference dfhb0020 end-call.
000902
000903                       WRITEQ-TS-EXIT.
000904 00224A                    EXIT.
000905                      ************************************************
000906                       DEBUG-CEBR SECTION.
000907                      * This section added to write debug messages without
000908                      * being done on them.
000909                      * To use:  MOVE 'text' TO DEBUG-WORDS  (max of 69 cha

MA■    a                                                              04/015
```

Figure 11-45 P01: Source Program Attribution (from E03)

Note: The Count value is the number of CPU usage sample.

11.2.4 E04: CICS mean service time by transaction

E04 shows an analysis of how time was spent by the CICS transactions that were executing during the observation session. An expanded line shows programs, CICS commands, SQL requests, DL/I requests, and MQSeries commands if they are part of the transaction, as shown in Figure 11-46.

```
  File  View  Navigate  Help
E04: CICS Mean Service Time by Txn (4807)                    Row 00001 of 00020
Command ===> _                                                Scroll ===> CSR

                                                 -------- Mean Time in Seconds ---------
Name     NTxns Description          Error  Execution + Suspend + Delay  =  Service

MYTD        46                     ±14.9%     0.001      0.000     0.000      0.002
→ CICS          System Services               0.000      0.000     0.000      0.000
→ MYTRADMV      CICS Program                  0.000      0.000     0.000      0.000
→ MYTRADS       EXEC CICS                     0.000      0.000     0.000      0.000
  → +1AAE       REWRITE FILE(CUSTFILE)        0.000      0.000     0.000      0.000
  → +223E       WRITEQ TS QUEUE(CEBR0000)     0.000      0.000     0.000      0.000
  → +1E6C       ASKTIME ABSTIME               0.000      0.000     0.000      0.000
  → +19C4       WRITE FILE(CUSTFILE)          0.000      0.000     0.000      0.000
  → +1B94       READ FILE(COMPFILE)           0.000      0.000     0.000      0.000
  → +1F6E       FORMATTIME                    0.000      0.000     0.000      0.000
  → +1862       READ FILE(CUSTFILE)           0.000      0.000     0.000      0.000

→ MYTRADS       CICS Program                  0.000      0.000     0.000      0.000
  → MYTRADS     Trader CICS                   0.000      0.000     0.000      0.000
  → DFHSIP      CICS Services                 0.000      0.000     0.000      0.000

→ MYTRADMV      EXEC CICS                     0.000      0.000     0.000      0.000
→ EQADCXXT      EXEC CICS                     0.000      0.000     0.000      0.000
→ EQADCCXR      EXEC CICS                     0.000      0.000     0.000      0.000
→ CEECCICS      EXEC CICS                     0.000      0.000     0.000      0.000

MA  a                                                                      04/015
```

Figure 11-46 E04: CICS Mean Service Time by Transaction (expanded)

Using the ++ line command (or by pressing Enter) on any line where *csect name* or where an *EXEC CICS* is displayed, Application Performance Analyzer displays the detail information shown in Figure 11-47 and Figure 11-48 for the hexadecimal offset +223E in the MYTRADS.

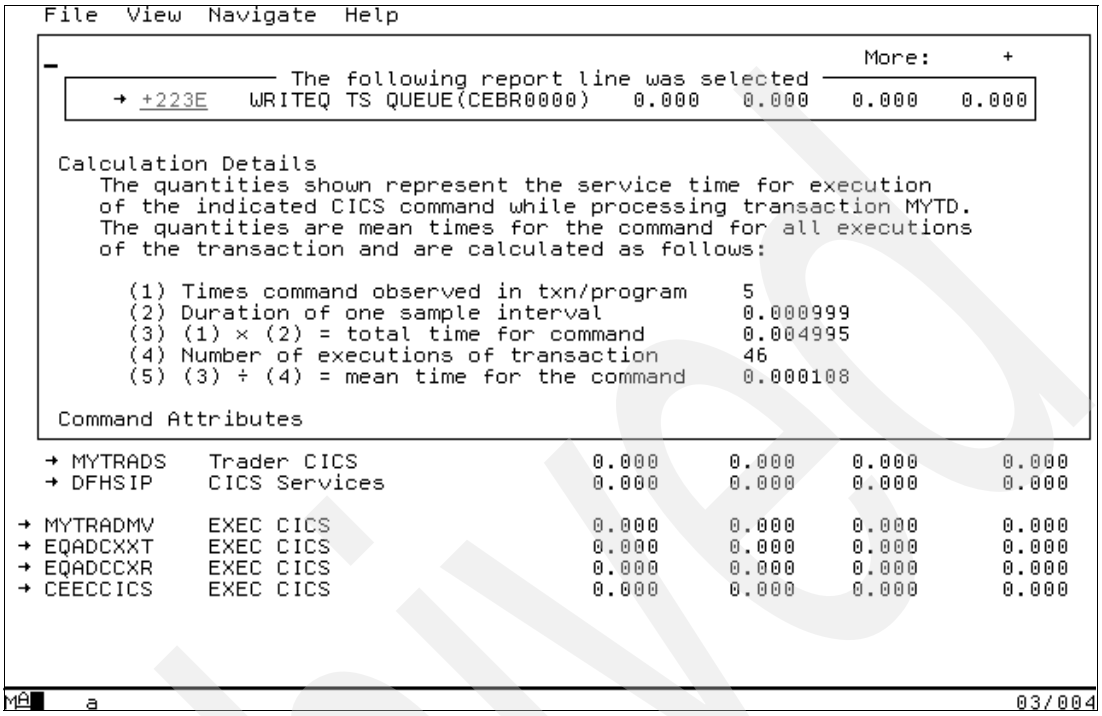


Figure 11-47 E04: Detail window for offset +223E in MYTRADS (1 of 2)

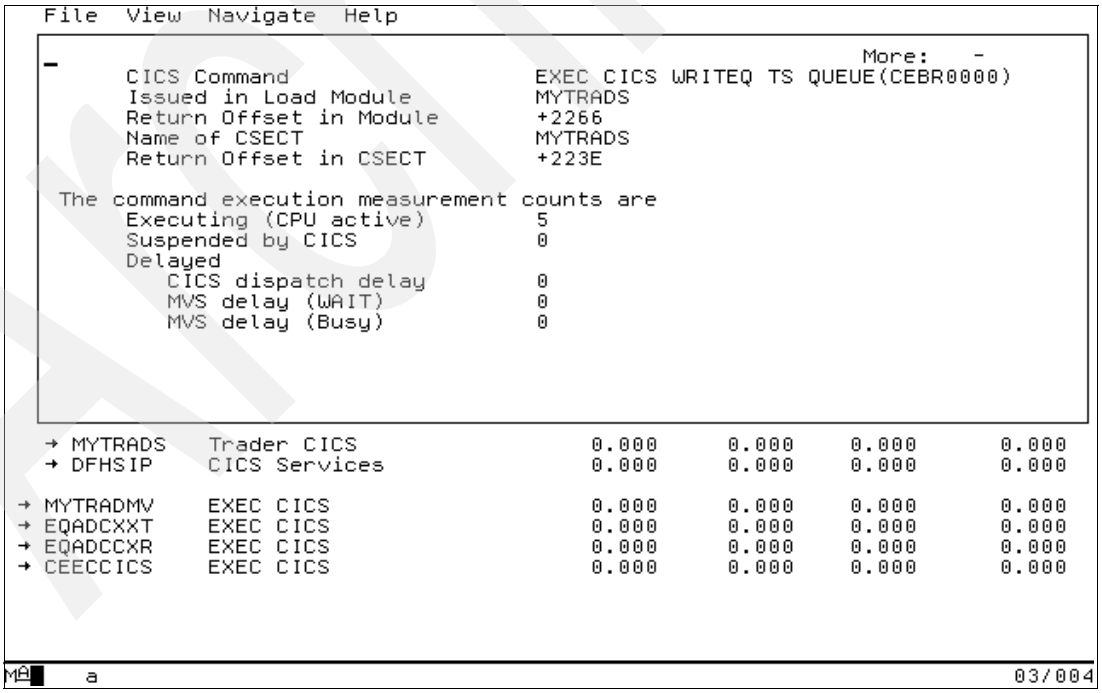


Figure 11-48 E04: Detail window for offset +223E in MYTRADS (2 of 2)

11.2.5 E05 - CICS total service time by transaction

E05 shows an analysis of how time was spent by the CICS transactions that were measured during the observation session. Expand a CICS transaction report line to see a further breakdown by programs, CICS commands SQL requests, DL/I request, and MQSeries commands, as shown in Figure 11-49.

```
  File  View  Navigate  Help
E05: CICS Total Service Time by Txn (4807)                  Row 00001 of 00020
Command ===> _____________________________________________  Scroll ===> CSR

                                                 -------- Total Time in Seconds --------
Name     NTxns Description          Error  Execution + Suspend + Delau  =  Service

MYTD        46                     ±14.9%     0.090      0.009     0.028      0.129
→ CICS          System Services               0.013      0.000     0.027      0.041
→ MYTRADMV      CICS Program                  0.023      0.000     0.000      0.024
→ MYTRADS       EXEC CICS                     0.014      0.009     0.000      0.024
  → +1AAE       REWRITE FILE(CUSTFILE)        0.000      0.006     0.000      0.007
  → +223E       WRITEQ TS QUEUE(CEBR0000)     0.004      0.000     0.000      0.004
  → +1E6C       ASKTIME ABSTIME               0.003      0.000     0.000      0.003
  → +19C4       WRITE FILE(CUSTFILE)          0.000      0.002     0.000      0.002
  → +1B94       READ FILE(COMPFILE)           0.001      0.000     0.000      0.001
  → +1F6E       FORMATTIME                    0.001      0.000     0.000      0.001
  → +1862       READ FILE(CUSTFILE)           0.000      0.000     0.000      0.000

→ MYTRADS       CICS Program                  0.010      0.000     0.000      0.010
  → MYTRADS     Trader CICS                   0.009      0.000     0.000      0.009
  → DFHSIP      CICS Services                 0.000      0.000     0.000      0.000

→ MYTRADMV      EXEC CICS                     0.009      0.000     0.000      0.009
→ EQADCXXT      EXEC CICS                     0.009      0.000     0.000      0.009
→ EQADCCXR      EXEC CICS                     0.003      0.000     0.000      0.003
→ CEECCICS      EXEC CICS                     0.002      0.000     0.000      0.002

MA  a                                                                      13/004
```

Figure 11-49 E05: CICS Total Service Time by Transaction (expanded)

Because source program mapping data is available, by entering the **P** line command on any line where csect name or where an EXEC CICS is displayed, Application Performance Analyzer displays the source code for the statement, as shown in Figure 11-50, for the hexadecimal offset +223E in MYTRADS.

Note: For CICS reports E05 and E06, the Count value is the number of service time sample (CPU + WAIT + Queued).

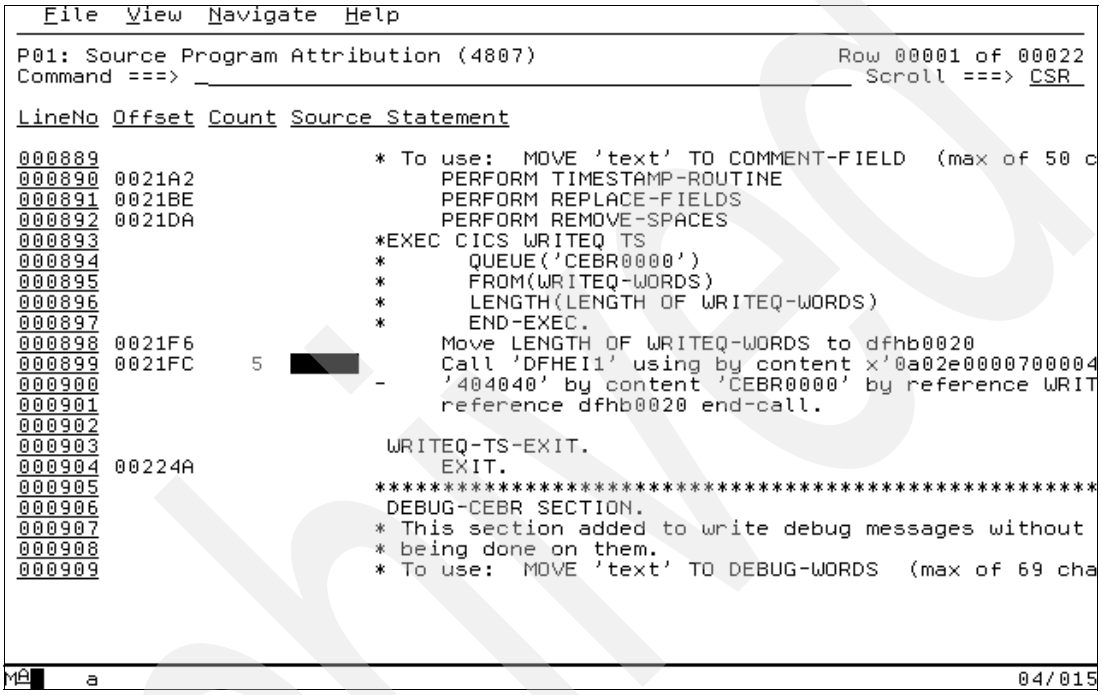
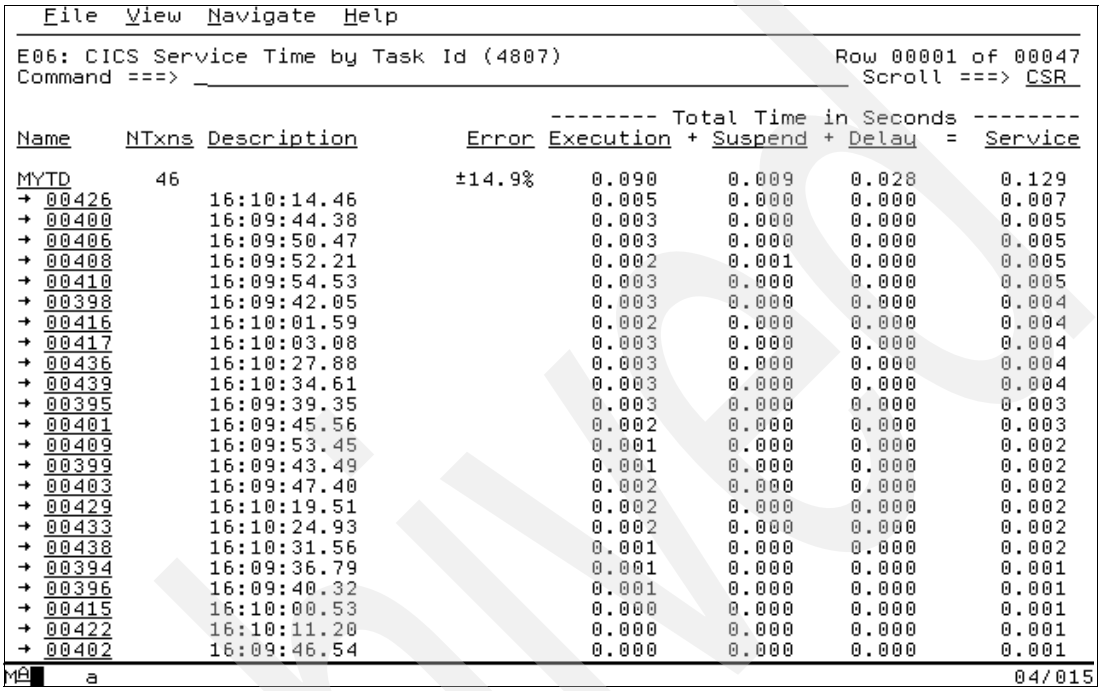


Figure 11-50 P01: Source Program Attribution (from E05)

11.2.6 E06 - CICS service time by task id

E06 shows a chronology of occurrences of CICS transactions. Expand a CICS transaction report line to see a separate line for each execution of the transaction. Expand the task number line to have a further breakdown by programs, CICS commands, SQL requests, DL/I requests, and MQSeries commands, as shown in Figure 11-51.



File View Navigate Help

E06: CICS Service Time by Task Id (4807) Row 00001 of 00047
Command ===> _____ Scroll ===> CSR

Name	NTxns	Description	Error	Execution	+ Suspend	+ Delay	= Service
				-------- Total Time in Seconds --------			
MYTD	46		±14.9%	0.090	0.009	0.028	0.129
→ 00426		16:10:14.46		0.005	0.000	0.000	0.007
→ 00400		16:09:44.38		0.003	0.000	0.000	0.005
→ 00406		16:09:50.47		0.003	0.000	0.000	0.005
→ 00408		16:09:52.21		0.002	0.001	0.000	0.005
→ 00410		16:09:54.53		0.003	0.000	0.000	0.005
→ 00398		16:09:42.05		0.003	0.000	0.000	0.004
→ 00416		16:10:01.59		0.002	0.000	0.000	0.004
→ 00417		16:10:03.08		0.003	0.000	0.000	0.004
→ 00436		16:10:27.88		0.003	0.000	0.000	0.004
→ 00439		16:10:34.61		0.003	0.000	0.000	0.004
→ 00395		16:09:39.35		0.003	0.000	0.000	0.003
→ 00401		16:09:45.56		0.002	0.000	0.000	0.003
→ 00409		16:09:53.45		0.001	0.000	0.000	0.002
→ 00399		16:09:43.49		0.001	0.000	0.000	0.002
→ 00403		16:09:47.40		0.002	0.000	0.000	0.002
→ 00429		16:10:19.51		0.002	0.000	0.000	0.002
→ 00433		16:10:24.93		0.002	0.000	0.000	0.002
→ 00438		16:10:31.56		0.001	0.000	0.000	0.002
→ 00394		16:09:36.79		0.001	0.000	0.000	0.001
→ 00396		16:09:40.32		0.001	0.000	0.000	0.001
→ 00415		16:10:00.53		0.000	0.000	0.000	0.001
→ 00422		16:10:11.20		0.000	0.000	0.000	0.001
→ 00402		16:09:46.54		0.000	0.000	0.000	0.001

MA a 04/015

Figure 11-51 E06: CICS Service Time by Task Id (expanded)

Note: The E06 report has been sorted, using the SV line command on the CICS transaction identification MYTD, by descending value of Service time.

The SV line command has been used twice because entering a sort command repeatedly toggles the report between sorting in descending and ascending values.

With an E06 sorted by descending value, is it easy to see which task ID is out of the target response time and to expand it to drill down to identify the cause.

11.2.7 E07 - CICS wait by transaction

E07 shows where CICS transactions were waiting. Expand a CICS transaction to see the breakdown by resource name, as shown in Figure 11-52.

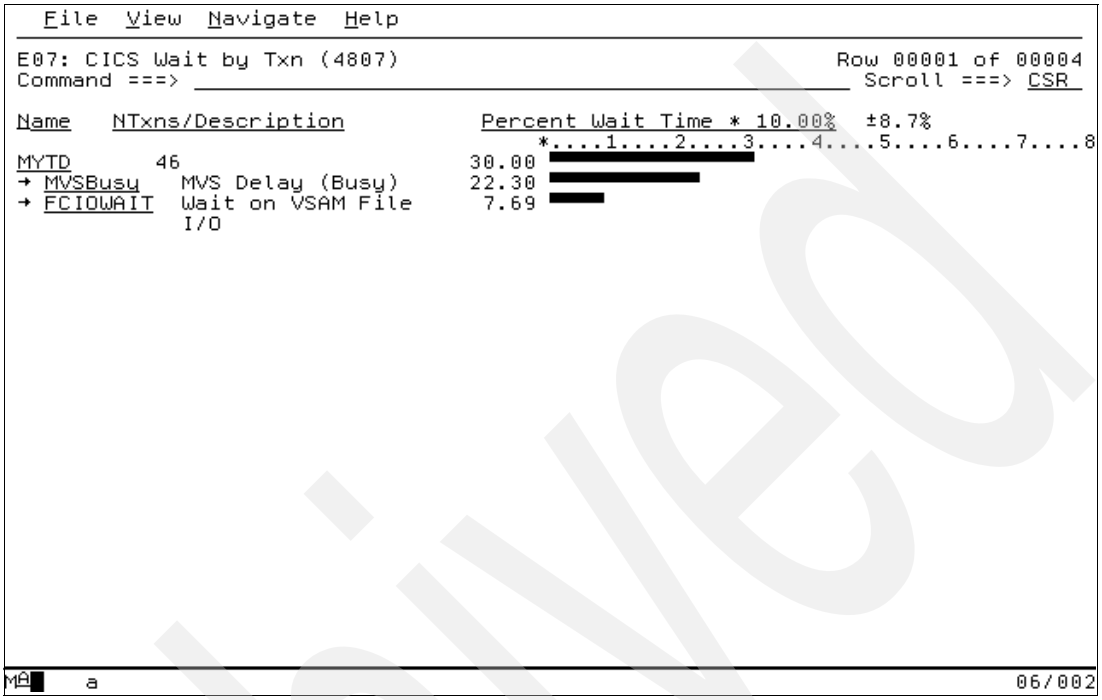


Figure 11-52 E07: Wait by Transaction (expanded)

11.3 DB2 measurement

To illustrate IMS reports, we use the TRADER application in batch mode using DB2 databases. This application is described in detail in Guided Tour of the Trader application.

We do not spend much effort on the things that are common to Application Performance Analyzer and that have been described in the previous chapter, but rather focus on the unique aspects of using Application Performance Analyzer with batch job accessing DB2 databases.

11.3.1 Observation session request definition

We have to monitor the TRADERD batch job with the following characteristics:

- We used a sampling rate of 1,000 samples per second for all its duration.
- TRADERD has a single step.
- The measurement file has to be retained for 30 days.
- All available DB2 data must be recorded.
- CHABERT has to be notified at the end of the measurement.

By entering the NEW command line on the Observation List (R02) panel, Application Performance Analyzer displays the Schedule New Measurement (R03) panel. Both DB2 and DB2+ data extractors have to be turned on to get all DB2 data.

When the DB2 data extractor is active, it collects measurement data directly related to SQL activity. During each sampling interval, Application Performance Analyzer interrogates DB2 to determine whether the application or transaction is currently performing a DB2 request. If it is, a DB2 measurement record is created that describes the request. If the request was to process an SQL statement then details of the SQL statement are also recorded.

When the DB2+ data extractor is active, it collects the data required to report exact SQL request counts, SQL CPU time, and SQL Service Time.

Note: The DB2+ data extractor has to be active to produce reports F10, F11, and F12.

Also, some fields in the F01 report require DB2+. Turning on DB2+ insures that the SQL text reported on SQL statements is accurate.

The settings required to monitor the TRADERD job are shown in Figure 11-53 for panel 1 (Job Information) and Figure 11-54 for panel 2 (Options).

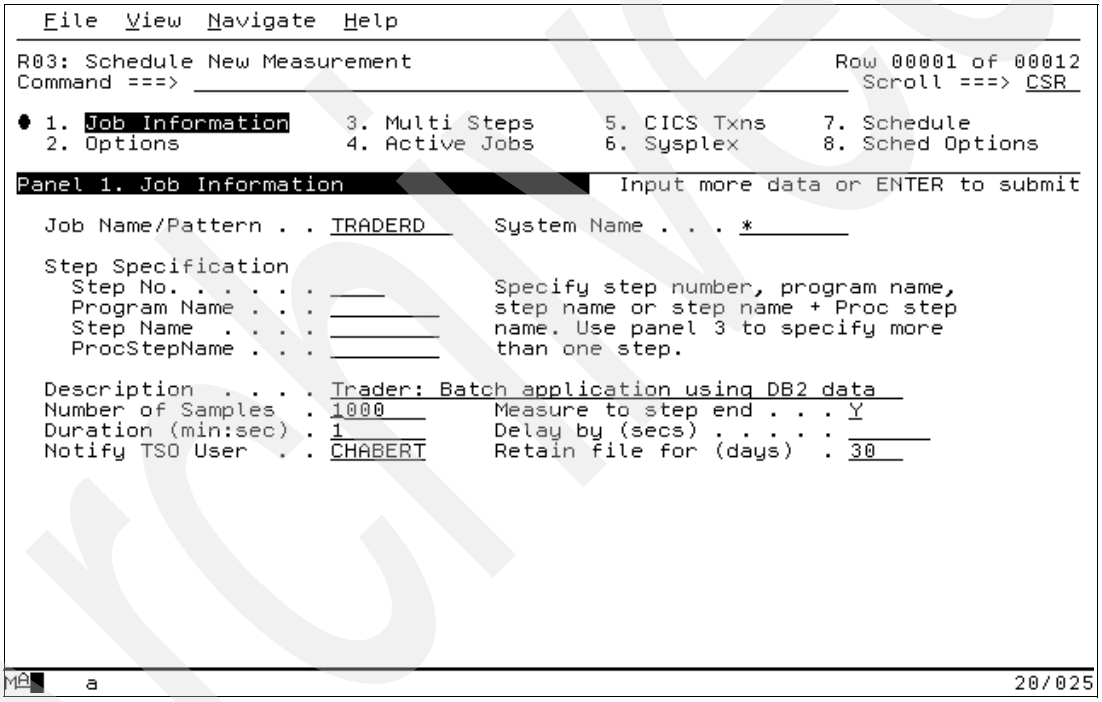


Figure 11-53 R03: Schedule New Measurement (panel 1: Job Information)

With panel 1 we specify:

- Job name
- Sample rate
- Retention period

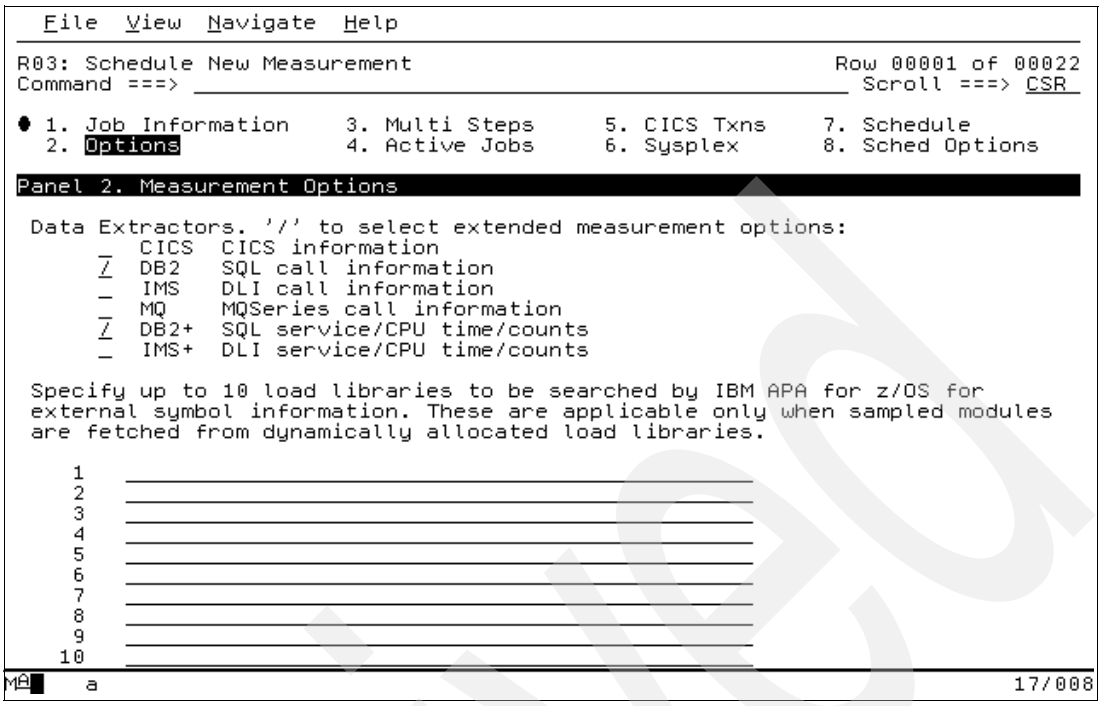


Figure 11-54 R03: Schedule New Measurement (panel 2: Options)

With panel 2 we turn on the DB2 data extractors.

When we press Enter to submit the previous definition, Application Performance Analyzer displays the Observation List (R02) panel, with our new measurement definition at the top of the list. Its number is 4811. We now start TRADERD, when the batch job is finished, by using the **R** line command on request number 4811. Application Performance Analyzer launches the associated performance reports.

Because `F DB2 Measurement` is displayed in red (in gray on our graphic), DB2 reports are available. To get the list of available DB2 reports, shown in Figure 11-55, just type `F` on the command line and press Enter.

```
  File  View  Navigate  Help
R01: IBM APA for z/OS Performance Reports (4811)          Row 00001 of 00008
Command ===> _                                             Scroll ===> CSR

  Select a category from     _ A Admin/Miscellaneous   _ I IMS Measurement
  the list to the right      _ S Statistics/Storage    _ E CICS Measurement
  to view the available      _ C CPU Usage Analysis    _ F DB2 Measurement
  reports in the selection   _ D DASD I/O Analysis     _ Q MQ Measurement
  list below.                _ W CPU WAIT Analysis      _ G Coupling Facility

Enter S to make a selection or enter the report code on the command line

  _ F01 DB2 Measurement Profile          _ F08 DB2 SQL Wait Time by Statement
  _ F02 DB2 SQL Activity Timeline        _ F09 DB2 SQL Wait Time by Plan
  _ F03 DB2 SQL Activity by DBRM         _ F10 DB2 SQL CPU/Svc Time by DBRM
  _ F04 DB2 SQL Activity by Statement    _ F11 DB2 SQL CPU/Svc Time by Stmt
  _ F05 DB2 SQL Activity by Plan         _ F12 DB2 SQL CPU/Svc Time by Plan
  _ F06 DB2 SQL Statement Attributes     _ F13 DB2 SQL Threads Analysis
  _ F07 DB2 SQL Wait Time by DBRM        _ F14 DB2 CPU by Plan/Stored Proc

MA  a                                                              04/015
```

Figure 11-55 R01: Performance Reports (DB2 reports selected)

The first report that we display is not a DB2 report, but a subset of report S01: Measurement Profile to see the additional information displayed when DB2 extractors have been turned on, as shown in Figure 11-56.

```
  File  View  Navigate  Help
S01: Measurement Profile (4811)                           Row 00029 of 00098
Command ===> _                                             Scroll ===> CSR

Most Active DB2 Plans
 Samples                  24,683  100.0%
 TRADERD                  22,676   91.8%

Most Active Package/DBRMs
 Samples                  24,683  100.0%
 TRADERD                  22,676   91.8%

Most Active SQL Statements                                  Reports:
 Samples                  24,683  100.0%                      F04
 TRADERD:00825 UPDATE      3,522   14.2%
 TRADERD:00742 FETCH       2,800   11.3%
 TRADERD:00854 FETCH       2,675   10.8%
 TRADERD:00774 FETCH       2,294    9.2%
 TRADERD:00761 CLOSE       2,020    8.1%

Request parameters
 Request number     4811
 Description        Trader: Batch application using DB2 Data
 Sample file DSN    ADTOOLS.APA.CHABERT.R4811.TRADERD.SF
 Retention          Fri Dec-09-2005
 Data extractors    DB2,DB2+

 Requesting user    CHABERT          Nbr of samples     1,000

MA  a                                                              04/015
```

Figure 11-56 S01: Measurement Profile (DB2 information)

Application Performance Analyzer gives us the following reports:

- Most Active DB2 Plans: Up to five DB2 plans are reported.
- Most Active Package/DBRMs: Up to five DB2 package names or DBRM names are reported. A DBRM name is shown instead of a package name in the event that the DBRM was bound directly to the plan instead of to a package.
- Most Active SQL Statements: Up to five SQL statements are reported. If we require more detail on that statement we can skip the cursor to the report code (F04) at the right part of the graph and press Enter.

We return to the previous panel by pressing the F3 key, and we select the first DB2 report.

11.3.2 F01 - DB2 measurement profile

F01 shows a general overview of the DB2 measurement data. This is a good report to examine first when analyzing DB2 information. It provides an at-a-glance summary of various aspects of the measurement data and helps us choose which other reports to concentrate on. It contains several sections, which are:

- Most Active DB2 Plans: Up to five DB2 plans are reported.
- Most Active Package/DBRMs: Up to five packages/DBRMs are reported.
- Most active SQL Statements: Up to five SQL statements are reported.
- Most CPU consumptive SQL: Total CPU time (if DB2+ was active):
  Up to five SQL statements are reported.
- Most Frequent SQL Statements (if DB2+ was active):
  Up to five SQL statements are reported.
- Most CPU consumptive SQL: Total SQL service time (if DB2+ was active):
  Up to five SQL statements are reported.
- DB2 Measurement Statistics: This is shown in Figure 11-57 and Figure 11-58.

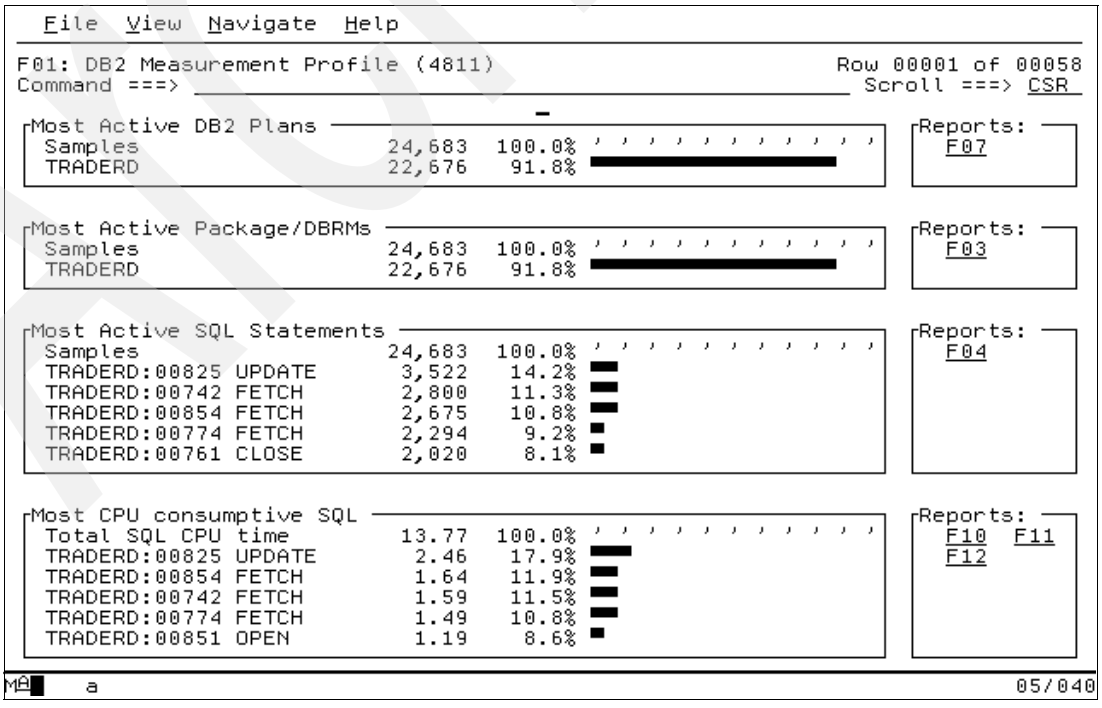


Figure 11-57 F01: DB2 Measurement Profile (1 of 2)

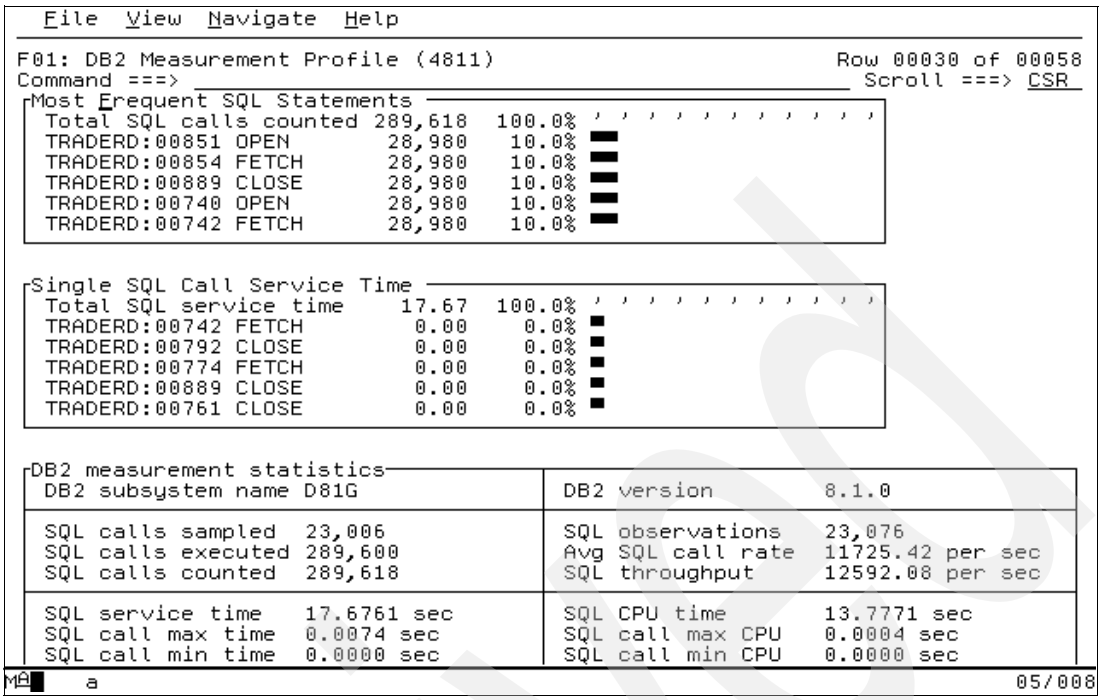


Figure 11-58 F01: DB2 Measurement Profile (2 of 2)

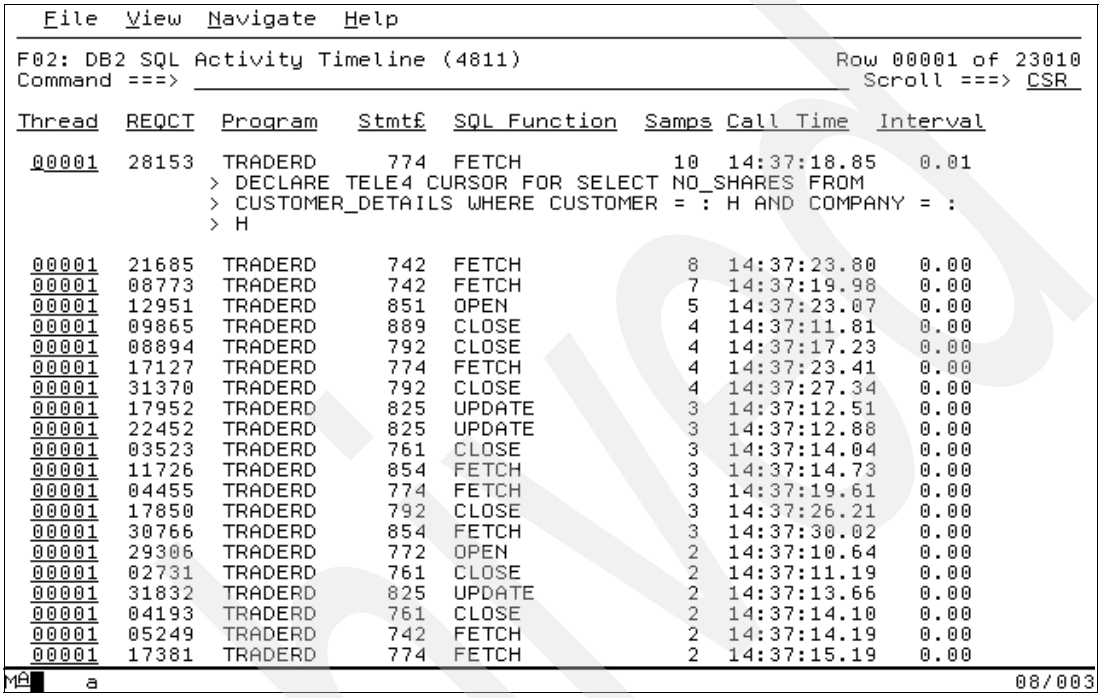
The relevant information about the main fields is:

SQL calls sampled Number of unique SQL requests in which samples were taken.
SQL call executed Number of SQL requests executed.
SQL call counted Exact number of SQL requests. DB2+ is required.
SQL observations Number of samples in which SQL activity was observed.
Avg SQL call rate Average number of SQL calls per second.
SQL throughput Theoretical SQL request throughput rate.

SQL service time, SQL CPU time, SQL max time, SQL max CPU, SQL min time, and SQL min CPU are available only if the DB2+ measurement option was selected for the measurement.

F02 - DB2 SQL activity timeline

F02 shows information about the chronology of SQL requests that were sampled over the duration of the measurement and to identify any calls with excessive service times, as shown in Figure 11-59. The report has been sorted in descending order by call time (default order is chronologically) by entering the SD line command on the Thread heading. This helps to identify SQL calls with an excessive service time. The first entry has been expanded.



```
  File  View  Navigate  Help
F02: DB2 SQL Activity Timeline (4811)                    Row 00001 of 23010
Command ===> _____________________________________________ Scroll ===> CSR

Thread  REQCT  Program   Stmt#  SQL Function  Samps Call Time  Interval

 00001  28153  TRADERD     774  FETCH            10  14:37:18.85   0.01
              > DECLARE TELE4 CURSOR FOR SELECT NO_SHARES FROM
              > CUSTOMER_DETAILS WHERE CUSTOMER = : H AND COMPANY = :
              > H

 00001  21685  TRADERD     742  FETCH             8  14:37:23.80   0.00
 00001  08773  TRADERD     742  FETCH             7  14:37:19.98   0.00
 00001  12951  TRADERD     851  OPEN              5  14:37:23.07   0.00
 00001  09865  TRADERD     889  CLOSE             4  14:37:11.81   0.00
 00001  08894  TRADERD     792  CLOSE             4  14:37:17.23   0.00
 00001  17127  TRADERD     774  FETCH             4  14:37:23.41   0.00
 00001  31370  TRADERD     792  CLOSE             4  14:37:27.34   0.00
 00001  17952  TRADERD     825  UPDATE            3  14:37:12.51   0.00
 00001  22452  TRADERD     825  UPDATE            3  14:37:12.88   0.00
 00001  03523  TRADERD     761  CLOSE             3  14:37:14.04   0.00
 00001  11726  TRADERD     854  FETCH             3  14:37:14.73   0.00
 00001  04455  TRADERD     774  FETCH             3  14:37:19.61   0.00
 00001  17850  TRADERD     792  CLOSE             3  14:37:26.21   0.00
 00001  30766  TRADERD     854  FETCH             3  14:37:30.02   0.00
 00001  29306  TRADERD     772  OPEN              2  14:37:10.64   0.00
 00001  02731  TRADERD     761  CLOSE             2  14:37:11.19   0.00
 00001  31832  TRADERD     825  UPDATE            2  14:37:13.66   0.00
 00001  04193  TRADERD     761  CLOSE             2  14:37:14.10   0.00
 00001  05249  TRADERD     742  FETCH             2  14:37:14.19   0.00
 00001  17381  TRADERD     774  FETCH             2  14:37:15.19   0.00
MA    a                                                          08/003
```

Figure 11-59 F02: DB2 SQL Activity Timeline (sorted by SQL call duration)

We can enter the ++ line command (or Enter) on any line to bring up a pop-up window containing additional information (as shown in Figure 11-59) for the first entry (as shown in Figure 11-60, Figure 11-61, and Figure 11-62).

```
File  View  Navigate  Help

 SQL Statement Information
    Subsystem name  D81G                Attach type      BATCH
    Plan name       TRADERD             Plan BIND time   Nov-09-05 14:34:04

    DBRM name       TRADERD             DBRM token       17BC7387 08CD6E00
    DBRM date/time  Nov-09-05 14:33:40

    SQL function    FETCH               Static/dynamic   Static
    Precmplr stmt£  774                 DBRM section£    4
    CSECT/module    TRADERD in TRADERD  Offset of call   0000295E
    Sample count    2294                SQL req count    28,889
    SQL CPU time    1.49                Service time     1.87

    SQL Statement:   DECLARE TELE4 CURSOR FOR SELECT NO_SHARES FROM
                     CUSTOMER_DETAILS WHERE CUSTOMER = : H AND COMPANY =
                     : H

00001  03523  TRADERD     761  CLOSE            3  14:37:14.04   0.00
00001  11726  TRADERD     854  FETCH            3  14:37:14.73   0.00
00001  04455  TRADERD     774  FETCH            3  14:37:19.61   0.00
00001  17850  TRADERD     792  CLOSE            3  14:37:26.21   0.00
00001  30766  TRADERD     854  FETCH            3  14:37:30.02   0.00
00001  29306  TRADERD     772  OPEN             2  14:37:10.64   0.00
00001  02731  TRADERD     761  CLOSE            2  14:37:11.19   0.00
00001  31832  TRADERD     825  UPDATE           2  14:37:13.66   0.00
00001  04193  TRADERD     761  CLOSE            2  14:37:14.10   0.00
00001  05249  TRADERD     742  FETCH            2  14:37:14.19   0.00
00001  17381  TRADERD     774  FETCH            2  14:37:15.19   0.00
MA    a                                                          03/004
```

Figure 11-60 F02: Additional details (1 of 3)

If we add the information from the first screen with the SQL source statement below, we have the information of the DB2 report F06: DB2 SQL Statements Attributes.

```
File  View  Navigate  Help

 SQL Source statement in: READ-SHARES-CUSTOMER
          *                                        **GET SHARES
          *****     EXEC SQL FETCH TELE4 INTO :NO-SHARES END-EXEC.
               PERFORM SQL-INITIAL UNTIL SQL-INIT-DONE
               CALL 'DSNHLI' USING SQL-PLIST12.


 DB2 Thread Information
     Thread sequence number          00001
     Attachment type                 CIB (DSN command)

     First REQCT value observed      00002
     Time of first REQCT             14:37:08.03

     REQCT reset from 32767 to 1     8 times

     Last REQCT value observed       27465
     Time of last REQCT              14:37:32.47

00001  30766  TRADERD     854  FETCH            3  14:37:30.02   0.00
00001  29306  TRADERD     772  OPEN             2  14:37:10.64   0.00
00001  02731  TRADERD     761  CLOSE            2  14:37:11.19   0.00
00001  31832  TRADERD     825  UPDATE           2  14:37:13.66   0.00
00001  04193  TRADERD     761  CLOSE            2  14:37:14.10   0.00
00001  05249  TRADERD     742  FETCH            2  14:37:14.19   0.00
00001  17381  TRADERD     774  FETCH            2  14:37:15.19   0.00
00001  29301  TRADERD     774  FETCH            2  14:37:16.19   0.00
00001  22905  TRADERD     761  CLOSE            2  14:37:18.41   0.00
00001  31878  TRADERD     854  FETCH            2  14:37:19.16   0.00
00001  18292  TRADERD     740  OPEN             2  14:37:20.77   0.00
MA    a                                                          03/004
```

Figure 11-61 F02: Additional details (2 of 3)

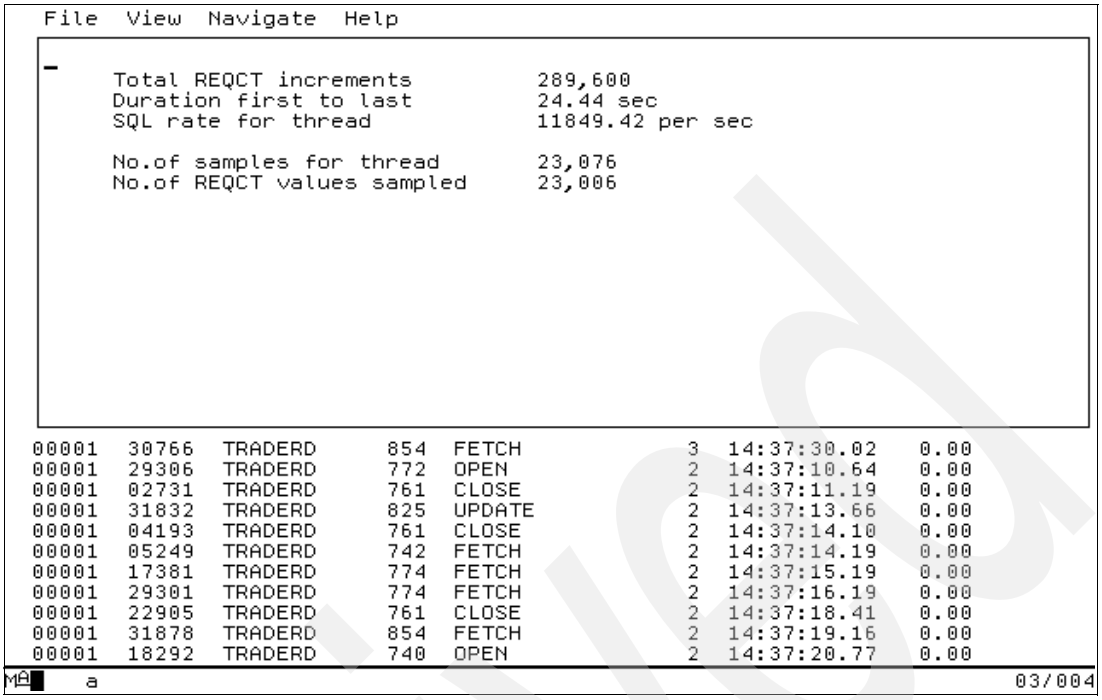


Figure 11-62 F02: Additional details (3 of 3)

11.3.3 F03 - DB2 SQL activity by DBRM

F03 shows how time was consumed by SQL request processing. The percentage of time is reported by each module that issued SQL requests. Expand a module line to see a further breakdown of time consumption by individual SQL request issued by the module, as shown in Figure 11-63.

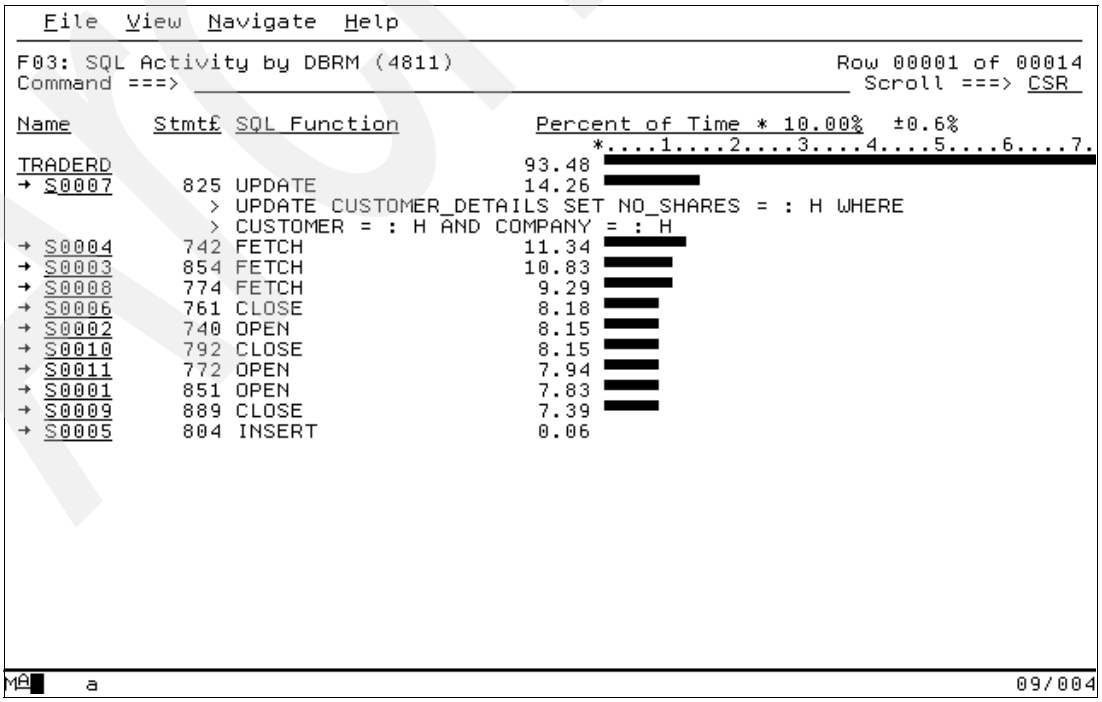


Figure 11-63 F03: DB2 SQL Activity by DBRM (expanded)

Because source program mapping data is available, by entering the **P** line command on any sequence number (a four-digit number assigned to uniquely identify the SQL request), Application Performance Analyzer displays the source code for the statement (as shown in Figure 11-64) for the sequence number S0007.

Note: Either S or D precedes the sequence number, indicating whether the SQL statement is static or dynamic.

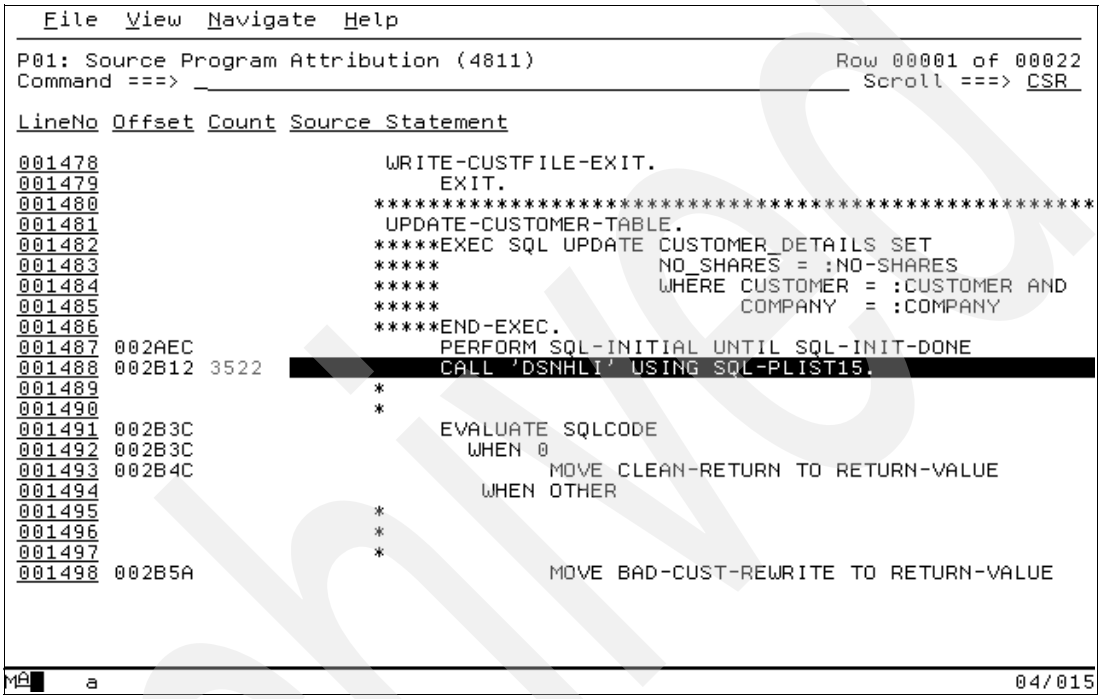


Figure 11-64 P01: Source Program Attribution (from F03)

Note: The Count value is the number of samples taken during the execution of the statement.

11.3.4 F04 - DB2 SQL activity by statement

F04 shows how time was consumed by SQL request processing, as shown in Figure 11-65.

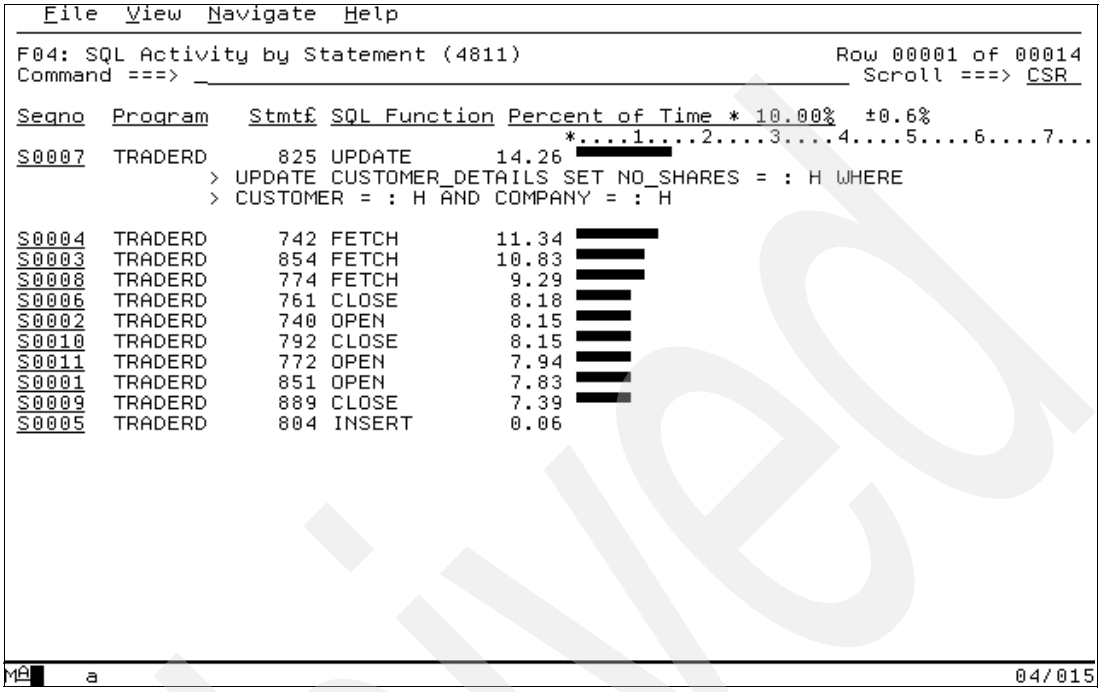


Figure 11-65 F04: SQL Activity by Statement (expanded)

By entering the **P** line command on any sequence number, Application Performance Analyzer displays the source code for the statement. The Count value is the number of samples taken during the execution of the statement.

By entering the **Ex** (Explain) line command on any sequence number, because Application Performance Analyzer has been customized to allow the Dynamic Explain function and because DB2+ was active, Application Performance Analyzer issues a dynamic EXPLAIN request on the SQL text of the statement you have selected, as shown in Figure 11-66.

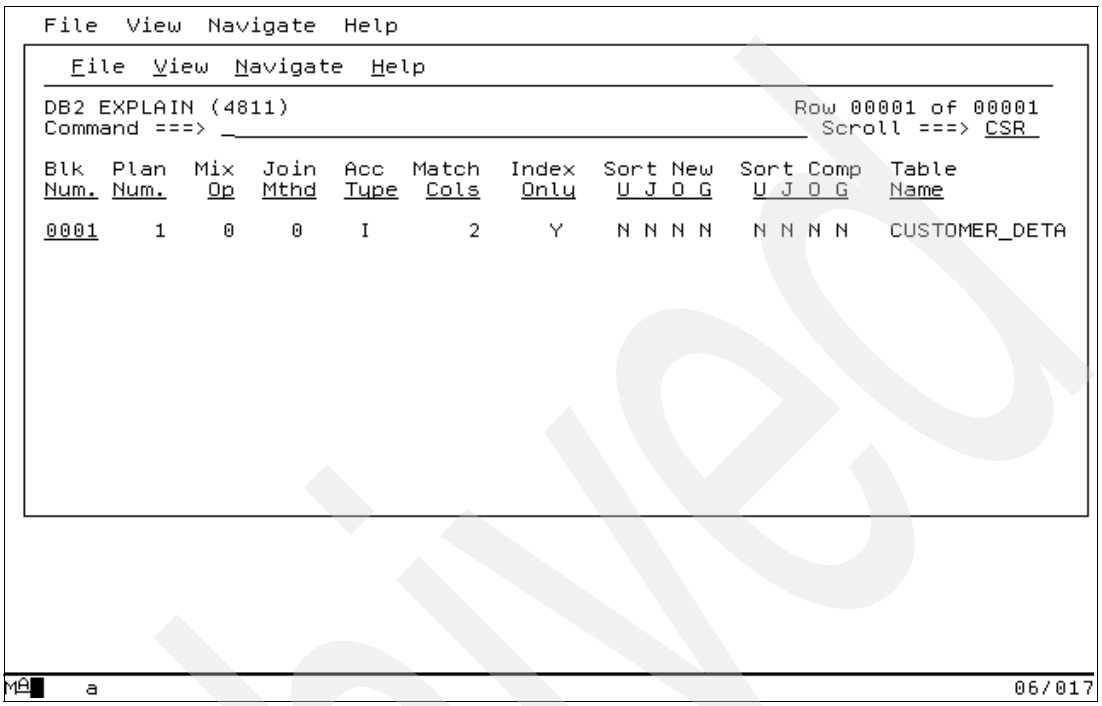


Figure 11-66 DB2 Dynamic Explain from F04

This report shows information from PLAN_TABLE built by a DB2 EXPLAIN request. Each report line represents a row in the result PLAN_TABLE.

Note: The EXPLAIN request is executed at the time you request it. It is not issued at the time of the measurement. Thus any changes made to the DB2 objects, since the measurement was requested, affect the EXPLAIN request.

By placing the cursor on any Blk Num (Query Block Number) and pressing Enter, additional details about this line are displayed (as shown in Figure 11-67 through Figure 11-71).

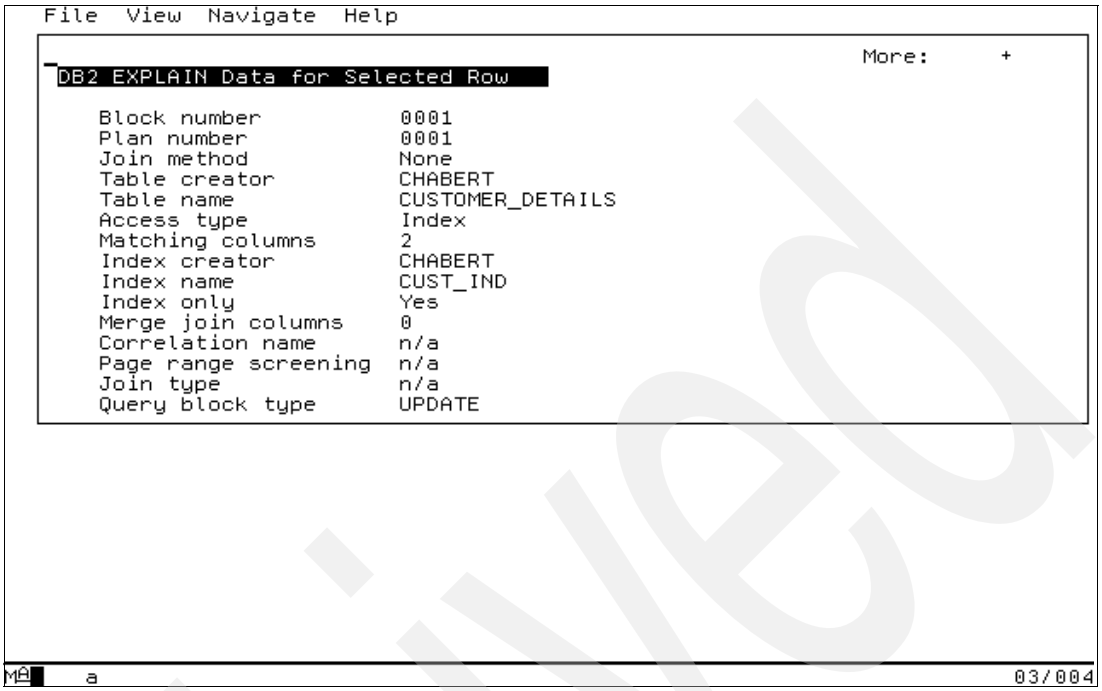


Figure 11-67 DB2 Explain (information detail 1 of 5)

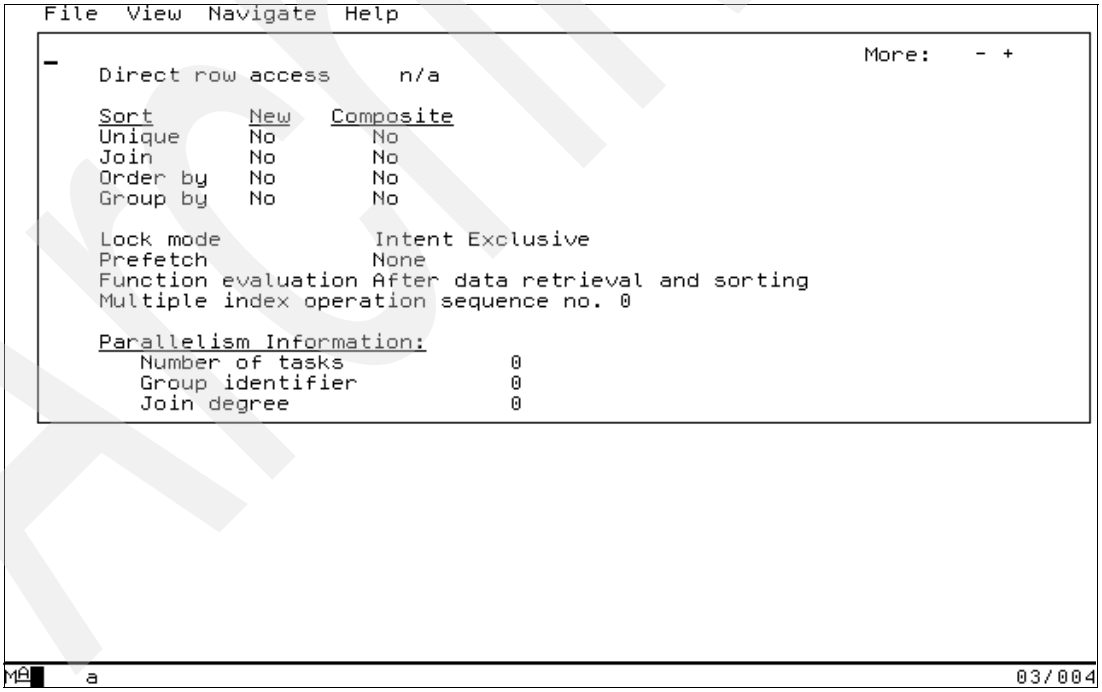


Figure 11-68 DB2 Explain (information detail 2 of 5)

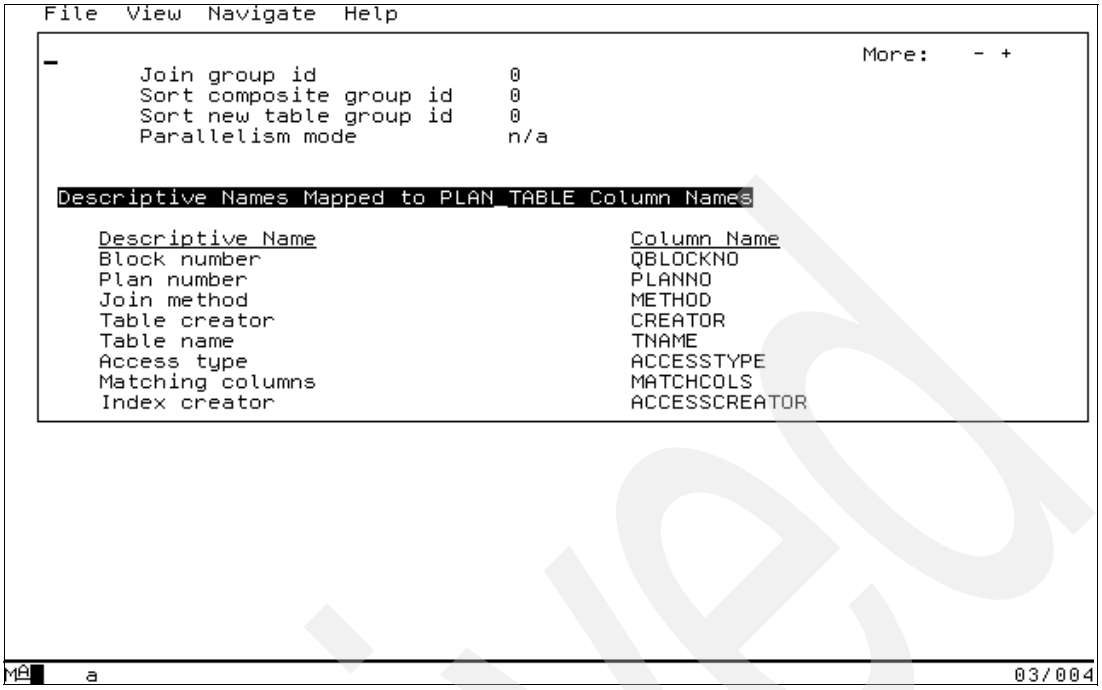


Figure 11-69 DB2 Explain (information detail 3 of 5)

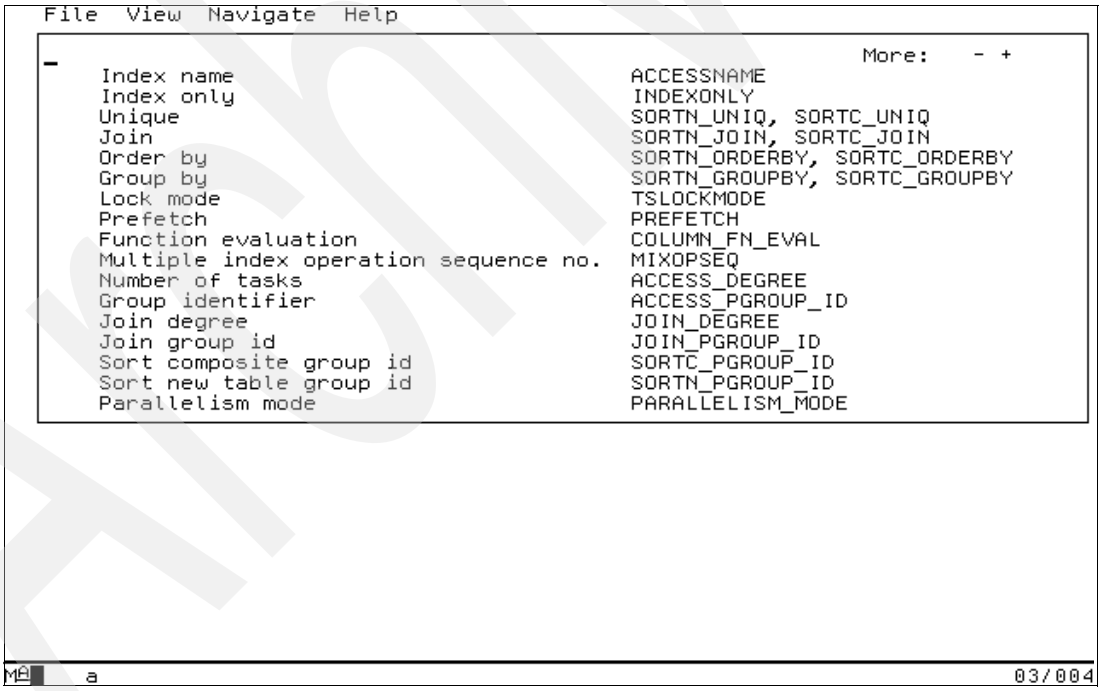


Figure 11-70 DB2 Explain (information detail 4 of 5)

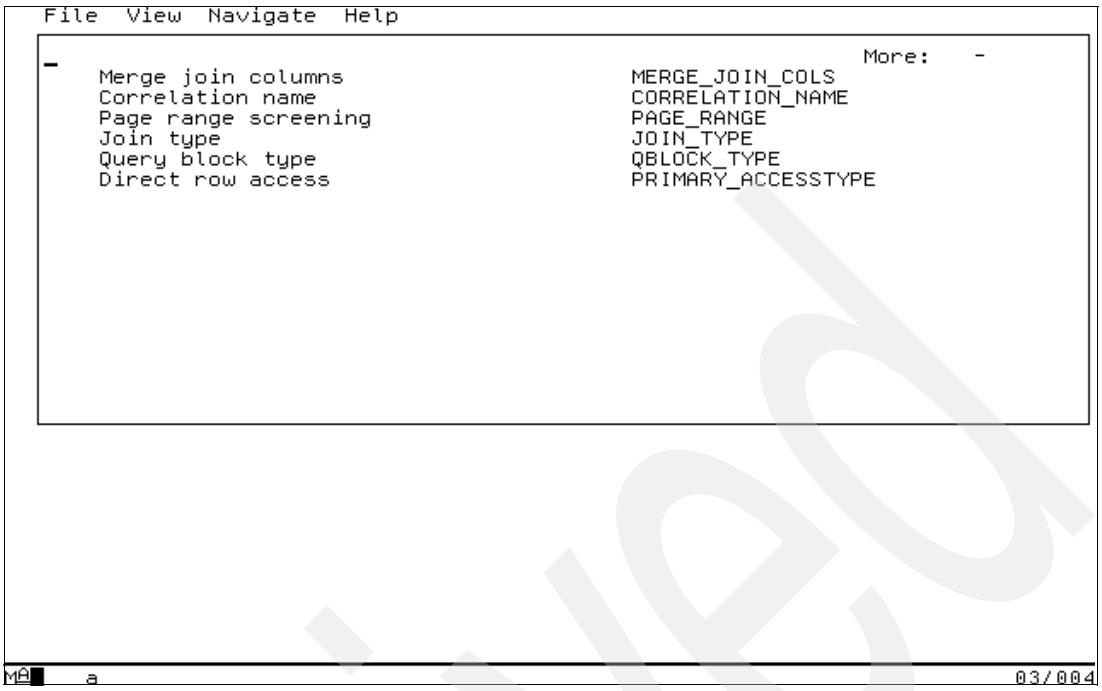


Figure 11-71 DB2 Explain (information detail 5 of 5)

11.3.5 F05 - DB2 activity by plan

F05 shows how time was consumed by SQL request processing. Expand a plan line to see a further breakdown of time consumption by individual SQL request, as shown in Figure 11-72.

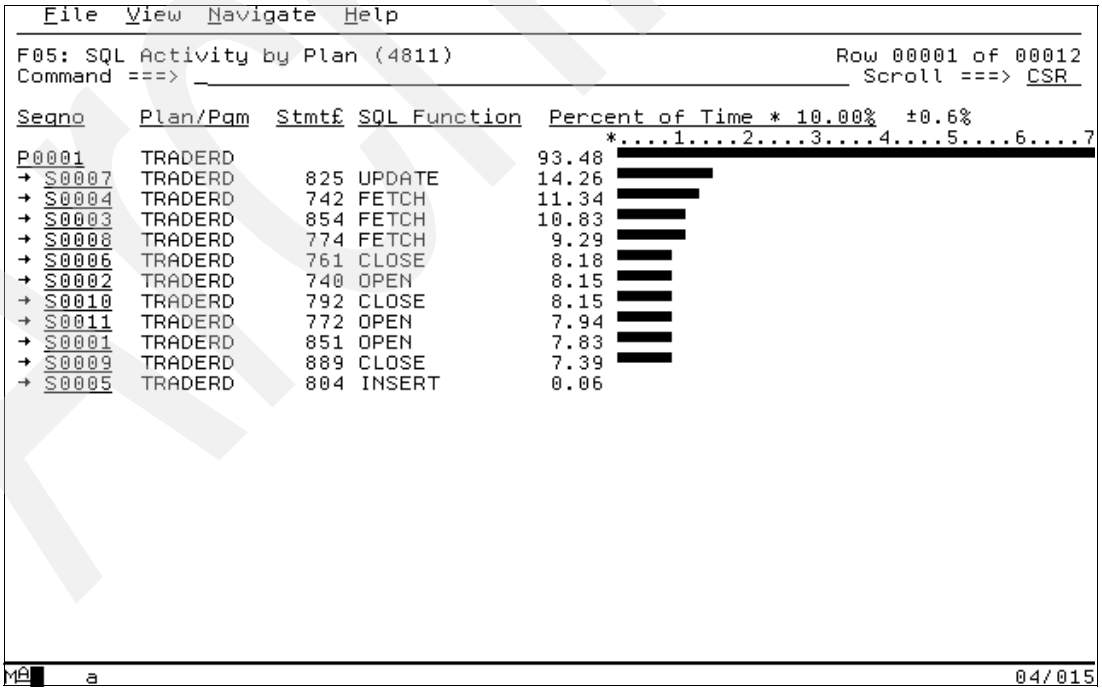


Figure 11-72 F05: DB2 Activity by Plan

By entering the **P** line command on any sequence number, Application Performance Analyzer displays the source code for the statement. The Count value is the number of samples taken during the execution of the statement.

By entering the **Ex** (Explain) line command on any sequence number, Application Performance Analyzer issues a dynamic EXPLAIN request on the SQL text of the statement you have selected.

11.3.6 F06 - DB2 SQL statement attributes

F06 shows detailed information about each of the measured SQL statements, as shown in Figure 11-73. This is useful as a reference report when working with printed copies of other DB2 reports that do not show full SQL statement details (online, when using the **++** line command, the pop-up detail windows show this information).

Figure 11-73 F06: DB2 SQL Statement Attributes

The relevant information about the main fields (when not self-explanatory) is:

SQL Statement ID A unique sequence number assigned by APA. It is shown in other DB2 reports.

Attachment Type The of DB2 attachment for the thread.

DBRM Token Consistency token.

Precmplr Stmt# Statement number assigned by the precompiler to the SQL statement.

DBRM Section# Section number assigned by the precompiler to the SQL statement. Groups of related statements (OPEN, FETCH, CLOSE) have the same section number.

Sample count Number of samples in which execution was measured.

SQL Req Count Number of SQL calls counted for the duration of the measurement (DB2+ must be active).

SQL CPU Time Accumulated CPU time used by the statement for the duration of the measurement (DB2+ must be active).

Service Time Accumulated service time used by the statement for the duration of the measurement (DB2+ must be active).

Note: If the source program mapping feature was not available, `SQL Source statement in` information is not displayed.

11.3.7 F07 - DB2 SQL WAIT time by DBRM

F07 shows information about WAIT time that occurred during the processing of SQL requests. We expand each module line to see a further breakdown and quantification by individual SQL statement, as shown in Figure 11-74.

```
  File  View  Navigate  Help
F07: SQL WAIT Time by DBRM (0339)                          Row 00001 of 00009

Name       Stmt£ SQL Function          Percent of Total Time * 10.00%  ±0.2%
                                          *....1....2....3....4....5....6....7.
TRADERD                                0.31
→ S0007      825 UPDATE                0.21
               > UPDATE CUSTOMER_DETAILS SET NO_SHARES = : H WHERE
               > CUSTOMER = : H AND COMPANY = : H
→ S0006      804 INSERT                0.05
→ S0002      854 FETCH                 0.02
→ S0001      851 OPEN                  0.02
→ S0003      740 OPEN                  0.00
→ S0009      772 OPEN                  0.00

Command ===>                                              Scroll ===> CSR
MA  c                                                          13/004
```

Figure 11-74 F07: SQL WAIT Time by DBRM (expanded)

Note: The SETUP command line lets us choose to see the WAIT time expressed as a percentage of SQL service time for the DBRM or as a percentage of the total measurement interval.

This helps to answer these questions about SQL processing wait time:

- For how much of the overall measurement interval was the address space in a WAIT during SQL processing?
- For how much of the SQL processing time was the address space in a WAIT?

Because source program mapping data is available, by entering the **P** line command on any sequence number, Application Performance Analyzer displays the source code for the statement (as shown in Figure 11-75) for the sequence number S0007.

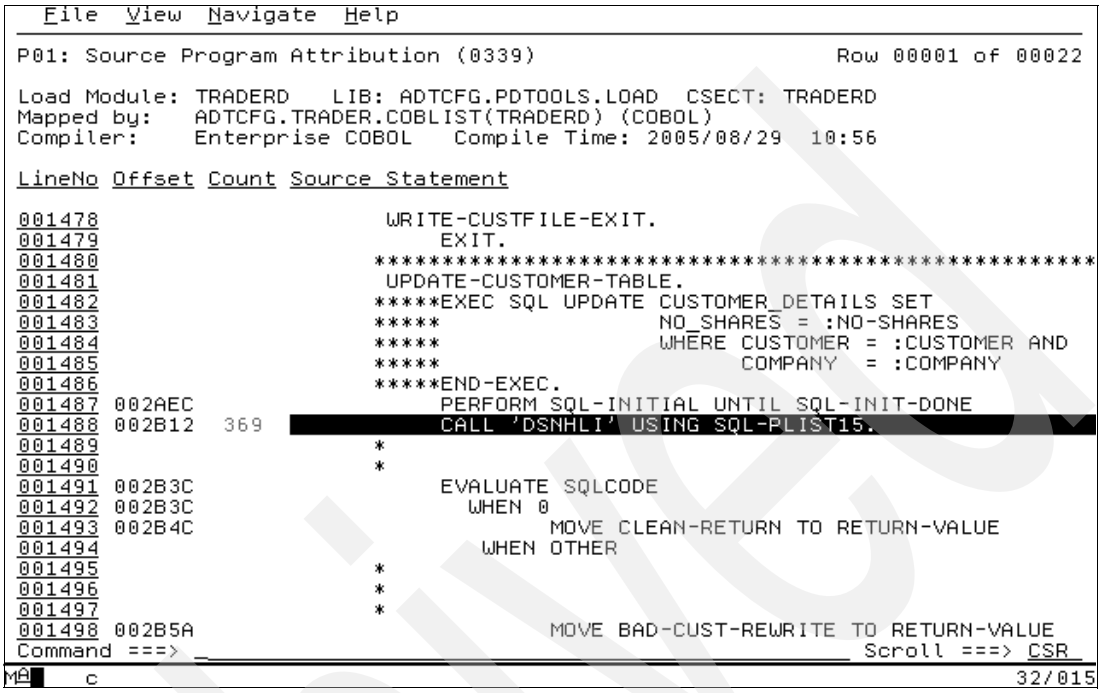


Figure 11-75 P01: Source Program Attribution (from F07)

Note: The Count value is the number of samples taken of execution of the SQL statement while in WAIT.

This applies to DB2 reports F07, F08, and F09.

Note: The SETUP line command has been used to show detailed information in the heading. This information includes:

- Load module name
- Load library name
- CSECT name
- Source mapping file name
- Compile date and time
- Code segment address range (when applicable)

It is usually preferable to turn off this option, as it occupies several lines at the top of the screen.

11.3.8 F08 - SQL WAIT time by statement

F08 shows information about WAIT time that occurred during the processing of SQL requests, as shown in Figure 11-76.

```
  File  View  Navigate  Help
F08: SQL WAIT Time by Statement (0339)                    Row 00001 of 00009

Seqno  Program   Stmt£ SQL Function Percent of Total Time * 10.00%  ±0.2%
                                      *....1....2....3....4....5....6....7...
S0007  TRADERD     825 UPDATE       0.21
               > UPDATE CUSTOMER_DETAILS SET NO_SHARES = : H WHERE
               > CUSTOMER = : H AND COMPANY = : H

S0006  TRADERD     804 INSERT       0.05
S0002  TRADERD     854 FETCH        0.02
S0001  TRADERD     851 OPEN         0.02
S0003  TRADERD     740 OPEN         0.00
S0009  TRADERD     772 OPEN         0.00

Command ===>                                              Scroll ===> CSR
MA    c                                                          07/002
```

Figure 11-76 F08: SQL WAIT Time by Statement (expanded)

Note: The SETUP command line lets us choose to see the WAIT time expressed as a percentage of SQL service time for the SQL statement or as a percentage of the total measurement interval.

This helps to answer these questions about SQL processing wait time:

- For how much of the overall measurement interval was the address space in a WAIT during SQL processing?
- For how much of the SQL processing time was the address space in a WAIT?

Because source program mapping data is available, by entering the **P** line command on any sequence number, Application Performance Analyzer displays the source code for the statement (as shown in Figure 11-76) for the sequence number S0007.

Note: The Count value is the number of samples taken of the execution of the SQL statement while in WAIT.

This applies to DB2 reports F07, F08, and F09.

11.3.9 F09 - DB2 SQL WAIT time by plan

F09 shows information about WAIT time that occurred during the processing of SQL requests, as shown in Figure 11-77.

```
  File  View  Navigate  Help
F09: SQL WAIT Time by Plan (0339)                          Row 00001 of 00010

Segno     Plan/Pgm  Stmt#  SQL Function  Percent of PLAN SQL Time * 10.00%  ±0.2%
                                              *....1....2....3....4....5....6....7
P0001     ERIC                           0.33
→ S0007   TRADERD      825 UPDATE        0.22
               > UPDATE CUSTOMER_DETAILS SET NO_SHARES = : H WHERE
               > CUSTOMER = : H AND COMPANY = : H

→ S0006   TRADERD      804 INSERT        0.06
→ S0002   TRADERD      854 FETCH         0.02
→ S0001   TRADERD      851 OPEN          0.02
→ S0003   TRADERD      740 OPEN          0.00
→ S0009   TRADERD      772 OPEN          0.00

Command ===>                                              Scroll ===> CSR
MA    c                                                            08/004
```

Figure 11-77 F09: DB2 SQL WAIT Time by Plan (expanded)

Note: The SETUP command line lets us choose to see the WAIT time expressed as a percentage of SQL service time for the PLAN or as a percentage of the total measurement interval.

This helps to answer these questions about SQL processing wait time:

- For how much of the overall measurement interval was the address space in a WAIT during SQL processing?
- For how much of the SQL processing time was the address space in a WAIT?

Because source program mapping data is available, by entering the **P** line command on any sequence number, Application Performance Analyzer displays the source code for the statement (as shown in Figure 11-77) for the sequence number S0007.

Note: The Count value is the number of samples taken of the execution of the SQL statement while in WAIT.

This applies to DB2 reports F07, F08, and F09.

11.3.10 F10 - DB2 SQL/Service time by DBRM

F10 shows exact SQL call counts, total SQL service time, and total SQL processing CPU time by embedded SQL statement by DBRM, as shown in Figure 11-78.

Note: DB2+ must be turned on.

```
  File  View  Navigate  Help
F10: SQL CPU/Service Time by DBRM (4811)                    Row 00001 of 00015
Command ===> _____________________________________________ Scroll ===> CSR

                                  Nbr of     --CPU Time--      --Svc Time--
Name      Stmt#  SQL Function    SQL Calls   Total    Mean      Total    Mean

TRADERD                         289,618     13.77   0.00004   17.67   0.00006
→ S0007     825  UPDATE          28,980       2.46   0.00008    3.12   0.00010
              > UPDATE CUSTOMER_DETAILS SET NO_SHARES = : H WHERE
              > CUSTOMER = : H AND COMPANY = : H

→ S0003     854  FETCH           28,980      1.64    0.00005    2.11   0.00007
→ S0004     742  FETCH           28,980      1.59    0.00005    2.06   0.00007
→ S0008     774  FETCH           28,889      1.49    0.00005    1.87   0.00006
→ S0001     851  OPEN            28,980      1.19    0.00004    1.52   0.00005
→ S0002     740  OPEN            28,980      1.12    0.00003    1.44   0.00004
→ S0011     772  OPEN            28,889      1.07    0.00003    1.37   0.00004
→ S0006     761  CLOSE           28,980      1.06    0.00003    1.40   0.00004
→ S0009     889  CLOSE           28,980      1.06    0.00003    1.36   0.00004
→ S0010     792  CLOSE           28,889      1.04    0.00003    1.36   0.00004
→ S0005     804  INSERT             91       0.01    0.00011    0.01   0.00015

MA  a                                                              10/004
```

Figure 11-78 F10: SQL CPU/Service Time by DBRM (expanded)

Note: Remember that measured CPU time applies only to the region being measured. DB2 executes in multiple address spaces and CPU could also be consumed in other DB2 regions not reflected in this report.

Because source program mapping data is available, by entering the **P** line command on any sequence number, Application Performance Analyzer displays the source code for the statement (as shown in Figure 11-79) for the sequence number S0007.

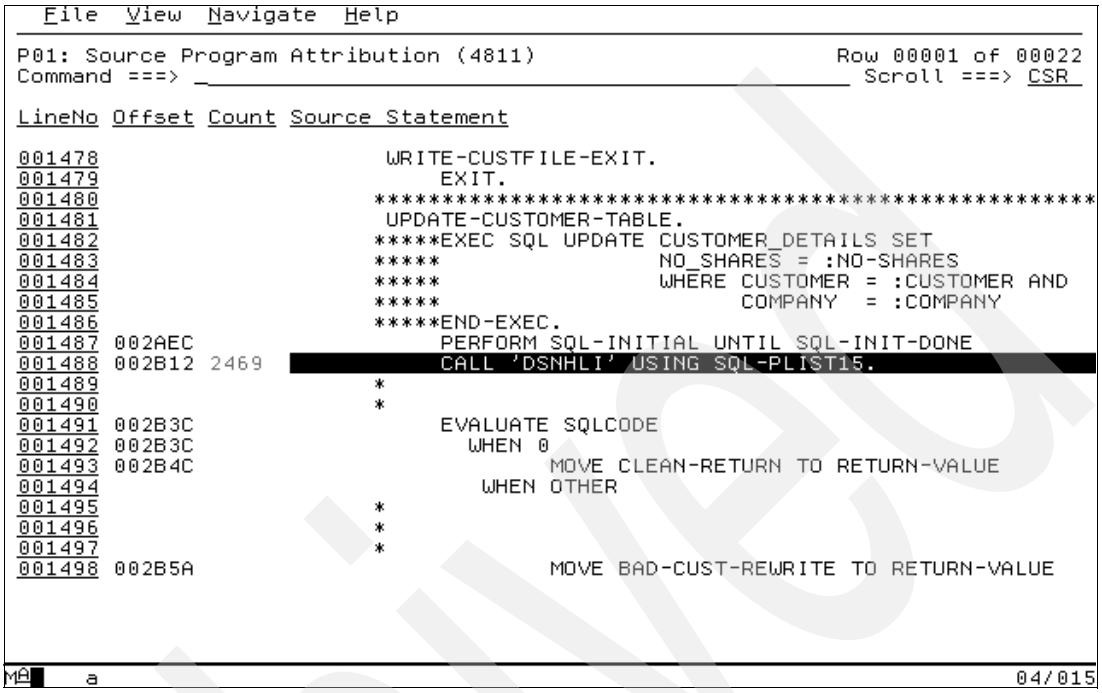


Figure 11-79 P01: Source Program Attribution (from F10)

Note: The Count value is the CPU time in milliseconds.

This applies to DB2 reports F10, F11, and F12.

11.3.11 F11 - DB2 SQL/Service time by statement

F11 shows exact SQL call counts, total SQL service time, and total SQL processing CPU time by embedded SQL statement, as shown in Figure 11-80.

Note: DB2+ must be turned on.

```
  File  View  Navigate  Help
F11: SQL CPU/Service Time by Statement (4811)              Row 00001 of 00014
Command ===> _____________________________________________ Scroll ===> CSR

                                    Nbr of    --CPU Time--      --Svc Time--
SeqNo Name      Stmt£ SQL Function SQL Calls  Total   Mean      Total   Mean

S0007 TRADERD     825 UPDATE         28,980    2.46   0.00008    3.12   0.00010
               > UPDATE CUSTOMER_DETAILS SET NO_SHARES = : H WHERE
               > CUSTOMER = : H AND COMPANY = : H

S0003 TRADERD     854 FETCH          28,980    1.64   0.00005    2.11   0.00007
S0004 TRADERD     742 FETCH          28,980    1.59   0.00005    2.06   0.00007
S0008 TRADERD     774 FETCH          28,889    1.49   0.00005    1.87   0.00006
S0001 TRADERD     851 OPEN           28,980    1.19   0.00004    1.52   0.00005
S0002 TRADERD     740 OPEN           28,980    1.12   0.00003    1.44   0.00004
S0011 TRADERD     772 OPEN           28,889    1.07   0.00003    1.37   0.00004
S0006 TRADERD     761 CLOSE          28,980    1.06   0.00003    1.40   0.00004
S0009 TRADERD     889 CLOSE          28,980    1.06   0.00003    1.36   0.00004
S0010 TRADERD     792 CLOSE          28,889    1.04   0.00003    1.36   0.00004
S0005 TRADERD     804 INSERT             91    0.01   0.00011    0.01   0.00015

MA    a                                                              09/002
```

Figure 11-80 F11: SQL CPU/Service Time by Statement

Because source program mapping data is available, by entering the **P** line command on any sequence number, Application Performance Analyzer displays the source code for the statement (as shown in Figure 11-80) for the sequence number S0007.

Note: The Count value is the CPU time in milliseconds.

This applies to DB2 reports F10, F11, and F12.

11.3.12 F12 - DB2 SQL/Service time by plan

F12 shows exact SQL call counts, total SQL service time, and total SQL processing CPU time by embedded SQL statement by DB2 Plan, as shown in Figure 11-81.

Note: DB2+ must be turned on.

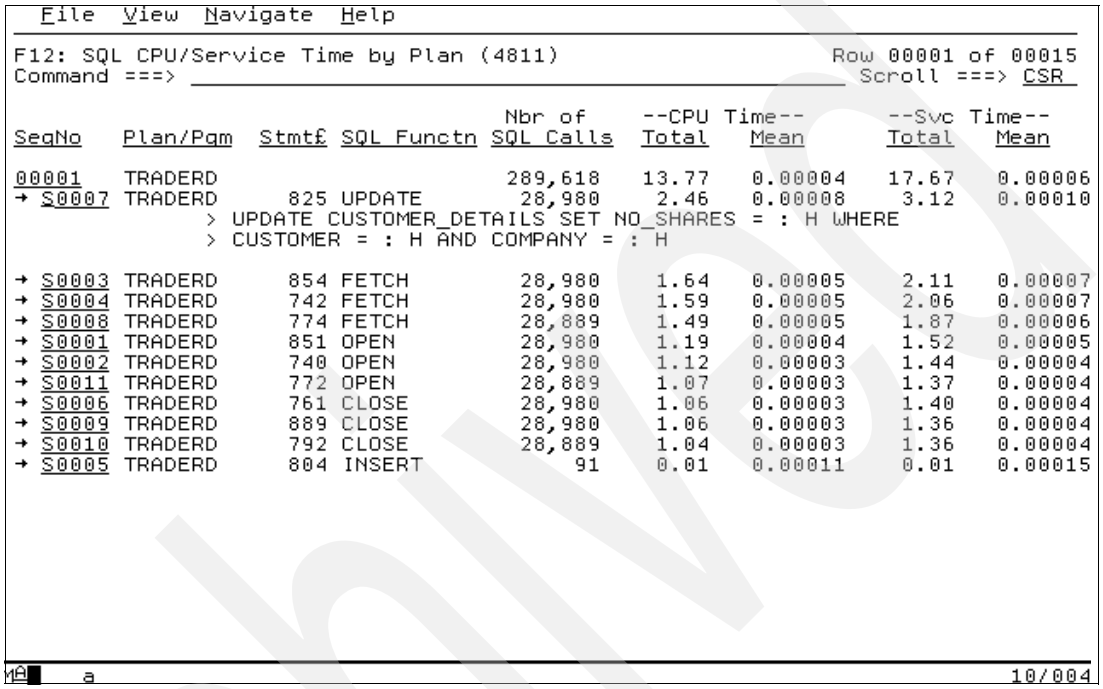


Figure 11-81 F12: SQL CPU/Service Time by Plan (expanded)

Because source program mapping data is available, by entering the **P** line command on any sequence number, Application Performance Analyzer displays the source code for the statement (as shown in Figure 11-81) for the sequence number S0007.

Note: The Count value is the CPU time in milliseconds.

This applies to DB2 reports F10, F11, and F12.

11.3.13 F13 DB2 - SQL threads analysis

F13 shows information about DB2 threads observed during the sampling of SQL call activity, as shown in Figure 11-82.

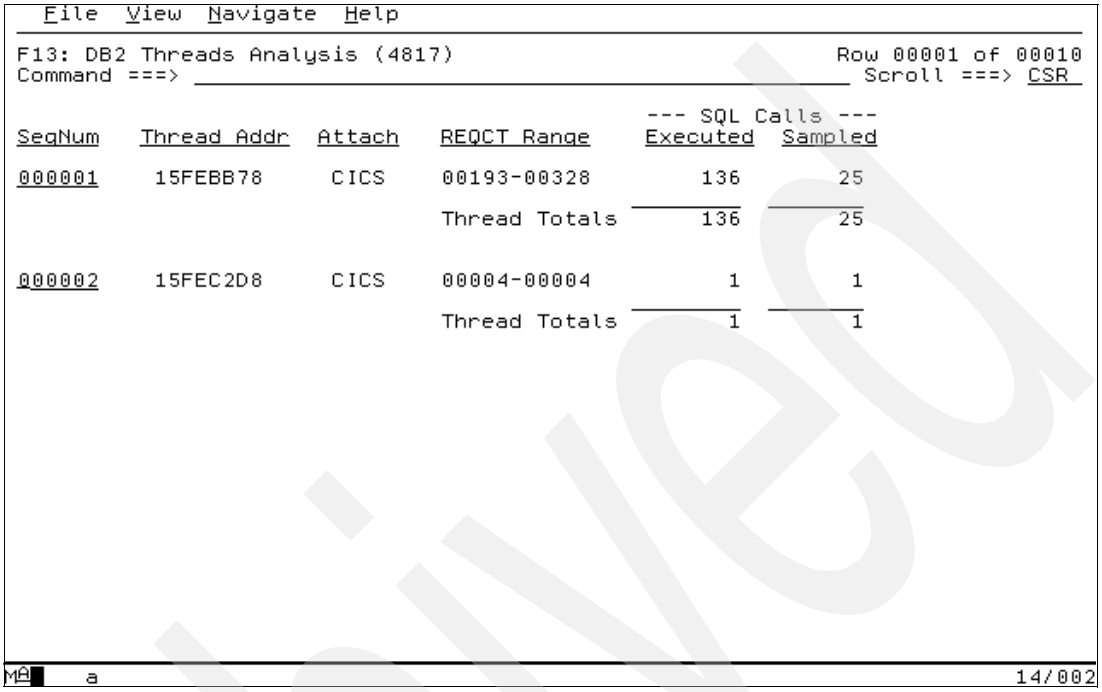


Figure 11-82 F13: DB2 - SQL Thread Analysis

11.3.14 F14 - DB2 CPU by plan/stored procedure

F14 shows how CPU resource was consumed by each stored procedure measured during the sampling interval. The percentage of time is reported for each DB2 plan under which measured stored procedure activity was recorded. Expand a plan line to see a further breakdown of time consumption by category, as shown in Figure 11-83.

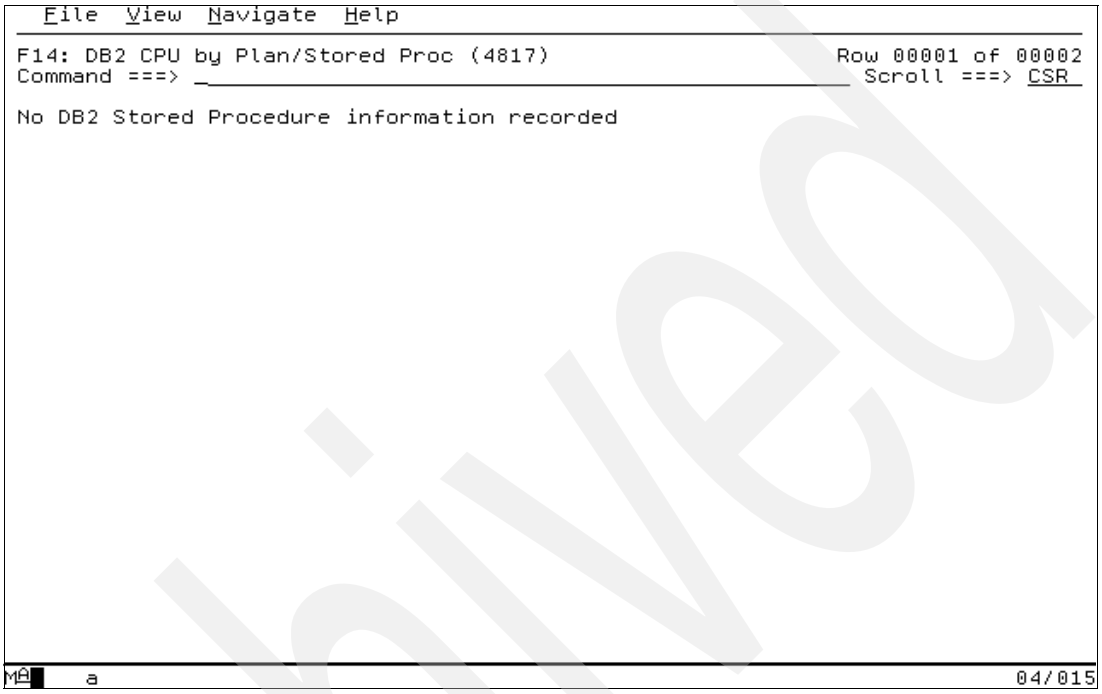


Figure 11-83 F14: DB2 CPU by Plan/Stored Procedure

11.4 MQ measurement

Because the TRADER application has no MQSeries command, to illustrate MQ reports, we use a specific CICS/MQSeries transaction.

We do not spend much effort on the things that are common to Application Performance Analyzer and that have been described in the previous chapter, but rather focus on the unique aspects of using Application Performance Analyzer with the batch job accessing MQSeries resources.

11.4.1 Observation session request definition

We have to monitor the CICS region named CICSC23G with the following characteristics:

- We use a sampling rate of 1,000 samples per second.
- We use a duration of 1 minute.
- The measurement file has to be retained for 30 days.
- Only MP* transactions have to be recorded.
- CHABERT has to be notified at the end of the measurement.

By entering the NEW command line on the Observation List (R02) panel, Application Performance Analyzer displays the Schedule New Measurement (R03) panel. CICS and MQseries data extractor has to be turned on to get both CICS and MQSeries data.

The MQ data extractor provides the ability to observe/sample and report on MQSeries interface calls (both dynamic and static) in batch, IMS, and CICS programs. More specifically, to show the CPU and wait time spent in MQSeries interface calls and to attribute the time spent to a particular MQSeries interface call. When the MQSeries data extractor is selected, Application Performance Analyzer records the following information in the sample file for each MQSeries call that it observes:

- Environment (batch, IMS, or CICS)
- Load module that issued the call
- Offset within the load module of the return address from the MQ call
- Type of MQI call (MQOPEN, MQGET, and so on)
- Queue Manager name
- Object name (for example, the queue name)
- Message size (actual length for MQPUT/MQPUT1, buffer length for MQGET)
- MQ Options (for MQOPEN, MQGET, MQPUT MQCLOSE)
- Message type, priority, and persistence
- Transaction ID (CICS and IMS)

The settings required to monitor the MP* CICS/MQSeries transactions are shown in Figure 11-84 for panel 1 (Job Information), Figure 11-85 for panel 2 (Options), and Figure 11-86 for panel 5 (CICS Txns).

```
 File  View  Navigate  Help
R03: Schedule New Measurement                          Row 00001 of 00012
Command ===> _____________________________________________ Scroll ===> CSR

● 1. Job Information    3. Multi Steps     5. CICS Txns     7. Schedule
● 2. Options            4. Active Jobs     6. Sysplex       8. Sched Options

Panel 1. Job Information                  Input more data or ENTER to submit

  Job Name/Pattern . . CICSC23F    System Name . . . *________
                       (Active)
  Step Specification
    Step No. . . . . . ____        Specify step number, program name,
    Program Name . . . ________    step name or step name + Proc step
    Step Name  . . . . ________    name. Use panel 3 to specify more
    ProcStepName . . . ________    than one step.

  Description  . . . . cics/mq_____________________________
  Number of Samples  . 1000___     Measure to step end . . . Y
  Duration (min:sec) . 1:00___     Delay by (secs) . . . . . ______
  Notify TSO User  . . CHABERT     Retain file for (days)  . ____

MA  a                                                          14/025
```

Figure 11-84 R03: Schedule New Measurement (panel 1: Job Information)

With panel 1 we specify:

- Job name
- Sample rate
- Retention period

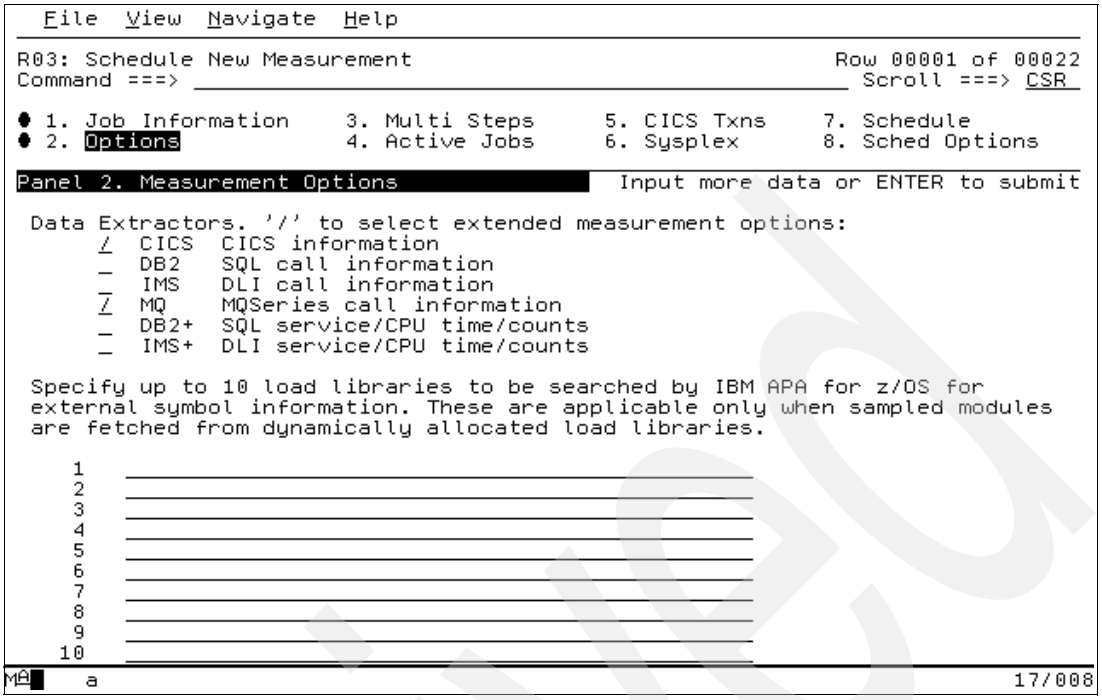


Figure 11-85 R03: Schedule New Measurement (panel 2: Options)

With panel 2 we turn on the CICS and MQSeries data extractors.

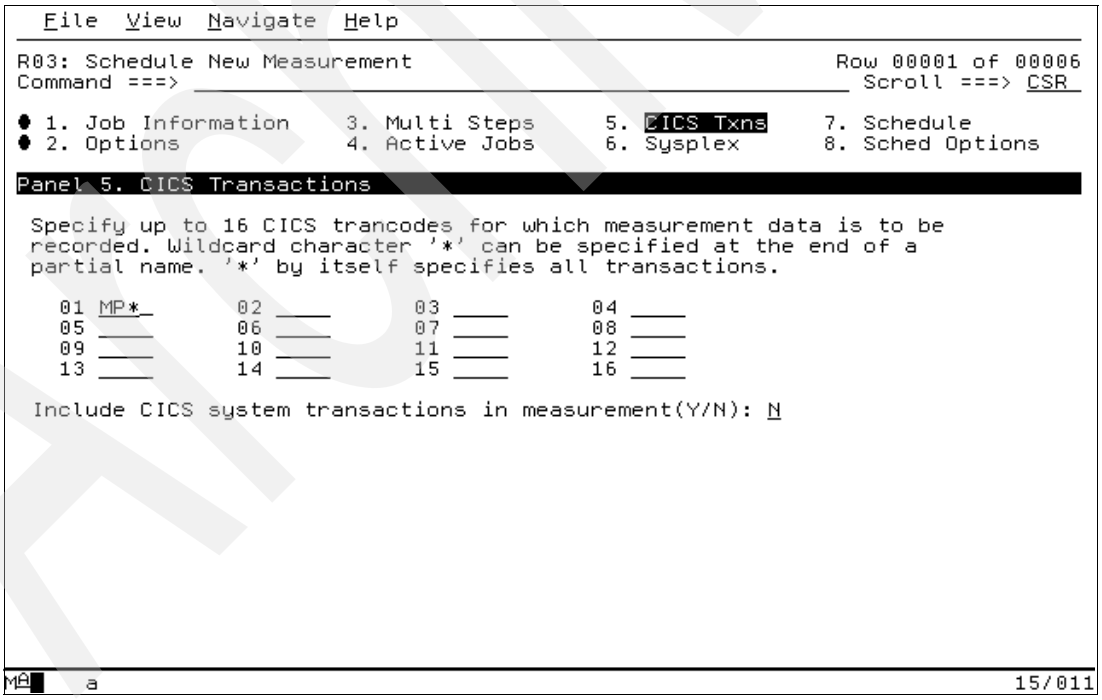


Figure 11-86 R03: Schedule New Measurement (panel 5: CICS Transactions)

With panel 5 we select which transactions have to be recorded.

From a CICS screen we start several MPUT transactions to run the application. Then using the **R** line command, Application Performance Analyzer launches the associated performance reports.

Because `Q MQ Measurement` is displayed in red (in gray on our graphic), MQSeries reports are available. To get the list of available MQ reports, shown in Figure 11-87, type **Q** on the command line and press Enter.

```
  File  View  Navigate  Help
R01: IBM APA for z/OS Performance Reports (4763)          Row 00001 of 00007
Command ===> _                                             Scroll ===> CSR

  Select a category from     _ A Admin/Miscellaneous  _ I IMS Measurement
  the list to the right      _ S Statistics/Storage   _ E CICS Measurement
  to view the available      _ C CPU Usage Analysis   _ F DB2 Measurement
  reports in the selection   _ D DASD I/O Analysis    _ Q MQ Measurement
  list below.                _ W CPU WAIT Analysis    _ G Coupling Facility

Enter S to make a selection or enter the report code on the command line

  _ Q01 MQSeries Activity Summary       _ Q07 MQSeries Serv Time by Txn
  _ Q02 MQSeries CPU Usage by Queue     _ Q08 MQSeries Wait Time by Queue
  _ Q03 MQSeries CPU Usage by Request   _ Q09 MQSeries Wait Time by Request
  _ Q04 MQSeries CPU Usage by Txn       _ Q10 MQSeries Wait Time by Txn
  _ Q05 MQSeries Serv Time by Queue
  _ Q06 MQSeries Serv Time by Request

MA  a                                                          04/015
```

Figure 11-87 R01: Performance Reports (MQ reports selected)

11.4.2 Q01 - MQSeries activity summary

Q01 shows a summary of the MQSeries requests (calls) issued during the observation session and a list of the MQSeries objects referenced by these requests, as shown in Figure 11-88.

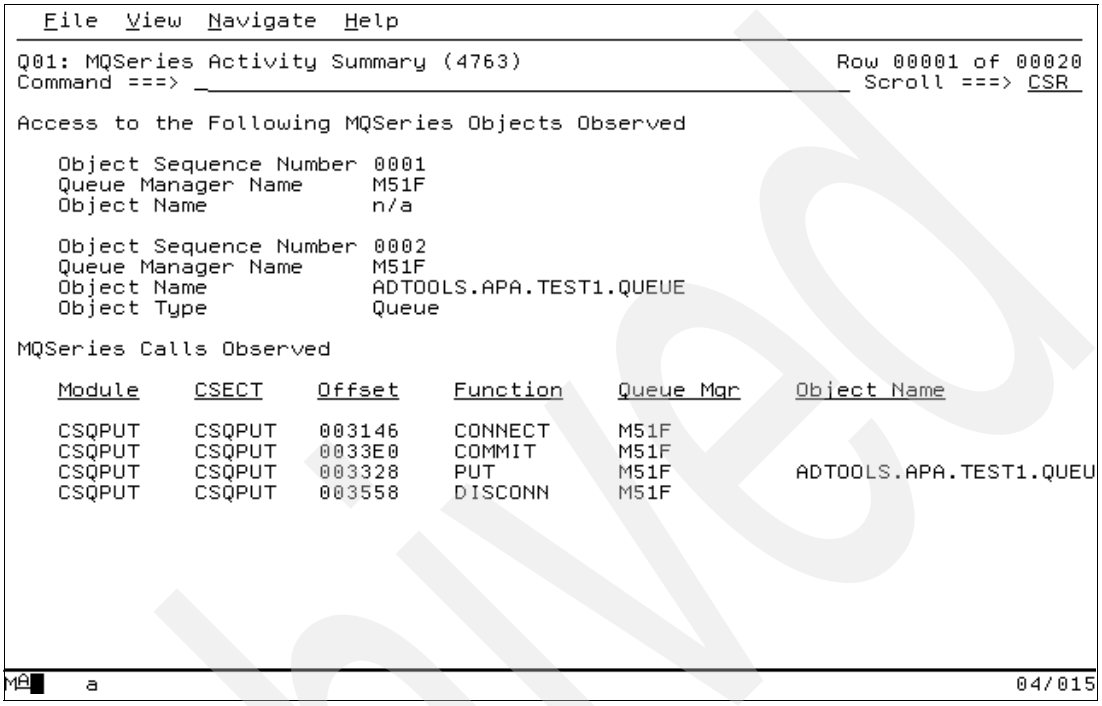


Figure 11-88 Q01: MQSeries Activity Summary

11.4.3 Q02 - MQSeries CPU usage by queue

Q02 shows how CPU resources were consumed by MQSeries requests. The percentage of CPU usage is reported by MQSeries Queue Name, as shown in Figure 11-89.

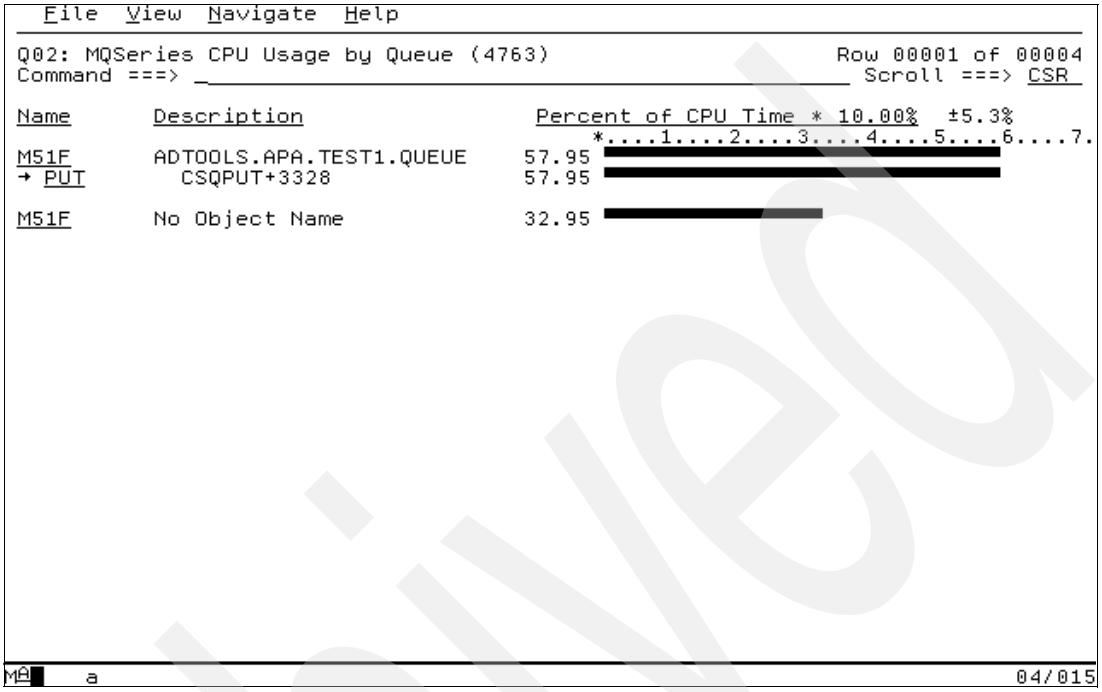


Figure 11-89 Q02: MQSeries CPU Usage by Queue (expanded)

We can enter the ++ line command (or press Enter) on any MQSeries function and a pop-up window containing additional information is displayed, as shown in Figure 11-90.

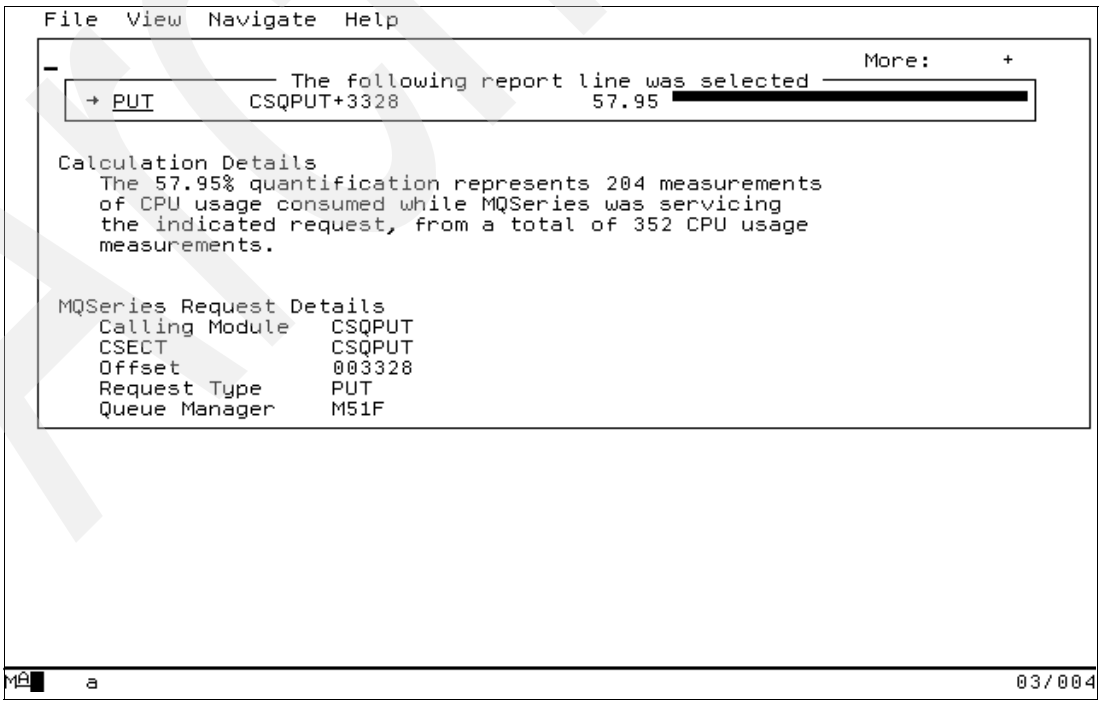


Figure 11-90 Detail about MQSeries PUT function

11.4.4 Q03 - MQSeries CPU usage by request

Q03 shows how CPU resources were consumed by MQSeries requests. The percentage of CPU usage is reported by MQSeries Request, as shown in Figure 11-91.

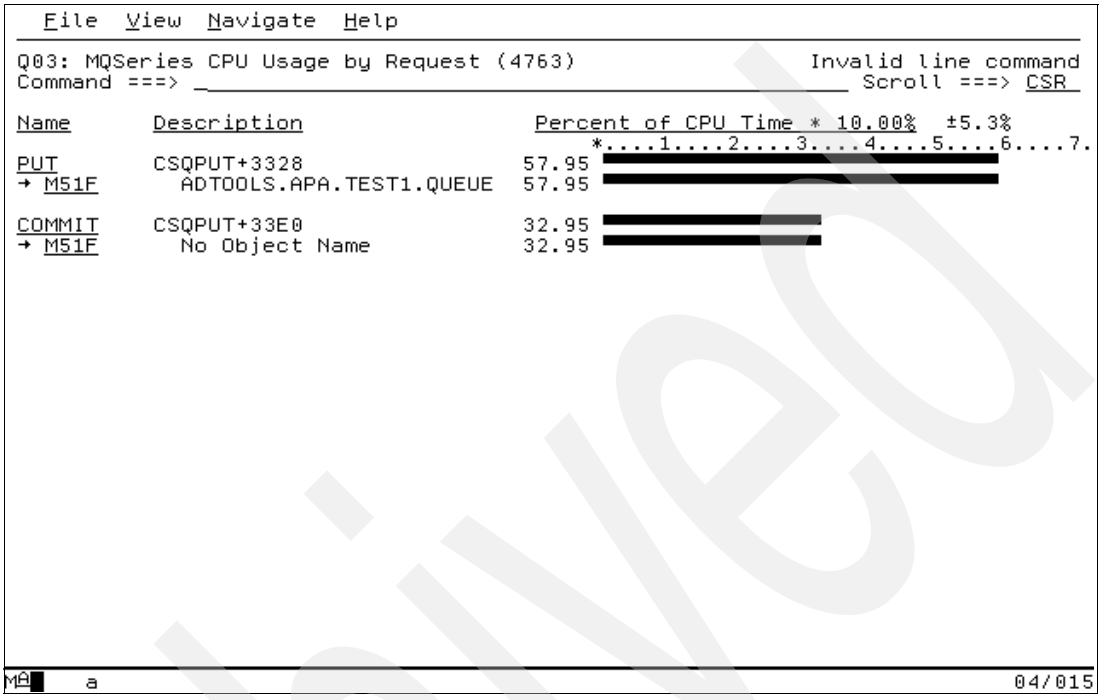


Figure 11-91 Q03: MQSeries CPU Usage by Request (expanded)

We can enter the ++ line command (or press Enter) on any MQSeries queue name and a pop-up window containing additional information is displayed, as shown in Figure 11-92.

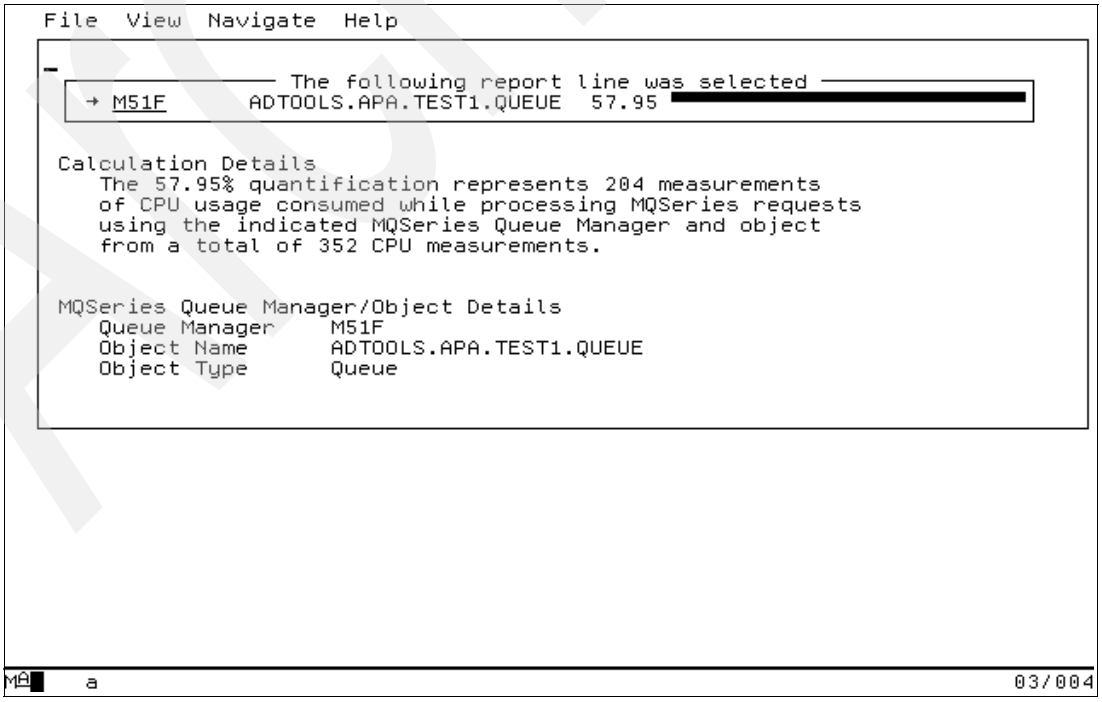


Figure 11-92 Detail about MQSeries M51F queue

11.4.5 Q04 - MQSeries CPU usage by transaction

Q04 shows how CPU resources were consumed by MQSeries Requests. The percentage of CPU usage is reported by transaction, as shown in Figure 11-93.

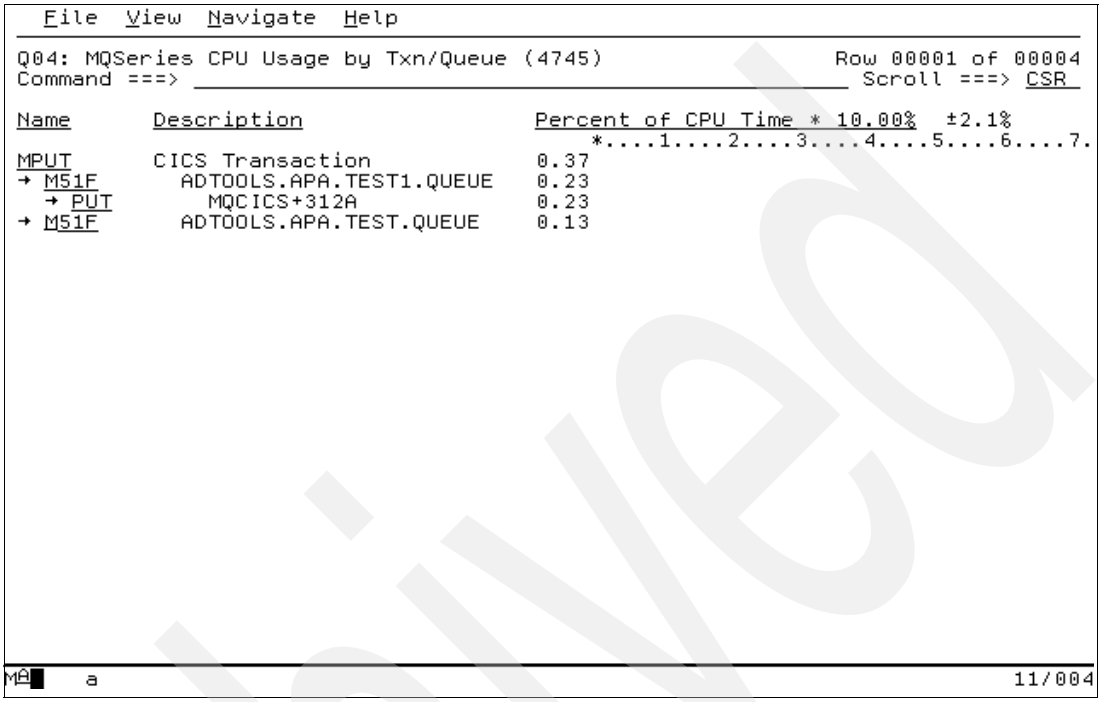


Figure 11-93 Q04: MQSeries CPU Usage by Transaction/Queue (expanded)

We can enter the ++ line command (or press Enter) on any CICS Transaction and a pop-up window containing additional information is displayed, as shown in Figure 11-94.

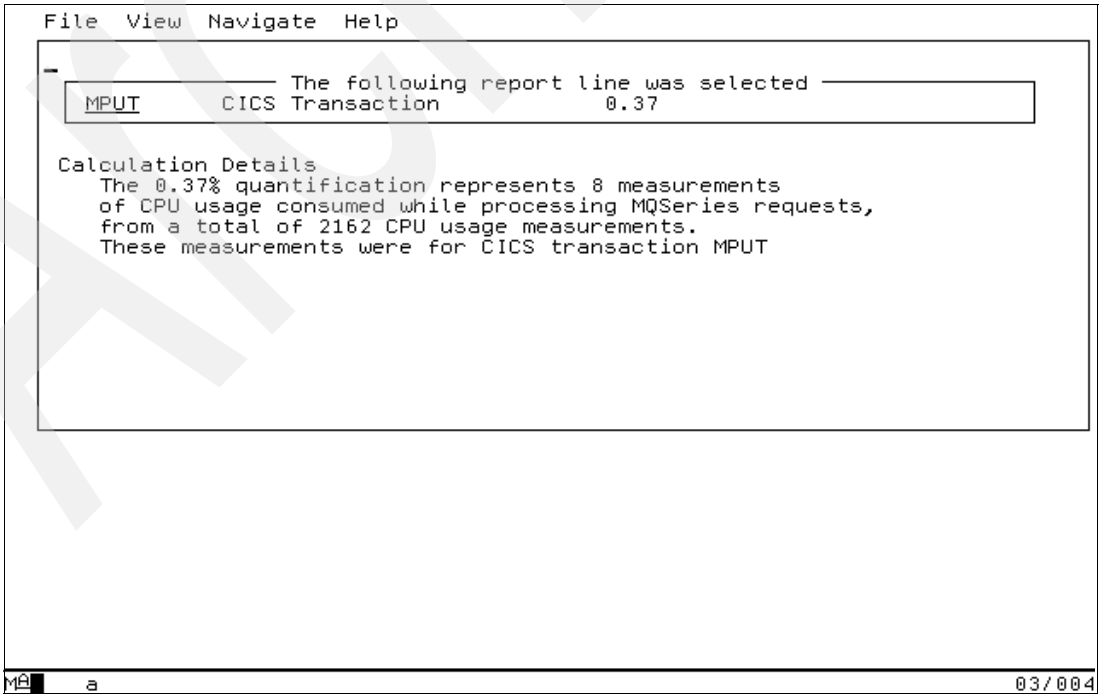


Figure 11-94 Detail about MPUT CICS transaction

11.4.6 Q05 - MQSeries service time by queue

Q05 shows how time was consumed by requests and by queue, as shown in Figure 11-95.

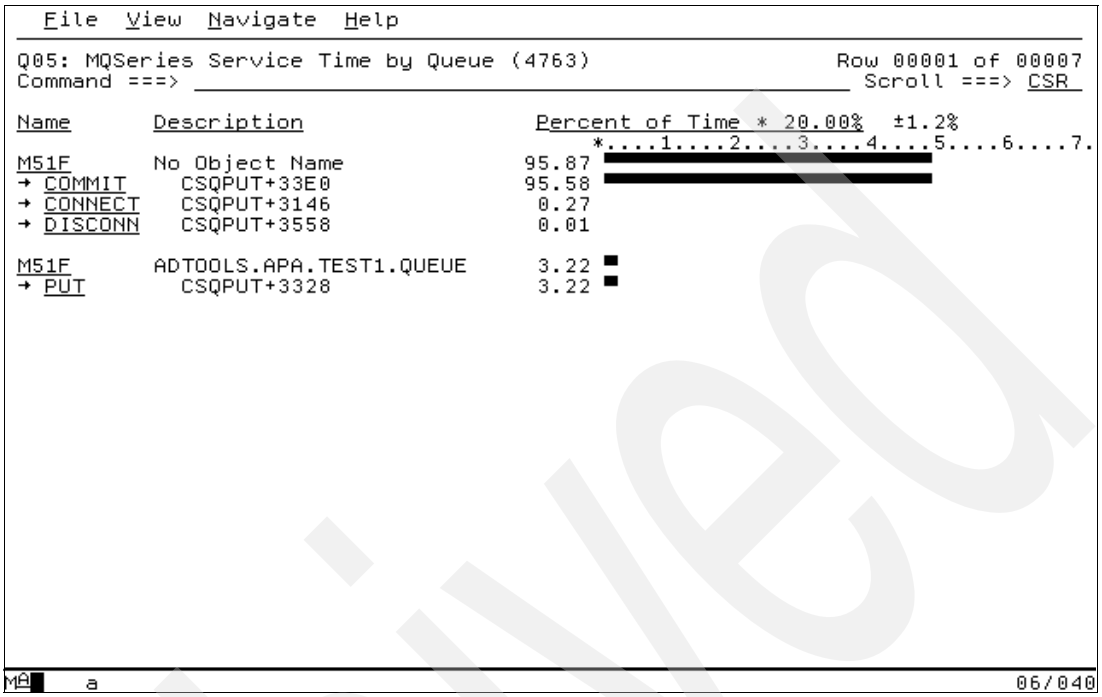


Figure 11-95 Q05: MQSeries Service Time by Queue (expanded)

11.4.7 Q06 - MQSeries service time by request

Q06 shows how time was consumed by requests reported by request, as shown in Figure 11-96.

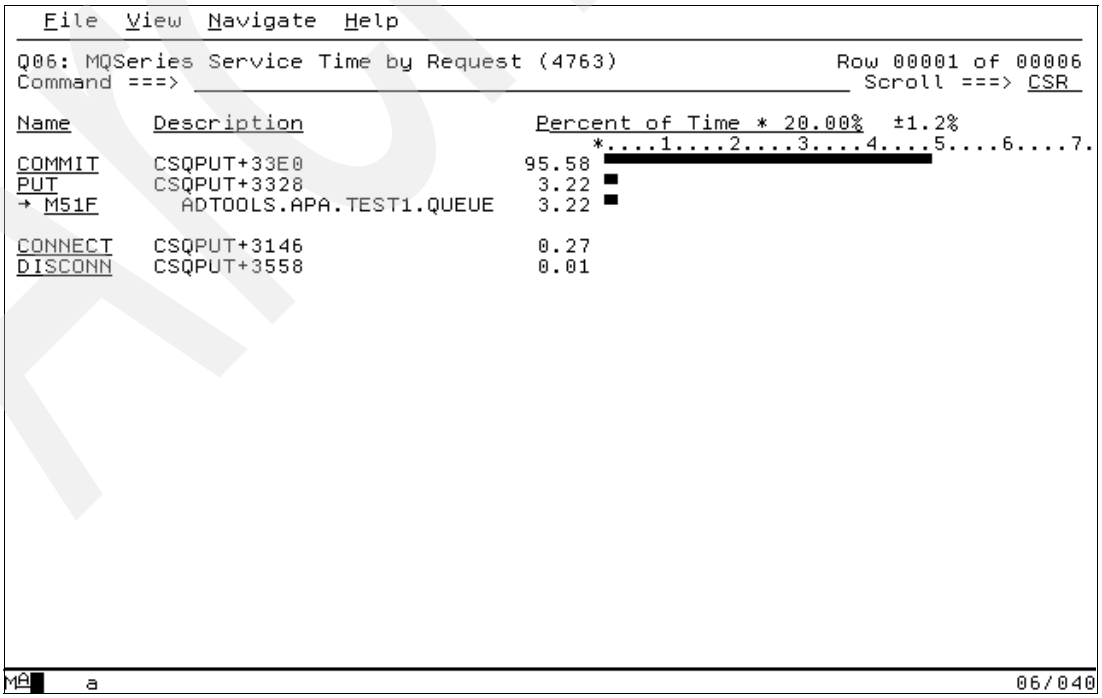


Figure 11-96 Q06: MQSeries Service Time by Request (expanded)

11.4.8 Q07 - MQSeries service time by transaction

Q07 shows how time was consumed by requests reported by Txn, as shown in Figure 11-97.

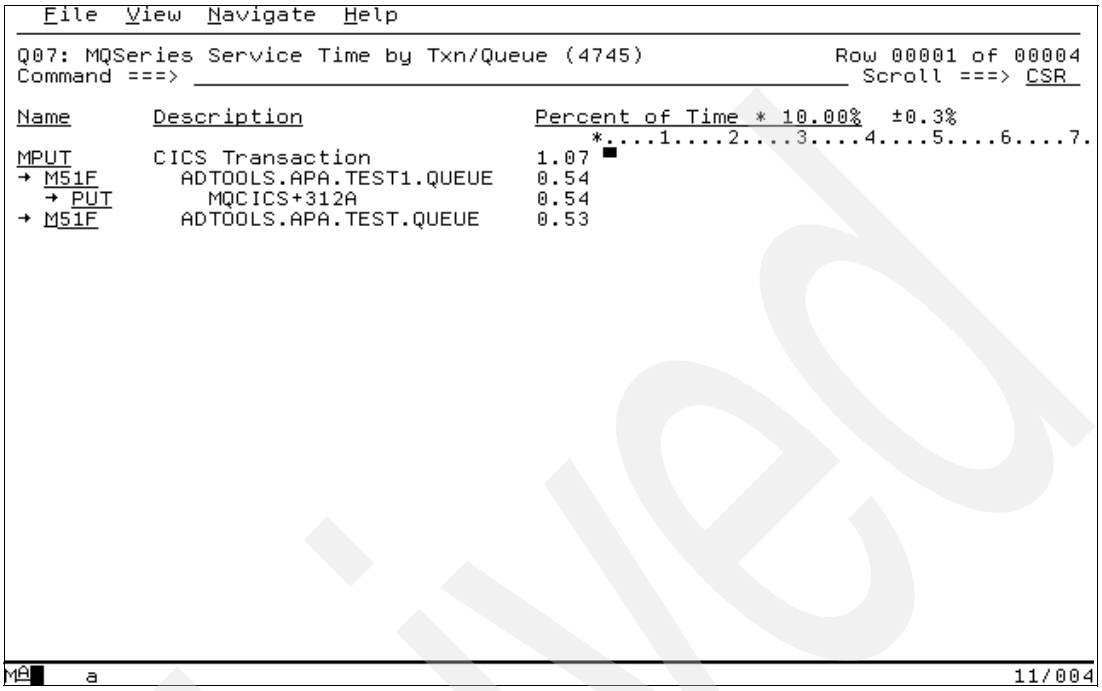


Figure 11-97 Q07: MQSeries Service Time by Transaction (expanded)

11.4.9 Q08 - MQSeries wait time by queue

Q08 shows how much wait time occurred during processing of MQSeries Requests reported by MQSeries Queue Name, as shown in Figure 11-98.

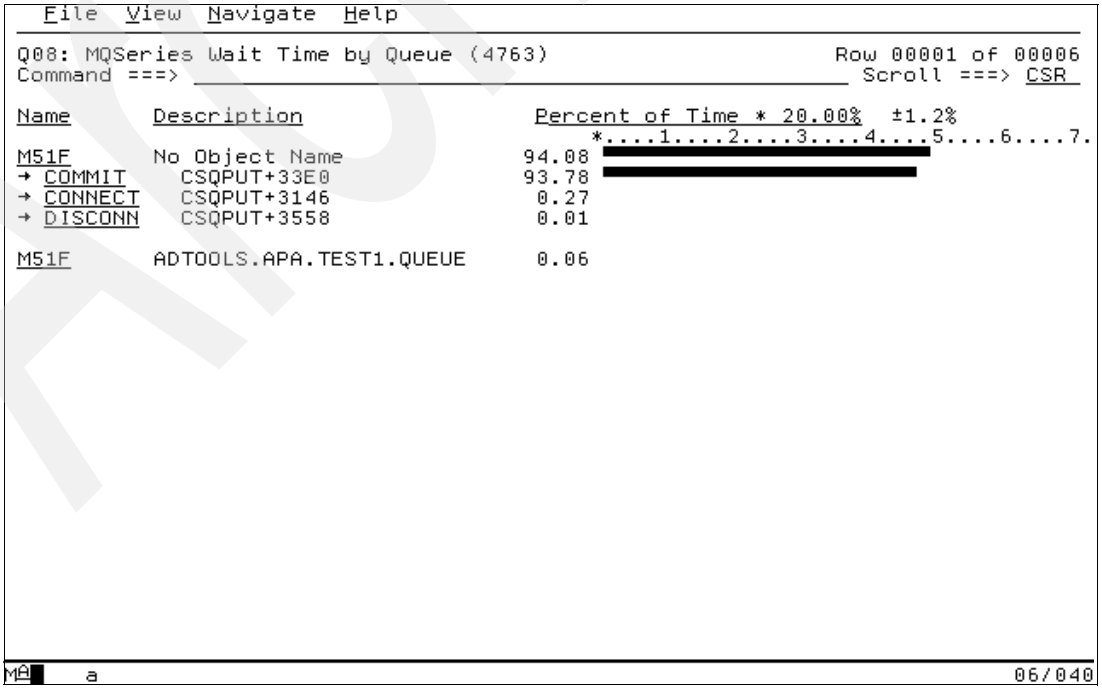


Figure 11-98 Q08: MQSeries Wait Time by Queue (expanded)

11.4.10 Q09 - MQSeries wait time by request

Q09 shows how much wait time occurred during processing of MQSeries Requests reported by MQSeries Request, as shown in Figure 11-99.

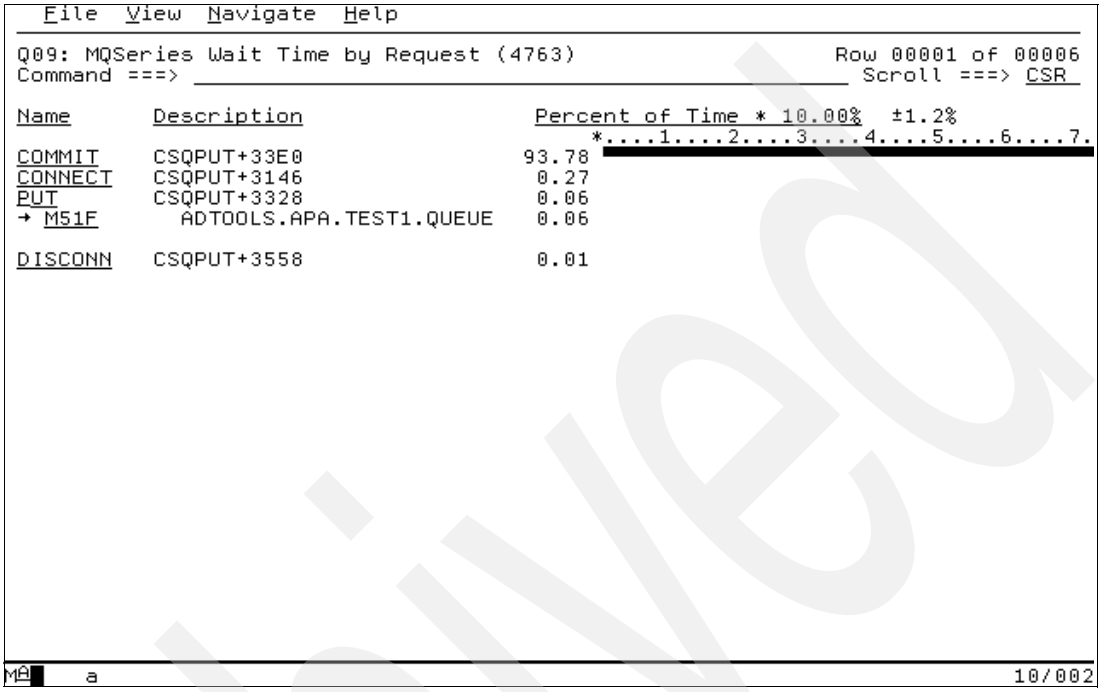


Figure 11-99 Q09: MQSeries Wait TIme by Request (expanded)

11.4.11 Q10 - MQSeries wait time by transaction

Q10 shows how much wait time occurred during processing of MQSeries Requests reported by transaction, as shown in Figure 11-100.

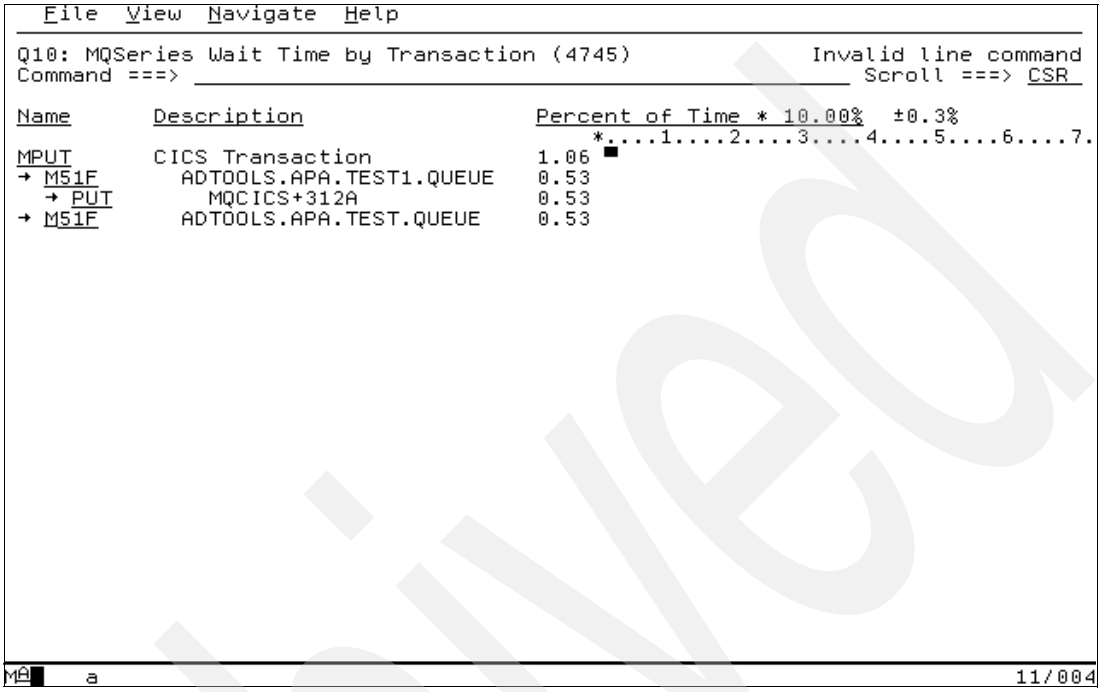


Figure 11-100 Q10: MQSeries Wait Time by Transaction (expanded)

12

Application Performance Analyzer: More about reports

In this chapter we describe the following topics:

- Real time monitoring
- How to print reports
- How to share reports between several sites
- How to start an observation session from Omegamon

12.1 Real-time analysis

In this section we discuss real-time analysis.

12.1.1 SETUP command line

The Realtime Monitor facility lets us view information about an in-progress measurement. Start this facility by selecting an active measurement from R02: Observation List using the **R** line command.

Note: The SETUP command line allows you to have the Realtime Monitor launched automatically when you start a measurement for an active job.

In this mode the panel is refreshed automatically to show changing data as it is measured. In auto-refresh mode, the keyboard is locked. To halt auto-refresh mode and unlock the keyboard, press the Attention key. You can then refresh the panel manually by repeatedly pressing Enter. Use the SETUP command line to set these options, as shown in Figure 12-1.

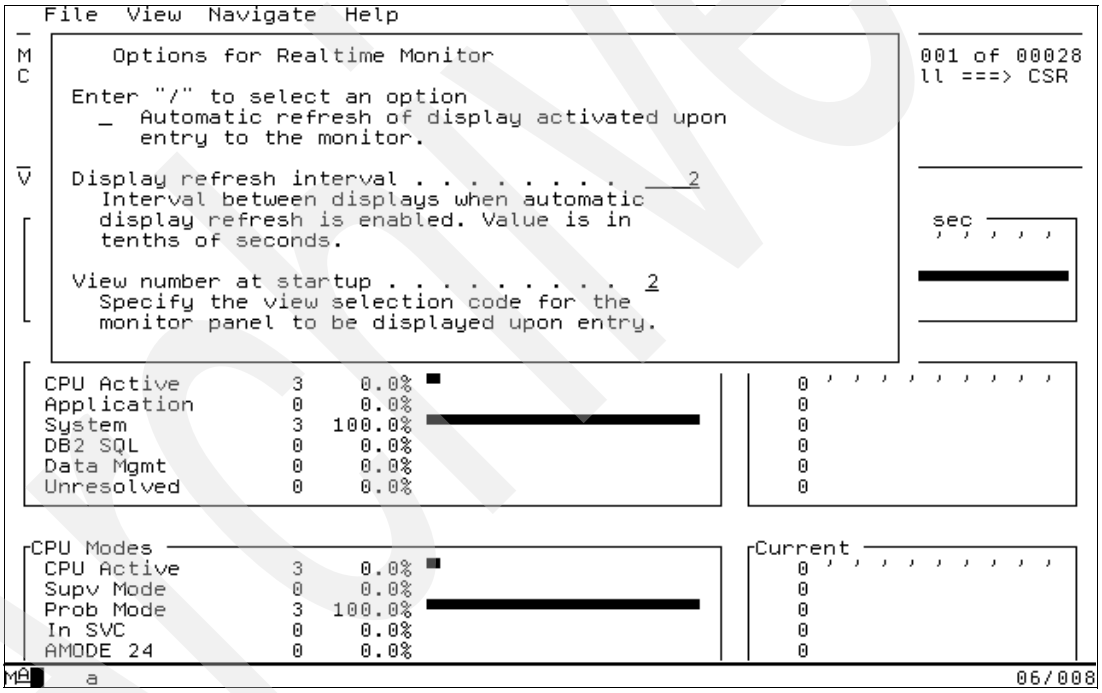


Figure 12-1 Real Time: SETUP command line

Use the SETUP command line to customize the following:

- Panel displayed at startup
- Display auto-refresh enabled
- Auto-refresh interval
- Length of current time slice

12.1.2 View 1: Measurement overview

View 1 provides an at-a-glance summary of the measurement status and shows a very high-level overview of observed resource usage, as shown in Figure 12-2.

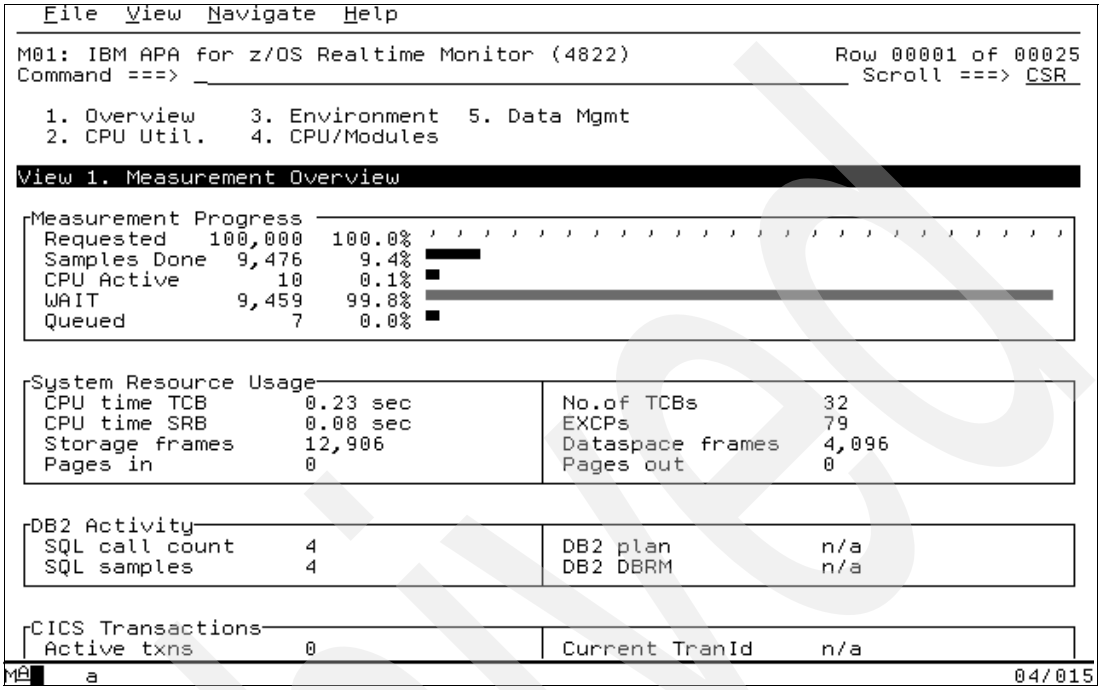


Figure 12-2 View 1: Measurement Overview (partial report)

Note: If in auto refresh mode, data in this screen refreshes automatically at the specified rate. Otherwise, press Enter to refresh the data.

This view has four sections:

- Measurement Progress:

  Shows the progress of the measurement by reporting the total number of samples completed. The sample counts are further subdivided by CPU active samples, TCB WAIT samples, and queued samples.

- System Resource Usage:

  Shows various aspects of general resource usage observed during the measurement interval. Resources quantified here are CPU time, storage usage, EXCPs, and paging.

- DB2 Activity:

  Shows information (SQL counts, SQL samples, DB2 plans, and DBRMs) about DB2 activity observed during the measurement interval.

- CICS Transaction:

  Shows information (active transactions, current TransId, suspended transactions, active Taskid) about CICS transactions that are currently active. This information is available only if the address space being measured is a CICS region and the CICS measurement feature is enabled.

12.1.3 View 2: CPU Utilization

Overall mode appears on the left side of the screen and shows accumulated quantifications based on the overall measurement. It is also expressed as a percentage and illustrated by a histogram.

Current mode appears on the right side of the screen and shows quantifications representing the last measured time slice. It is also illustrated by a histogram, as shown in Figure 12-3.

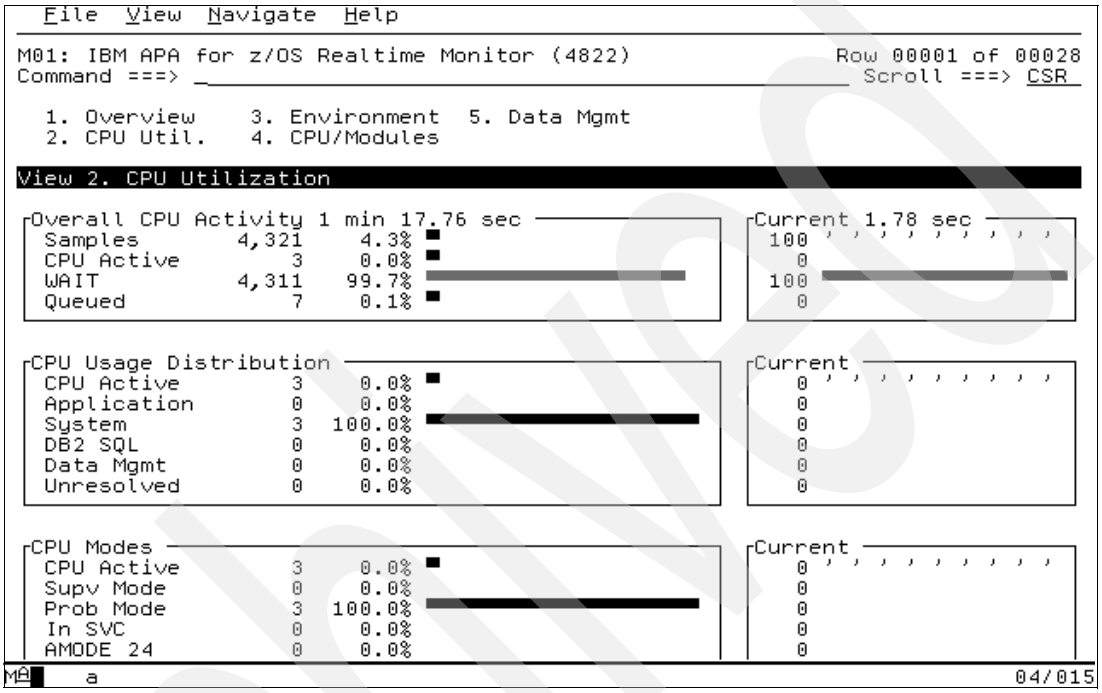


Figure 12-3 View 2: CPU Utilization (partial report)

This views has three sections:

- CPU Activity:

  Shows sample counts for the overall measurement and for the current time slice. These are categorized as CPU Active, WAIT, and Queued.

- CPU Usage Distribution:

  Shows a breakdown of CPU active TCB observations. CPU active observations are broken down by categories:

  - Application code
  - System services
  - DB2
  - Data management

- CPU Modes:

  Shows a breakdown of CPU active TCB observations by mode of CPU execution.

12.1.4 View 3: Measurement environment

The data reported here is static and shows the measurement request parameters and information about the measurement environment, as depicted in Figure 12-4.

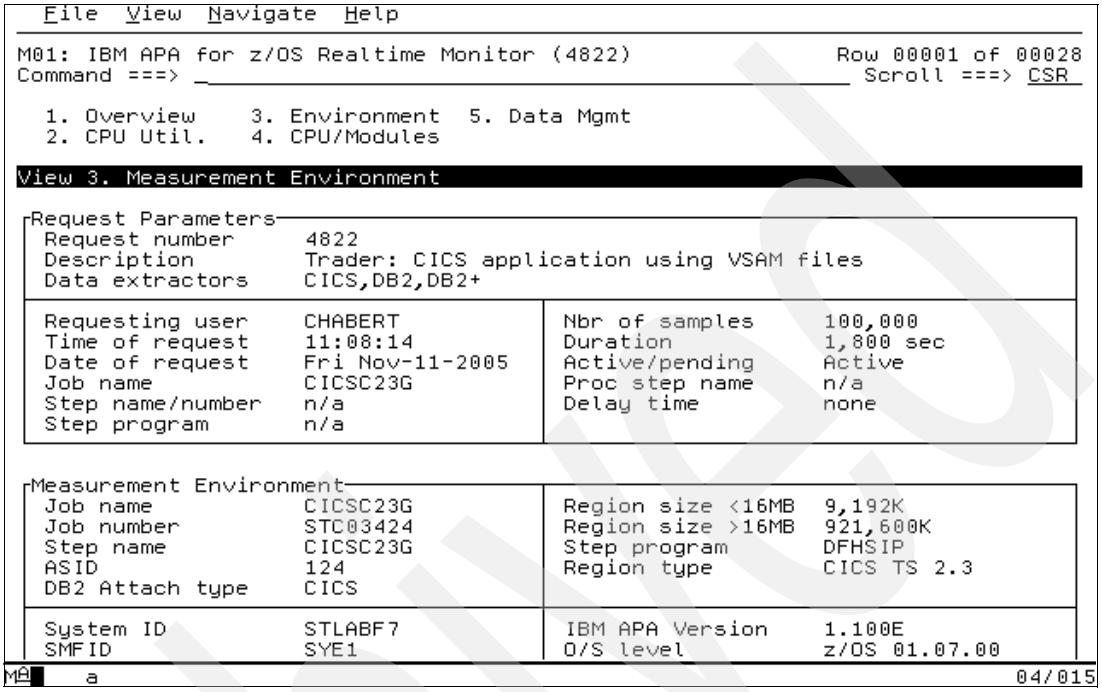


Figure 12-4 View 3: Measurement Environment (partial report)

12.1.5 View 4: CPU active modules

This view shows the number of samples in the current time slice and the number of these samples in which CPU activity was observed. The heading includes the elapsed time of the time slice, as shown in Figure 12-5.

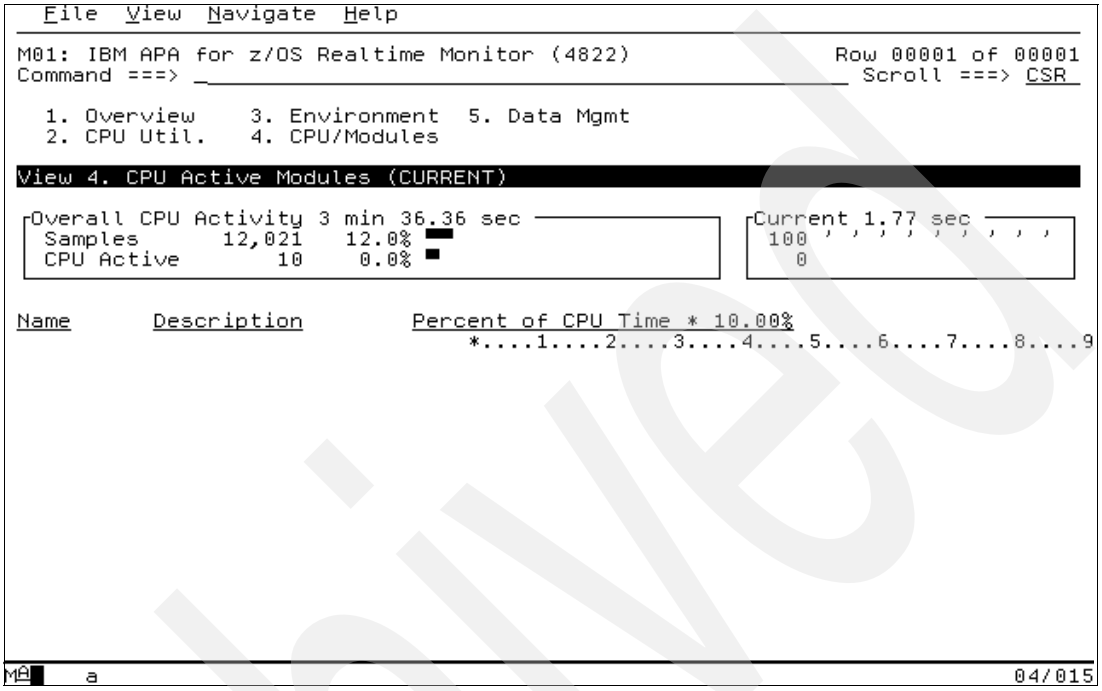


Figure 12-5 View 4: CPU Active Modules

12.1.6 View 5: Data management service time

In ACCUM mode, files for which EXCPs were processed during the measurement are shown.

In CURRENT mode, files for which EXCPs were processed since the last data refresh are shown.

In both cases, the detail lines are sorted in descending sequence by EXCP count since the last data refresh, as shown in Figure 12-6.

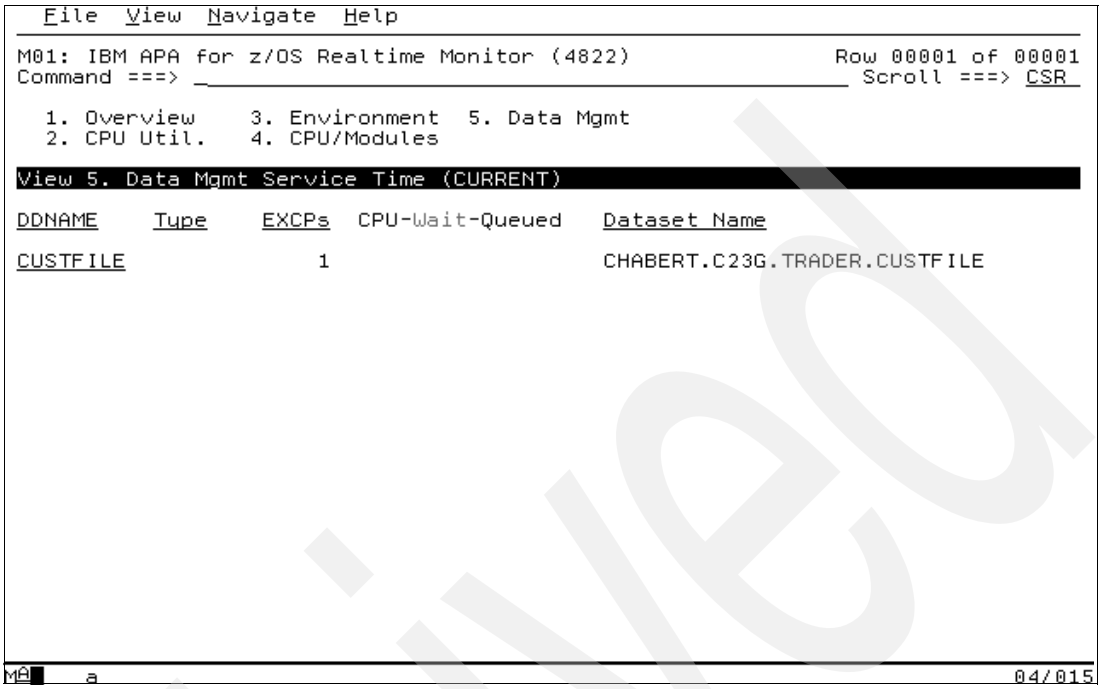


Figure 12-6 View 5: Data Mgnt Service Time

By using the ++ line command on any DDname, Application Performance Analyzer provides detail information, as shown in Figure 12-7.

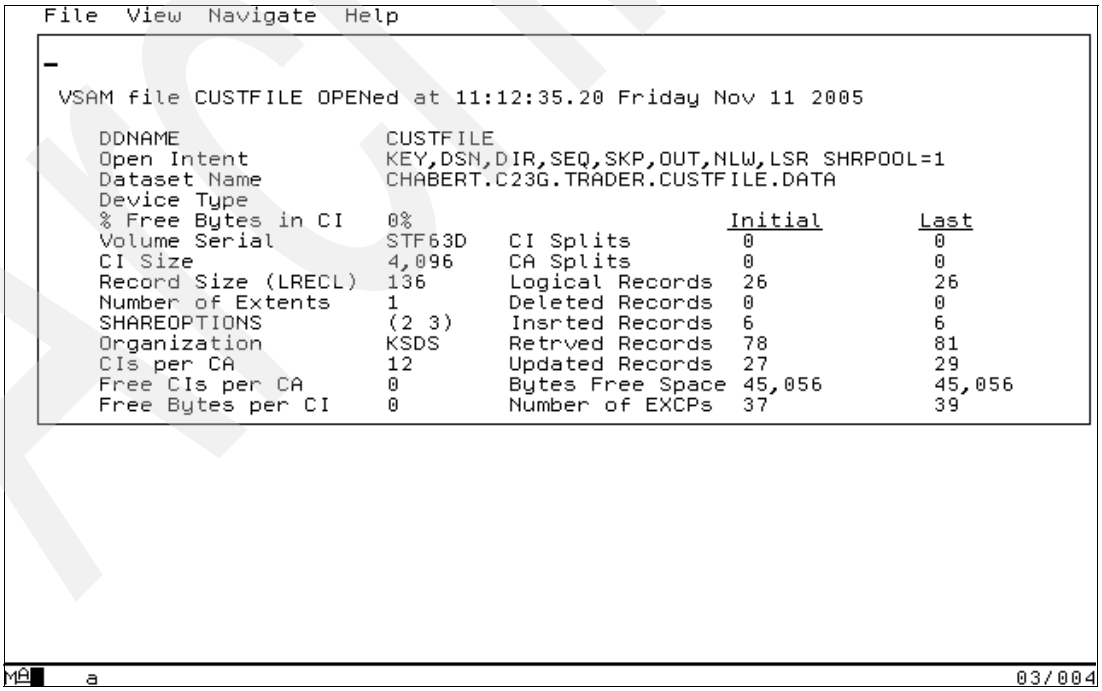


Figure 12-7 Data set information from view 5

12.2 How to print reports

We can generate a printable report, in batch, by submitting JCL. The Application Performance Analyzer (APA) ISPF interface provides ISPF screens to help us to generate and submit the JCL. We discuss this facility in the following two sections.

12.2.1 About APA's report printing feature

Performance analysis reports are available in a format suitable for printing. Application Performance Analyzer allows us to generate reports by submitting JCL that executes the program CAZPRINT.

We can use the ISPF-based report request facility to build the necessary JCL and CAZPRINT control statements. CAZPRINT can produce report output in two different formats.

Line printer format

This is a JES SYSOUT file produced by specifying a PRINT control statement and by specifying a DD name in the PRINT statement and supplying a DD statement for that name.

PDF format

We produce a PDF file by specifying a CONVERT control statement and specifying a DD name in the CONVERT statement and supplying a DD statement for that name that defines the output file.

We then have to transfer the file to a PC platform on which we can view and print the report.

12.2.2 ISPF report request facility

We first have to select an observation session by using the **R** line command on the request number from the R02: Observation List screen. Then, from report R01: Performance Reports, we select **A: Admin/Miscellaneous**, as shown in Figure 12-8.

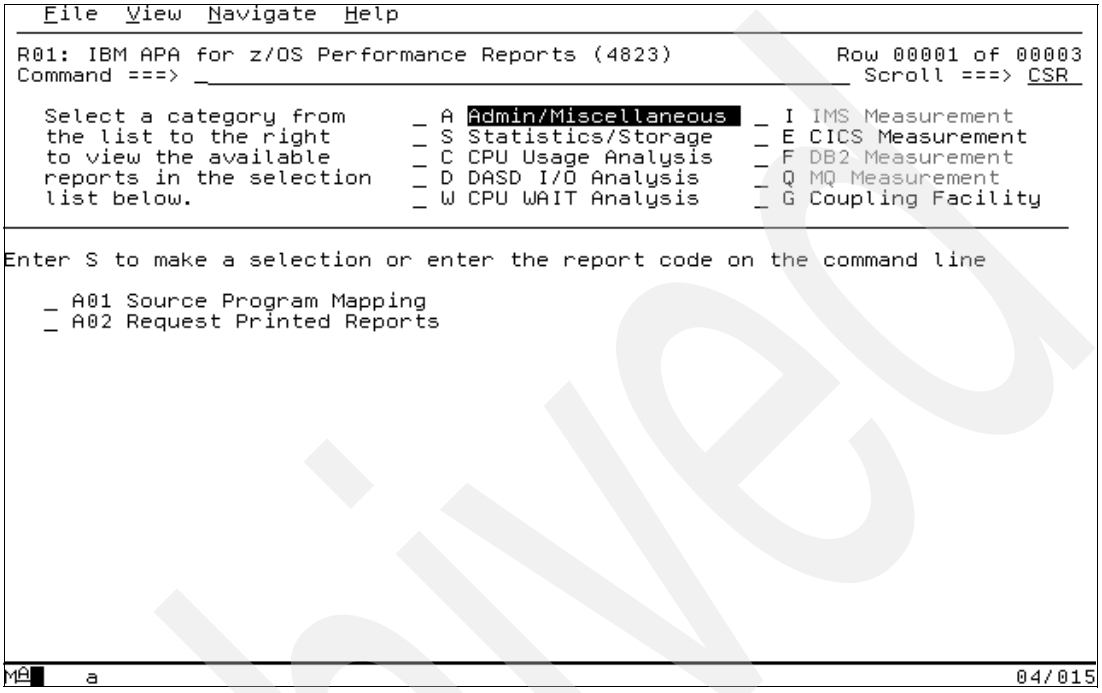


Figure 12-8 R01: Performance Reports (Admin/Miscellaneous selected)

From here we select **A02: Request Printed Reports**. A screen is then displayed listing the available report sections that we can select to be included in the report, as shown in Figure 12-9.

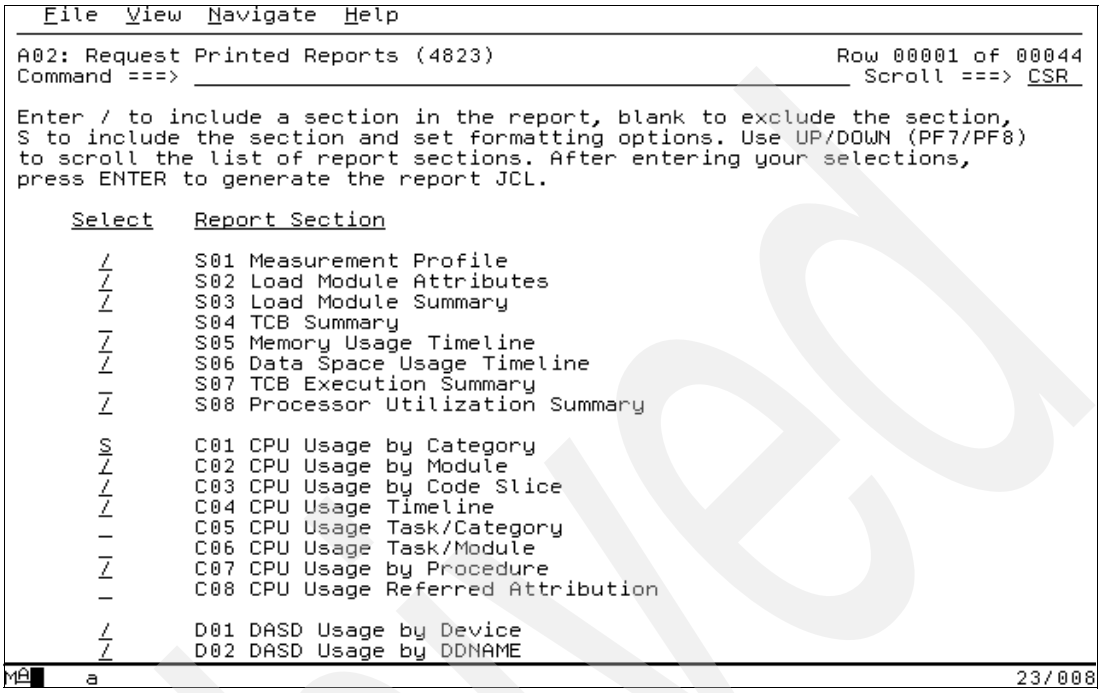


Figure 12-9 A02: Request Printed Reports

To print a report, type the / line command in front of its name. To specify formatting options, select the report section with the **S** line command instead of a / line command (as shown in Figure 12-9 on page 278) for section C01: CPU Usage by Category. A pop-up window appears in which we can select our options, as shown in Figure 12-10.

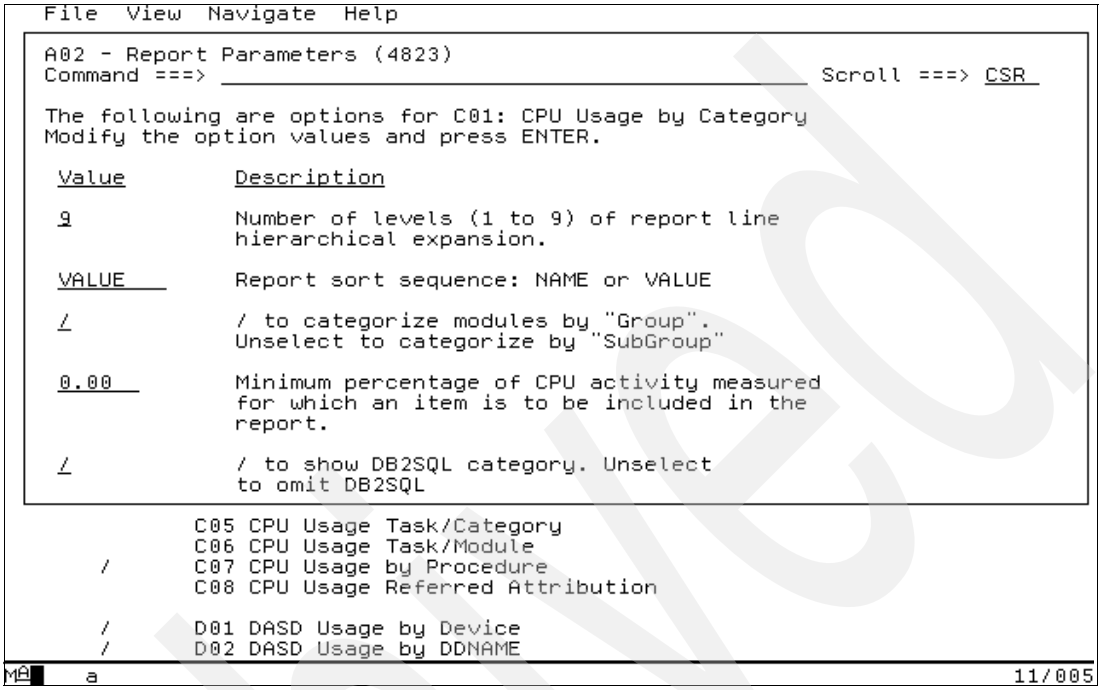


Figure 12-10 A02: Report Parameters

When all options have been specified for all desired sections, press Enter. A pop-up window appears with the JCL submission dialog, as shown in Figure 12-11.

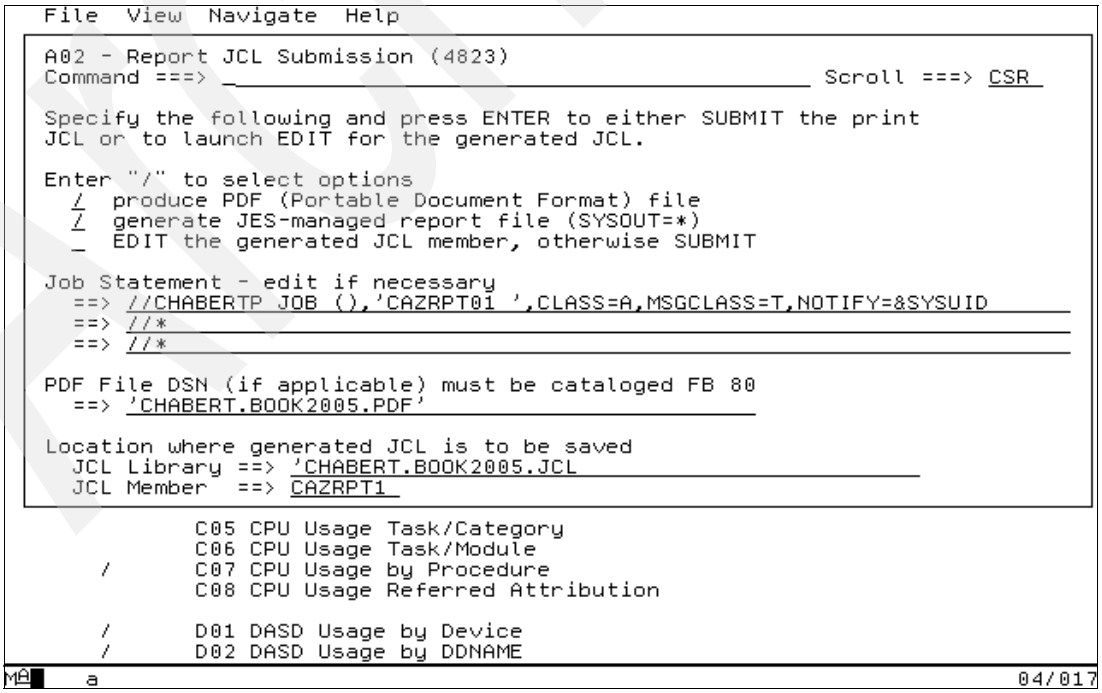


Figure 12-11 A02: Report JCL - Submission

The JCL library should exist.

The PDF file has to be pre-allocated, as shown in Figure 12-12.

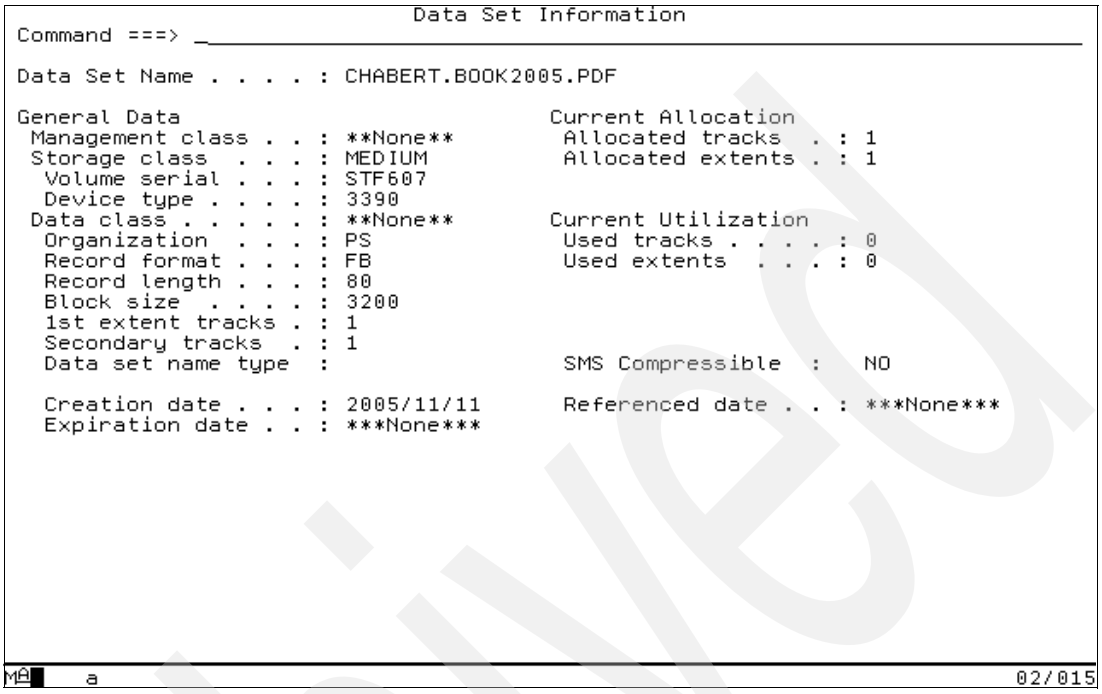


Figure 12-12 PDF file definition

Because the option, Edit the generated JCL, has been selected, pressing Enter gives us a pop-up window with the JCL generation, as shown in Example 12-1.

Example 12-1 Generated JCL to create PDF file

```
//CHABERTP JOB (),'CAZRPT01 ',CLASS=A,MSGCLASS=T,NOTIFY=&SYSUID
//*
//STEP1    EXEC PGM=CAZPRINT
//STEPLIB  DD  DISP=SHR,DSN=ADTOOLS.APA.V1R1M0.SCAZAUTH
//SYSABEND DD  SYSOUT=*
//CAZLOG   DD  SYSOUT=*
//SFILE01  DD  DISP=SHR,DSN=ADTOOLS.APA.CHABERT.R4807.CICSC23G.SF
//REPORT1  DD  SYSOUT=*
//PDF1     DD  DISP=SHR,DSN=CHABERT.BOOK2005.PDF
//CAZCTL   DD  *
 PROFILE 01 INPUT=SFILE01
 MAP TYPE=L FILE=CHABERT.BOOK2005.COBLIST MEMBER=MYTRADS
 MAP TYPE=L FILE=CHABERT.BOOK2005.COBLIST MEMBER=MYTRADMV
 SECTION S01
 SECTION S02 Sequence=NAME
             OmitESD=Y
             OmitNUC=N
             OmitPLPA=N
 SECTION S03 Sequence=NAME
 SECTION S04
 SECTION S05 Intervals=45
 SECTION S06 Intervals=45
 SECTION S07
 SECTION S08
 SECTION C01 Levels=9
```

```
            Sequence=VALUE
            DPAGroup=Y
            ShowDB2=Y
            Minimum=0.00
            MLD=Y
            Datamg=Y
            ShowIMS=Y
SECTION C02 Levels=9
            Sequence=VALUE
            Minimum=0.00
            MLD=Y
SECTION C03 Levels=9
            Sequence=VALUE
            Minimum=0.00
            SliceSize=64
SECTION C04 Intervals=45
SECTION C05 Levels=9
            Sequence=VALUE
            DPAGroup=Y
            ShowDB2=Y
            ShowInact=Y
            MLD=Y
            Datamg=Y
            ShowIMS=Y
SECTION C06 Levels=9
            MLD=Y
SECTION C07
SECTION C08 Levels=9
            MLD=Y
SECTION D01 Levels=9
            Minimum=0.00
SECTION D02 Levels=9
            Minimum=0.00
SECTION D03 Levels=9
            Minimum=0.00
SECTION D04
SECTION D05
SECTION D06
SECTION D07
SECTION D08 Levels=9
            MLD=Y
SECTION D09
SECTION W01 Levels=9
            Sequence=VALUE
            DPAGroup=Y
            ShowDB2=Y
            ShowInact=Y
            MLD=Y
            Datamg=Y
            ShowIMS=Y
SECTION W02 Levels=9
            MLD=Y
SECTION W03 Levels=9
            MLD=Y
SECTION E01
SECTION E03 Levels=9
            Minimum=0.00
            MLD=Y
SECTION E04 Levels=9
SECTION E05 Levels=9
```

```
 SECTION E06 Levels=9
 SECTION E07 Levels=9
 SECTION E08 Levels=9
 SECTION G01
 SECTION G02 Levels=9
 SECTION G03 Levels=9
 SECTION P01 Program=MYTRADS
             AdjLines=4
             AllSource=N
             AsmObj=Y
             Header=Y
 SECTION P01 Program=MYTRADMV
             AdjLines=4
             AllSource=N
             AsmObj=Y
             Header=Y
 PRINT DDNAME=REPORT1
 CONVERT DDNAME=PDF1 Format=PDF
/*
//
```

The last line of the JCL shows that the Source Program Mapping takes place for both MYTRADS and MYTRDMV.

The last step is to download the generated file onto our workstation in binary format, as shown in Figure 12-13.

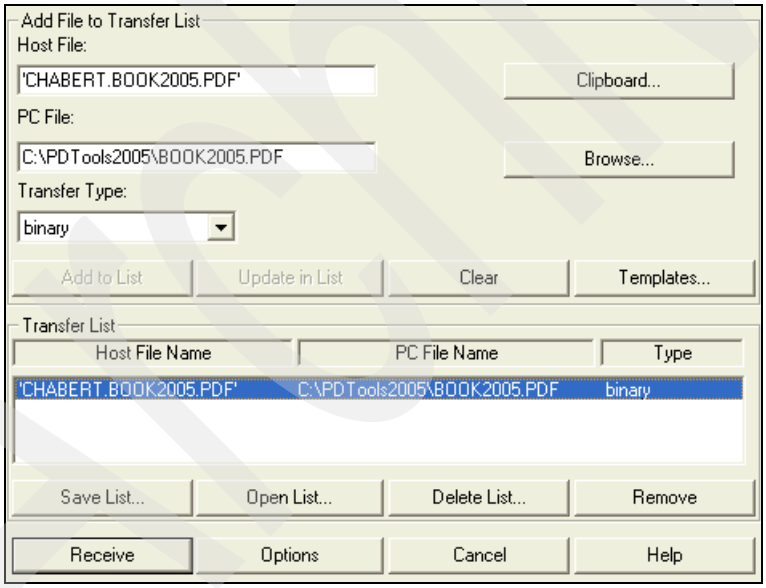


Figure 12-13 Download options

We now have to open the generated PDF file to read the Application Performance Analyzer's reports it contains, as shown in Figure 12-14.

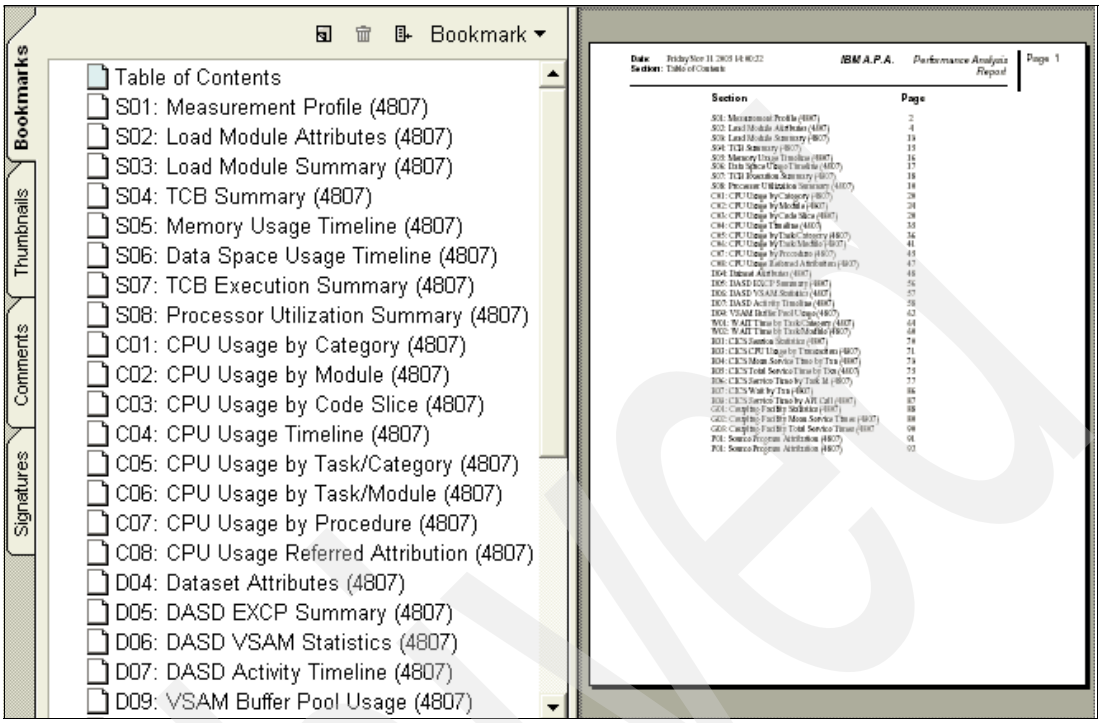


Figure 12-14 PDF file

12.3 How to share reports

If you do not want to print reports but to allow other sites, having Application Performance Analyzer properly installed and customized, you can use the Export and Import commands.

From the R02: Observation List, we enter the **Exp** line command for the report we want to export and press Enter, as shown in Figure 12-15.

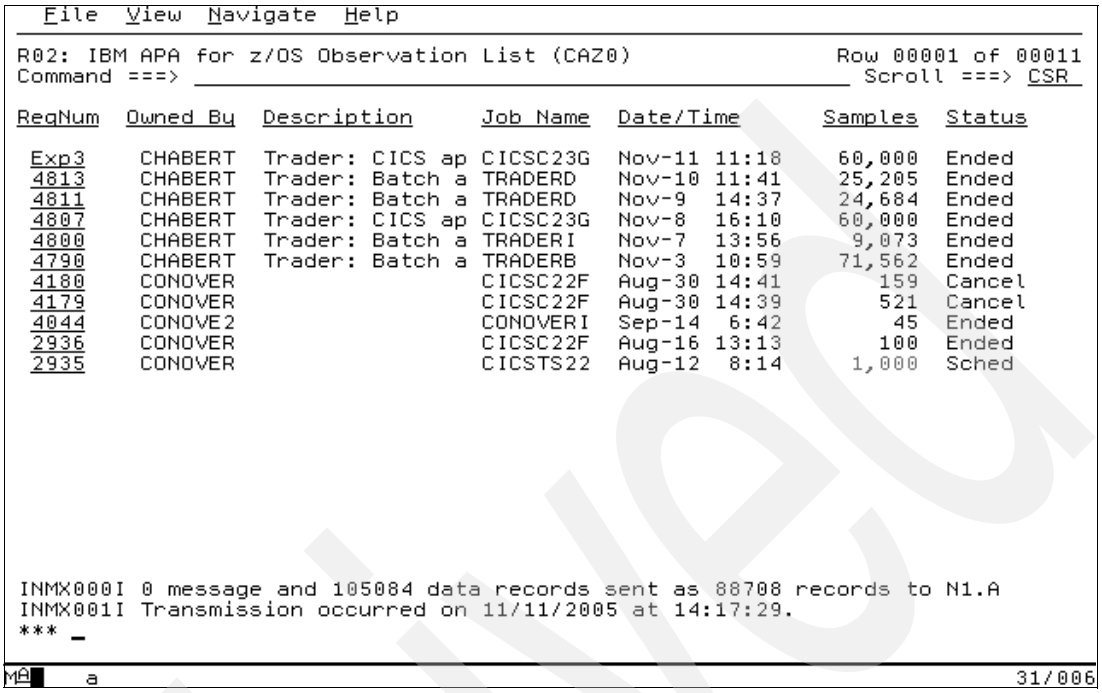


Figure 12-15 R02: Observation List EXPort line command

The first two messages show that 88,708 records have been sent. By pressing Enter, Application Performance Analyzer brings up a pop-up window with the name of the file that has been generated, as shown in Figure 12-16.

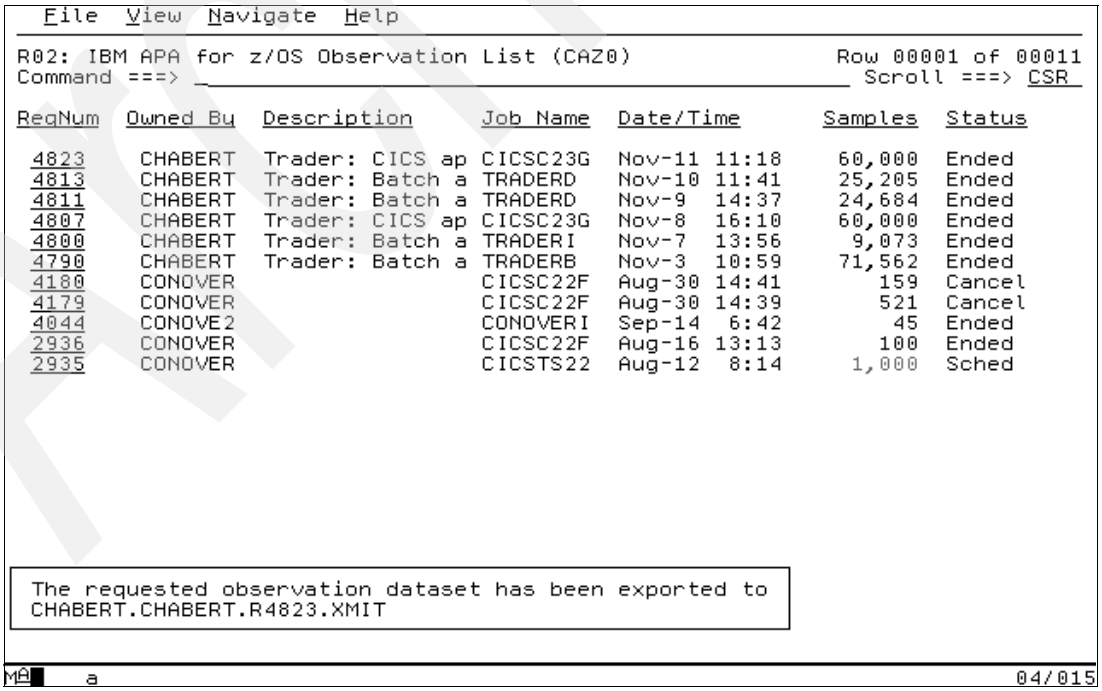


Figure 12-16 R02: EXPort message

We can now send the generated XMIT file to another location, from where it is possible to import it (using the IMPORT command line from report R02: Observation List, as shown as Figure 12-17) to perform an analysis.

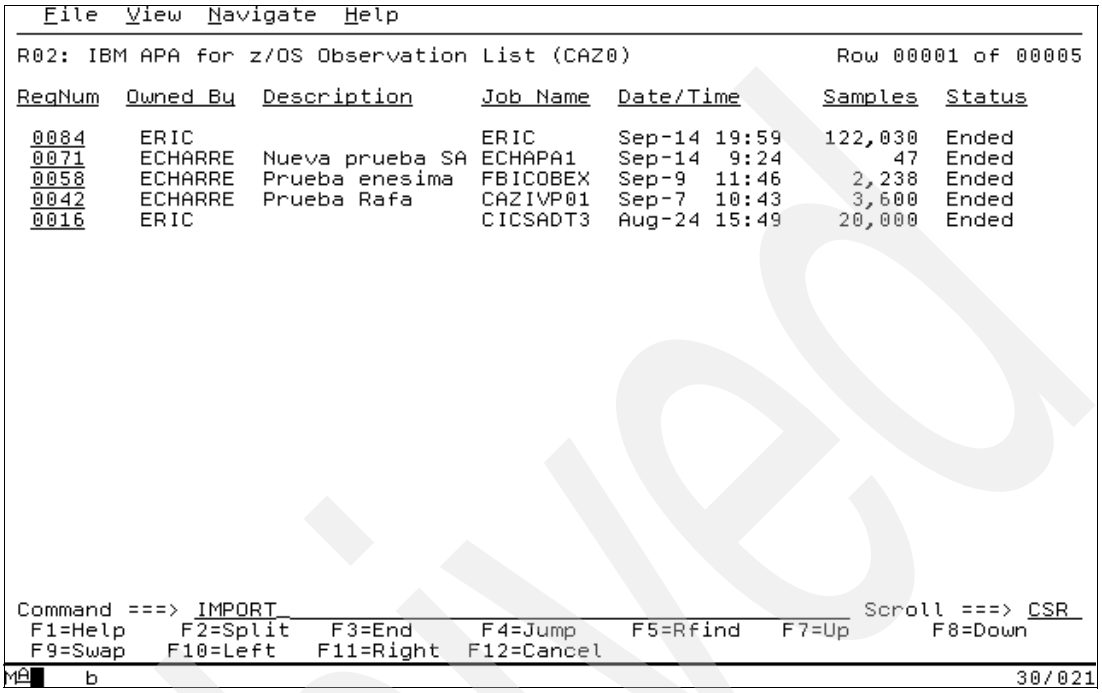


Figure 12-17 R02: Observation List IMPORT command line

By pressing Enter, a pop-up window appears, as shown in Figure 12-18.

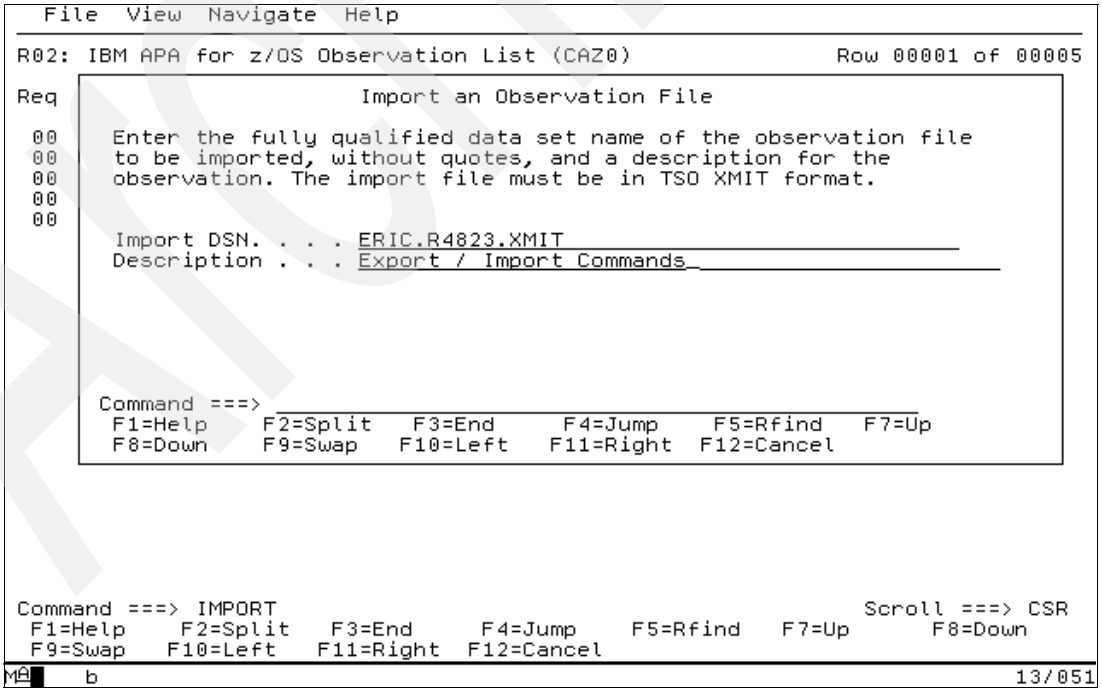


Figure 12-18 R02: Import an Observation File

When we have specified the Import DSN and the Description, by pressing Enter, Application Performance Analyzer imports the XMIT file and creates a new measurement file, as shown in Figure 12-19.

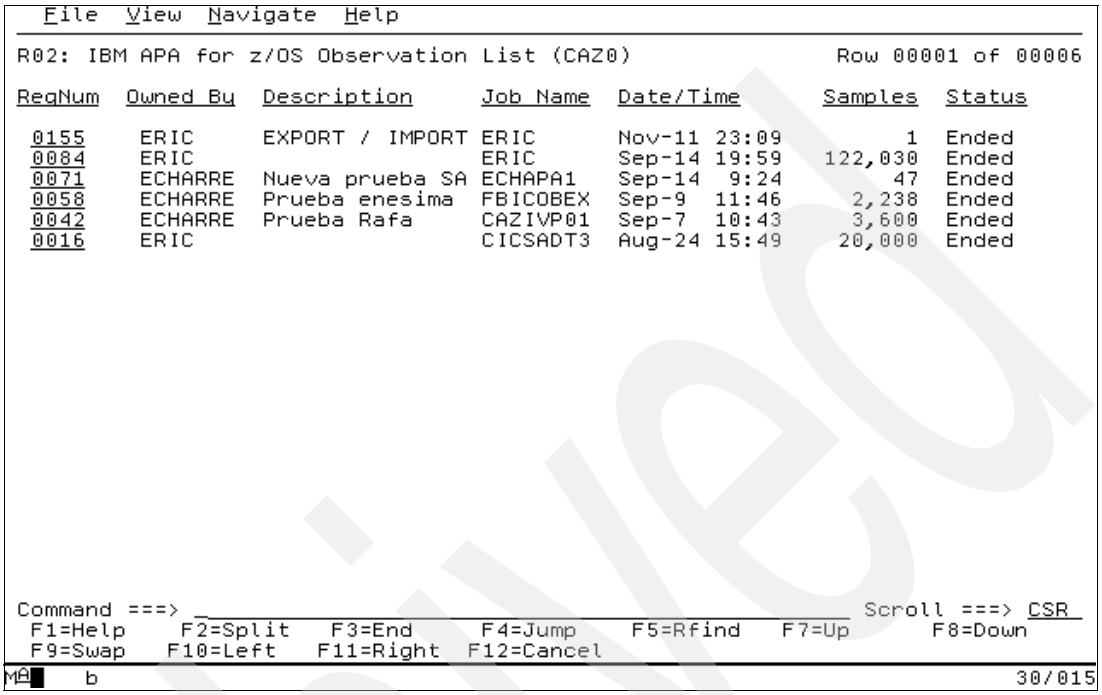


Figure 12-19 R02: Observation List after the IMPORT command line

Application Performance Analyzer automatically allocates a new request number.

Note: When we import a measurement file, its samples number is always 1.

We can now analyze, from this site, a report that has been created on another site. The XMIT files contain all the data required (except the Listing/Sidefile/Sysdebug/ADATA files required to perform the Source Program Mapping) to display any APA reports that were available on the original site.

12.4 How to start an observation session from Omegamon

It is possible to automatically start an observation session from any Omegamon feature, as shown in Figure 12-20.

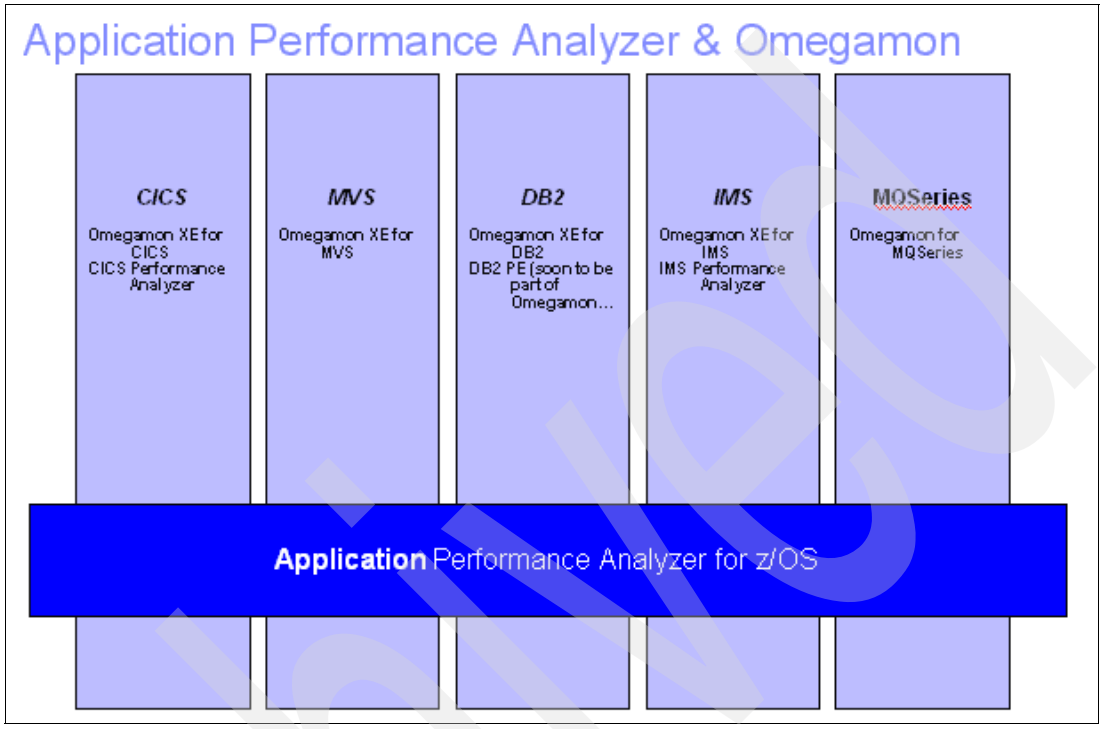


Figure 12-20 Application Performance Analyzer and Omegamon integration overview

To allow Omegamon to be coupled to Application Performance Analyzer, we have to:

- Define *Situation* in Omegamon DE or ITM where user thresholds (EXCP, duration, CPU usage, and so on) can be reached. It can be simple or complex situations to drive APA observation requests.
- Create a started task and program to create and submit a batch job to Application Performance Analyzer.

Figure 12-21 shows a basis for a sample situation from Omegamon XE on z/OS and its address space table.

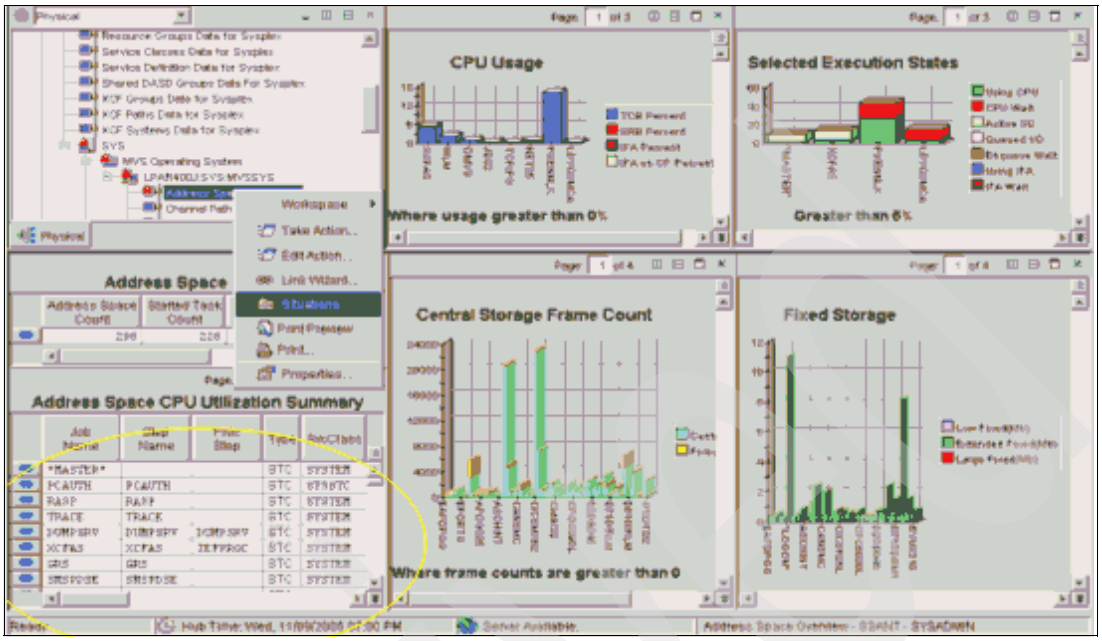


Figure 12-21 OMEGAMON® XE on z/OS address space table

We want to monitor an address space if its CPU usage is greater than 1.5% of the total CPU. We have to create a *situation* and to set a *condition* from the Address Space CPU Utilization Summary of Omegamon, as shown in Figure 12-22.

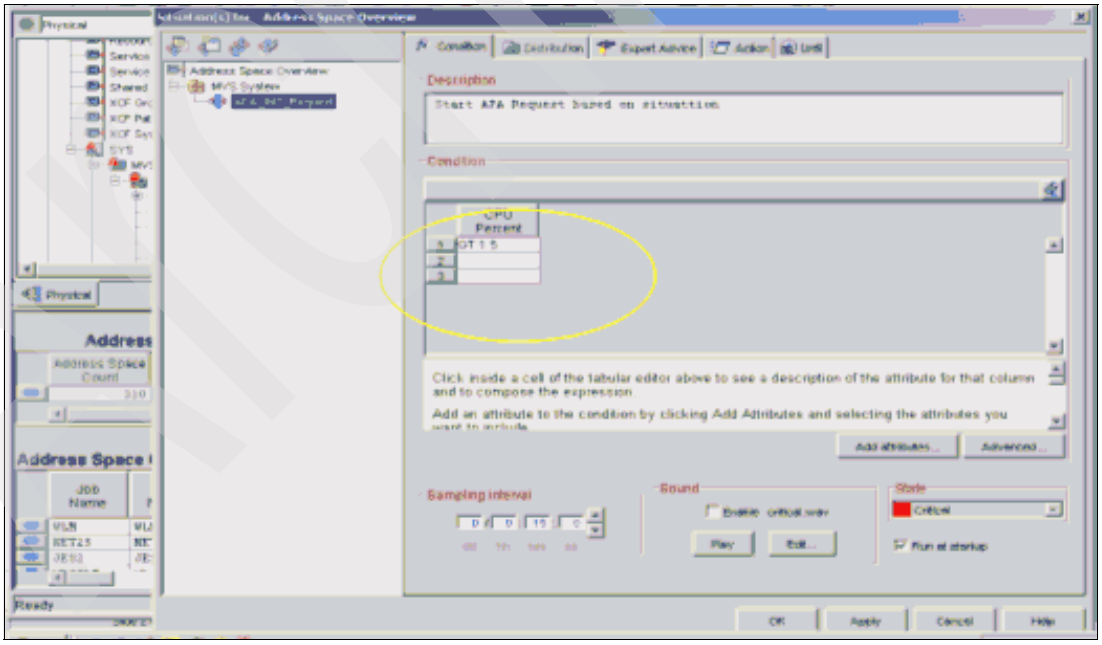


Figure 12-22 APA_INIT_Request for active job when CPU% > 1.5%

The *Situation Take Action* of Omegamon executes a z/OS start command to launch the APA request initiation. It is the HLPDPRC1 started task and starts with some parameters (Address Space Job_Name), as shown in Figure 12-23.

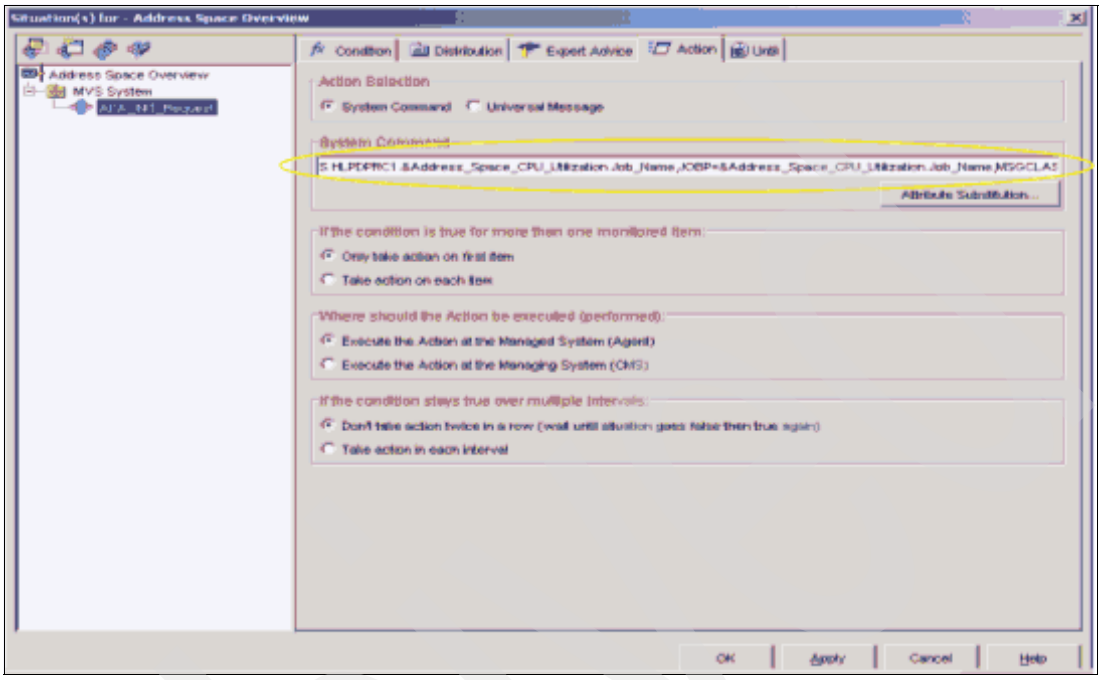


Figure 12-23 Situation: system command to be executed

Then the HLPDPRC1 started task runs a TSO batch with REXX exec to create the Application Performance Analyzer request, as shown in Example 12-2.

Example 12-2 HLPDPRC1

```
//*******************************************************************
//*
//* Purpose: Started task proc to initial a CAZBATCH run
//*
//*******************************************************************
//HLPDPRC1 PROC HILEV='TDZOST.HAB511O',
//              LOWLEV='JCL',
//              JOBP='',
//              MCLIST=CAZCC,
//              MSG=X
//*
//APAREQI EXEC PGM=IKJEFT01,REGION=4096K,
// PARM='%&MCLIST &JOBP'
//SYSPROC  DD DISP=SHR,
//            DSN=&HILEV..&LOWLEV
//SYSPRINT DD SYSOUT=&MSG
//SYSTSPRT DD SYSOUT=&MSG
//RDRFIL   DD SYSOUT=(&MSG,INTRDR)
//SYSTSIN  DD DUMMY
```

Figure 12-3 shows an extract of the CAZCC input file used.

Example 12-3 Extract of the CAZCC

```
/* REXX */
/*********************************************************************/
/** LICENSED MATERIALS - PROPERTY OF IBM                             */
/** (c) COPYRIGHT IBM CORP. 1992, 2005 ALL RIGHTS RESERVED           */
/**                                                                  */
/*                                                                   */
/*  PURPOSE: GENERATE AND SUBMIT A BATCH JOB TO queue a sample       */
/*           request to APA.                                         */
/*                                                                   */
/*  EXEC PARAMETERS:                                                 */
/*                                                                   */
/*    REQUIRED POSITIONAL PARAMETER(S):                              */
/*     JOBMASK  - JOB MASK FOR SAMPLING REQUEST                      */
/*                                                                   */
/*                                                                   */
/*********************************************************************
parse arg  jobmask junk
/*********************************************************************/
/* Create the job input                                              */
/*********************************************************************/
"ALLOC FI(CCJOBI) DSN('TDZOST.HAB5110.JCL(CAZSYSIN)') SHR"
queue "* This is a comment line  "
queue " // This is also a comment line  "
queue " NEW JOBNAME=("jobmask")       // specify job name  "
queue "     SYSTEMS=(SYSG)            // specify target system  "
queue "     ACTIVE=Y                  // job is active  "
queue "  ;                            // end of command  "
QUEUE " // END OF FILE  "
"EXECIO * DISKW CCJOBI (FINIS"
"FREE F(CCJOBI)"
/*********************************************************************/
/* Copy the job card to the JES internal reader.                     */
/*********************************************************************/
"ALLOC FI(CCJOBC) DSN('TDZOST.HAB5110.JCL(CAZJOBC)') SHR"
"EXECIO 1 DISKR CCJOBC"
RetCode = RC
............
............
............
End
“EXECIO 0 DISKR CCJOBC (FINIS”
“FREE F(CCJOBC)”
/*********************************************************************/
/* Submit job                                                        */
/*********************************************************************/
queue "//*************************************************************/"
queue "//* Licensed Materials - Property of IBM                     */"
queue "//*                                                           */"
queue "//* (c) Copyright IBM Corp. 2005. All rights reserved.       */"
queue "//*                                                           */"
queue "//* US Government Users Restricted Rights - Use,             */"
queue "//* duplication or disclosure restricted by GSA ADP          */"
queue "//* Schedule Contract with IBM Corp.                        */"
queue "//*************************************************************/"
queue "//*-------------------------------------------------------"
```

```
queue "//* Copyright (c) 2005 by BankNet Technologies Inc. "
queue "//* All Rights Reserved. No part of the contents of this   "
queue "//* source program may be reproduced or transmitted in any "
queue "//* form or by any means without the written permission of "
queue "//* the publisher. "
queue "//*-------------------------------------------------------"
queue "//*-------------------------------------------------------"
queue "//* "
queue "//* Function: JCL for submitting sampling requests via the "
queue "//*  IBM Application Performance Analyzer batch interface. "
queue "//* "
queue "//S1 EXEC PGM=CAZBATCH,REGION=50K,PARM='STCID=CAZ0' "
queue "//STEPLIB DD DISP=SHR,DSN=TDZOST.HAB5110.SCAZAUTH   "
queue "//SYSPRINT DD SYSOUT=*  "
queue "//SYSIN DD DISP=OLD,DSN=TDZOST.HAB5110.JCL(CAZSYSIN)  "
queue "/*EOF"
queue ""
"EXECIO * DISKW RDRFILE (FINIS"
"FREE F(RDRFILE)"
return 0
```

Part 4

Debug Tool, Debug Tool Utilities, and Advanced Functions

In this part of the book, we describe how to customize and use Debug Tool and Debug Tool Utilities and Advanced Functions. This part is divided into four chapters that deal with the following topics:

- Debugging programs using Debug Tool
- Using Debug Tool under UNIX System Services
- Debug Tool and subsystems
- Debug Tool Utilities and Advanced Functions

13

Debugging programs using Debug Tool

In this chapter we describe how to debug programs using Debug Tool. Sections on debugging programs written in many languages include:

- C/C++ programs
- PL/I programs
- COBOL programs
- Non-LE programs
- Assembler programs

13.1 Debugging C/C++ programs using Debug Tool

Before debugging a C/C++ program, the program must meet the following requirements:

- All the data sets required to debug your program must comply with the guidelines described in this book. The source is in a single file and not a concatenation of files.
- All the libraries that your program requires are available.
- Your program is compiled with the appropriate compiler options. When a program is under development, you can get the full capability of Debug Tool by compiling your program with the TEST(ALL) compiler option.

13.1.1 Preparing the program

Compile the IVP program from the SCCNSAM1 library using the instructions found in the Debug Tool User's Guide, V7.1. The JCL to compile the IVP program is shown in Example 13-1.

Example 13-1 JCL to compile the IVP program

```
//jobcard
//COMPILE EXEC EDCCB,
// INFILE='CBC.SCCNSAM(CCNYIV1)',
// OUTFILE='MLDUCKW.DEBUG.TESTLOAD(CIVP1),DISP=SHR',
// CPARM='OPTFILE(DD:OPTIONS)'
//* GPARM='OOOOOOO VERIFICATION SUCCESSFUL OOOOOOO'
//COMPILE.OPTIONS DD *
TEST
/*
```

13.1.2 Using full-screen 3270 debugger

Perform the following steps to use the full-screen 3270 debugger:

1. Invoke the program from batch using the appropriate test run-time option.

   In this example, we are debugging the C IVP that was already compiled, as shown in Example 13-2. The C IVP program can run with any number of command line parameters. For our initial run, we used no parameters.

   It might be to your advantage to allocate an INSPLOG data set. This allows the setup commands from this run to be captured, edited, and used as a preferences file for the next run of this program.

Example 13-2 Invoking DT for program CIVP1.

```
CALL 'MLDUCKW.DEBUG.TESTLOAD(CIVP1)' 'TEST(,,,*)/'
```

2. The 3270 full-screen mode looks as shown in Figure 13-1 on entry to a C debug session.

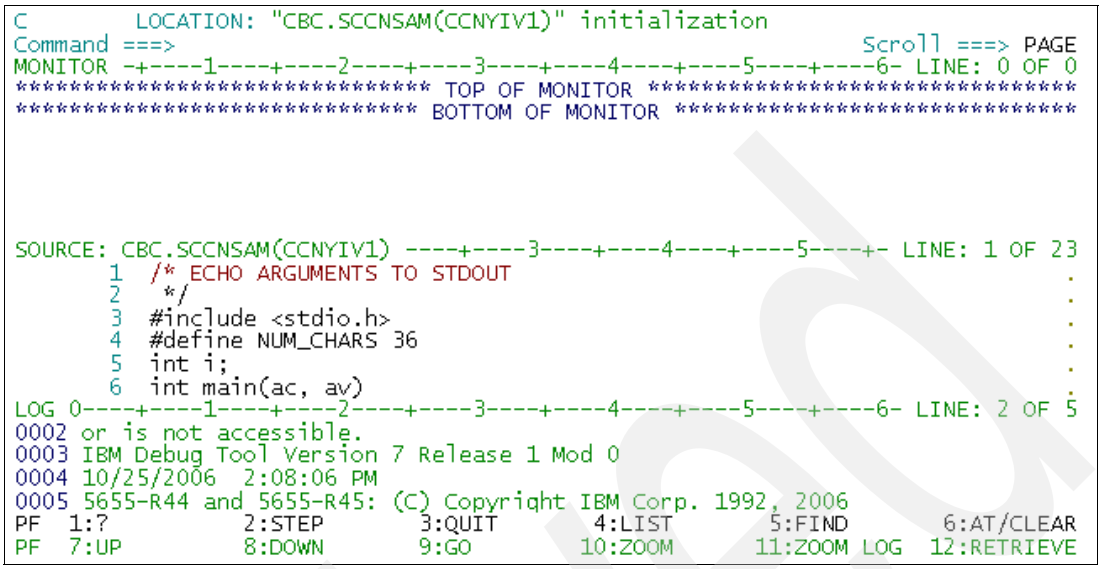


Figure 13-1 Full screen display on entry to a C debug session

3. To stop at the call to printf and at the exit of the program, issue the breakpoint commands:

```
AT CALL printf
AT EXIT main
```

To monitor the variable **i** and stop at the begging of the program (**main)**, enter the following commands:

```
MON LIST i
AT ENTRY main
```

4. Run the program by issuing the **GO** command. You should stop at the beginning of the program at the breakpoint set by AT ENTRY main as shown in Figure 13-2.

Note that the location where Debug Tool has stopped appears as the red line in the display.

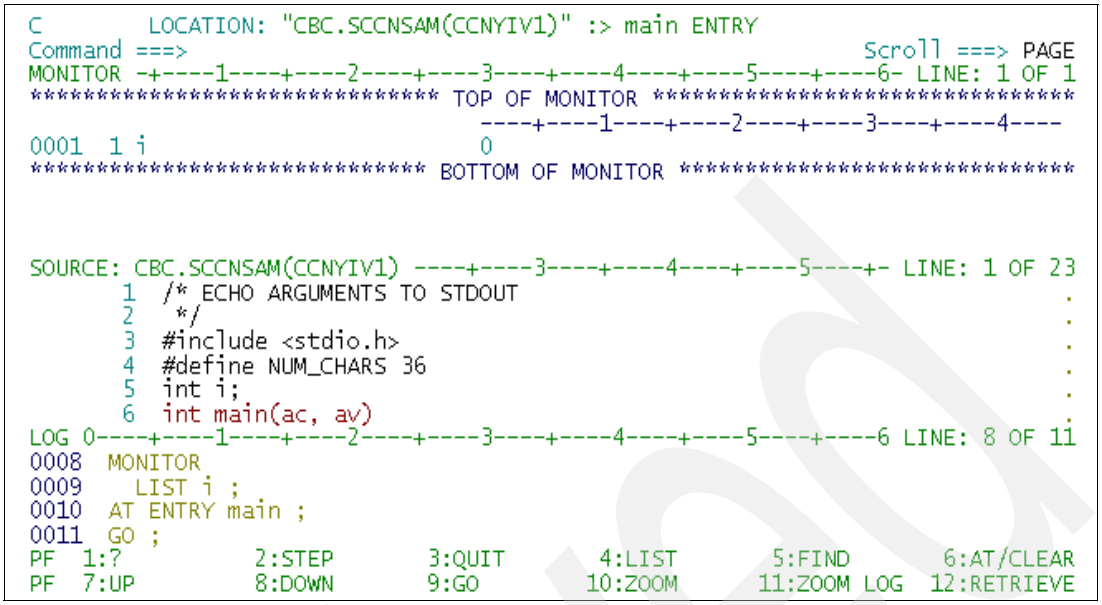


Figure 13-2 Display at the first breakpoint

5. At this point you want to add additional monitor statements for ac, av, and *av. Ideally, one should be able to put these monitor statements in one entry, but some experimentation has found that they are not always accepted when the variable is out of scope. For this reason, we recommend putting a breakpoint at the appropriate scope and turning on the monitoring at that point. The new monitor commands are:

   ```
   MON LIST ac;
   MON LIST av;
   MON LIST *av;
   ```

   The monitor section at the top of now has entries for i, ac, av, and *av.

   If you did not set up an INSPLOG, this would be a good time to cut the commands from the log file to create a preference file.

6. Issue **GO** again and the panel in Figure 13-3 is displayed.

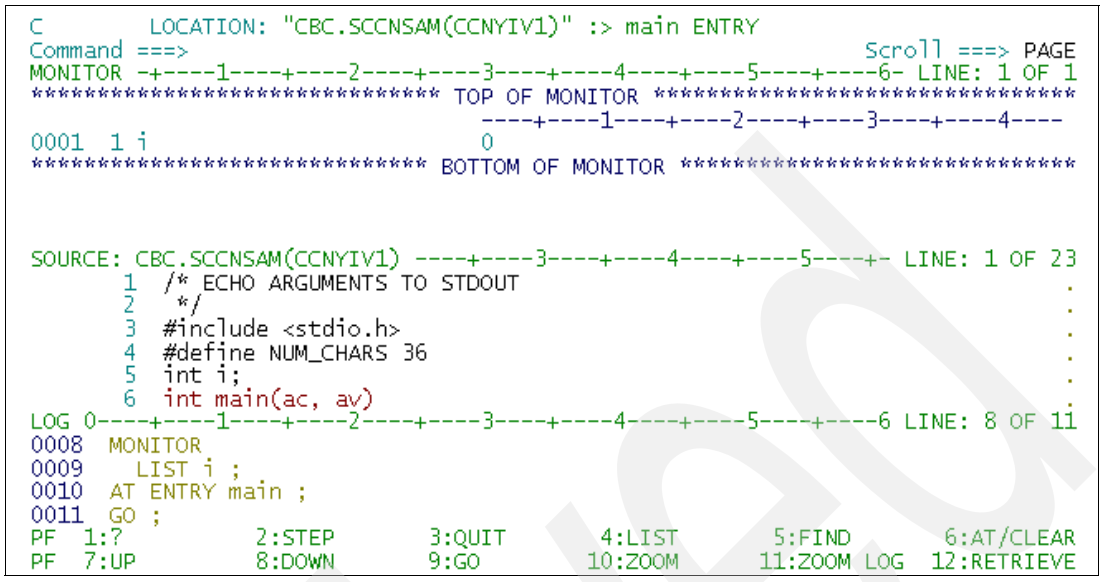


Figure 13-3 At the first breakpoint

Note that the printf breakpoint was never triggered because the printf was never executed. Type **qq** to quit this session.

7. If you rerun the program using the INSPREF file and with the PARMs shown in Example 13-3, it first stops at the call to printf for the first parameter.

Example 13-3 Command to rerun the CIVP1 program

```
CALL 'SE65273.DEBUG.TESTLOAD(CIVP1)' 'TEST/Testing one two'
```

13.2 Debugging PL/I programs using Debug Tool

This section describes how to:

- Prepare a PL/I application for debugging.
- Debug a PL/I application with the full-screen 3270 interface.
- Debug a PL/I application with the remote interface.

This section is based on an example given in the manual *OS PL/I Version 2 Programming: Using PLITEST*.

13.2.1 Preparing a PL/I program for debugging

For this section, we are using the sample PLI IVP program, IQIVPF, found in the DT sample library in member, EQAWIVPF. We are taking the defaults except for using the SEPARATE sub-option of the TEST compile option.

The TEST option is used by the PL/I compiler to enable debugging.

Note: Using the SEPARATE sub-option of the TEST compile option allows the debug information to be stored external to the executable rather than inside the load module itself.

13.2.2 Debugging a PL/I application using a full-screen interface

To begin debugging the IQIVPF program, perform the following steps:

1. Compile and link the sample program EQAWIVPF changing the TEST option as shown in Example 13-4.

Example 13-4 Modified TEST option to include the SEPARATE sub-option

```
TEST(ALL,SYM,SEP,NOHOOK)
```

2. Invoke the program from batch using the appropriate test runtime option.
3. It might be to your advantage to allocate an INSPLOG data set, since you might wish to review the debug session after it has completed.
4. Set breakpoints:

```
AT ENTRY IQIVPF
AT EXIT initialize_Rankings
```

5. Now run the program to that breakpoint by typing `GO` or pressing F9 and Figure 13-4 is displayed.

```
PL/I     LOCATION: DD:SYSIN :> IQIVPF EXIT
Command ===>                                                  Scroll ===> CSR
MONITOR -+----1----+----2----+----3----+----4----+----5----+----6- LINE: 0 OF 0
******************************** TOP OF MONITOR ********************************
******************************** BOTTOM OF MONITOR *****************************




SOURCE: DD:SYSIN -1----+----2----+----3----+----4----+----5----+-- LINE: 5 OF 9
       5     Do I = 1 to 2;                                                    .
       6       Put Skip List ('IQIVPF   incrementing I');                      .
       7     End;                                                              .
       8     Put Skip List ('IQIVPF   terminating');                           .
       9   END;                                                                .
******************************** BOTTOM OF SOURCE ******************************
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 12 OF 15
0012  AT EXIT IQIVPF ;
0013  GO ;
0014  GO ;
0015  SET DEFAULT SCROLL CSR ;
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 13-4 Display at the end of execution.

13.2.3 Debugging a PL/I application with the remote interface

To show off the remote debugger function, we describe how to finish debugging this problem with the IBM WDDz Distributed Debugger.

1. We start WDDz on our workstation as the distributed debugger. It is listening on port 8001.

   The results are shown in Figure 13-5.

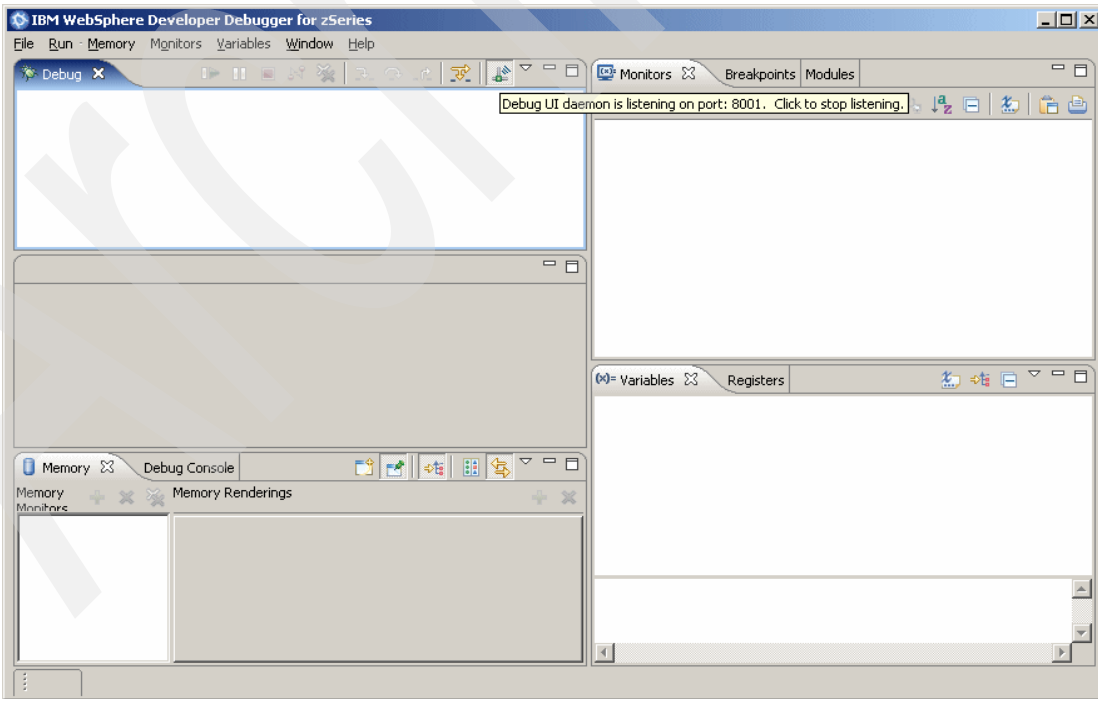


Figure 13-5 WDDz remote debugger initialization window

Using the **Ipconfig** command from a Windows Command prompt window, we display the IP address of our workstation, as shown in Figure 13-6.

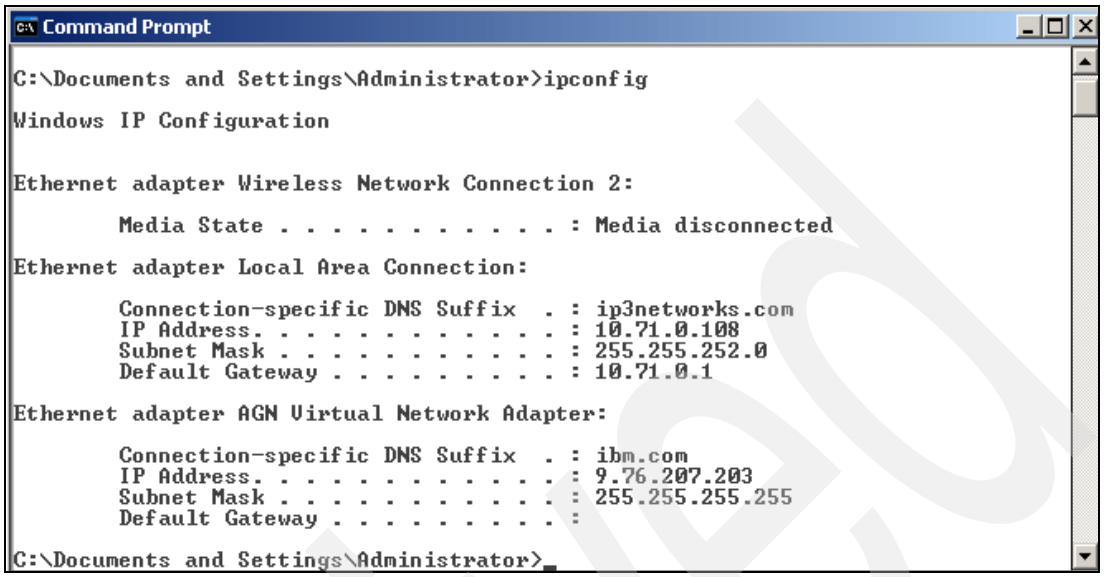


Figure 13-6 Identifying the IP address for this workstation

Because this is workstation is working outside the local area connection, we use the AGN Virtual Network's IP address (9.76.207.203).

2. We have recompiled the program after making the required changes to the TEST compile option. Invoke the remote debugger to debug it using the job shown in Figure 13-7.

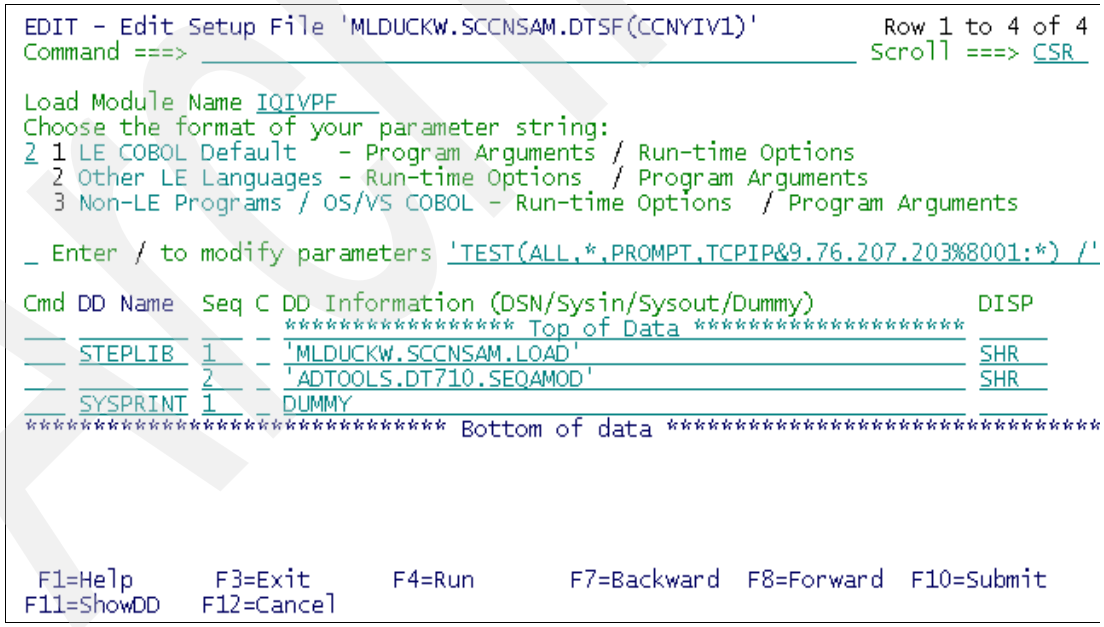


Figure 13-7 JCL to initialize remote debug session

Notice the IP address in the TEST compile option above. Pressing F4 now starts the program execution for debugging.

At this point the remote debugger is displayed, as shown in Figure 13-8.

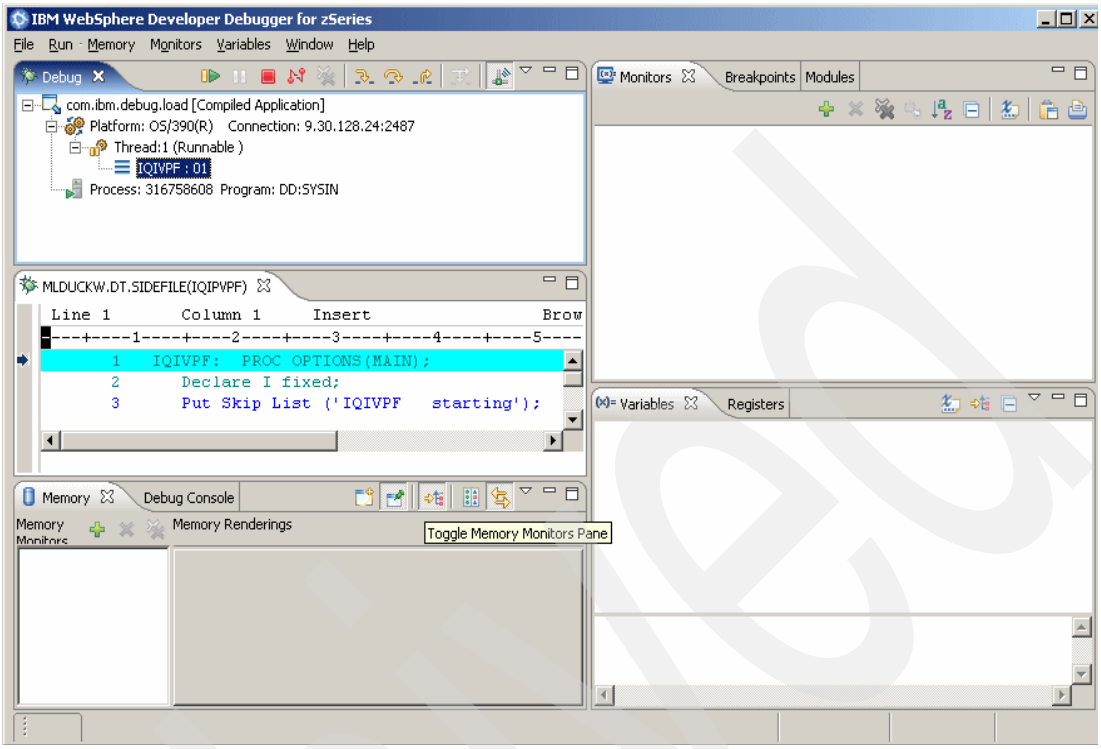


Figure 13-8 Initial look of the Remote Debugger for the PL/I program, IQIVPF

Place breakpoints in the program by either double-clicking in the area to the left of the statement or by right-clicking in the same area and selecting **Add Breakpoint**, as shown in Figure 13-9.

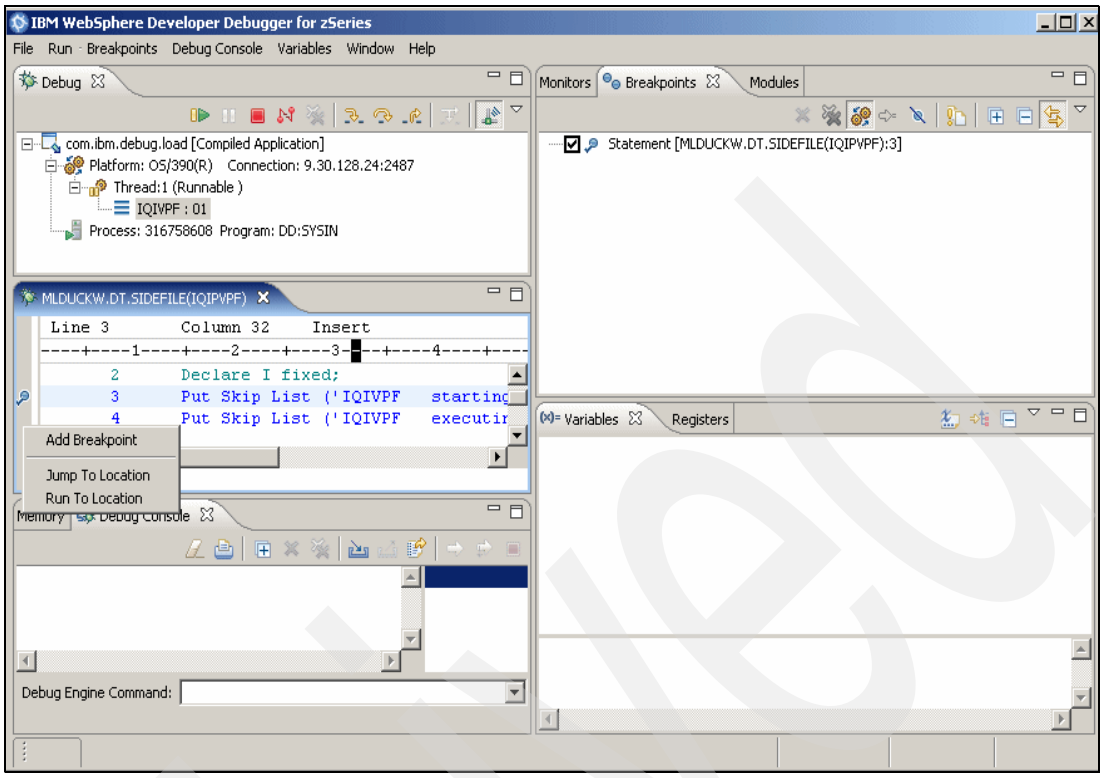


Figure 13-9 Debug Tool - PLI - Remote debugger - after setting breakpoint

3. Click the **Step Over** button twice — located in the top left window below. The program runs to the breakpoint that you set previously, as shown in Figure 13-10.

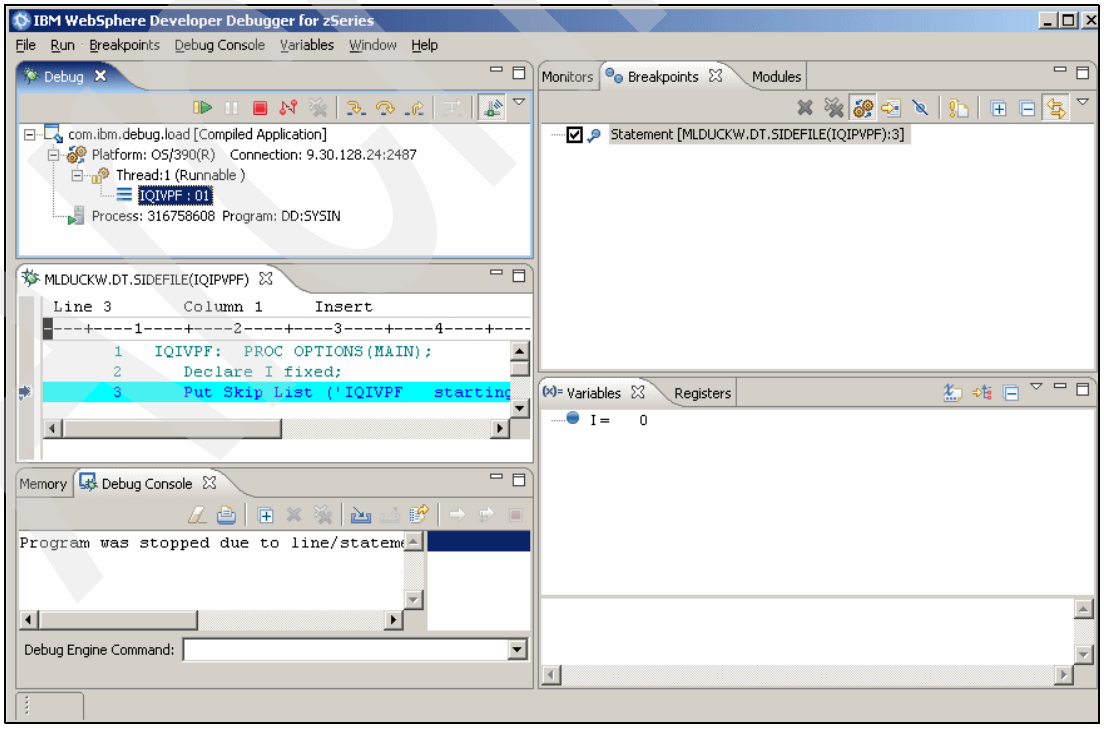


Figure 13-10 Remote debugger display at the breakpoint

4. In the top left window are controls that allow you to manage the execution of your procedure, step-by-step if you want. Then the screen is divided into four parts:
   - The top left part is shown below. This area displays high level information about what is being debugged. A sample of this is shown in Figure 13-11.
   -

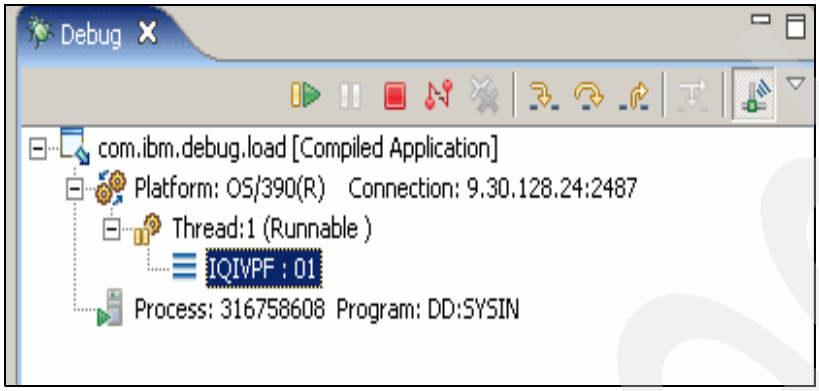


Figure 13-11 Remote Debugger top left portion of the screen

   - In the source code part of the main window, an arrow shows the current statement being executed, as shown in Figure 13-12.

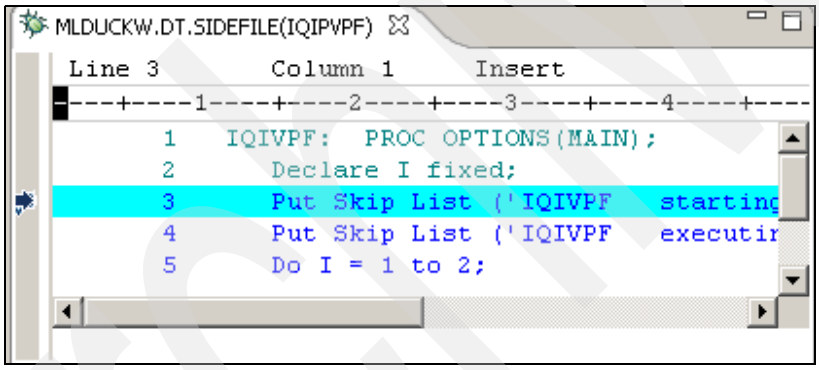


Figure 13-12 Source window

     You can only set breakpoints in lines that actually execute some code, so you are not able to set breakpoints in lines with comments or in storage definitions.
   - In the monitor part of the main window, you can monitor and change values of variables and parameters. To start monitoring the values of a variable, right-click in the Monitor window and enter the expression for evaluation in the pop-up window that appears.

– Figure 13-13 shows a Monitor window watching the integer, I.

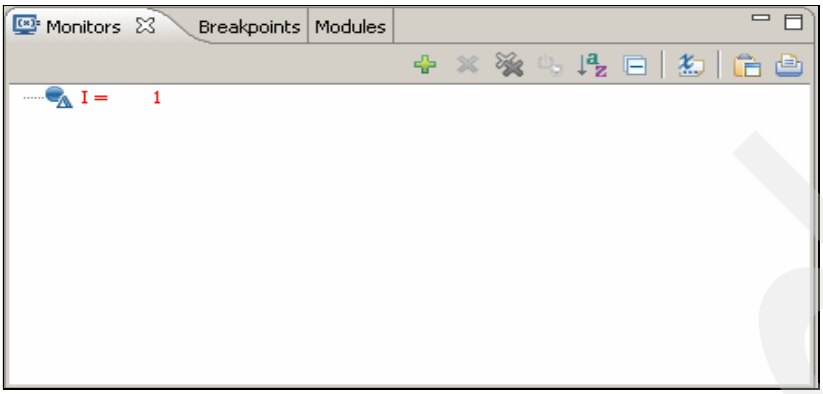


Figure 13-13 Monitor window

– The last part is the Debug Console log window, as shown in Figure 13-14.

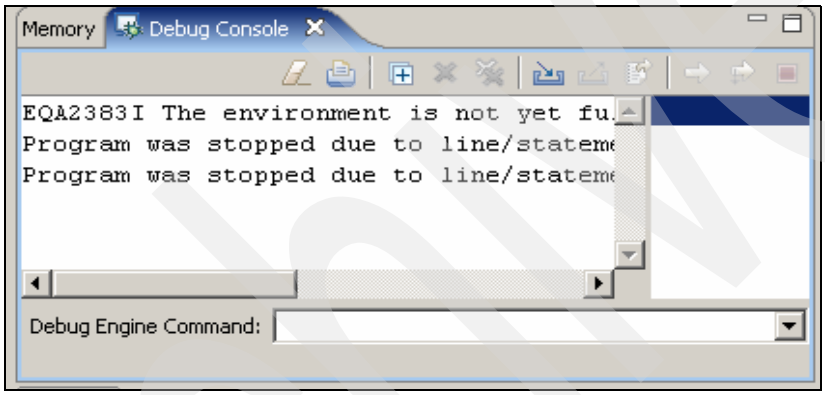


Figure 13-14 Debug console log window

13.3 Debugging COBOL programs using Debug Tool

This section describes how to:

- Prepare an OS/VS COBOL application for debugging using JCL.
- Prepare a COBOL for z/OS and OS/390 application for debugging using Debug Tool Utilities and Advanced Functions (DTU&AF).
- Debug a COBOL for z/OS and OS/390 application using Debug Tool Set Utility (DTSU) to start the full-screen 3270 interface.
- Debug a COBOL for z/OS and OS/390 application using Debug Tool Set Utility (DTSU) to start the remote debugger interface.

13.3.1 Preparing an OS/VS COBOL application for debugging using JCL

When you debug an OS/VS program, you can use most of the Debug Tool commands. Debugging an OS/VS COBOL program is much like debugging an assembler program. However, there are differences between debugging an OS/VS COBOL or assembler program and debugging programs written in other programming languages supported by Debug Tool.

The main considerations and differences are as follows:

- After you compile your program, you must create a debug information file, also called the EQALANGX file. Debug Tool uses this file to obtain information about your OS/VS COBOL program.
- You must inform Debug Tool that a compile unit is an OS/VS COBOL compile unit and instruct Debug Tool to load the OS/VS COBOL compile unit's debug information. Do this by entering the LOADDEBUGDATA (or LDD) command.
- OS/VS COBOL expressions must be enclosed in single quotation marks. In addition, only a subset of the OS/VS COBOL operators and figurative constants are supported by Debug Tool.

Note: OS/VS COBOL is part of Debug Tool Utilities and Advanced Functions.

Program preparation

The following compiler options are required:

- SOURCE
- DMAP
- PMAP
- VERB
- XREF

You cannot have LSTONLY or LSTCOMP.

EQALANGX generation

The process of generating the EQALANGX file for OS/VS COBOL is the same as generating the EQALANGX for assembler with two exceptions:

1. The PARMs are different. The required PARMs for OS/VS COBOL are:

   ```
   PARM='(COBOL LOUD ERROR 64K CREF'
   ```

2. Instead of using the SYSADATA output of the assembler, the input to EQALANGX is the OS/VS COBOL listing.

Example 13-5 is a sample JCL that can be used to generate an EQALANGX file for OS/VS COBOL.

Example 13-5 EQALANGX generation

```
//IDILANGX EXEC PGM=EQALANGX,REGION=32M,
// PARM='(COBOL LOUD ERROR 64K CREF'
//STEPLIB  DD  DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL71.SEQAMOD
//LISTING  DD  DISP=SHR,DSN=ADTOOLS.COBOSVS.LISTING(&MEM.)
//IDILANGX DD  DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL71.OSVS.EQALANX
```

The variables used in this example have the following meanings:

COBOL	Indicates that an OS/VS COBOL module is being processed.
LOUD	This parameter is required.
ERROR	This parameter is optional. If you specify it, additional information is displayed when an error is detected.
64K	This parameter is required.
CREF	This parameter is required.

hlq.EQALANGX The name of the data set where the EQALANGX debug file is to be placed. This data set must have variable block record format (RECFM=VB) and a logical record length of 1562 (LRECL=1562).

Debug Tool searches for the EQALANGX debug file in a partitioned data set with the name hlq.EQALANGX and a member name that matches the name of the OS/VS COBOL program. If you want the member name of the EQALANGX debug file to match the name of the OS/VS COBOL program, you do not have to specify a member name on the DD statement.

EQALANGX usage

Before you can debug an OS/VS COBOL program, the compilation unit (CU) must meet the following requirements:

- If the CU is known to Debug Tool, it must be a disassembly CU.
- If the CU is not known to Debug Tool, it must appear later as a disassembly CU.

If the CU is part of a load module that has not yet been loaded when you issue the LDD command, Debug Tool issues a message indicating that the CU is not found and that the LDD has been deferred. If the CU later appears as a disassembly CU, the LDD is done at that time.

In either case, you must define the CU as an OS/VS COBOL CU and load the debug data that is associated with that program. To accomplish these objectives, use the LOADDEBUGDATA command (abbreviated as LDD) as follows:

- If your debug data is in a partitioned data set where the high-level qualifier is the current user ID, the low-level qualifier is EQALANGX, and the member name is the same as the name of the CU that you want to debug, enter the following command:

  ```
  LDD membername
  ```

- If your debug data is in a different partitioned data set than userid.EQALANGX but the member name is the same as the name of the CU that you want to debug, enter the following command before or after you enter LDD membername:

  ```
  SET DEFAULT LISTINGS
  ```

- If your debug data is in a sequential data set or is a member of a partitioned data set but the member name is different from the CU name, enter the following command before or after you enter LDD membername:

  ```
  SET SOURCE
  ```

Link-edit

Example 13-6 is sample JCL that link-edits an OS/VS COBOL program with the OS/VS COBOL run-time.

Example 13-6 JCL to link-edit an OS/VS COBOL program with OS/VS COBOL run-time

```
//OSCOBLNK JOB (USER01),'USER01',MSGCLASS=H,MSGLEVEL=(1,1),
//            NOTIFY=&SYSUID,REGION=32M,CLASS=A,TIME=1
//*
//* JCL to link edit Cobol testcase COBO2O. COBO2O was compiled with
//* the OS/VS Cobol compiler.
//*
//* Link edit library   : OS/VS Cobol
//* Input  object       : ADTOOLS.DEBUG.TOOL71.OSVS.OBJ
//* Output RUNLIB module: ADTOOLS.DEBUG.TOOL71.OSVS.LOAD
//*
//LKED   EXEC PGM=IEWL,PARM=(LET,MAP,LIST)
```

```
//SYSLIB   DD DISP=SHR,DSN=ESFLINT.OSVSCOB.VSCLLIB
//SYSPRINT DD SYSOUT=*
//SYSLMOD  DD DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL71.OSVS.LOAD
//SYSUT1   DD UNIT=SYSDA,SPACE=(TRK,(10,10))
//OBJECT   DD DISP=OLD,DSN=ADTOOLS.DEBUG.TOOL71.OSVS.OBJ
//SYSLIN   DD *
 INCLUDE OBJECT(COBO2O)
 NAME COBO2O(R)
/*
//
```

Example 13-7 is sample JCL that link-edits an OS/VS COBOL program with the LE run-time.

Example 13-7 JCL to link-edit an OS/VS COBOL program with the LE runtime

```
//OSCOBLNK JOB (USER01),'USER01',MSGCLASS=H,MSGLEVEL=(1,1),
//             NOTIFY=&SYSUID,REGION=32M,CLASS=A,TIME=1
//*
//* JCL to link edit Cobol testcase COBO2O. COBO2O was compiled with
//* the OS/VS Cobol compiler.
//*
//* Link edit library   : OS/VS Cobol
//* Input  object       : ADTOOLS.DEBUG.TOOL71.OSVS.OBJ
//* Output RUNLIB module: ADTOOLS.DEBUG.TOOL71.OSVS.LOAD
//*
//LKED   EXEC PGM=IEWL,PARM=(LET,MAP,LIST)
//SYSLIB   DD DISP=SHR,DSN=CEE.SCEELKED
//SYSPRINT DD SYSOUT=*
//SYSLMOD  DD DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL71.OSVS.LOAD
//SYSUT1   DD UNIT=SYSDA,SPACE=(TRK,(10,10))
//OBJECT   DD DISP=OLD,DSN=ADTOOLS.DEBUG.TOOL71.OSVS.OBJ
//SYSLIN   DD *
 INCLUDE OBJECT(COBO2O)
 NAME COBO2OL(R)
/*
//
```

Execution

These are some considerations for the execution:

- If you link with the OS/VS COBOL run-time, your execution JCL should STEPLIB to the OS/VS COBOL runtime library.
- If you link with the LE run-time, your execution JCL should STEPLIB to LE's SCEERUN library

Syntax for OS/VS COBOL expressions

The syntax used for debugging OS/VS COBOL programs is a subset of OS/VS COBOL syntax and a subset of the syntax used for debugging assembler programs.

In general, whenever you enter an OS/VS COBOL expression (such as the operand of LIST expression, assignment, IF, and so forth), you must enclose the OS/VS COBOL expression in single quotation marks. For example:

```
LIST 'A-B IN C';
'A' = 'B';
IF 'A = 22' THEN …
```

There are some Debug Tool commands that can be used for debugging OS/VS COBOL programs that use the assembler syntax for OS/VS COBOL programs. For example, while debugging an OS/VS COBOL program you might use the following command:

```
STORAGE(X"1B4C0",3) = X"0102FC";
```

Restrictions on OS/VS COBOL expressions

In addition to the requirement that OS/VS COBOL expressions be enclosed in single quotations, the following restrictions apply to OS/VS COBOL expressions:

- The following operators are supported by Debug Tool in OS/VS COBOL expressions:
  - IN or OF
  - Subscript/index

    In a subscript or index list, the subscript or index expressions must be separated by a comma. A space is not sufficient for separating subscript or index expressions.
  - LENGTH OF

    Returns the length of a symbol
  - +, -, *, /

    Addition, subtraction or prefix minus, multiplication, and division
  - //

    Remainder
  - ||

    Concatenation
  - (...)

    Parenthesis to control the order of operation, specify the subscript of an array, or select a substring:
    - symbol(subscript,subscript,…)

      Parenthesis to specify a subscript or index for an array
    - symbol(substrstart:substrend)

      Parenthesis to select a substring of the bytes from substrstart to substrend from a character variable
    - symbol(substrstart::substrlen)

      Parenthesis to select a substring of substrlen bytes
  - &

    Logical *and* operation
  - |

    Logical *or* operation
  - =, <, >, <=, >=
- Lower-case letters are accepted in contexts other that non-numeric literals as a substitute for (and equivalent to) upper-case letters.
- The use of COBOL special registers such as DAY, DATE, TIME, and so forth is not supported in Debug Tool OS/VS COBOL expressions.
- All non-numeric literals must be enclosed in double quotation marks. Single quotation marks cannot be used.

- Only the following subset of figurative constants is supported in Debug Tool OS/VS COBOL expressions:
  - HIGH-VALUE, HIGH-VALUES
  - LOW-VALUE, LOW-VALUES
  - QUOTE, QUOTES
  - SPACE, SPACES
  - ZERO, ZEROES, ZEROS

Common symbol

Debug Tool implicitly defines the following symbols in all OS/VS COBOL compilation units:

- _STORAGE

  This symbol is implicitly defined as a symbol representing all main memory.

- %symbol

  A valid Debug Tool variable or built-in function, for example, %ADDRESS or %HEX(expression).

Restrictions

The restrictions are:

- Limited Path Table, %PATHCODE, or AT PATH support (Entry and Label path points only)
- No AT CALL support
- No AT EXIT support
- No STEP RETURN support

Assignment command

The Assignment command assigns the value of an expression to a specified variable. It is the equivalent of the OS/VS COBOL COMPUTE statement. A sample is shown in Example 13-8.

Example 13-8 Assignment

```
>>-- 'receiver' -- = -- 'sourceexpr'--; ><
```

receiver A valid Debug Tool OS/VS COBOL reference enclosed in single quotation marks

sourceexpr A valid Debug Tool OS/VS COBOL expression enclosed in single quotation marks

Usage:

- When the assignment receiver is an arithmetic variable, the source can be a hexadecimal string of the same length as the receiver.
- When the assignment receiver is a non-numeric string, the source can be a hexadecimal string of any length.
- When the assignment receiver is a COBOL INDEX variable, the source is assumed to be a subscript value and is converted to the proper offset before the value is stored into the receiver.

Note: The Assignment command cannot be used while you replay recorded statements by using the PLAYBACK commands.

13.3.2 Preparing a COBOL for z/OS and OS/390 application using DTU

Debug Tool Utilities is only available if you have installed Debug Tool Utilities and Advanced Functions. It allows you to set up programs to run with Debut Tool and save these setup characteristics for future use.

Perform the following steps to compile and link with program preparation:

1. From the main DTU panel, select **1 - Program Preparation** as shown in Figure 13-15.

```
---------------------------- Debug Tool Utilities ----------------------------
Option ===> ____________________________________________________________________

                                                                        More:     +
0  Manage Job Card
   For Program Preparation and Setup File Management

1  Program Preparation
   Compile old or new COBOL programs with newer compilers, convert old COBOL
   source into new COBOL source, use other compilers, and link edit.

2  Manage and Use Debug Tool Setup Files
   You can manage setup files and use them to run your program interactively
   with Debug Tool in TSO Foreground or submit your program to run in
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-15 Main Debug Tool Utilities and Advanced Functions panel

2. Select option **1 - COBOL Compile** from the Debug Tool Program Preparation panel as shown in Figure 13-16.

```
---------------------- Debug Tool Program Preparation ----------------------
Option ===> ____________________________________________________________________

                                                                        More:     +
1  COBOL Compile
   Using 5655-G53 IBM Enterprise COBOL for z/OS

2  COBOL Convert and Compile
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid
   and   5655-G53 IBM Enterprise COBOL for z/OS

3  PL/I  Compile
   Using 5655-H31  IBM(R) Enterprise PL/I for z/OS

4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-16 Program Preparation panel

This compile utility lets you select either for a Foreground (**F**) or Background (**B**) compile as shown in Figure 13-17. In the foreground compile, the processing is run in TSO, and the return codes are displayed on a panel. In the background compile, a job is created and submitted in batch. You must specify (using FP8) the SYSLIB data set used for COPY statement processing to search for COPY members if required. You can see how to do this later. For the time being, we omit this step, which generates a compile error.

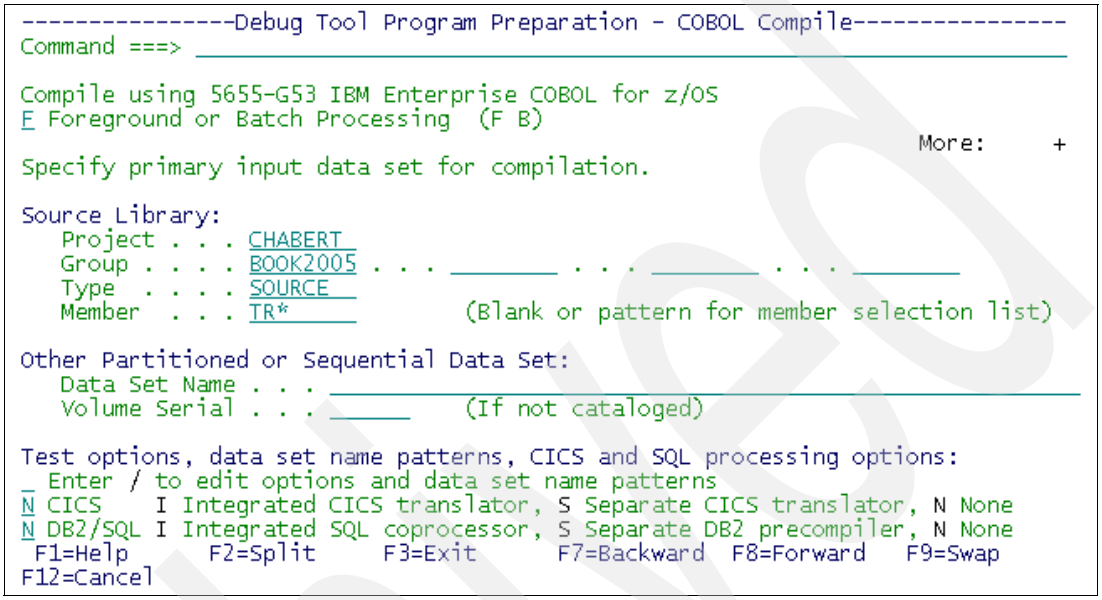


Figure 13-17 COBOL Compile selection panel

3. Specify your source library. If you do not specify the source member or if it is a generic value, a member selection list is presented as shown in Figure 13-18.

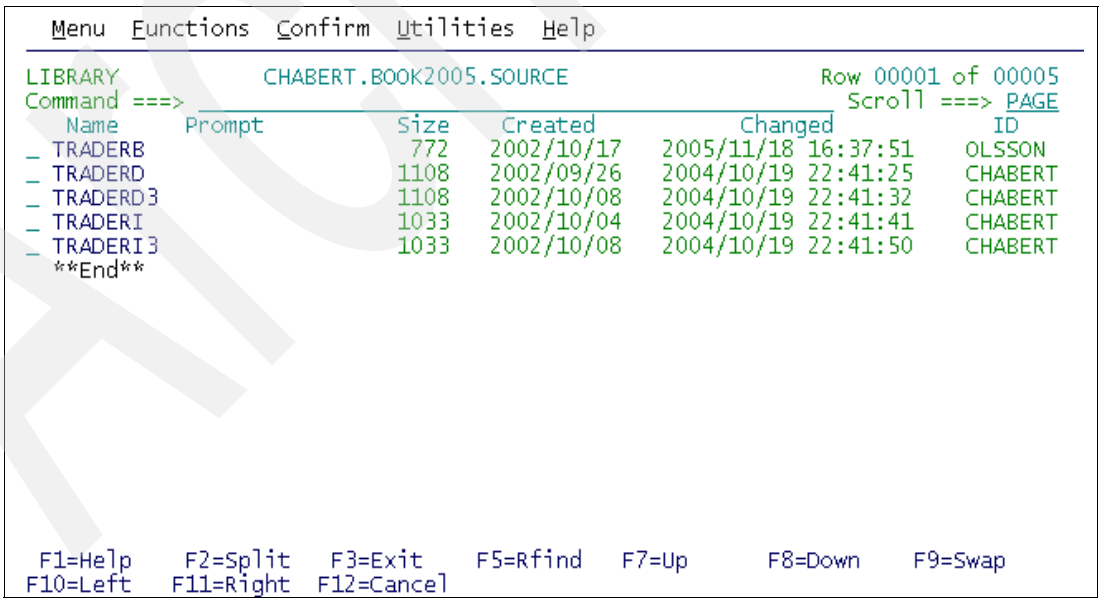


Figure 13-18 COBOL Compile member selection panel

4. Select the member **TRADERB** and press Enter. Figure 13-19 is displayed listing all the input data sets (your source), final options, data set names for the listing, object, sidefile, DB2 compiler, and CICS translator options and data sets. You can overwrite any or all of these fields.

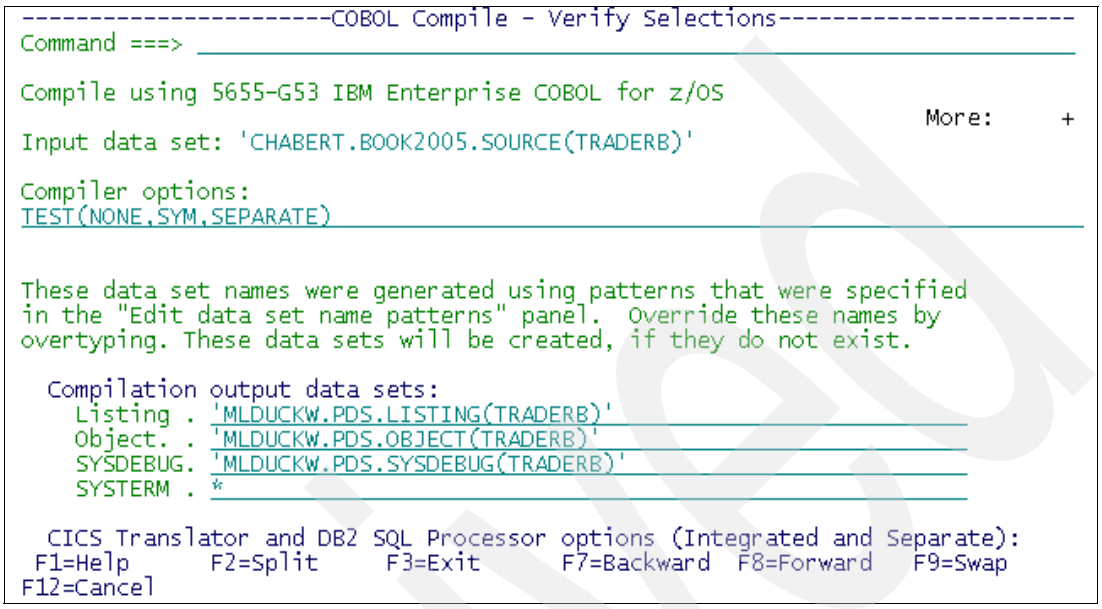


Figure 13-19 COBOL Compile - Verify Selections

TEST compiler option

The suboptions you specify when you compile your COBOL program with the TEST compiler option affect the size and performance of your program and the debugging capabilities available. Depending on the suboptions you specify, the compiler does the following tasks:

- Creates the symbol tables
- Creates debugging information
- Inserts hooks at selected points in your program

Debug Tool uses the symbol tables to obtain information about program variables. Programs compiled with one of the following compilers and with the SEPARATE suboption store debugging information and symbol tables in a separate file:

- Enterprise COBOL for z/OS and OS/390 Version 3
- COBOL for OS/390 and VM Version 2 Release 2
- COBOL for OS/390 and VM Version 2 Release 1 with APAR PQ40298

The file, called a separate debug file, must be a non-temporary file and must also be available during the debug session. If you move or rename the separate debug file, specify the new location using one of the following methods:

- Enter the SET SOURCE command with the name of the new location.
- Enter the SET DEFAULT LISTINGS command with the name of the new location.
- Specify the EQADEBUG DD statement with the name of the new location.
- Code the EQAUEDAT user exit with the new location.

Debug Tool uses hooks to gain control of your program at selected points during its execution. The hooks do not modify your source. The hooks are inserted into your program at one of the following times:

- At compile time, when you specify the TEST compiler option with any suboption except NONE.
- At run time, if the Dynamic Debug facility is activated, which is the default. To use the Dynamic Debug facility while you debug programs that do not have compiled-in hooks, compile your programs with the TEST(NONE) compiler option.

If you want to use the DATA suboption of the PLAYBACK ENABLE command, you must specify the SYM suboption of the TEST compiler option when you compile your program.

For VS COBOL II programs, in addition to the TEST compiler option, you must specify:

- The SOURCE compiler option. This option is required to generate a listing file.
- The RESIDENT compiler option. This option is required by Language Environment to ensure that the necessary Debug Tool routines are loaded dynamically at run time.

If a data set (such as Listing, Object, or SYSDEBUG) is missing when you press Enter from the Verify Selections screen, in either foreground or background mode, the Create data set panel shown in Figure 13-20 is displayed. You can overwrite any values on this panel.

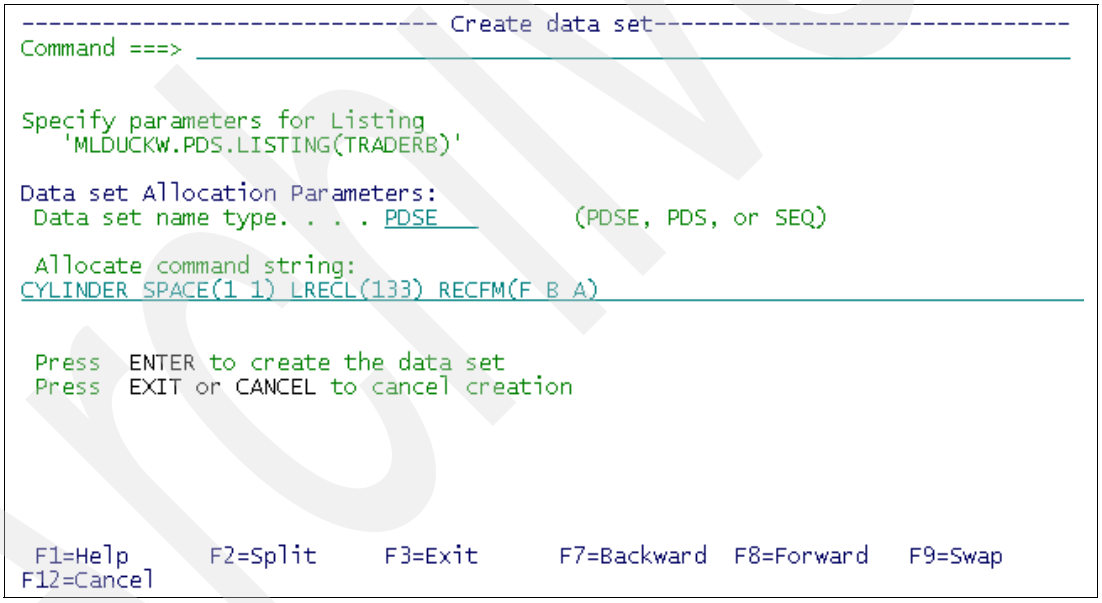


Figure 13-20 Create data set

Once the data sets are defined, the compile is started. If you defined the compile to be run in batch, you first get a panel showing the JCL (where you also might want to overwrite any statements), as shown in Figure 13-21.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       MLDUCKW.DTPPTEMP.C1151520                    Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
000008 //*******************************************************************
000009 //*
000010 //*  Debug Tool Utilities - COBOL Compile
000011 //*
000012 //*******************************************************************
000013 //EQASPBC1 EXEC PGM=IGYCRCTL,
000014 // PARM='TEST(NONE,SYM,SEPARATE) LIB'
000015 //STEPLIB    DD  DISP=SHR,
000016 //       DSNAME=IGY.V3R4M0.SIGYCOMP
000017 //SYSIN    DD  DISP=SHR,
000018 //       DSNAME=CHABERT.BOOK2005.SOURCE(TRADERB)
000019 //SYSPRINT DD  DISP=OLD,
000020 //       DSNAME=MLDUCKW.BOOK2005.LISTING(TRADERB)
000021 //SYSTERM  DD  SYSOUT=*
000022 //SYSDEBUG DD  DISP=OLD,
000023 //       DSNAME=MLDUCKW.BOOK2005.SYSDEBUG(TRADERB)
000024 //SYSLIN   DD  DISP=OLD,
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 13-21 Portion of the JCL created for the background job

You can save this as a model to be reused, after customization, for other COBOL compilations. Submit the job using the **SUB** command from the command line.

After the compile is completed and if you are using foreground mode, you get a slightly different panel, as shown in Figure 13-22.

```
------------------------COBOL Compile - View Outputs-------------------------
Command ===> ____________________________________________________________
                                                                More:     +
Compile using 5655-G53 IBM Enterprise COBOL for z/OS

DB2 Precompiler return code = NOT RUN
CICS Translator return code = NOT RUN
COBOL Compiler  return code = 12

The following data sets were used for this compilation.
Enter E to Edit, V to View, or B to Browse these data sets.

  _ Input . : 'CHABERT.BOOK2005.SOURCE(TRADERB)'

  Compilation output data sets:
    Listing : 'MLDUCKW.BOOK2005.LISTING(TRADERB)'
  _ Object. : 'MLDUCKW.BOOK2005.OBJECT(TRADERB)'
  _ SYSDEBUG: 'MLDUCKW.BOOK2005.SYSDEBUG(TRADERB)'
  _ SYSTERM : *

N DB2 SQL output DBRM data set:
    DBRMLIB :
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-22 COBOL Compile - View Outputs

You can browse the listings data set directly from here by entering a **'B'** just before the data set name. If you find that the compile did not work because of a syntax problem, then edit the source from here by entering **'E'** just before the input data set name.

As you can see in Figure 13-22, the return code from the compiler was 12. Enter **B** to browse the Listing data set to find the reason for the return code of 12, as shown in Figure 13-23.

```
 BROWSE    MLDUCKW.BOOK2005.LISTING(TRADERB)          Line 00001422 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
PP 5655-G53 IBM Enterprise COBOL for z/OS  3.4.1              TRADERB   Date 10
LineID  Message code  Message text

    71  IGYGR1216-I   A "RECORDING MODE" of "F" was assumed for file "REPORT-FIL

    74  IGYGR1216-I   A "RECORDING MODE" of "F" was assumed for file "TRAN-REPOR

   111  IGYDS1159-E   A "PICTURE" clause was not found for elementary item "CUST

   112  IGYDS0010-S   A "COPY" statement was found but the "LIB" compiler option
                      item following the next period.

                      Same message on line:    114    116

   113  IGYDS1159-E   A "PICTURE" clause was not found for elementary item "COMP

   115  IGYDS1159-E   A "PICTURE" clause was not found for elementary item "TRAN

  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 13-23 COBOL compile - section of listing data set showing errors

After correcting all errors, you can link-edit the program. In this case, go back to the COBOL Compile program preparation screen, scroll down, and add the copy libraries) to the SYSLIB concatenation. Then, repeat the steps for a clean compile.

After correcting all errors, link edit the object code. The link utility, option L from the Debug Tool Program Preparation panel, lets you select either a foreground or a background link. You must specify your object library. You can define the object member, but are not required to. If you do not specify the object member, you are presented with a member selection list later.

The panels shown for the link-edit are quite similar to those used for compiling the program and are not included here.

Managing and using the Debug Tool setup file

Setup files can save you time when you are debugging a program that must be restarted multiple times. Setup files store information required to run your program and start Debug Tool.

You can create several setup files for each program. Each setup file can store information about starting and running your program in different circumstances. To create and manage files, use Debug Tool Setup Utility (DTSU), which is part of Debug Tool. You do not require Debug Tool Utilities and Advanced Functions to use this tool.

From the Debug Tool Utilities panel shown in Figure 13-24, select option **2 - Manage and Use Debug Tool Setup Files**.

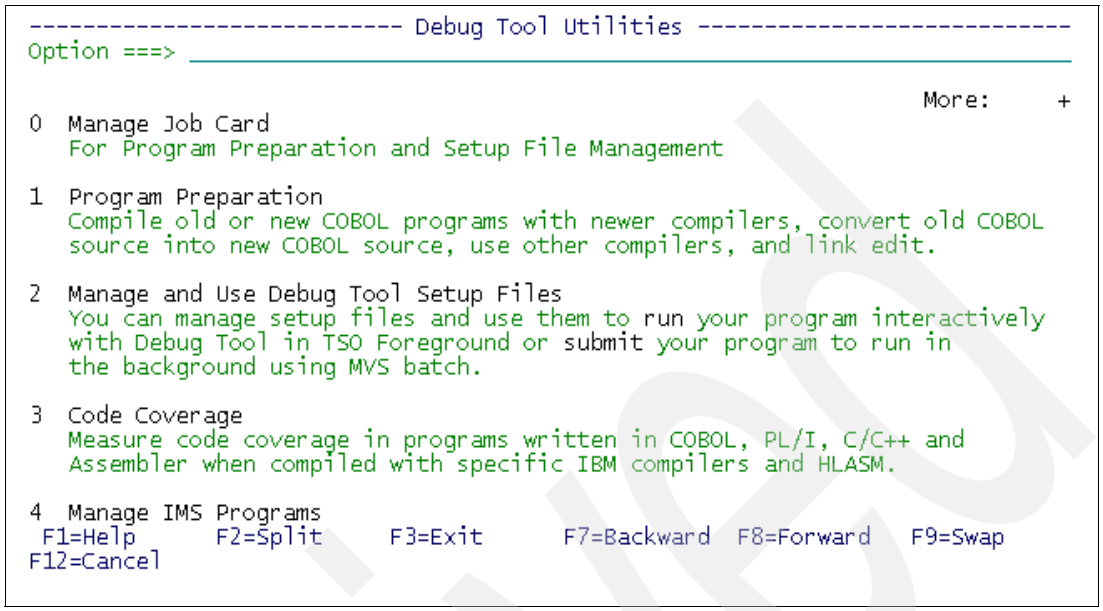


Figure 13-24 Debug Tool Utilities

In this example we describe how to create a new setup file member:

1. Enter the new member name DEBUGTR, as shown in Figure 13-25, and press Enter.

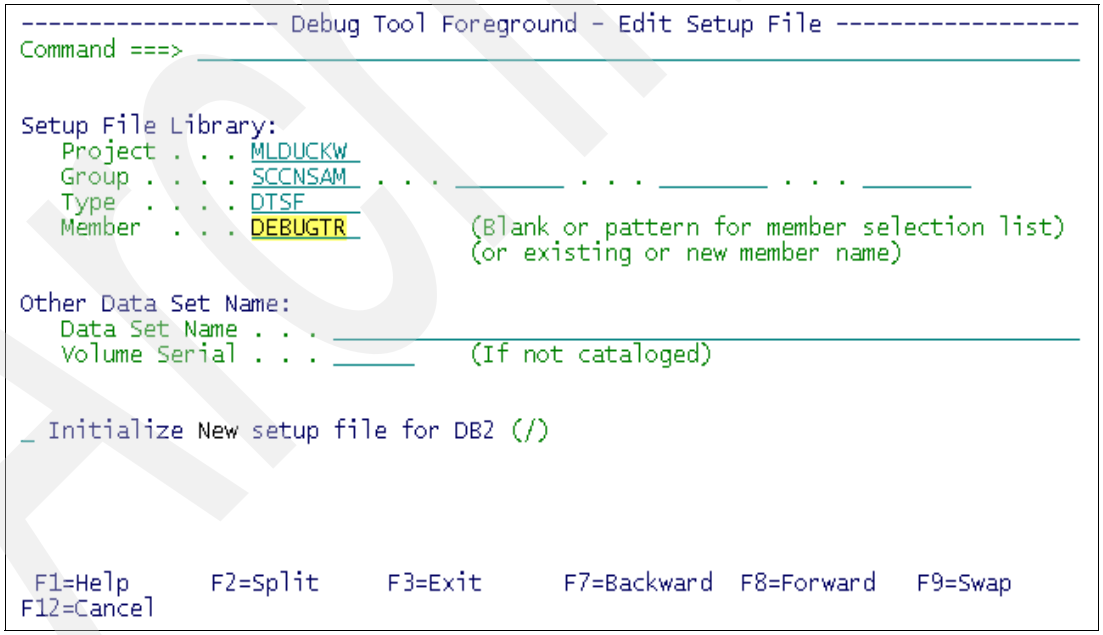


Figure 13-25 Debug Tool Foreground - Edit Setup File

After pressing Enter, the panel in Figure 13-26 is displayed.

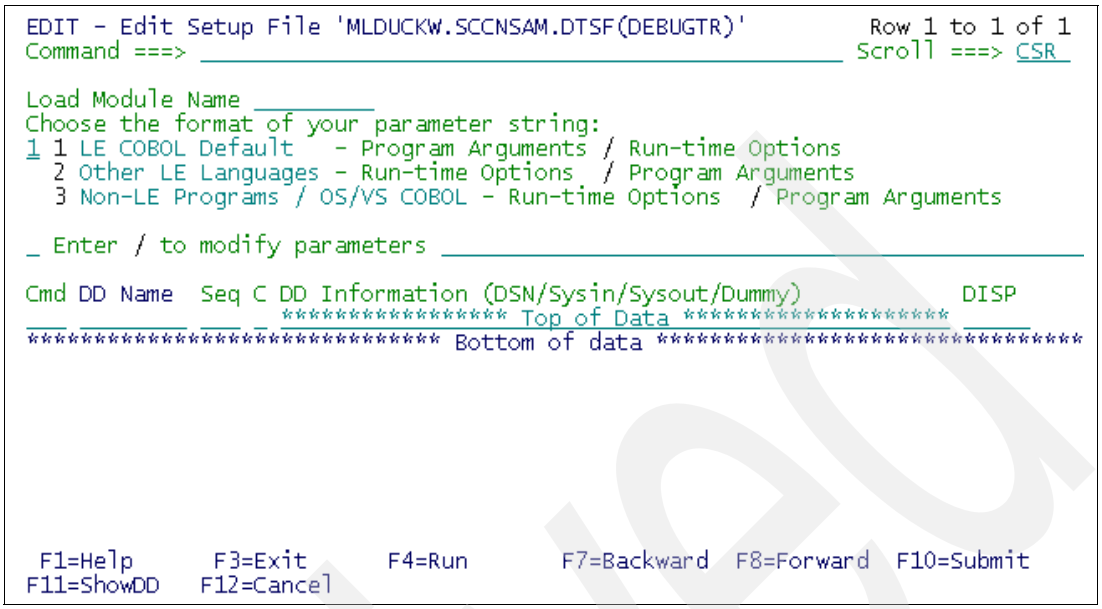


Figure 13-26 Debug Tool - Edit Setup File

The top part of the Edit Setup File panel contains the name of the program (load module) that you want to run and the run-time parameter string. If the setup file is for a DB2 program, the panel also contains fields for the DB2 System identifier and the DB2 plan.

The bottom part of the Edit Setup File panel contains the file allocation statements. This part of the panel is similar to an ISPF edit panel. You can insert new lines, copy (repeat) a line, delete a line, and type over information about a line.

2. You can enter the COPY command to copy information from another setup file or JCL data set into the current setup file as shown in Figure 13-27.

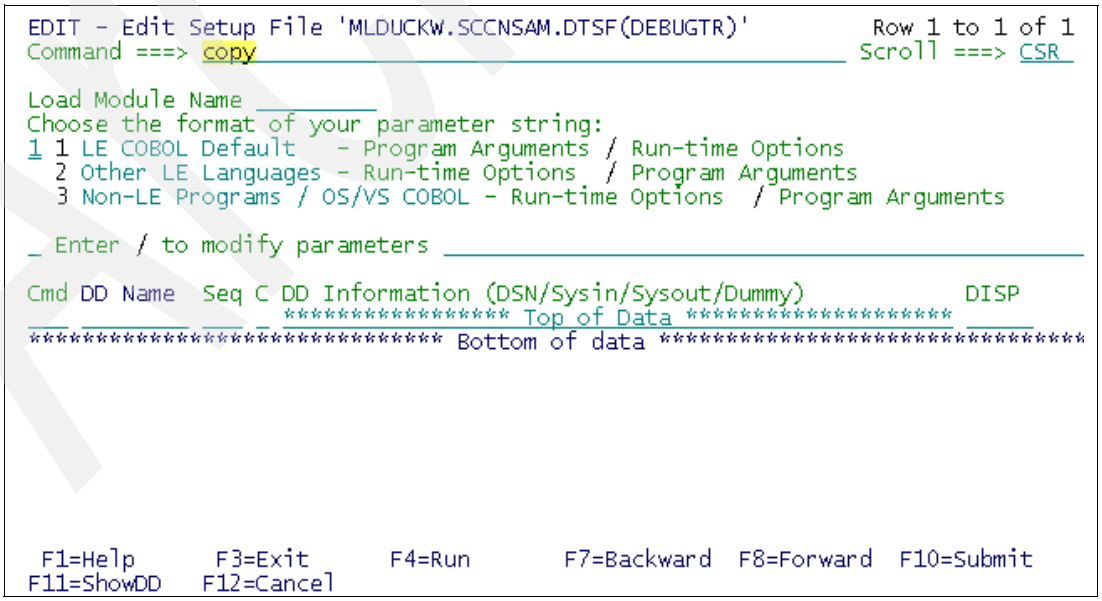


Figure 13-27 Entering the COPY command

3. The JCL to execute the program we want to test resides at ABAPHEN.TRADER.JCL(TRADERB).

   Fill in the library name where the JCL resides, put an asterisk in the Member name if you do not know it, and press Enter, as shown in Figure 13-28.

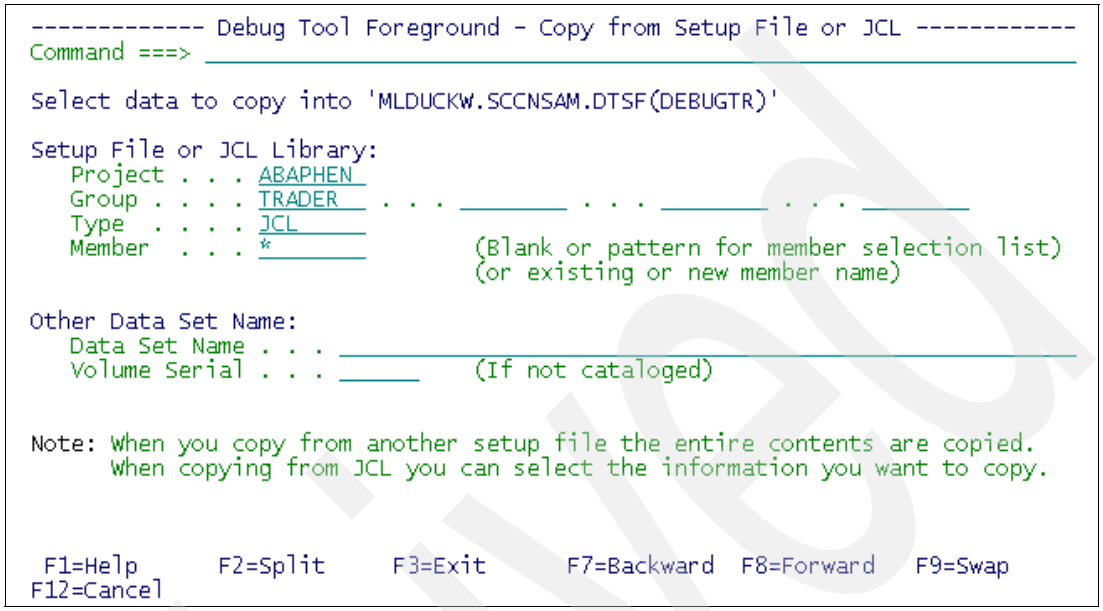


Figure 13-28 Debug Tool - Copy from Setup File or JCL

4. Figure 13-29 is displayed. Select the member TRADERB and press Enter.

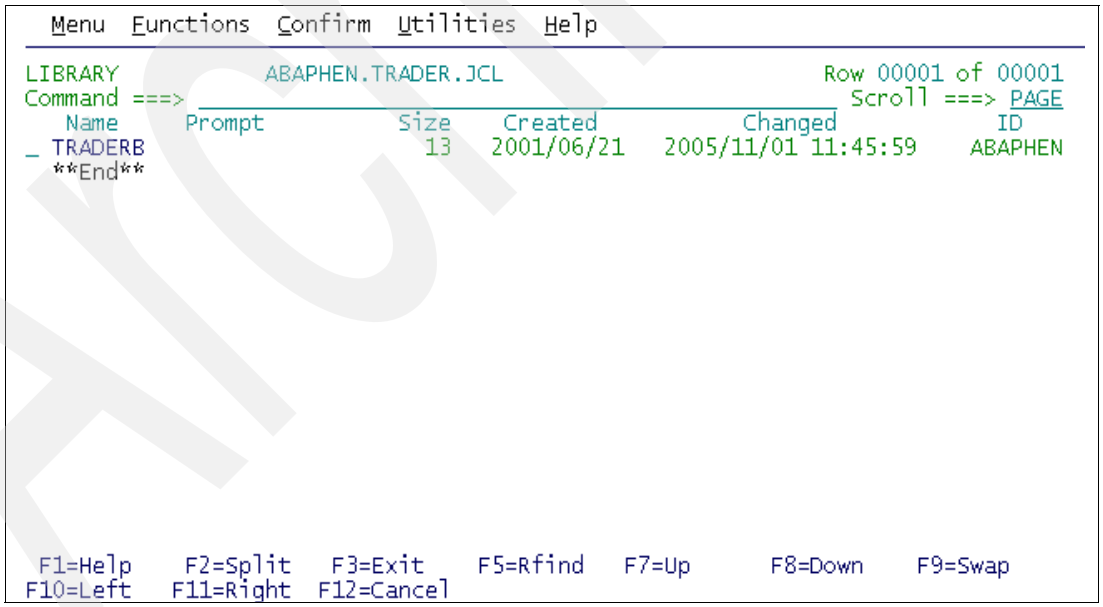


Figure 13-29 Library members

5. Figure 13-30] is displayed. Type **S*** on the command line to select all the JCL lines and press Enter.

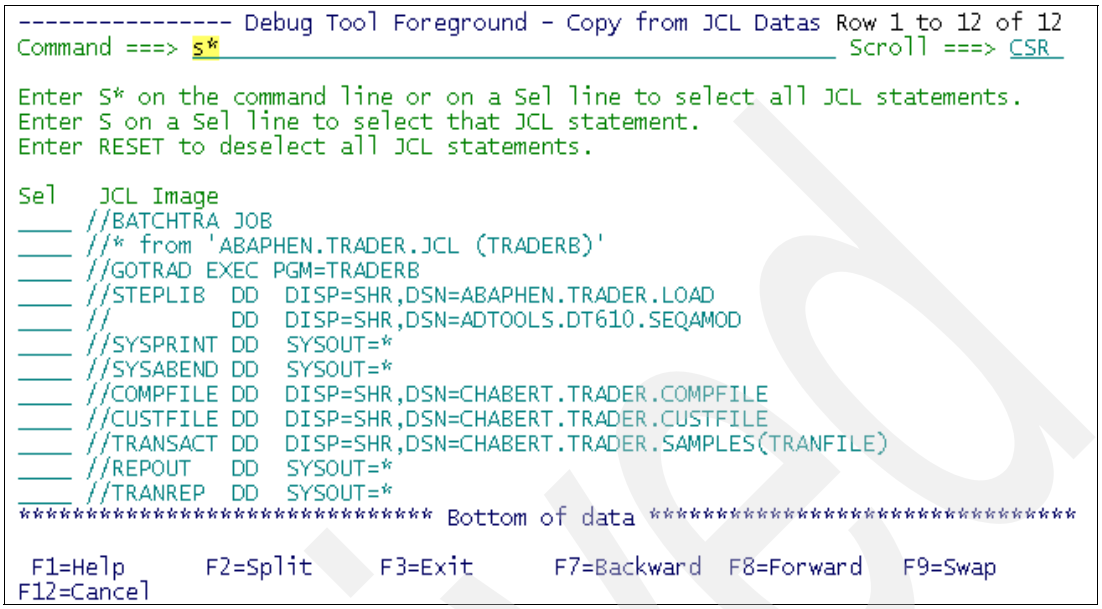


Figure 13-30 JCL copied from the TRADERB member

6. All JCL statements are selected, as shown in Figure 13-31.

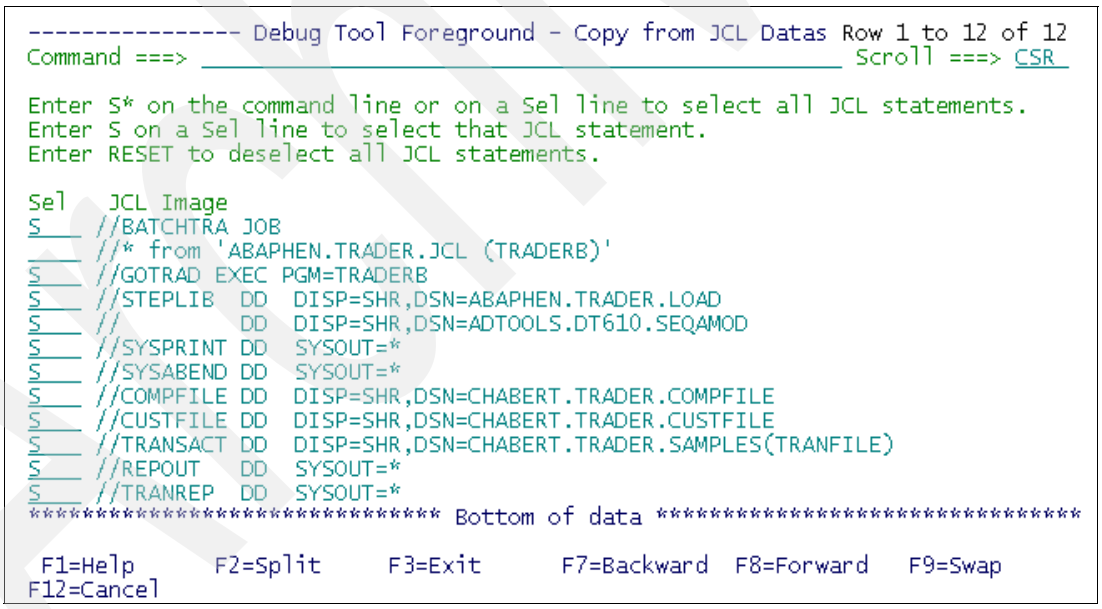


Figure 13-31 Selected JCL statements

7. All JCL statements have now been selected. Press PF3 and the panel for editing the setup is displayed, as shown in Figure 13-32. Notice that the JCL statements have been moved to the edit setup file.

```
EDIT - Edit Setup File 'MLDUCKW.SCCNSAM.DTSF(DEBUGTR)'        Row 1 to 10 of 10
Command ===> ____________________________________________________ Scroll ===> CSR

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs / OS/VS COBOL - Run-time Options  / Program Arguments

_ Enter / to modify parameters ____________________________________________

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)             DISP
___ ________ ___ _ ****************** Top of Data **********************
___ COMPFILE 1   _ 'CHABERT.TRADER.COMPFILE'                           SHR
___ CUSTFILE 1   _ 'CHABERT.TRADER.CUSTFILE'                           SHR
___ REPOUT   1   _ SYSOUT=*                                            ____
___ STEPLIB  1   _ 'ABAPHEN.TRADER.LOAD'                               SHR
___ ________ 2   _ 'ADTOOLS.DT610.SEQAMOD'                             SHR
___ SYSABEND 1   _ SYSOUT=*                                            ____
___ SYSPRINT 1   _ SYSOUT=*                                            ____
___ TRANREP  1   _ SYSOUT=*                                            ____
___ TRANSACT 1   _ 'CHABERT.TRADER.SAMPLES(TRANFILE)'                  SHR
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F11=ShowDD   F12=Cancel
```

Figure 13-32 Edit Setup panel with allocations copied from the JCL member, TRADERB

8. You can set parameters for debugging by putting a dash in front of the field, *Enter / to modify parameters*, and pressing Enter. Figure 13-33 is the resulting screen.

```
--------------- Debug Tool Foreground - Modify Parameter String --------------
Command ===> ____________________________________________________________
                                                                More:     +
Modify Test options, other run-time options, and Program arguments

Select Test Options:

Test Option     ==> TEST           Test/Notest
Test Level      ==> ALL            All/Error/None
Commands File                      *, DDname, or Data Set Name
                ==> *______________________________________________________
Prompt Level    ==> PROMPT         Prompt/NoPrompt
Preference File                    *, DDname, or Data Set Name
                ==> *______________________________________________________

Select (/) a session type and provide parameters:

 / Full-screen mode
   Terminal LU     ==> ____________ blank or VTAM Terminal LU

 _ Full-screen mode using the Debug Tool Terminal Interface Manager
   User ID         ==> ____________ User ID
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-33 Debug Tool - Edit TEST parameters

If you were planning on using the remote debugger, you would scroll down to the next screen and fill in the appropriate port number. In this case though, we are going to use the defaults.

9. Press PF3 to accept the defaults. This brings you back to the previous screen, which has been updated with the TEST PARM values as shown in Figure 13-34.

```
EDIT - Edit Setup File 'MLDUCKW.SCCNSAM.DTSF(DEBUGTR)'        Row 1 to 10 of 10
Command ===> _____________________________________________ Scroll ===> CSR

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs / OS/VS COBOL - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,*,PROMPT,MFI:*) '

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ****************** Top of Data **********************
___ COMPFILE 1   _ 'CHABERT.TRADER.COMPFILE'                          SHR
___ CUSTFILE 1   _ 'CHABERT.TRADER.CUSTFILE'                          SHR
___ REPOUT   1   _ SYSOUT=*                                           ____
___ STEPLIB  1   _ 'ABAPHEN.TRADER.LOAD'                              SHR
___ ________ 2   _ 'ADTOOLS.DT610.SEQAMOD'                            SHR
___ SYSABEND 1   _ SYSOUT=*                                           ____
___ SYSPRINT 1   _ SYSOUT=*                                           ____
___ TRANREP  1   _ SYSOUT=*                                           ____
___ TRANSACT 1   _ 'CHABERT.TRADER.SAMPLES(TRANFILE)'                 SHR
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F11=ShowDD   F12=Cancel
```

Figure 13-34 Debug Tool - Edit setup file with TEST parameters filled in

10. A preference file, INSPPREF, was not included in the JCL above, so we must add this file manually to the DD Name allocations. Put an **I** in the Cmd field to the left and press Enter, as shown in Figure 13-35.

```
EDIT - Edit Setup File 'MLDUCKW.SCCNSAM.DTSF(DEBUGTR)'        Row 1 to 10 of 10
Command ===> _____________________________________________ Scroll ===> CSR

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs / OS/VS COBOL - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,*,PROMPT,MFI:*) '

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
i__ ________ ___ _ ****************** Top of Data **********************
___ COMPFILE 1   _ 'CHABERT.TRADER.COMPFILE'                          SHR
___ CUSTFILE 1   _ 'CHABERT.TRADER.CUSTFILE'                          SHR
___ REPOUT   1   _ SYSOUT=*                                           ____
___ STEPLIB  1   _ 'ABAPHEN.TRADER.LOAD'                              SHR
___ ________ 2   _ 'ADTOOLS.DT610.SEQAMOD'                            SHR
___ SYSABEND 1   _ SYSOUT=*                                           ____
___ SYSPRINT 1   _ SYSOUT=*                                           ____
___ TRANREP  1   _ SYSOUT=*                                           ____
___ TRANSACT 1   _ 'CHABERT.TRADER.SAMPLES(TRANFILE)'                 SHR
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F11=ShowDD   F12=Cancel
```

Figure 13-35 Adding a line so we can specify the INSPPREF allocation

11. The panel shown in Figure 13-36 is displayed.

```
EDIT - Edit Setup File 'MLDUCKW.SCCNSAM.DTSF(DEBUGTR)'        Row 1 to 10 of 11
Command ===> ____________________________________________________ Scroll ===> CSR

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs / OS/VS COBOL - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,*,PROMPT,MFI:*) '

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ****************** Top of Data ********************* _____
___ ________ 1   _ ** Select Detail DD Information **                  _____
___ COMPFILE 1   _ 'CHABERT.TRADER.COMPFILE'                           SHR
___ CUSTFILE 1   _ 'CHABERT.TRADER.CUSTFILE'                           SHR
___ REPOUT   1   _ SYSOUT=*                                            _____
___ STEPLIB  1   _ 'ABAPHEN.TRADER.LOAD'                               SHR
___ ________ 2   _ 'ADTOOLS.DT610.SEQAMOD'                             SHR
___ SYSABEND 1   _ SYSOUT=*                                            _____
___ SYSPRINT 1   _ SYSOUT=*                                            _____
___ TRANREP  1   _ SYSOUT=*                                            _____
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F11=ShowDD   F12=Cancel
```

Figure 13-36 Debug Tool - Edit

12. As shown above, a line has been added with the text `**Select Detail DD Information.**` On this line, add the data set and member that contains the initial commands for the debug session, as shown in Figure 13-37. After pressing Enter, the new DD Name is moved so that all the DD Names are now in ascending alphabetical order.

```
EDIT - Edit Setup File 'MLDUCKW.SCCNSAM.DTSF(DEBUGTR)'        Row 1 to 10 of 11
Command ===> ____________________________________________________ Scroll ===> CSR

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs / OS/VS COBOL - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,*,PROMPT,MFI:*) '

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ****************** Top of Data ********************* _____
___ COMPFILE 1   _ 'CHABERT.TRADER.COMPFILE'                           SHR
___ CUSTFILE 1   _ 'CHABERT.TRADER.CUSTFILE'                           SHR
___ INSPPREF 1   _ 'CHABERT.DEBUG.PREF(TRAD1)'                         SHR
___ REPOUT   1   _ SYSOUT=*                                            _____
___ STEPLIB  1   _ 'ABAPHEN.TRADER.LOAD'                               SHR
___ ________ 2   _ 'ADTOOLS.DT610.SEQAMOD'                             SHR
___ SYSABEND 1   _ SYSOUT=*                                            _____
___ SYSPRINT 1   _ SYSOUT=*                                            _____
___ TRANREP  1   _ SYSOUT=*                                            _____
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F11=ShowDD   F12=Cancel
```

Figure 13-37 INSPPREF file added

Now that the setup file is read, you can press PF4 or issue the RUN command on the command line to begin debugging the program.

13.4 Debugging non-LE programs using Debug Tool

There are several considerations when you debug programs that do not run under the Language Environment. Some of these are unique to programs that contain no Language Environment routines, others pertain only when the initial program does not execute under control of the Language Environment, and still others apply to all programs that have mixtures of non-Language Environment and Language Environment programs.

13.4.1 Debugging exclusively non-LE programs

When Language Environment is not active, you can debug only assembler and disassembly programs. Debugging programs written in other languages requires the presence of an active LE runtime environment. In the case of OS/VS COBOL, it can be run with LE only if you link-edit with LE first, before running with LE runtime library.

13.4.2 Debugging MVS batch or TSO non-LE initial programs

If the initial program that is invoked does not run under Language Environment and you want to begin debugging before Language Environment is initialized, you must use the EQANMDBG program to start both Debug Tool and your user program.

You do not have to use EQANMDBG to initiate a Debug Tool session if the initial user program runs under control of the Language Environment, even if other parts of the program do not run under the Language Environment.

13.4.3 Debugging CICS non-LE initial programs

The initial program that you request in a DTCN or CADP debug profile to start up a non-Language Environment debugging session must be the first program started for the CICS transaction or the initial program that runs for an EXEC CICS LINK or XCTL.

If you set the debug profile to a non-Language Environment program that meets both of the following requirements, Debug Tool does not start if:

- The program is called only statically.
- The program is called dynamically and does not cross a link boundary by using EXEC CICS LOAD/CALL.

13.5 Debugging assembler programs using Debug Tool

This section describes debugging assembler programs using Debug Tool.

13.5.1 Preparing the program

In order to utilize Debug Tool's full capabilities to debug an assembler program, you must first prepare the program using the following steps:

1. Assemble your program with the proper options.
2. Create the EQALANGX file. This can also be referred to as a debug file or sidefile. While not directly related to debugging, this file can also be used by Fault Analyzer to provide source code information about an abend.
3. Link-edit your program.

If you use Debug Tool Utilities to prepare your assembler program, you can combine steps 1 and 2 into a single step.

You must have Debug Tool Utilities and Advanced Functions installed on your system to prepare and debug assembler programs.

You can debug an assembler program several ways:

- Using the full-screen 3270 interface
- Using the Distributed Debugger
- Using the Compiled Language Debugger component of WebSphere Developer Debugger for zSeries

Before beginning to debug

When you debug an assembler program, you can use most of Debug Tool commands. There are three differences between debugging an assembler program and debugging programs written in other programming languages supported by Debug Tool:

- After you assemble your program, you must create a debug information file, also called the EQALANGX file. Debug Tool uses this file to obtain information about your assembler program.
- Debug Tool assumes all compile units are written in some high-level language (HLL). You must inform Debug Tool that a compile unit is an assembler compile unit and instruct Debug Tool to load the assembler compile unit's debug information. Do this by entering the LOADDEBUGDATA (or LDD) command.
- Assembler does not have language elements you can use to write expressions. Debug Tool provides assembler-like language elements you can use to write expressions for Debug Tool commands that require an expression. See *Debug Tool for z/OS Reference and Messages* for a description of the syntax of the assembler-like language.

After you verify that your assembler program meets these requirements, prepare your assembler program.

13.5.2 Preparing programs and modifying setup files with Debug Tool Utilities

The examples in the following sections show you how to use Debug Tool Utilities to prepare your programs and how to create, manage, and use a setup file. The examples guide you through the following tasks:

- Creating personal data sets with the correct attributes.
- Starting Debug Tool Utilities.
- Compiling or assembling your program using Debug Tool Utilities. You must have Debug Tool Utilities and Advanced Functions installed on your system to run the steps in this task. If you do not have this product installed, you can build your program through your usual methods and resume the example with the next step.
- Modifying and using a setup file to run your program in the foreground or in batch.

Creating sample data sets for debugging an assembler program

Create the data sets with the names and attributes described in Table 13-1. Allocate five tracks for each of the data sets. Partitioned data sets should be specified with five blocks for the directory.

Table 13-1 Names and attributes to use when you create your own data sets

Data set name	LRECL	BKLSIZE	RECFM	DSORG
hlq.SAMPLE.ASM	80	*	FB	PO

Copy the members of the SEQASAMP data set described in Table 13-2 into the personal data sets you just created. This dataset is provided as the Debug Tool sample library.

Table 13-2 Description of samples

SEQASAMP member name	Sample data set name	Description of member
EQAWPP5	*hlq*.sample.asm(wpp5)	assembler source code
EQAWSU5	*hlq*.sample.asm(wsu5)	setup file for EQAWPP5

Starting Debug Tool Utilities

To start Debug Tool Utilities, use one of these methods, as appropriate:

- If Debug Tool Utilities was installed as an option on an existing ISPF panel, select that option.
- If Debug Tool Utilities data sets were installed as part of your log-on procedure, enter the following command from ISPF option 6:

  ```
  EQASTART
  ```

- If Debug Tool Utilities was installed as a separate application, enter the following command from ISPF option 6:

  ```
  EX 'hlq.SEQAEXEC(EQASTART)'
  ```

Debug Tool Utilities primary panel (EQA@PRIM) is displayed. On the command line, enter the `PANELID` command. This command displays the name of each panel in the upper left corner of the screen. These names are used as navigation aids in the instructions provided in this section. After you complete these examples, you can stop the display of these names by entering the PANELID command.

13.5.3 Assembling your program using Debug Tool Utilities

1. Figure 13-38 shows the Main Panel of Debug Tool. Select option **1** to begin program preparation.

```
---------------------------- Debug Tool Utilities ----------------------------
Option ===> ____________________________________________________________________

                                                                        More:     +
0  Manage Job Card
   For Program Preparation and Setup File Management

1  Program Preparation
   Compile old or new COBOL programs with newer compilers, convert old COBOL
   source into new COBOL source, use other compilers, and link edit.

2  Manage and Use Debug Tool Setup Files
   You can manage setup files and use them to run your program interactively
   with Debug Tool in TSO Foreground or submit your program to run in
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-38 Assembler program preparation selection

2. Select option **5** and press Enter to assemble using the high-level assembler as shown in Figure 13-39.

```
----------------------- Debug Tool Program Preparation -----------------------
Option ===> ____________________________________________________________________

                                                                        More:     +
1  COBOL Compile
   Using 5655-G53 IBM Enterprise COBOL for z/OS

2  COBOL Convert and Compile
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid
   and   5655-G53 IBM Enterprise COBOL for z/OS

3  PL/I  Compile
   Using 5655-H31  IBM(R) Enterprise PL/I for z/OS

4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-39 Selecting option 5 to assemble using high-level assembler

3. Prepare an assembler program. As shown in Figure 13-40, the following inputs are required:
   - Choose either foreground (F) or background (B) mode. In this example background was selected.
   - Source Library names:
     - Project: *hlq* (in this example, it is **gracine**)
     - Group: **sample**
     - Type: **asm**
     - Member: **wpp5**
   - Select **Enter / to edit options and data set name patterns** to specify a naming pattern for the output data sets in the field Data set naming pattern as shown in Figure 13-40. Press Enter.

     Notice that in this sampling, CICS and DB2 are not part of this test. Therefore, option N for *no* has been selected.

```
-------------Debug Tool Program Preparation - High Level Assembler------------
Command ===> ____________________________________________________________

Assemble using High Level Assembler
F Foreground or Batch Processing  (F B)
                                                                More:     +
Specify primary input data set for compilation.
Source Library:
   Project . . . GRACINE
   Group . . . . SAMPLE   . . . ________ . . . ________ . . . ________
   Type  . . . . ASM
   Member  . . . WPP5     (Blank or pattern for member selection list)

Other Partitioned or Sequential Data Set:
   Data Set Name . . . ____________________________________________________
   Volume Serial . . . ______    (If not cataloged)

Data set name patterns, CICS and SQL processing options:
/ Enter / to edit options and data set name patterns
N CICS    S Separate CICS translator, N None
N DB2/SQL S Separate DB2 precompiler, N None

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-40 Options panel for asm program preparation (part 1 of 2)

4. Scroll down by pressing F8 to see the remainder of the panel as shown in Figure 13-41. Enter LE's SCEEMAC library in the SYSLIB DD allocation.

```
-------------Debug Tool Program Preparation - High Level Assembler------------
Command ===> 

Assemble using High Level Assembler
F Foreground or Batch Processing  (F B)
                                                             More:    - +

Assembler Options Data Set(ASMAOPT):

Assembler Options String:


Specify secondary input data sets for Copy processing:
DDname         Syslib data set Names
  SYSLIB   . . . 'cee.sceemac'
           . . .
           . . .
           . . .
           . . .
           . . .
           . . .
 F1=Help      F2=Split      F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-41 Options panel for asm program preparation (part 2 of 2)

5. Press Enter to display the edit options and data set name patterns, as shown in Figure 13-42. This is part 1 of a two-part panel displaying the options and data set name patterns.

```
--------High Level Assembler - Edit options and data set name patterns--------
Command ===> 

Assemble using High Level Assembler
                                                             More:     +

Specify data set name patterns for output data sets:
Pattern characters: /n, /B, /L, /M, /U, /P  Enter Help for usage information.
Assembler output data sets:
   Listing . '/U./B.ASMLIST(/M)'
   Object. . '/U./B.OBJECT(/M)'
   SYSADATA. '/U./B.SYSADATA(/M)'
   LangFile. '/U.EQALANGX(/M)'
   SYSTERM . *

CICS Translator and DB2 SQL Processor options (Integrated and Separate):
   CICS. . . CICS
   DB2 . . . HOST(ASM)

DB2 SQL Output DBRM data set (Integrated and Separate):
   DBRMLIB . '/U./B.DBRMLIB(/M)'

 F1=Help      F2=Split      F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-42 Options and data set patterns panel (part 1 of 2)

6. Press PF8 to scroll forward and display the second portion of this panel, as shown in Figure 13-43.

```
--------High Level Assembler - Edit options and data set name patterns--------
Command ===> 

Assemble using High Level Assembler
                                                            More:    -
   LangFile. '/U.EQALANGX(/M)'
   SYSTERM . *

CICS Translator and DB2 SQL Processor options (Integrated and Separate):
   CICS. . . CICS
   DB2 . . . HOST(ASM)

DB2 SQL Output DBRM data set (Integrated and Separate):
   DBRMLIB . '/U./B.DBRMLIB(/M)'

CICS Translator data set name patterns (Separate):
   Listing . '/U./B.CICSCC(/M)'
   SYSTERM . *

DB2 Precompiler data set name patterns (Separate):
   Listing . '/U./B.DB2CC(/M)'
   SYSTERM . *
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-43 Options and data set patterns panel (part 2 of 2)

7. Press PF3, followed by Enter. The Verify Selections panel is displayed, as shown in Figure 13-44.

```
-------------------High Level Assembler - Verify Selections-------------------
Command ===> 

Assemble using High Level Assembler
                                                              More:    +
Input data set: 'GRACINE.SAMPLE.ASM(WPP5)'

Assembler Options Data Set(ASMAOPT):


Assembler Options:
ADATA


These data set names were generated using patterns that were specified
in the "Edit data set name patterns" panel.  Override these names by
overtyping. These data sets will be created, if they do not exist.

  Assembler output data sets:
    Listing . 'MLDUCKW.SAMPLE.ASMLIST(WPP5)'
    Object. . 'MLDUCKW.SAMPLE.OBJECT(WPP5)'
    SYSADATA: 'MLDUCKW.SAMPLE.SYSADATA(WPP5)'
    LangFile: 'MLDUCKW.EQALANGX(WPP5)'
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-44 Verification selection panel displayed

8. Next, we scroll down (**F8**) and specify **N** for CICS translator and **N** for the DB2 SQL output DBRM and DB2 precompiler datasets. None of these programs contain CICS or DB2 instructions as shown in Figure 13-45. Press Enter.

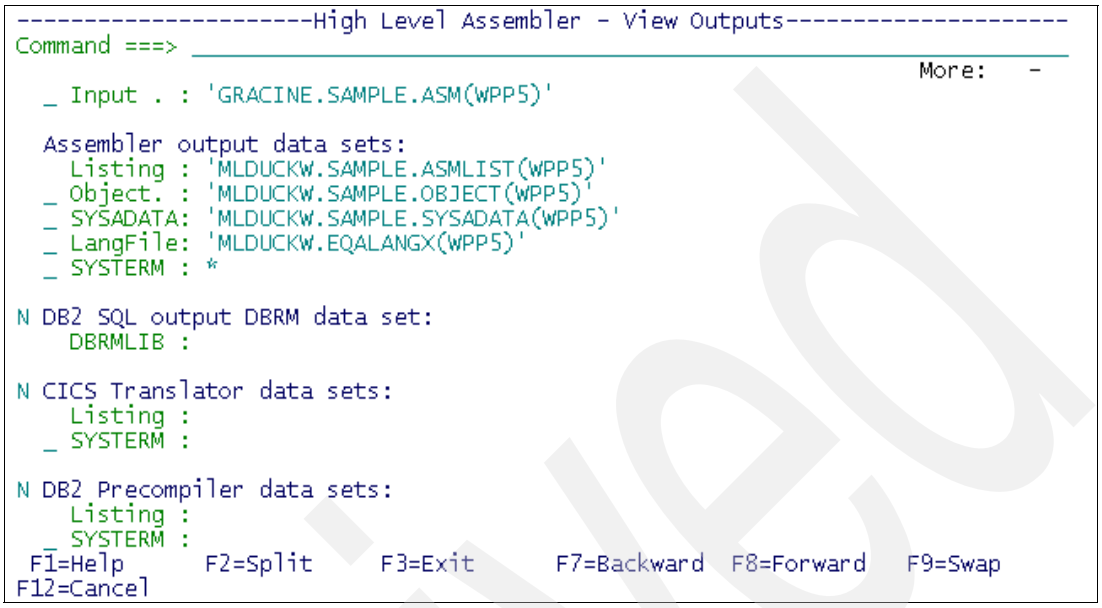


Figure 13-45 Selecting to not generate CICS or DB2 datasets.

9. Make a note of the data set name for object compilation output. For an ASSEMBLER program, the data set name looks similar to the name: *hlq*.SAMPLE.OBJECT(WPP5). You use this name when you link your object modules (in this scenario, the *hlq* is **gracine**).
10. Press Enter. The create dataset panel for your assembler listing is displayed, as shown in Figure 13-46.

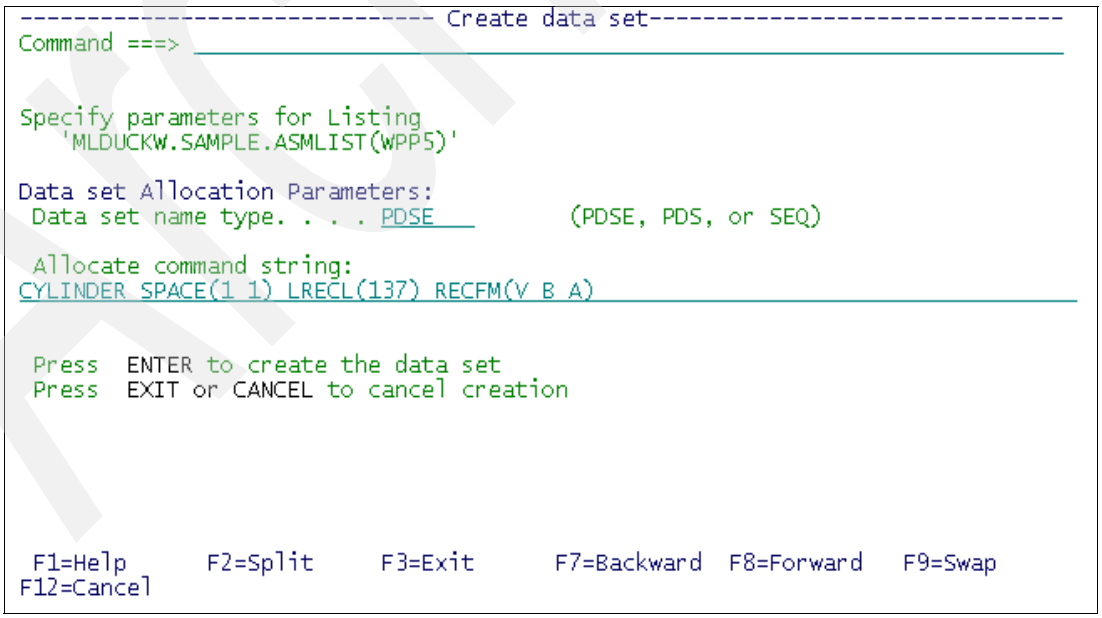


Figure 13-46 Panel display EQAPPA1, creating listing PDSE data set

11. Continue to press Enter to view and edit the specified parameters for the remainder of newly created data sets.
12. Once all datasets have been addressed, the following log screen is displayed as shown in Figure 13-47.

```
EQAZ040I High Level Assembler started for 'GRACINE.SAMPLE.ASM(WPP5)' using 'GRA
CINE.SAMPLE.ASM(WPP5)'.
EQAZ041I High Level Assembler library: ASMA90 invoked.
EQAZ047I EQALANGX started for 'MLDUCKW.SAMPLE.SYSADATA(WPP5)'.
EQAZ048I EQALANGX EQALANGX invoked from 'ADTOOLS.DT710.SEQAMOD(EQALANGX)'.
***
```

Figure 13-47 Pop-up message screen

13. Press Enter to continue. The View Outputs screen is displayed, as shown in Figure 13-48.

```
----------------------High Level Assembler - View Outputs---------------------
Command ===> ____________________________________________________________________
                                                                        More:     +
Assemble using High Level Assembler

DB2 Precompiler      return code = NOT RUN
CICS Translator      return code = NOT RUN
High Level Assembler return code = 0

The following data sets were used for this compilation.
Enter E to Edit, V to View, or B to browse these data sets.

  _ Input . : 'GRACINE.SAMPLE.ASM(WPP5)'

  Assembler output data sets:
    Listing : 'MLDUCKW.SAMPLE.ASMLIST(WPP5)'
  _ Object. : 'MLDUCKW.SAMPLE.OBJECT(WPP5)'
  _ SYSADATA: 'MLDUCKW.SAMPLE.SYSADATA(WPP5)'
  _ LangFile: 'MLDUCKW.EQALANGX(WPP5)'
  _ SYSTERM : *

N DB2 SQL output DBRM data set:
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-48 View outputs panel

As you see, the return code result is displayed. A return code of 0 (zero) is always a very good sign. If the result is greater than a return code of 4 (four), proper problem determination is necessary to assure a successful execution.

Important: Accepted return codes are 0 (zero) and 4 (four).

14. At this point, you can view the various output files. When done viewing, back out and link the program as shown in the other language examples earlier in this chapter.

14

Using Debug Tool under UNIX System Services

In this chapter we provide a brief example of compiling for and using Debug Tool in the UNIX System Services (USS) environment. Enterprise PL/I, Enterprise COBOL, and all C/C++ compilers shipped with the operating system support compilation under the UNIX System Services shell.

This chapter covers compilation, invoking Debug Tool in this environment, and restrictions on what can be done. It does not go into debugging the application.

14.1 Common considerations for all languages

The UNIX System Services shell enforces re-entrancy, so non-reentrant programs should not be run under the USS shell.

The default single screen debugging cannot be used when debugging USS applications. The VTAM interface or the remote debugger interface must be used.

14.1.1 Specifying the TEST runtime option

Under UNIX System Services, the LE runtime options are specified using the _CEE_RUNOPTS environment variable. This variable can be set using an export command in the shell, for example:

```
export _CEE_RUNOPTS='TEST(,,,TCPIP&9.30.62.149%8000:*)'
```

This runtime option now applies to every job that runs until you unset the environment variable using the command:

```
unset _CEE_RUNOPTS
```

Be very careful about what you run in the shell when the TEST runtime option is specified in _CEE_RUNOPTS. Commands such as **tar**, **c89**, and **make** can cause the tar, c89, and make utilities to appear in the remote debugger screen (in an assembler view since no debug information is available). To check what Language Environment runtime options you currently have set, you can use the command:

```
echo $_CEE_RUNOPTS
```

Note: When running under USS, the test option can always be specified in _CEE_RUNOPTS regardless of the language used to compile the program entry point. No slashes (/) are required.

Other methods of specifying runtime options

CEEUOPTS and the C #pragma runopts directive can also be used under the shell to invoke the debugger. However, the _CEE_RUNOPTS environment variable tends to be the preferred method because TCP/IP addresses and VTAM terminal LU IDs can change, and the environment variable allows the person debugging the program to change his TEST runtime option without having to recompile or relink their program.

14.1.2 Special considerations for running DLL applications from the shell

When running a USS module and looking for DLLs to load, the loader looks in the path specified by the LIBPATH environment variable. This should be set to point to your DLLs prior to running your DLL application.

14.2 Using C/C++ and Debug Tool under USS

In this example we rebuild the DLL application used in the C/C++ example. In this case our source is called:

- main.c
- setup.c
- process.c
- report.c
- cleanup.c

Each file is compiled into its own DLL and its export file is saved.

14.2.1 Compiling a C/C++ application

The following commands were used to build the DLL application. Notice the -g to ask for debug information and the LANGLVL(EXTENDED) to allow the use of the ctest function.

```
c89 -g -Wc,EXPORTALL -Wl,DLL -o setupd  setup.c
c89 -g -Wc,EXPORTALL,DLL,LANGLVL\(extended\) -Wl,DLL -o processd process.c setupd.x
c89 -g -Wc,EXPORTALL,DLL -Wl,DLL -o reportd  report.c setupd.x
c89 -g -Wc,EXPORTALL,DLL -Wl,DLL -o cleand cleanup.c setupd.x
c89 -g -Wc,DLL  -o linecount main.c setupd.x processd.x reportd.x cleand.x
```

14.2.2 Debugging a USS application

Next we give examples of debugging a USS application.

Setup

We are running the program in a different directory than we built it in, so we need to ensure that the PATH and LIBPATH are set up correctly. Our main and DLLs are in /u/redbk1/dllexmp/:

```
$ echo $PATH
/usr/lpp/java/IBM/J1.3/bin:/usr/lpp/Printsrv/bin:/bin:.
$ echo $LIBPATH
/lib:/usr/lib:/usr/lpp/Printsrv/lib:.
```

The current directory is in both paths, but our dllexmp directory is not in either of them. We must modify the PATH and LIBPATH as follows:

```
$ export PATH=/u/redbk1/dllexmp:$PATH
$ export LIBPATH=/u/redbk1/dllexmp:$LIBPATH
$ echo $PATH
/u/redbk1/dllexmp:/usr/lpp/java/IBM/J1.3/bin:/usr/lpp/Printsrv/bin:/bin:.
$ echo $LIBPATH
/u/redbk1/dllexmp:/lib:/usr/lib:/usr/lpp/Printsrv/lib:.
```

Now we can do a test run of the linecount program:

```
$ linecount bigtestfile.txt
Report on File bigtestfile.txt
Total number of records: 127169
```

Invocation

First we set up the TEST runtime option. From the C/C++ example, we see that the appropriate test option is:

```
NOTEST(ALL,*,NOPROMPT,TCPIP&9.30.62.149%8000:*)
```

We export our runtime option as follows:

```
export _CEE_RUNOPTS='NOTEST(ALL,*,NOPROMPT,TCPIP&9.30.62.149%8000:*)'
```

Now we start the remote debugger and run the program:

```
$ linecount bigtestfile.txt
```

As before, the remote debugger is triggered right at the ctest() call in our program.

15

Debug Tool and subsystems

In this chapter we describe how Debug Tool can help you debug your application while it runs in a host environment, such as CICS, IBM IMS, or IBM DB2 Universal Database (including IBM DB2 stored procedures) environments.

15.1 Debug Tool and CICS

This topic is divided into four parts:

- How to add support for debugging under CICS
- How to prepare a CICS program
- How to start Debug Tool under CICS
- How to debug CICS programs

15.1.1 How to add support for debugging under CICS

To debug applications that run in CICS, the system administrator must perform several steps. Some of them are mandatory, and others are only required in certain circumstances.

Basic CICS customization

The steps for basic CICS customization are as follows:

1. CICS definitions:

   Several Debug Tool resources have to be defined and installed into the CICS region. The CICS definitions are in EQACCSD and EQACDCT members of the hlq.SEQASAMP data set.

   EQACDCT contains six Destination Control Table macro definitions as part of Debug Tool under CICS. If your policy is to define CICS resources using CSD, you do not have to use these definitions. Note DCT support was dropped in CICS TS 2.2.

   EQACCSD contains the CICS definitions that are part of Debug Tool under CICS. You have to remove comments around the DEFINE TDQUEUE statements.

   Note: To use a COBOL side file, the Transient Data Queue CIGZ is required. This queue might also exist in Language Environment CICS resource definitions. In that case, you should be sure which one will be used.

   The group (EQA) that contains Debug Tool run time routines, as well as transactions and TD queues, must be in the group list used during CICS start-up.

   Note: If your policy is to autoinstall program definitions, you have to add comments around the DEFINE PROGRAMS statements.

2. JCL updates:

   Include Debug Tool's hlq.SEQAMOD data set in the DFHRPL concatenation.

   In the DFHRPL concatenation, include the Language Environment runtime libraries if not aready there (SCEECICS, SCEERUN, and, if required by your applications, SCEERUN2).

   Include EQA00DYN and EQA00HFS from Debug Tool's hlq.SEQAMOD data set in the STEPLIB concatenation in one of the following ways:

   – Use the Authorized Program Facility to authorize the hlq.SEQAMOD data set and add it to the STEPLIB concatenation.

   – Copy the EQA00DYN and EQA00HFS modules from the hlq.SEQAMOD data set to a library that is already to the STEPLIB concatenation.

   Ensure that no DD statements exist for CINSPIN, CINSPLS, CINSPOT, or IGZDBGIN.

3. For any terminal that Debug Tool uses to display a debugging session, verify that the CICS TYPETERM definition for that terminal specifies a minimum value of 4096 for the RECEIVESIZE parameter or sets the BUILDCHAIN parameter to YES.
4. Verify that users are able to run the CDT# transaction without receiving any errors.
5. If you are running your CICS programs in a distributed environment, the DTCN transaction name must be defined the same across all local and remote systems. If the DTCN transaction name is changed, or if a DTCN transaction is duplicated and given a different name, the name must be changed on all systems.
6. If you are planning to debug command-level assembler application programs that do not run under or use Language Environment services, you must activate the CICS Non-Language Environment Exits as described in the next section.

Activating CICS non-Language Environment exits

To debug non-Language Environment assembler programs that run under CICS, you must start the required Debug Tool global user exits before you start the programs. Debug Tool provides the following global user exits to help you debug non-Language Environment applications: XPCFTCH, XEIIN, XEIOUT, XPCTA, and XPCHAIR.

DTCX transaction

You can turn the exits on and off by using the transaction DTCX.

To activate all of the exits, from a clear CICS terminal screen, enter `DTCXXO`. You must activate the exits only once. After you enter `DTCXXO`, if all exits are activated successfully, the following message is displayed:

```
EQA9970I - CICS exit activation successful.
```

To deactivate all of the exits, enter `DTCXXF`. When you enter `DTCXXF`, the following message is displayed:

```
EQA9971I - CICS exit deactivation successful.
```

If there is a problem starting or activating one of the exits, an error message like the following one is displayed:

```
EQA9977E - OOSVC is backlevel. Exits not enabled.
EQA9974I Error enabling XPCFTCH - EQANCFTC
```

If you see this error message, verify that the CICS CSD is properly updated to include the latest Debug Tool resource definitions, and that Debug Tool SEQAMOD data is in the DFHRPL DD concatenation for the CICS region. You can start the exits during region initialization by using a sequential terminal. You are not required to shut down the exits prior to or during a region shutdown.

PLT solution

The exits can also be activated automatically during CICS region startup by using Debug Tool-supplied CICS Program List Table (PLT) post-initialization program EQANCPLT. This program must run in the second or third phase of the CICS PLT process. See the sample provided in the hlq.SEQASAMP member EQACXITP for more details.

Sharing the DTCN repository profile items among CICS systems

The DTCN debug profile repository is a CICS temporary storage queue called EQADTCN2. If you must share the repository among CICS systems, you can use either of the following methods:

- Designate a single CICS region as the Queue-Owning Region (QOR) and define the queue as REMOTE in a TSMODEL resource definition on regions that must access it remotely. This makes the queue profile items owned by one CICS system accessible to other CICS systems, as shown in Figure 15-1.

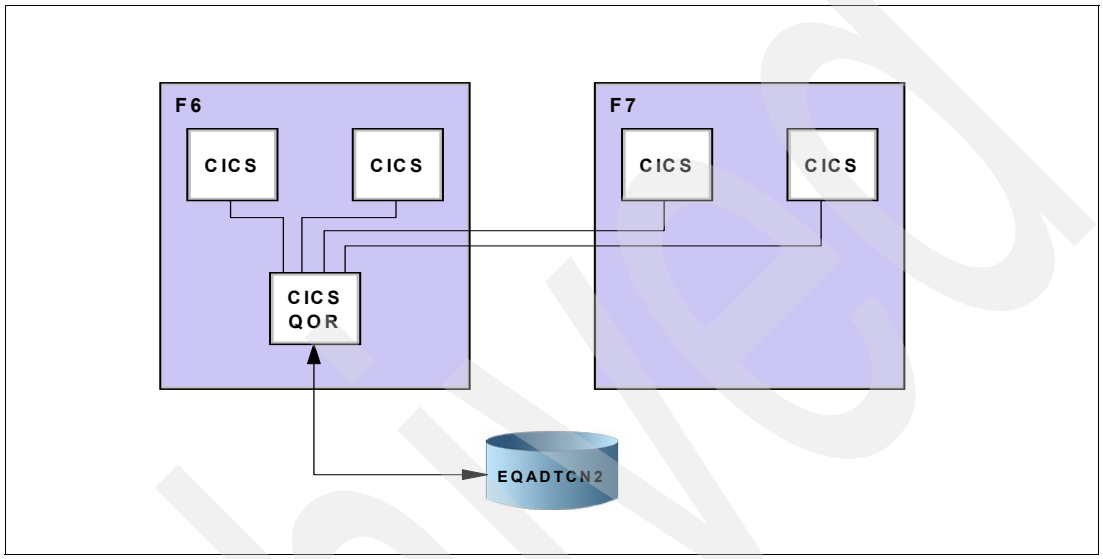


Figure 15-1 Sharing DTCN profile using remote TSQ

Example 15-1 is a sample resource definition that shows how to define Debug Tool EQADTCN2 temporary storage queue in a region that will use it remotely.

Example 15-1 TSmodel remote sample definition

```
VIEW TSMODEL(DTCN1) GR(DTCNREM)
 OBJECT CHARACTERISTICS                                  CICS RELEASE = 0630
  CEDA  View TSmodel( DTCN1    )
   TSmodel         : DTCN1
   Group           : DTCNREM
   Description     : TEST DTCN TSQ REMOTE
   PRefix          : EQADTCN2
   XPrefix         :
   Location        : Main               Auxiliary | Main
  RECOVERY ATTRIBUTES
   RECovery        : No                 No | Yes
  SECURITY ATTRIBUTES
   Security        : No                 No | Yes
  SHARED ATTRIBUTES
   POolname        :
  REMOTE ATTRIBUTES
   REMOTESystem    : C23F
   REMOTEPrefix    : EQADTCN2
   XRemotepfx      :
```

Note: To optimize the performance of Debug Tool, you can choose to define this queue as Location MAIN.

- If you have a Parallel Sysplex® environment, you can update the Coupling Facility Resource Manager (CFRM) policy to define a new structure named DFHXQLS_*poolname* and start a CICS TS data sharing server to share TS queues across your CICSplex, as shown Figure 15-2.

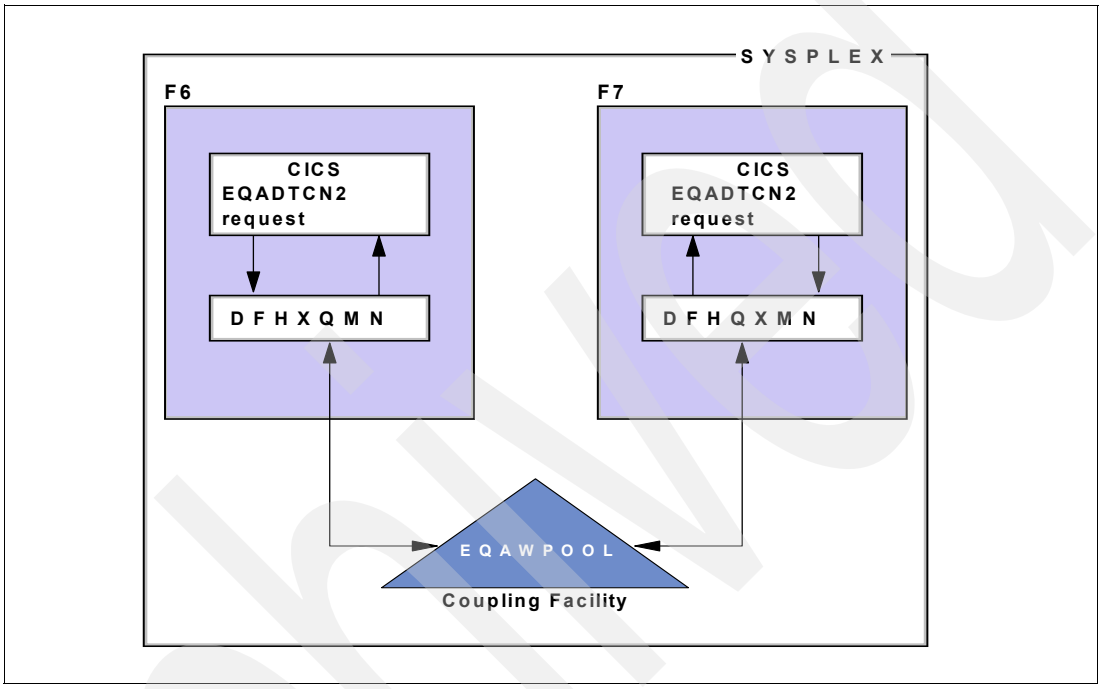


Figure 15-2 Sharing DTCN profile using TSQ in a coupling facility

Example 15-2 shows how to define the DFHXQLS_*poolname* structure.

Example 15-2 DFHXQLS_poolname definition

```
STRUCTURE
   NAME (DFHXQLS_EQAWPOOL)
   SIZE(50000)
   INITSIZE(20000)
   PREFLIST(FACIL01,FACIL02)
```

Example 15-3 shows how to write the job to manage the DFHQXLS_*poolname*.

Example 15-3 TS data sharing server JCL

```
//CTSTSHF6 JOB 1,RACFUSER,TIME=1440,NOTIFY=&SYSUID,REGION=4M,
//            CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//*    Start CICS TS data sharing server
//TSSERVER EXEC PGM=DFHXQMN,REGION=64M
//STEPLIB DD DSN=CICSVS.TS310.CICS.SDFHAUTH,DISP=SHR
//*    Authorized library containing DFHXQMN
//SYSPRINT DD SYSOUT=*
//*    Options
//SYSIN DD    *
```

```
FUNCTION=SERVER
POOLNAME=EQAWPOOL  Pool name
MAXQUEUES=1000     Allow up to 1003 List Structures
```

Example 15-4 shows how to define Debug Tool EQADTCN2 temporary storage queue in a region that will use it shared.

Example 15-4 TSmodel shared

```
CEDA View TSmodel(DTCN1)
TSmodel      ==> DTCN1
Group        ==> DTCNREM
Description  ==> TEST DTCN TSQ REMOTE
PRefix       ==> EQADTCN2
XPrefix      ==>
Location     ==> Main          Auxiliary | Main
RECOVERY ATTRIBUTES
RECovery     ==> No            No | Yes
SECURITY ATTRIBUTES
Security     ==> No            No | Yes
SHARED ATTRIBUTES
POolname     ==> EQAWPOOL
REMOTE ATTRIBUTES
REMOTESystem ==>
REMOTEPrefix ==>
   XRemotepfx   ==>
```

These settings store a profile item in one CICS system, or one TS server region, and make it readable to other CICS systems.

Enabling communication between Debug Tool and a remote debugger

If you use a remote debugger, Debug Tool communicates with the remote debugger by using TCP/IP Sockets.

If you have CICS Transaction Server for z/OS Version 2 Release 3 or later, Debug Tool uses the CICS Sockets Domain. To start the Sockets Domain, ensure that the CICS system initialization parameter TCPIP is set to YES. You also have to define a TCPIPSERVICE resource.

Example 15-5 shows a sample resource definition.

Example 15-5 TCpiservervice definition for the remote debugger

```
CEDA  ALter TCpipservice( REMDEBUG )
 TCpipservice   : REMDEBUG
 GROup          : DEBUGTOL
 DEscription  ==> REMOTE DEBUG TOOL TCPIP PORT
 Urm          ==> DFHWBADX
 POrtnumber   ==> 08001              1-65535
 STatus       ==> Open               Open ! Closed
 PROtocol     ==> Http               Iiop ! Http ! Eci ! User
 TRansaction  ==> CWXN
 Backlog      ==> 00005              0-32767
 TSqprefix    ==>
 Ipaddress    ==>
 SOcketclose  ==> No                 No ! 0-240000 (HHMMSS)
 Maxdatalen   ==> 006000             3-536870
SECURITY
 SSl          ==> No                 Yes ! No ! Clientauth
 CErtificate  ==>
 (Mixed Case)
 PRIvacy      ==>                    Notsupported ! Required ! Supported
 CIphers      ==>
 AUthenticate ==> No                 No ! Basic ! Certificate ! AUTORegister
                                     ! AUTOMatic ! ASserted
 ATtachsec    ==>                    Local ! Verify
DNS CONNECTION BALANCING
 DNsgroup     ==>
 GRPcritical  ==> No                 No ! Yes
```

If you have an earlier version of CICS, Debug Tool uses the TCP/IP Socket Interface for the CICS feature of TCP/IP for MVS.

You can use EZAC, a TCP/IP-supplied CICS transaction, to inquire about the status of TCP/IP connections in the CICS region.

EZAC,DISPLAY,LISTENER provides information about the TCP/IP listener task that has to be running.

EZAC,DISPLAY,CICS displays information about the CICS-TCP/IP configuration.

Enabling the CADP transaction

If you have CICS Transaction Server for z/OS Version 2 Release 3, and you want to use the CADP transaction to manage debugging profiles, set the DEBUGTOOL system initialization parameter to YES.

The default setting of DEBUGTOOL=NO results in the DTCN transaction being used for Debug Tool profiles.

You can still use CADP to define profiles with DEBUGTOOL=NO, but they will not be used by Debug Tool.

You can dynamically switch between the CADP and DTCN debug profiles that are used by Debug Tool. Use the following commands to switch between the profiles after the CICS region is started:

- CEMT SET DEBUG - use CADP profiles
- CEMT SET NODEBUG - use DTCN profiles

Enabling the CADP Web interface

If you have CICS Transaction Server for z/OS Version 2 Release 3 or later, Debug Tool uses the CICS Sockets Domain. To start the Sockets Domain, ensure that the CICS system initialization parameter TCPIP is set to YES. You also have to define a TCPIPSERVICE resource. Example 15-6 shows how to define it.

Example 15-6 TCpipservice definition for the CADP Web interface

```
OBJECT CHARACTERISTICS                                    CICS RELEASE = 0630
 CEDA  View TCpipservice( DEBUGWEB )
  TCpipservice   : DEBUGWEB
  GROup          : DEBUG
  DEscription    : TCPIP SERVICE FOR CADP Web interface
  Urm            : DFHWBADX
  POrtnumber     : 05000              1-65535
  STatus         : Open               Open | Closed
  PROtocol       : Http               Iiop | Http | Eci
  TRansaction    : CWXN
  Backlog        : 00005              0-32767
  TSqprefix      :
  Ipaddress      :
  SOcketclose    : No                 No | 0-240000 (HHMMSS)
 SECURITY
  SSl            : No                 Yes | No | Clientauth
  Certificate    :
  (Mixed Case)
  PRIvacy        :                    Notsupported | Required | Supported
  AUthenticate   : No                 No | Basic | Certificate | AUTORegister
                                      | AUTOMatic | ASserted
  ATtachsec      :                    Local | Verify
 DNS CONNECTION BALANCING
  DNsgroup       :
  GRPcritical    : No                 No | Yes

                                                 SYSID=ADT2 APPLID=A6T1ADT2
```

Start the application debugging profile manager by typing its URL in your Web browser. The URL that you enter depends upon how CICS Web support is configured. For example, if your Web browser connects directly to CICS, and your system is configured to use the default analyzer program (DFHWBADX), the URL is:

```
http://mvs_address:5000/CICS/CWBA/dfhdpwb
```

Your system administrator will tell you the URL to use for your system.

Sharing the CADP repository profile items among CICS systems

The CADP debug profile repository is VSAM files called DFHDPFMB and DFHDPFMP. If you want to share the repository among CICS systems, use either of the following methods:

- Designate a single CICS region as the File-Owning Region (FOR) and define the file as REMOTE in a FILE resource definition on regions that must access it remotely. This makes the file profile items owned by one CICS system accessible to other CICS systems, as shown in Figure 15-3.

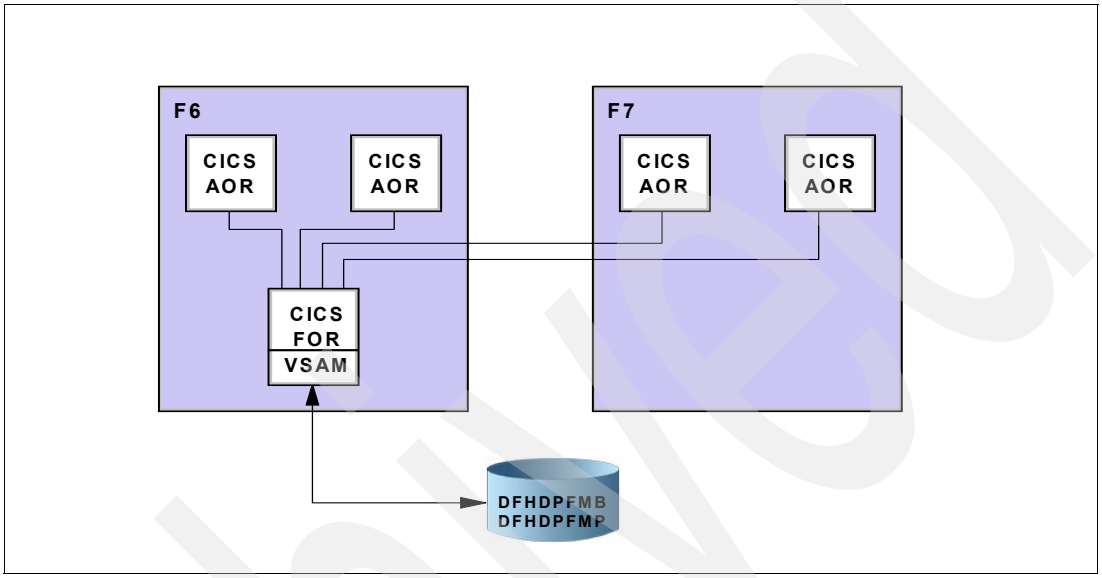


Figure 15-3 Sharing the CADP debug profile repository as a remote file

Example 15-7 is a sample resource definition that shows how to define Debug Tool DFHDPFMB and DFHDPFMP files in a region that will use them remotely.

Example 15-7 DFHDPFMB - remote definition

```
CEDA  ALter File( DFHDPFMB )
 File          : DFHDPFMB
 Group         : ERCEQAW
 DEScription  ==>
VSAM PARAMETERS
 DSNAme        ==>
 Password      ==>                  PASSWORD NOT SPECIFIED
 RLsaccess     ==> No               Yes ! No
 LSrpoolid     ==> 1                1-8 ! None
 READInteg     ==> Uncommitted      Uncommitted ! Consistent ! Repeatable
 DSNSharing    ==> Allreqs          Allreqs ! Modifyreqs
 STRings       ==> 001              1-255
 Nsrgroup      ==>
REMOTE ATTRIBUTES
 REMOTESystem ==> C31F
 REMOTEName   ==> DFHDPFMB
REMOTE AND CFDATATABLE PARAMETERS
 RECORDSize   ==>                   1-32767

                                                    SYSID=C31G APPLID=CICSC31G
```

The same definition has to be created for DFHDPFMP.

- The two files can be shared across a sysplex using a coupling facility if the VSAM Record Level Sharing is available on your sysplex. The following information is required:
  - VSAM server - SMSVSAM
  - SHaring Control Data Set (SHCDS)
  - Common buffers and control blocks management:
    - SMSVSAM data space
    - Coupling facility data caches structure
    - Coupling facility lock structure named IGWLOCK00

  Figure 15-4 illustrates the basic structure of this configuration.

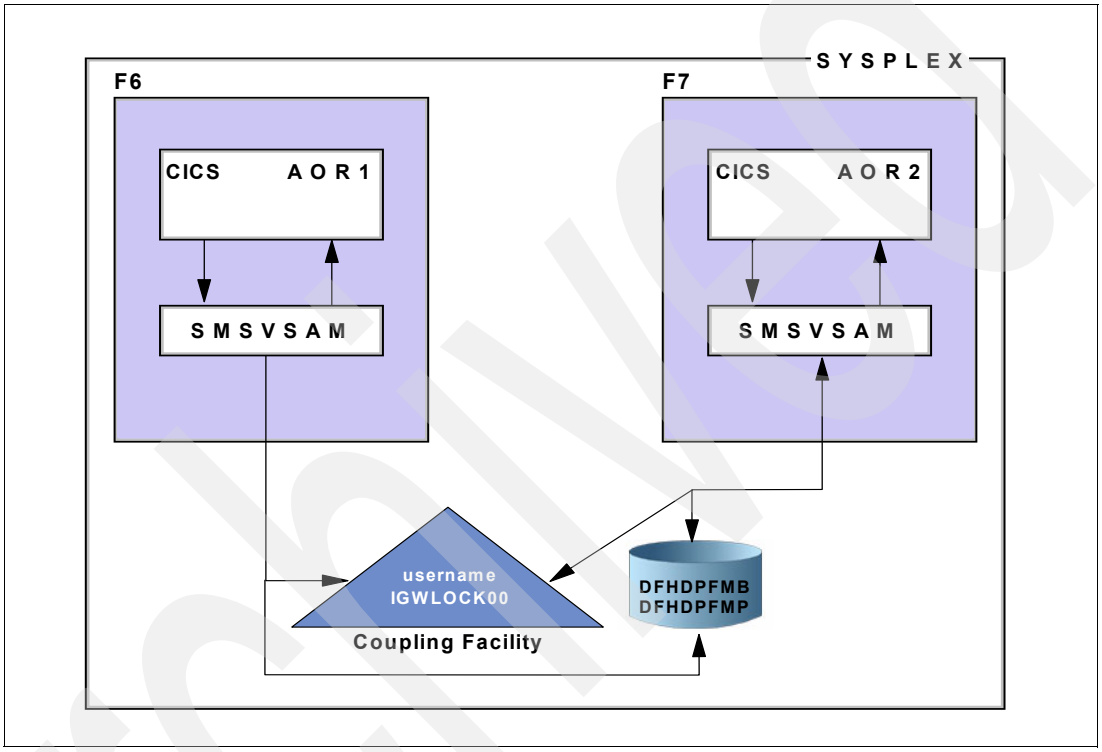


Figure 15-4 Sharing the CADP debug profile repository in a parallel sysplex

15.1.2 How to prepare a CICS program

To prepare a CICS program for debugging, you must do the following tasks:

1. Complete the program preparation tasks for assembler, C, C++, COBOL, or PL/I.
2. If you are using debugging profiles to indicate which programs to debug, choose one of the following CICS transactions to manage your debugging profiles:
   - If you are using versions of CICS earlier than CICS Transaction Server for z/OS Version 2 Release 3, use DTCN to manage debugging profiles.

     Link-edit EQADCCXT into the CICS program that you want to debug. This step is required only for PL/I and C/C++ programs.
   - If you are using CICS Transaction Server for z/OS Version 2 Release 3 or later, you can use either CADP or DTCN to manage debugging profiles.

     If you use CADP, you do not have to link-edit EQADCCXT into the CICS program that you want to debug. You cannot use both CADP and DTCN, you must choose only one.

3. Create and store a profile that specifies the combination of resource IDs that you want to debug.
4. Run your program.

Determine whether you must link-edit EQADCCXT into your program

Debug Tool provides a Language Environment CEEBXITA assembler exit called EQADCCXT to help you activate, by using the DTCN transaction, a debugging session under CICS.

You do not have to use this exit if you are running either of the following options:

- You are running under CICS Transaction Server for z/OS Version 2 Release 3 or later and you use the CADP transaction to define debug profiles.
- You are using the DTCN transaction and you are debugging COBOL programs.

If you activate a debug session by using DTCN and you are debugging a C, C++, or PL/I programs, you must link the EQADCCXT exit into the main load module of the program. When you use EQADCCXT, be aware of the following conditions:

- If your site does not use a Language Environment assembler exit (CEEBXITA), then link-edit member EQADCCXT, which contains the CSECT CEEBXITA and is in library hlq.SEQAMOD, into your main program.
- If your site uses an existing CEEBXITA, the EQADCCXT exit provided by Debug Tool must be merged with it. The source for EQADCCXT is in hlq.SEQASAMP(EQADCCXT). Link the merged exit into your main program.

After you link-edit your program, use the DTCN transaction to create a profile that specifies the combination of resource IDs that you want to debug.

How to create and store a DTCN profile

The DTCN transaction stores one profile for each DTCN terminal in a repository. Each profile is retained in the repository until one of the following events occurs:

- The profile is explicitly deleted by the terminal that entered it.
- DTCN detects that the terminal that created the profile has been disconnected.
- The CICS region is terminated, except if you share EQADTCN2 through the coupling facility.

To create and store a DTCN profile, perform the following steps:

1. Log on to a CICS terminal and enter the transaction ID `DTCN`. The DTCN transaction displays the main DTCN screen, shown in Figure 15-5.

```
DTCN              Debug Tool CICS Control - Primary Menu            QXPE1ASX

Select the combination of resources to debug (see Help for more information)

 Terminal Id      ==> 1339
 Transaction Id   ==>
 Program Id(s)    ==>          ==>          ==>          ==>
                  ==>          ==>          ==>          ==>
 User Id          ==> SIMCOCK
 NetName          ==>

Select type and ID of debug display device

 Session Type     ==> MFI               MFI, TCP
 Port Number      ==>                   TCP Port
 Display Id       ==> 1339
 Connection Type  ==> SINGLE            Single, Multiple

Generated String:  TEST(ALL,'*',PROMPT,'MFI%1339:*')

Repository String: No string currently saved in repository


PF1=HELP 2=GHELP 3=EXIT 4=SAVE 6=DELETE 7=SHOW 9=OPTIONS
```

Figure 15-5 DTCN - Primary Menu

2. Specify the combination of resource IDs that you want to debug:

   Terminal Id Specify the CICS terminal to debug. By default, this ID is set to the terminal that is currently running DTCN.

   Transaction Id Specify the CICS transaction to debug. If you specify a transaction ID without any other resource, Debug Tool is started every time that transaction is run, including times when other users run the transaction.

   Program Id(s) Specify the CICS program or programs to debug. If you specify a program ID without any other resource, Debug Tool is started every time the program is run, including times when other users run the program.

   Specifying a CICS program in the Program Id(s) field is similar to setting a breakpoint by using the AT ENTRY command.

   If Debug Tool is already running and it cannot find the separate debug file, then Debug Tool does not stop at the CICS program specified in the Program Id(s) field. Use the AT APPEARANCE or AT ENTRY command to stop at this CICS program.

   User Id Specify the CICS user ID to debug. All programs that are run by this user will start Debug Tool.

   NetName Specify the NETNAME of a CICS terminal. All programs that are running by this Netname will start Debug Tool.

3. Specify the type of debugging and the ID of the display device:

   Session Type Select one of the following options:

   - MFI indicates that Debug Tool initializes on a 3270 type of terminal.

- TCP indicates that you want to interact with Debug Tool from your workstation using TCP/IP and a remote debugger.

Port Number Specifies the TCP/IP port number that is listening for debug sessions on your workstation. By default, IBM Distributed Debugger uses port 8000. The Compiled Language Debugger component of WebSphere Developer for zSeries uses port 8001.

Display Id Identifies the target destination for Debug Tool information. Depending on the session type that you selected, the display ID is one of the following:

- If you selected MFI, the display ID is a CICS 3270 terminal ID. This ID is set by default to the terminal ID that is currently running DTCN, but you can change this to direct full-screen 3270 screens to a different CICS terminal.
- If you selected TCP, enter either the IP address or host name of the workstation that will display the debug screens. For the debug session to start, the appropriate software must be running on that workstation.

Connection Type Specifies whether you are using a single or multiple socket connection to the workstation.

Note: If you are using Distributed Debugger Version 9.2 or later, or the Compiled Language Debugger component of either WebSphere Developer for zSeries or WebSphere Developer Debugger for zSeries, we recommend that you specify a single socket connection.

If you are using a version of the Distributed Debugger that is earlier than Version 9.2 (copyright date 2003/10/19), we recommend that you specify a multiple socket connection.

4. Specify the debugging options by pressing PF9 to display the secondary options menu, shown in Figure 15-6.

```
DTCN              Debug Tool CICS Control - Menu 2                QXPE1ASX

Select Debug Tool options

Test Option     ==> TEST                Test/Notest
Test Level      ==> ALL                 All/Error/None
Commands File   ==> *
Prompt Level    ==> PROMPT
Preference File ==> *

Any other valid Language Environment options
==>









PF1=HELP 2=GHELP 3=RETURN
```

Figure 15-6 DTCN - menu 2

The options are:

Test Option TEST/NOTEST specifies the conditions under which Debug Tool assumes control during the initialization of your application.

Test Level ALL/ERROR/NONE specifies what conditions must be met for Debug Tool to gain control.

Command File A valid fully qualified data set name that specifies the primary commands file for this run. Do not enclose the name of the data set in single or double quotation marks.

Prompt Level Specifies whether Debug Tool is started at Language Environment initialization.

Preference File A valid fully qualified data set name that specifies the preference file to be used. Do not enclose the name of the data set in single or double quotation marks.

Any other valid Language Environment option

You can change any Language Environment option that your site has defined as over-rideable except the STACK option.

5. Press PF3 to return to the main DTCN panel.
6. Press PF4 to save the profile. DTCN performs data verification on the data that you entered in the DTCN panel. When DTCN discovers an error, it places the cursor in the erroneous field and displays a message. You can use context-sensitive help (PF1) to find what is wrong with the input.

Now any tasks that run in the CICS system and match the resource IDs that you specified in the previous steps will start Debug Tool.

To display all of the active DTCN profiles in the CICS region, press PF7. Debug Tool CICS Control - All Sessions screen is displayed, as shown in Figure 15-7.

```
DTCN              Debug Tool CICS Control - All Sessions            QXPE1ASX
Overtype "_" with a "D" to delete a profile.
  Owner   Term  Tran  User      Netname   Applid   Options String
_ 1339    1339  ____  SIMCOCK   ________  QXPE1ASX TEST(ALL,'*',PROMPT,'MFI%1339
          Program(s)  ________  ________  ________  ________
                      ________  ________  ________  ________

PF1=HELP 2=GHELP 3=RETURN 7=BACK 8=FORWARD
```

Figure 15-7 DTCN - All Sessions

The column titles are defined as follows:

Owner The ID of the terminal that created the profile by using DTCN.

Term The value that was entered on the main DTCN screen in the Terminal Id field.

Tran The value that was entered on the main DTCN screen in the Transaction Id field.

User The value that was entered on the main DTCN screen in the User Id field.

Netname The value that was entered on the main DTCN screen in the Netname field.

Applid The application identifier associated with this profile.

Options String The value of the Repository String field on the main DTCN screen. It is created based on the values that the user enters in the other fields.

Program(s) The values that were entered on the main DTCN screen in the Program Ids field.

DTCN also reads the Language Environment NOTEST option supplied to the CICS region in CEECOPT or CEEROPT. You can supply suboptions, such as the name of a preference file, with the NOTEST option to supply additional defaults to DTCN.

How to create and store a CADP profile

First you have to start the CICS Application Debugging Profile (CADP) manager 3270 interface by running the CADP transaction. CICS displays the panel shown in Figure 15-8.

```
CADP      -    CICS Application Debugging Profile Manager   -     CICSC31F

List Debugging Profiles    (A=Activate,I=Inactivate,D=Delete,C=Copy)

  Owner    Profile  S Tran Program  Compile Unit Applid   Userid   Term Type
_ VIKRAMM  ALLTXS   I *    *        *            CICSC31F VIKRAMM  *    Comp
_ SIMCOCK  ANS      A *    MYTRADS  *            CICSC31F SIMCOCK  *    Comp
_ WACARTE  BC       I NSPE *        *            CICSC31F WACARTE  0019 Comp
_ ZHONG    BLUEFOX  A OS*  *        *            CICSC31F ZHONG    *    Comp
_ $EXAMPLE CORBA    I T*                         *        IORWERTH      Corb
_ $EXAMPLE EJB      I *                          *        *             EJB
_ CHABERT  ERC1#TRN I ERC1 *        *            CICSC31F *        *    Comp
_ CHABERT  ERC2#PGM I *    ERC2PGM* *            CICSC31F CHABERT  *    Comp
_ $EXAMPLE JAVA     I TR*                        *        PENFOLD*     Java
_ CHABERT  JAVA     I TR*                        *        PENFOLD*     Java
_ KPHUME   KPHTEST1 I KPH1 *        *            CICSC31C *        *    Comp
_ $EXAMPLE LE1      I T*   P*       *            CICSREG1 PANDREWS TTT1 Comp
_ $EXAMPLE LE2      I TR   *        SAMPCOMPUN + CICSREG2 DRBEARD* TTT2 Comp
_ $EXAMPLE LE3      I TRN3 PROG3    *            CICSREG3 *        TTT2 Comp
_ NICK     NIX      I *    CUSC*    *            CICSC31F *        *    Comp
_ TIMOTHY  TIMP     I PS*  *        *            CICSC31F TIMOTHY  *    Comp
20  profile(s). All profiles shown
Enter=Process PF1=Help 2=Filter 3=Exit 4=View 5=Create Comp 6=Create Java
      8=Forward 9=Set display device 10=Edit 11=Sort
```

Figure 15-8 CADP Main panel

The List Debugging Profiles screen

When you use the debugging profile manager for the first time, CICS displays all the profiles that you own. Subsequently, CICS displays profiles that were selected when you last used it.

The list contains selected information from the debugging profiles. Columns listed are:

Owner	The user ID of the profile owner (that is, of the user who created the profile).
Profile	The name of the profile.
S	The status of the profile (A for active or I for inactive).

The following columns display information specified when the profile is created:

Tran	Displays the contents of the transaction field.
Program	Displays the contents of the program field.
Compile Unit	Displays the first ten characters of the Compile Unit field. If the Compile Unit name is longer, a plus sign (+) to the right of the name shows that only part of the name is displayed.
Applid	Displays the contents of the Applid field.
Userid	Displays the contents of the Userid field.
Term	Displays the contents of the Terminal field.
Type	Displays the type of program specified in the debugging profile: **Corb** - CORBA object **EJB™** - Enterprise bean **Java** - Java program **LE** - Compiled language program

Not all of the information in the debugging profile is displayed on this screen. To display the additional information, move the cursor to the line that contains the profile, and press PF4.

You can change the way CICS displays information about the List Debugging Profiles screen by the following criteria:

- Selecting which profiles are displayed. Use PF2 to cycle through the available options in turn. The options are:
  - Display all the profiles in the system. This is the setting the first time you use the debugging profile manager.
  - Display the profiles that you own.
  - Display all active profiles.
- Sorting the list. Use PF11 to cycle through the available options in turn. The options are:
  - Re-display the profiles in sequence of profile name. This is the sequence the first time you use the debugging profile manager.
  - Re-display the profiles in sequence of transaction ID.
  - Re-display the profiles in sequence of program name.
  - Re-display the profiles in sequence of owner.

In each case, CICS uses the EBCDIC sorting sequence.

Your choice of which profiles are displayed, and your chosen sequence, is preserved and used the next time you use the debugging profile manager.

Creating a new profile for a compiled language program

To create a new profile for a compiled language program, complete the following steps:

1. From the List Debugging Profiles screen, press PF5. CICS displays the Create Compiled Debugging Profile screen, as shown in Figure 15-9.

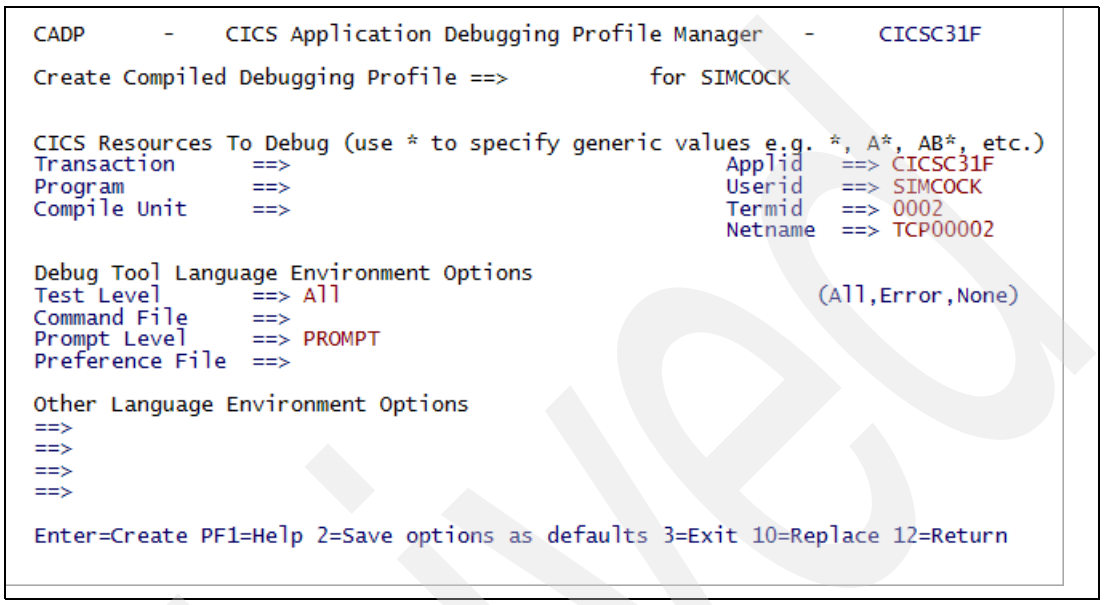


Figure 15-9 CADP new LE profile

The fields on the screen are:

Create Compiled Debugging Profile

Specify the name of the profile. If you change the name of an existing profile, CADP creates a new profile with the new name and leaves the original profile unchanged.

Transaction Specify a value in this field when you want to debug only those programs that run under the specified transaction. You can specify a generic value.

Program Specify a value in this field when you want to debug only the specified program. You can specify a generic value.

Compile unit Specify a value in this field when you want to debug only the specified compile unit. You can specify a generic value.

Applid Specify a value in this field when you want to confine debugging to programs that run in the specified CICS region. You can specify a generic value.

Userid Specify a value in this field when you want to confine debugging to programs that are being run by the specified user. You can specify a generic value.

Termid Specify a value in this field when you want to confine debugging to programs that are being run by the specified terminal. You can specify a generic value.

Netname Specify a value in this field when you want to confine debugging to programs that are being run by the specified netname. You can specify a generic value.

Test level Specifies which conditions raised by your program will cause Debug Tool to gain control. You can enter the following values: All, Error, or None.

Command file Specifies the primary commands file associated with the profile. You can specify the fully qualified name of a sequential dataset or a member of a partitioned data set.

Prompt level Specifies whether an initial command list is unconditionally executed during program initialization. Enter one of the following: PROMPT, NOPROMPT, or command.

Preference file Specifies the preference file that Debug Tool uses when debugging programs that match this profile. You can specify the fully qualified name of a sequential dataset or a member of a partitioned data set.

Other Language Environment options

Specifies Language Environment runtime options for programs that match this profile. When a program is selected for debugging because it matches the profile, the runtime options specified will override other runtime options that you might have in effect.

2. Press Enter. CICS checks that you have entered valid data.
3. Press PF12 to return to the List Debugging Profiles screen.

Specifying default values for Debug Tool and LE options

You can specify default values for the following Debug Tool options, and the Language Environment options. The saved values are used by default each time you create a debugging profile for a compiled language program. Debug Tool options are:

- Test level
- Command file
- Prompt level
- Preference file

To save the default values, start with the Create LE Debugging Profile screen, and follow these steps:

1. Type the default values that you want to specify for Debug Tool options and Language Environment options.
2. Press PF2. CICS saves the values that you have specified.

The values that you save are used by default each time you create a new profile.

Creating a new profile for a Java program

To create a new profile for a Java program, complete the following steps:

1. From the List Debugging Profiles screen, press PF6. CICS displays the Create Java Debugging Profile screen, as shown in Figure 15-10 and Figure 15-11.

```
CADP      -    CICS Application Debugging Profile Manager   -     CICSC31F

Create Java Debugging Profile ==>            for SIMCOCK

CICS Resources To Debug (use * to specify generic values e.g. *, A*, AB*, etc.)
Transaction      ==>                          Applid       ==> CICSC31F
                                              Userid       ==> SIMCOCK

Debugging Options
JVM Profile      ==>

Java Resources To Debug
Type             ==> J       (J=Java Applications, E=Enterprise Beans, C=Corba)

Class (Java Applications or Corba)
==>
==>
==>
==>

Press PF8 to set Bean and Method

Enter=Create PF1=Help 2=Save options as defaults 3=Exit 8=Forward
10=Replace 12=Return
```

Figure 15-10 CADP New Java profile (part 1 of 2)

```
CADP      -    CICS Application Debugging Profile Manager   -     CICSC31F

Java Resources To Debug

Bean (Enterprise Beans only)
==>
==>
==>
==>

Method (Enterprise Beans or Corba)
==>
==>
==>
==>




Enter=Create PF1=Help 3=Exit 7=Back 10=Replace 12=Return
```

Figure 15-11 CADP New Java profile (part 2 of 2)

The fields on the screen are:

Create Java Debugging Profile

Specify the name of the profile. If you change the name of an existing profile, CADP creates a new profile with the new name and leaves the original profile unchanged.

Transaction Specify a value in this field when you want to debug only those programs that run under the specified transaction. You can specify a generic value.

Applid Specify a value in this field when you want to confine debugging to programs that run in the specified CICS region. You can specify a generic value.

Userid Specify a value in this field when you want to confine debugging to programs that are being run by the specified user. You can specify a generic value.

JVM™ Profile Specify the name of the JVM profile that is used for Java programs that match this profile. The profile should specify that the Java program is to run in debug mode.

The following fields specify which Java resources should trigger the start of a debugging session when the profile is active:

– **Type**

Specifies the type of Java resource that you want to debug:

- J: Enter this value when you want to debug a Java program.
- E: Enter this value when you want to debug an enterprise bean.
- C: Enter this value when you want to debug a stateless CORBA object.

– **Class**

For Java programs and stateless CORBA objects only, specify a value in this field when you want to debug only the specified class. You can specify a generic value.

– **Bean**

For enterprise beans only, specify a value in this field when you want to debug only the specified bean. You can specify a generic value.

– **Method**

For enterprise beans and CORBA objects only, specify a value in this field when you want to debug only the specified method. When an inbound request initiated by a Java remote method invocation is received, the value specified is compared with the mangled name in the inbound request to determine whether the profile matches the request. If it is possible that mangling can take place; do not specify a method name in the debugging profile, but specify a generic method instead.

2. Complete the fields required to specify your profile.
3. Press Enter. CICS checks that you have entered valid data:
   – If you have specified valid data, the profile is saved.
   – If your data contains an error, CICS displays a message. Re-enter the data and press Enter again.
4. Press PF12 to return to the List Debugging Profiles screen.

How to create and store a profile using the CADP Web interface

When you start the application debugging profile manager's Web interface, CICS displays the List profiles page shown in Figure 15-12.

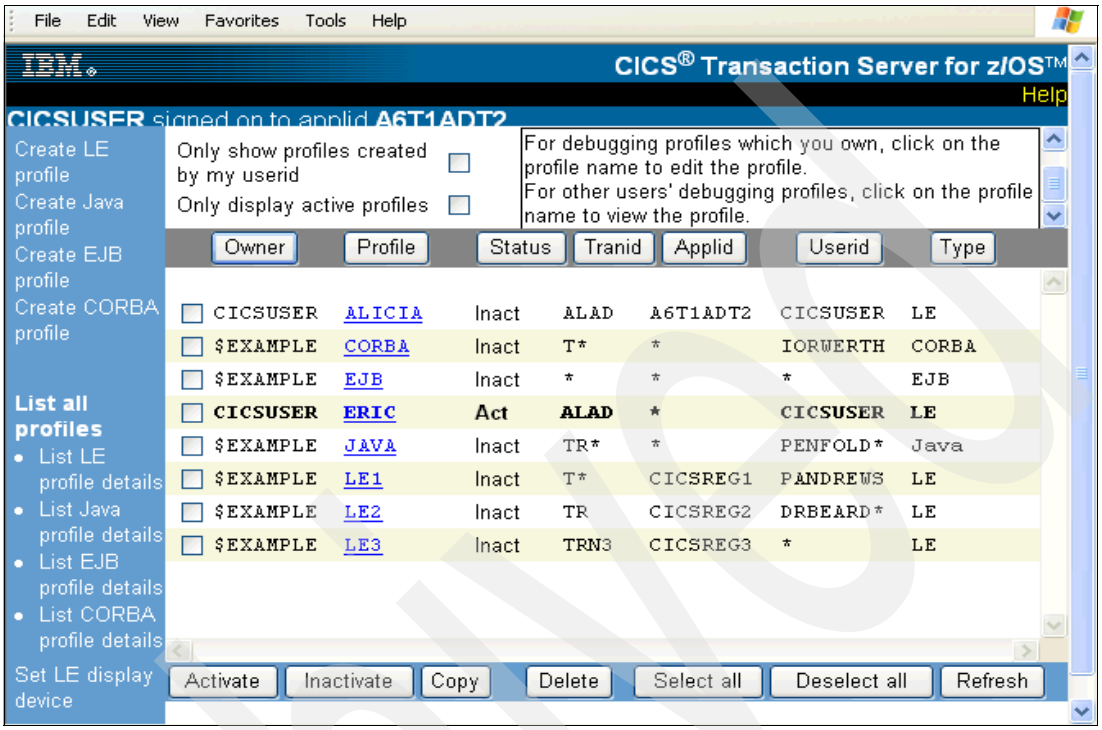


Figure 15-12 CADP Web interface - The List profiles page

When you use the debugging profile manager for the first time, CICS displays all profiles. Subsequently, CICS displays the profiles that were selected when you last used it.

If there are more profiles than can be displayed in the window, use the scrollbars to scroll backwards and forwards through the list. If you have no profiles, CICS displays an empty list.

There are four variants of the List profiles page, as shown in the following figures:

List LE profiles Lists only the compiled language profiles, as shown in Figure 15-13.

List Java profiles Lists only the profiles for Java programs, as shown in Figure 15-14.

List EJB profiles Lists only the profiles for enterprise beans, as shown in Figure 15-15.

List CORBA profiles Lists only the profiles for stateless CORBA objects, a shown in Figure 15-16.

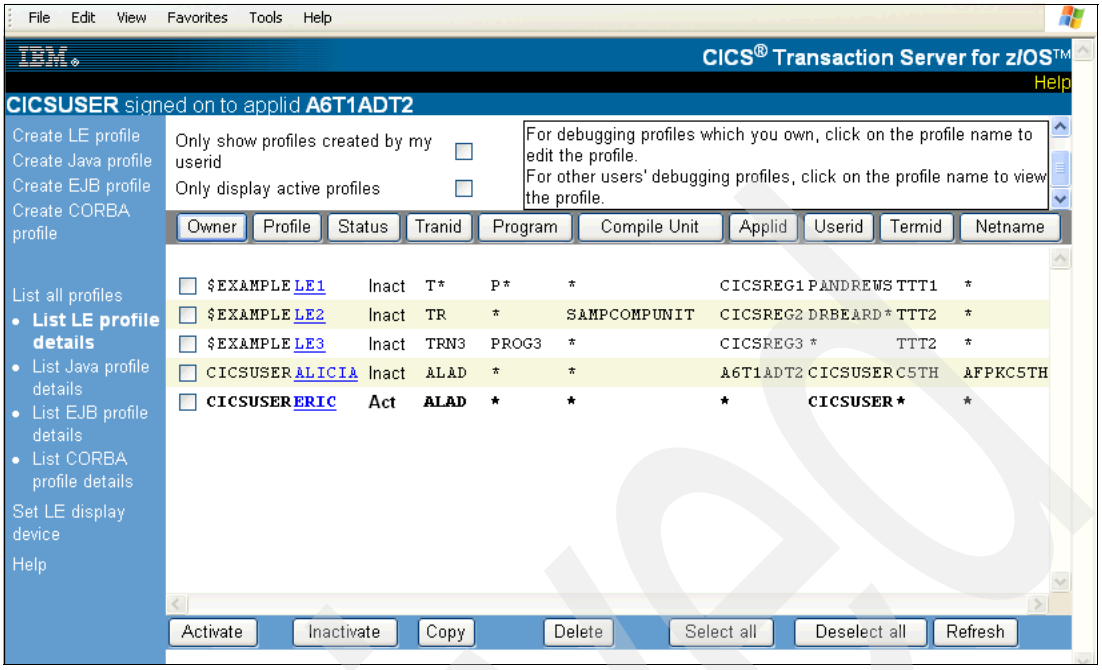


Figure 15-13 CADP - Web interface - LE profiles

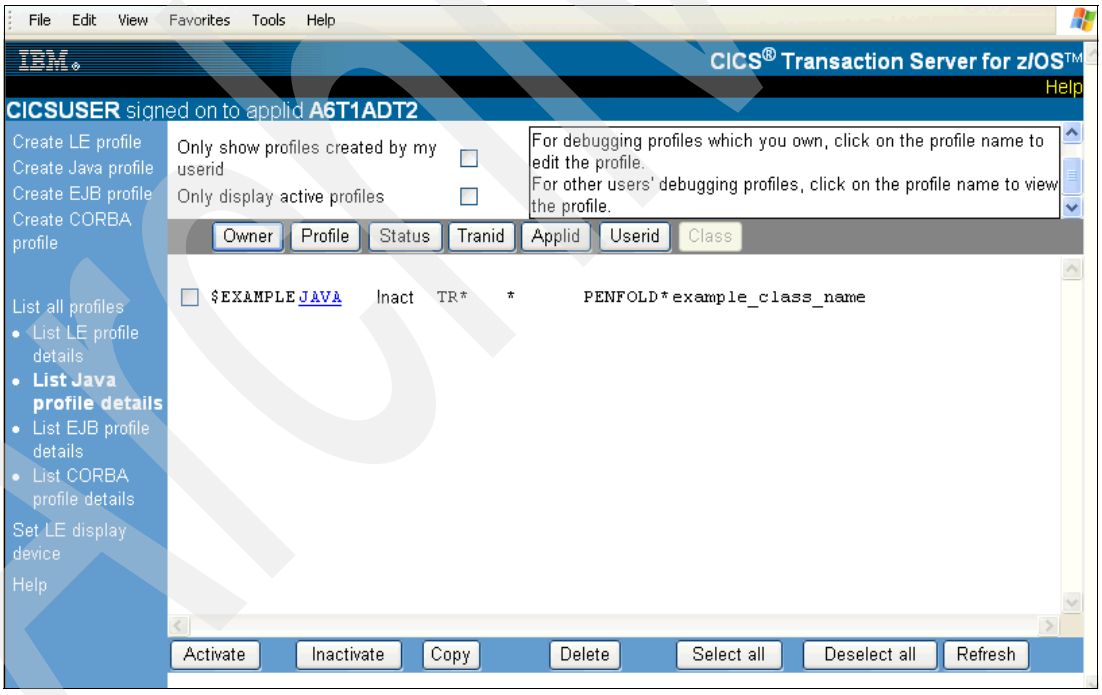


Figure 15-14 CADP Web Interface - Java profiles

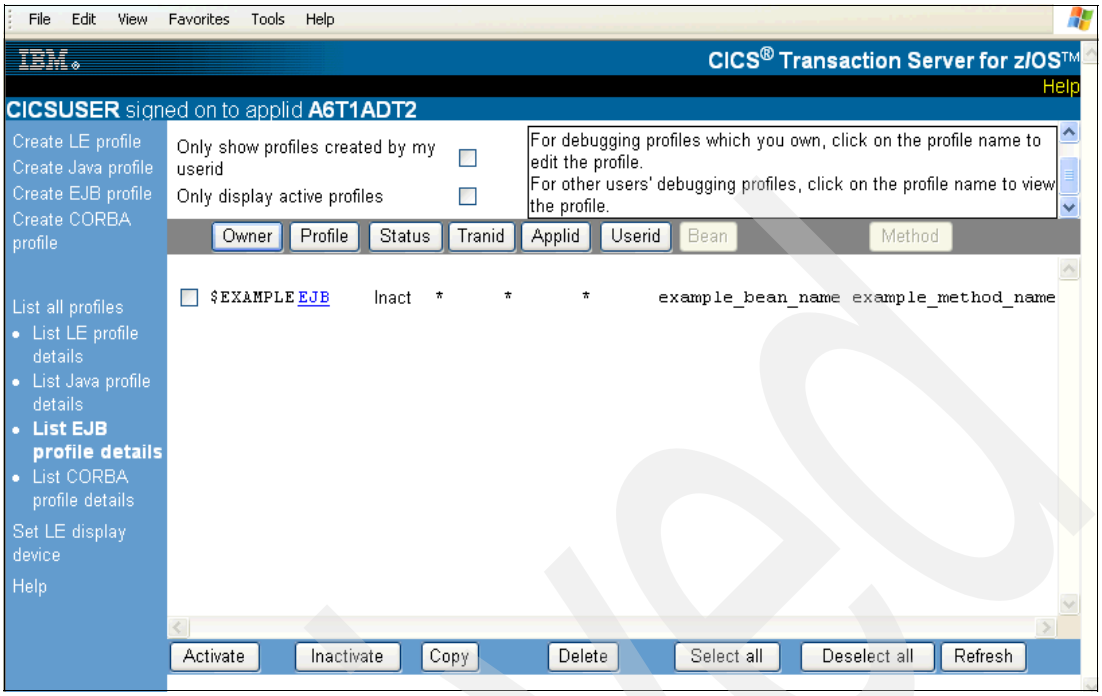


Figure 15-15 CADP Web interface - EJB profiles

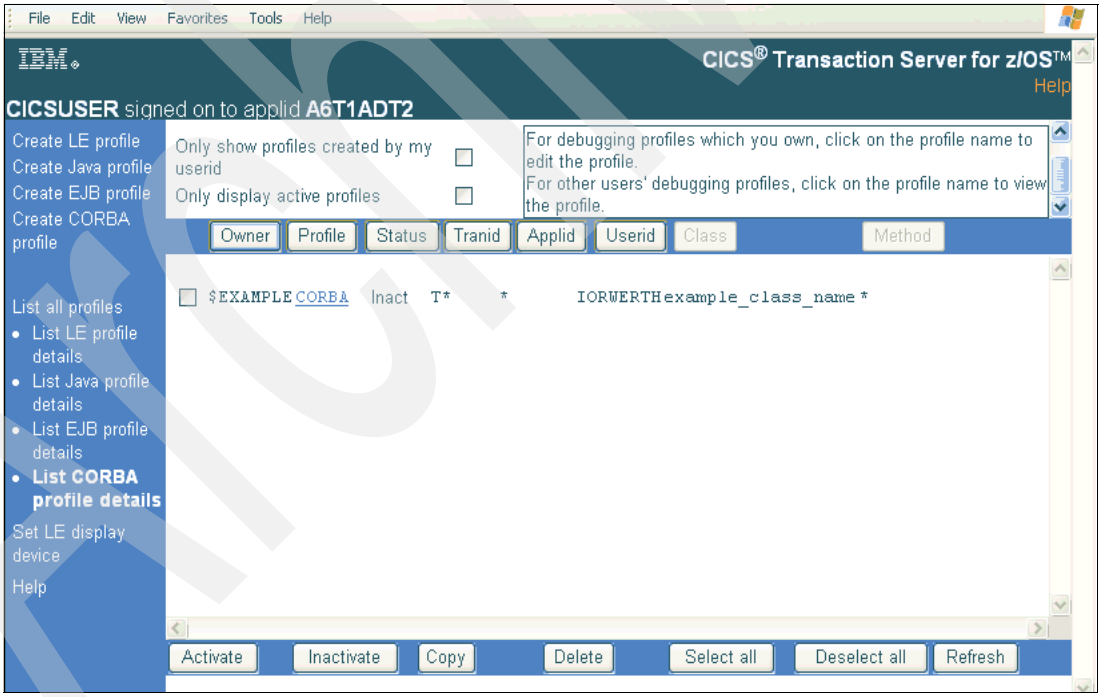


Figure 15-16 CADP Web interface - CORBA profiles

The behavior of these pages is identical to the List profiles page. However, the information displayed on each is specific to the type of profile.

The list contains selected information from the debugging profile. Columns listed are:

Owner The user ID of the profile owner (that is, of the user who created the profile)

Profile The name of the profile

Status The status of the profile (Act for active or Inact for inactive)

The following columns display information specified when the profile is created:

Tranid Displays the contents of the transaction field.

Program On the List profiles and List LE profiles pages only, displays the contents of the program field.

Compile Unit On the List profiles and List LE profiles pages only, displays the contents of Compile Unit field.

If the Compile Unit name is too long to display in the available space, the leading characters are displayed, followed by an ellipsis (...). To display the Compile Unit name in full, select the profile name.

Applid Displays the contents of the Applid field.

Userid Displays the contents of the Userid field.

Termid On the List profiles and List LE profiles pages only, displays the contents of the Terminal field.

Type On the List profiles page only, displays the type of program specified in the debugging profile:

- CORBA - CORBA object
- EJB - enterprise bean
- Java - Java program
- LE - Compiled language program

Netname On the List LE profiles page only, displays the contents of the Netname field.

Class On the List Java profiles and List CORBA profiles pages only, displays the contents of the Class field.

If the Class name is too long to display in the available space, the trailing characters are displayed, preceded by an ellipsis (...). To display the Class name in full, click the profile name.

Bean On the List EJB profiles page only, displays the contents of the Bean field.

If the bean name is too long to display in the available space, the leading characters are displayed, followed by an ellipsis. To display the bean name in full, click the profile name.

Method On the List EJB profiles and List CORBA profiles pages only, displays the contents of the Method field. If the Method name is too long to display in the available space, the leading characters are displayed, followed by an ellipsis. To display the Method name in full, click the profile name.

You can change the way information is displayed on the List profiles page, specifying which profiles are displayed and the order in which entries appear.

- Selecting which profiles are displayed:

  Use the check boxes at the top of the page to select which debugging profiles are displayed. The options are:

  - Display all profiles
  - Display all profiles that you created
  - Display all active profiles
  - Display only active profiles that you created

- Sorting the list:

  Use the buttons above each column to re-display the list in the sequence determined by the contents of the column. CICS uses the EBCDIC sorting sequence when it re-displays the list.

Your choice of which profiles are displayed, and your chosen sequence, are preserved and used the next time you use the debugging profile manager.

Buttons on the List Profiles page

The buttons at the head of the following columns are used to re-display the list of profiles in sequence:

- Owner
- Profile
- Status
- Tranid
- Program
- Compile
- Unit
- Applid
- Userid
- Termid
- Netname
- Type

The following buttons are inactive, and cannot be used to re-sequence the list of profiles:

- Class
- Bean
- Method

Other actions are performed using the buttons at the bottom of the List profiles page:

Activate	Activate selected profiles.
Inactivate	Inactivate selected profiles.
Copy	Copy selected profiles.
Delete	Delete selected profiles.
Select all	Select all the profiles in the list.
Deselect all	Deselect all the profiles in the list.
Refresh	Refresh the List profiles page. The list is updated to show any changes that you, and other users, have made.

Creating a new LE debugging profile with the Web interface

To create a new LE debugging profile with the Web interface, complete the following steps:

1. Click **Create LE profile**. CICS displays the Create LE profile page, as shown in Figure 15-17.
2. Click **Create EJB profile**. CICS displays the Create EJB profile page, as shown in Figure 15-18 on page 366.

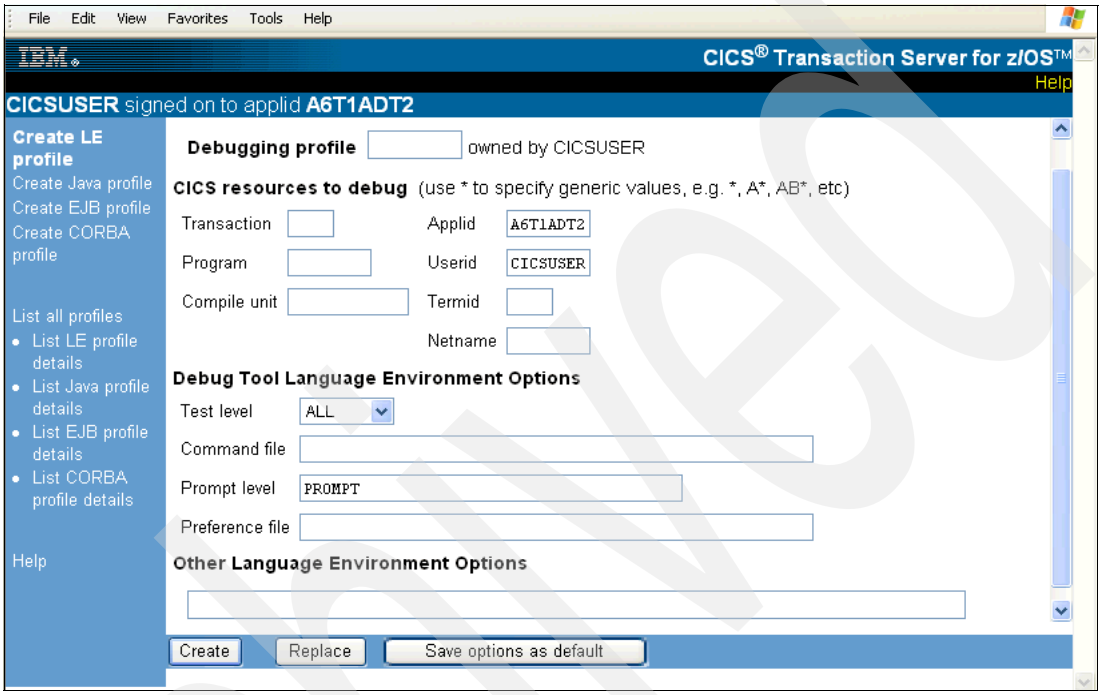


Figure 15-17 CADP - Web interface - Create LE profile page

Fields on the Create LE profile page are:

– Debugging profile

  Specify the name of the profile. If you change the name of an existing profile, CADP creates a new profile with the new name, and leaves the original profile unchanged.

The following fields specify which programs should trigger the start of a debugging session when the profile is active. You can specify a generic value for any of these fields.

– Transaction

  Specify a value in this field when you want to debug only those programs that run under the specified transaction.

– Program

  Specify a value in this field when you want to debug only the specified program.

– Compile unit

  Specify a value in this field when you want to debug only the specified compile unit.

– Applid

  Specify a value in this field when you want to confine debugging to programs that run in the specified CICS region.

- Userid

  Specify a value in this field when you want to confine debugging to programs that are being run by the specified user.

- Termid

  Specify a value in this field when you want to confine debugging to programs that are being run by the specified terminal.

- Netname

  Specify a value in this field when you want to confine debugging to programs that are being run by the specified netname.

The following fields specify options that are passed to Debug Tool:

- Test level

  Specifies which conditions raised by your program will cause Debug Tool to gain control. You can enter the following values: All, Error, or None.

- Command file

  Specifies the primary commands file associated with the profile. You can specify the fully qualified name of a sequential dataset or a member of a partitioned data set.

- Prompt level

  Specifies whether an initial commands list is unconditionally executed during program initialization. Enter one of the following PROMPT, NOPROMPT, or *command*.

- Preference file

  Specifies the preference file that Debug Tool uses when debugging programs that match this profile. You can specify the fully qualified name of a sequential dataset or a member of a partitioned data set.

- Other Language Environment options

  Specifies Language Environment runtime options for programs that match this profile. When a program is selected for debugging because it matches the profile, the runtime options specified will override other runtime options that you might have in effect.

3. Click **Create**. CICS checks that you have entered valid data.
4. Click **List profiles** to return to the List profiles page.

Buttons on the Create LE profile page

The buttons on the Create LE profile page are:

- Create

  Create a new profile using the information entered on the page.

- Replace

  Update an existing profile using the information entered on the page.

- Save options as default

  Save the contents of the following fields. The saved values are used by default each time you create an LE debugging profile.

  - Test level
  - Command file
  - Prompt
  - Preference file
  - Language Environment options

Creating a new Java, EJB, or CORBA profile with the Web interface

To create a new Java, EJB, or CORBA profile with the Web interface, click the appropriate item on the screen:

- Create Java profile.

  CICS displays the Create Java profile page shown in Figure 15-18.

- Create EJB profile.

  CICS displays the Create EJB profile page shown in Figure 15-19.

- Create CORBA profile.

  CICS displays the Create CORBA profile page shown in Figure 15-20.

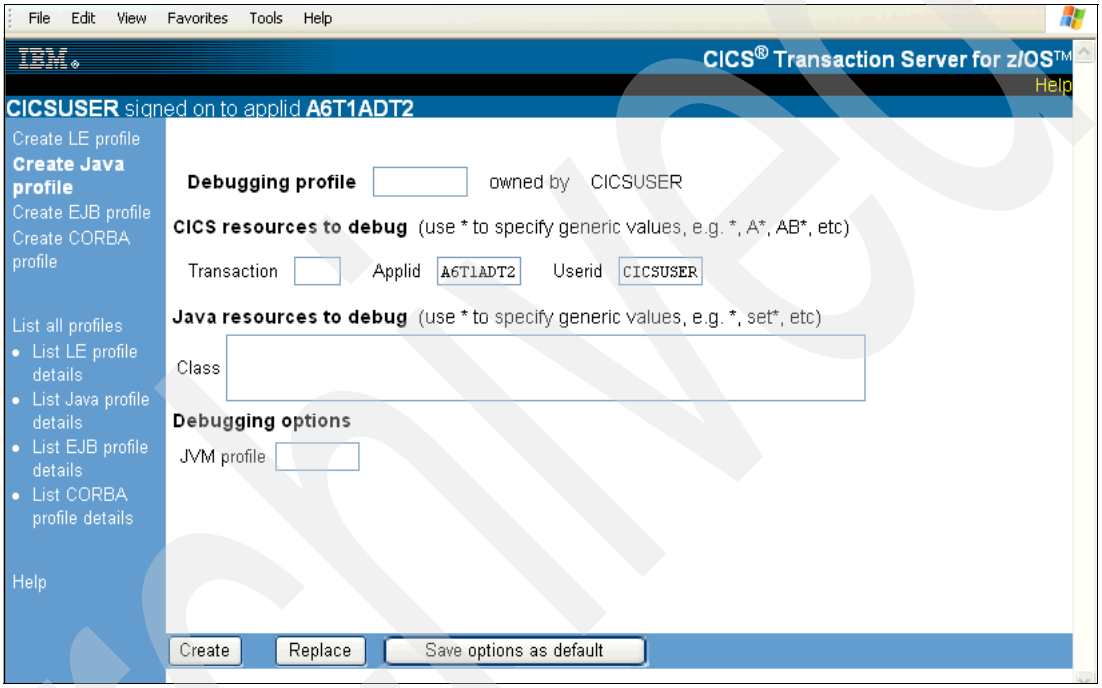


Figure 15-18 CADP - Web interface - Create Java profile page

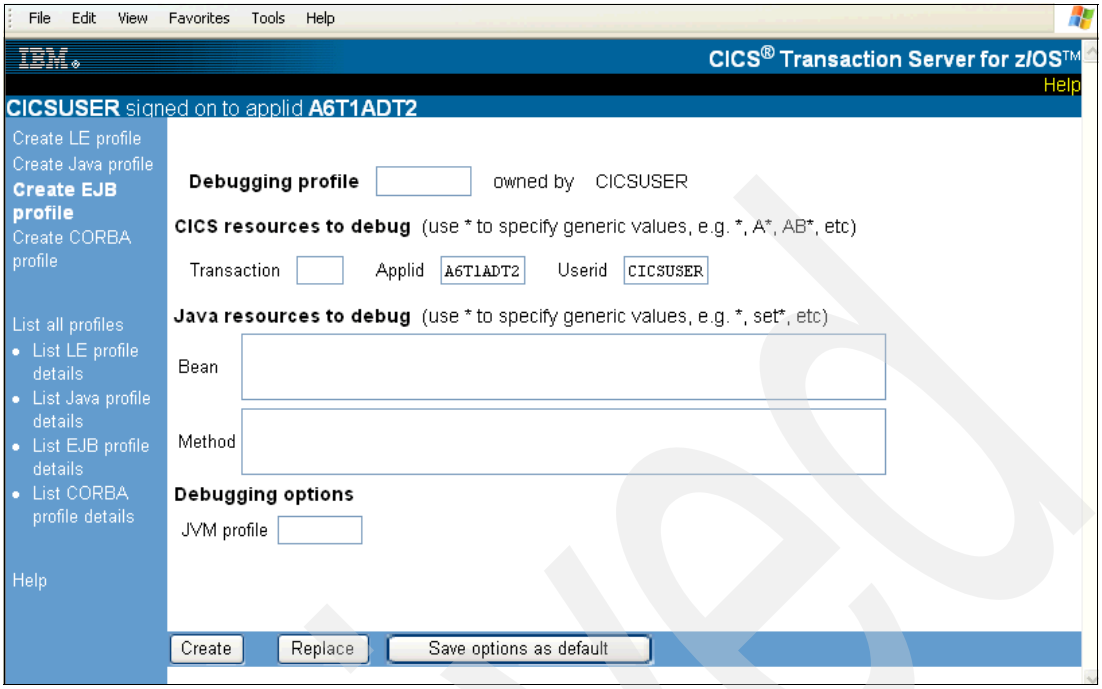


Figure 15-19 CADP - Web interface - Create EJB profile page

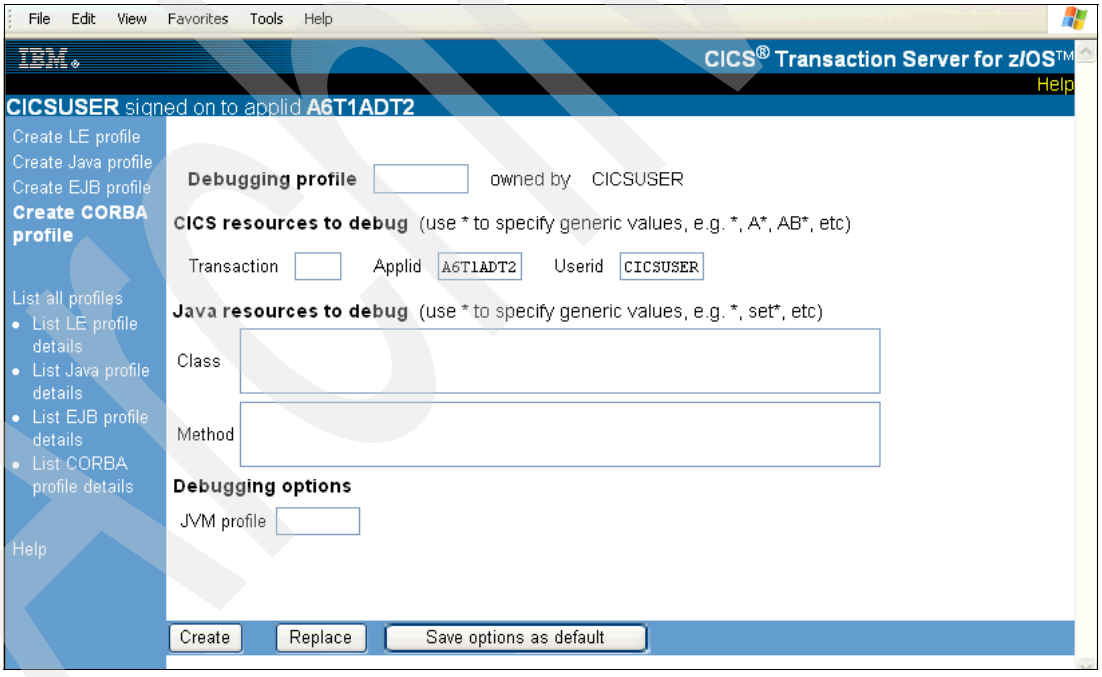


Figure 15-20 CADP - Web interface - Create CORBA profile page

- Fields on the previous three panels are:
  - Debugging Profile

    Specify the name of the profile. If you change the name of an existing profile, CADP creates a new profile with the new name, and leaves the original profile unchanged.
- The following fields specify which programs should trigger the start of a debugging session when the profile is active. You can specify a generic value for any of these fields.
  - Transaction

    Specify a value in this field when you want to debug only those programs that run under the specified transaction.
  - Applid

    Specify a value in this field when you want to confine debugging to programs that run in the specified CICS region.
  - Userid

    Specify a value in this field when you want to confine debugging to programs that are being run by the specified user.
- The following fields specify which Java resources should trigger the start of a debugging session when the profile is active:
  - Type

    Specifies the type of Java resource that you want to debug:
    - Java: Enter this value when you want to debug a Java program.
    - EJB: Enter this value when you want to debug an enterprise bean.
    - CORBA: Enter this value when you want to debug a stateless CORBA object.
  - Class

    For Java programs and stateless CORBA objects only, specify a value in this field when you want to debug only the specified class. You can specify a generic value if you want to debug a set of similarly named classes.
  - Bean

    For enterprise beans only, specify a value in this field when you want to debug only the specified bean. You can specify a generic value if you want to debug a set of similarly named beans.
  - Method

    For enterprise beans and stateless CORBA objects only, specify a value in this field when you want to debug only the specified method.

    When an inbound request initiated by a Java remote method invocation is received, the value specified is compared with the mangled name in the inbound request to determine whether the profile matches the request. If it is possible that mangling can take place, do not specify a method name in the debugging profile, but specify a generic method instead.
- The following field specifies the debugging options for this profile. You can save the value that you specify. The saved value is used by default each time you create a Java debugging profile.
  - JVM profile

    The JVM profile specifies the name of the JVM profile that is used for Java programs that match this profile. The profile should specify that the Java program is to run in debug mode. You cannot specify a generic value for this parameter.

Buttons on the Create LE profile page

The buttons on the three previous create profile pages are:

- Create

  Create a new profile using the information entered on the page.

- Replace

  Update an existing profile using the information entered on the page.

- Save options as default

  Save the contents of the following fields. The saved values are used by default each time you create a Java debugging profile.

How to start a Non-Language Environment Debug Tool under CICS

To debug CICS non-Language Environment programs, Debug Tool Non-Language Environment Exits must have been previously started.

Passing runtime parameters into the non-LE debug session on startup

When you define your debugging profile using DTCN or CADP, you can pass a limited set of runtime options that will take effect in your non-Language Environment debugging session. These runtime options and their settings include the following:

- TEST/NOTEST: must be TEST
- TEST LEVEL: must be ALL
- Commands file
- Prompt Level: must be PROMPT
- Preference file
- You can also specify the following runtime options in a TEST string:
  - COUNTRY: Specifies a country code for Debug Tool. Valid country codes are:
    - US: United States of America
    - JP: Japan
  - NATLANG: Specifies the national language used to communicate with Debug Tool. Valid national language identifiers are:
    - ENU: English
    - UEN: Upper-case English
    - JPN: Japanese
  - TRAP: Specifies whether Debug Tool is to intercept abends of a valid value. It is one of:
    - ON: Enable Debug Tool to trap ABENDs.
    - OFF: Prevent Debug Tool from trapping ABENDs. An ABEND causes abnormal termination of both Debug Tool and the program under test.

15.1.3 How to start Debug Tool under CICS

To use Debug Tool under CICS, you must ensure that all of the required installation and configuration steps for CICS Transaction Server, Language Environment, and Debug Tool have been completed.

Choosing a debug mode

You can start Debug Tool in one of the following ways:

- Single terminal mode:

  Debug Tool displays its screens on the same terminal as the application. This can be set up using CADP, DTCN, CEETEST, pragma, or CEEUOPT(TEST).

- Dual terminal mode:

  Debug Tool displays its screens on a different terminal than the one used by the application. This can be set up with CADP, DTCN, or CEDF.

  If you are using Debug Tool in a multiple-CICS region environment and sharing the EQADTCN2 temporary storage queue for DTCN or DFHDPFMB and DFHDPFMP for DCADP, set your profile to a debugging Display ID that is located in the same CICS region that the task you want to debug will run in.

- Batch mode:

  Debug Tool does not have a terminal, but uses a commands file for input and writes output to the log. This can be set up using CADP, DTCN, CEETEST, pragma, or CEEUOPT(TEST).

- Remote debug mode:

  Debug Tool works with a remote debugger to display results on a graphical user interface. This can be set up using CADP, DTCN, CEETEST, pragma, or CEEUOPT(TEST).

Methods for starting Debug Tool under CICS

There are several different mechanisms available to start Debug Tool under CICS. Each mechanism has its own advantages. The mechanisms are as follows:

- DTCN is a full-screen CICS transaction that allows you to dynamically modify any Language Environment TEST or NOTEST runtime option with which your application was originally link-edited. You can also use DTCN to modify other Language Environment runtime options that are not specific to Debug Tool.
- CADP is a CICS transaction that enables you to manage debugging profiles. This transaction is available with CICS Transaction Server for z/OS Version 2 Release 3. CADP has the following advantages over DTCN:
  - With CADP, multiple profiles with a single program name can be added from the same display device. There is no limit to the number of profiles supported. With DTCN, a single profile, with up to eight program IDs, can be added from a single display device. In either case, the program names can be specified with wild cards.
  - CADP provides the same abilities as DTCN for managing debug profiles for Language Environment applications. CADP can also help manage debug profiles for Java applications, Enterprise Java Beans (EJBs), and CORBA stateless objects.
  - CADP profiles are persistent, and are kept in VSAM files. Persistence means that if the CADP profile was present before a CICS region is restarted, the CADP profile is present after the CICS region is restarted. For DTCN profiles, if the CICS region that owns the temporary storage queue where the debugging profiles were defined is restarted, the DTCN profiles must be added again after the region is restarted.
- Language Environment CEEUOPT module link-edited into your application, containing an appropriate TEST option, tells Language Environment to start Debug Tool every time the application is run. This mechanism can be useful during initial testing of new code when you will want to run Debug Tool frequently.
- A *compiler directive* within the application, such as #pragma runopts(test) (for C and C++) or CALL CEETEST.

These directives can be useful when you must run multiple debug sessions for a piece of code that is deep inside a multiple enclave or multiple CU application. The application runs without Debug Tool until it encounters the directive, at which time Debug Tool is started at the precise point that you specify. With CALL CEETEST, you can even make the invocation of Debug Tool conditional, depending on variables that the application can test.

- CEDF utility where you can start a debug session in Dual Terminal mode alongside CEDF, using a special option on the CEDF command.

  This mechanism does not require you to change the application link-edit options or code, so it can be useful if you must debug programs that have been compiled with the TEST option, but do not have invocation mechanisms built into them.

Starting Debug Tool by using DTCN

DTCN profiles contain the identifiers (IDs) of CICS resources to debug. These resource IDs can be terminal, transaction, program, or user.

Note: To debug a PL/I or C/C++ CICS program by using DTCN to start Debug Tool, update the link-edit step to include the EQADCCXT member from Debug Tool library hlq.SEQAMOD into the program's main load module. If you are using CICS TS 2.3 and CADP, you do not have to include EQADCCXT.

When CICS programs are started, Debug Tool monitors the tasks to find a task that contains a combination of terminal ID, transaction ID, program ID, and user ID that matches a repository profile item. Debug Tool selects the best matching profile (the one with the greatest number of resource IDs that match the active task).

If two tasks have an equal number of matching resource IDs, the older debug profile is selected. If this situation occurs, an error message is sent to the system console, suggesting that DTCN users should specify additional resource qualification.

DTCN not only provides the capability to specify *what* to debug by specifying debug resource IDs, DTCN also provides the capability to specify *how* the debug session will run, for example, whether a mainframe (MFI) or workstation (VAD) debug session is desired.

When a DTCN profile is active for a full-screen mode debugging session, Debug Tool preserves in the profile most of the breakpoint information for that session. When the DTCN profile is deleted, the breakpoint information is deleted.

How to end your CICS debugging session

After you have finished debugging your program, use DTCN again to turn off your debug profile by pressing PF6 to delete your debug profile and then pressing PF3 to exit. You do not have to remove EQADCCXT from the load module. In fact, it is a good idea to leave it there for the next time you want to start Debug Tool.

Starting Debug Tool using CADP

CADP is an interactive transaction supplied by CICS Transaction Server for z/OS Version 2 Release 3. CADP helps you maintain persistent debugging profiles. When you start the CICS Transaction Server for z/OS Version 2 Release 3 region and you set the DEBUGTOOL system initialization parameter to YES, Debug Tool uses the CADP profile repository instead of the DTCN profile repository to find a matching debugging profile.

If you start the CICS region with DEBUGTOOL=YES, CADP will be the method for defining profiles. Those defined in DTCN will not be used to start Debug Tool sessions.

You can use the CEMT transaction to set DEBUGTOOL dynamically as follows:

- To use CADP:

  ```
  CEMT SET DEBUG
  ```

- To use DTCN:

  ```
  CEMT SET NODEBUG
  ```

Activating, inactivating, copying, and deleting profiles with the 3270 interface

To activate, inactivate, copy, or delete debugging profiles, start with the List debugging profiles screen, and follow these steps:

1. Press PF2 to ensure that the display includes the profiles that you want to activate.
2. Press PF7 and PF8 to scroll to a profile that you want to activate.
3. Type A (for activate), I (for inactivate), C (for copy), or D (for delete) in the field to the left of the profile name.
4. Repeat steps 2 and 3 to select all the profiles you want to activate. Press Enter.

By default, if any of the selected profiles to be activated is for a compiled language program, CICS displays the Set LE debugging display device screen.

If none of the selected profiles is for a compiled language program, CICS refreshes the List Debugging Profiles screen.

You can choose not to see the Set LE debugging display device screen when you activate profiles.

Note: If you change a profile while it is active, the changes take effect immediately. The next time a program is started, the changed parameters are used to decide whether the program should run under the debugger's control.

From the List Debugging Profiles screen, you can activate, inactivate, delete, and copy debugging profiles by typing the appropriate action character (A, I, D, and C, respectively) in the field to the left of the profile name. You can combine these actions on the List Debugging Profiles screen, as shown in Figure 15-21.

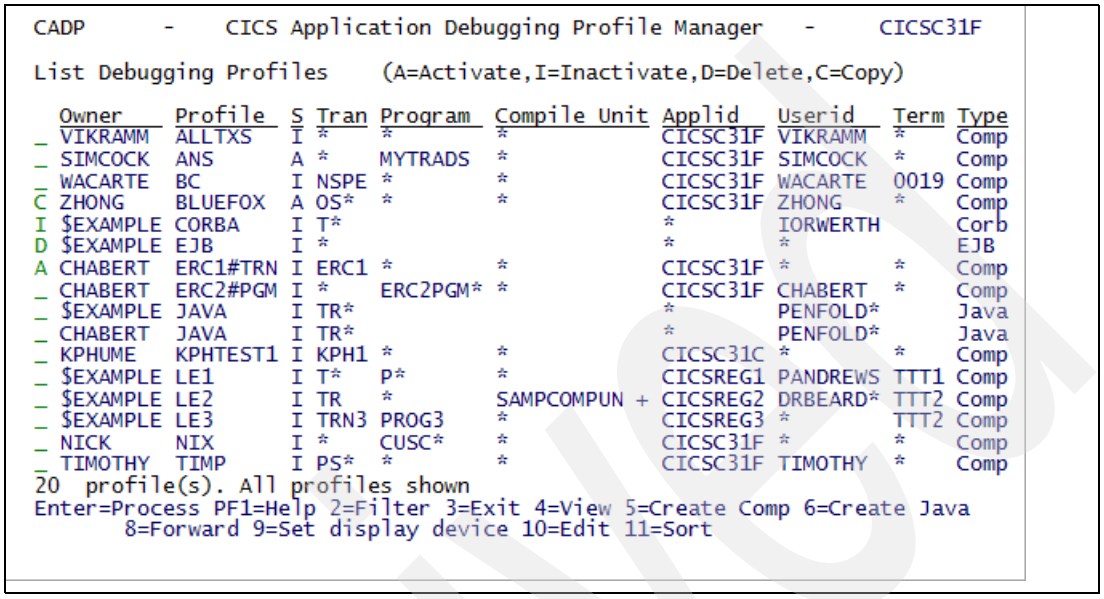


Figure 15-21 CADP - 3270 interface - profile management

CICS displays the Set Compiled Debugging Display Device to specify which display device you will use to interact with the debugger for the profile you want to activate, as shown in Figure 15-22.

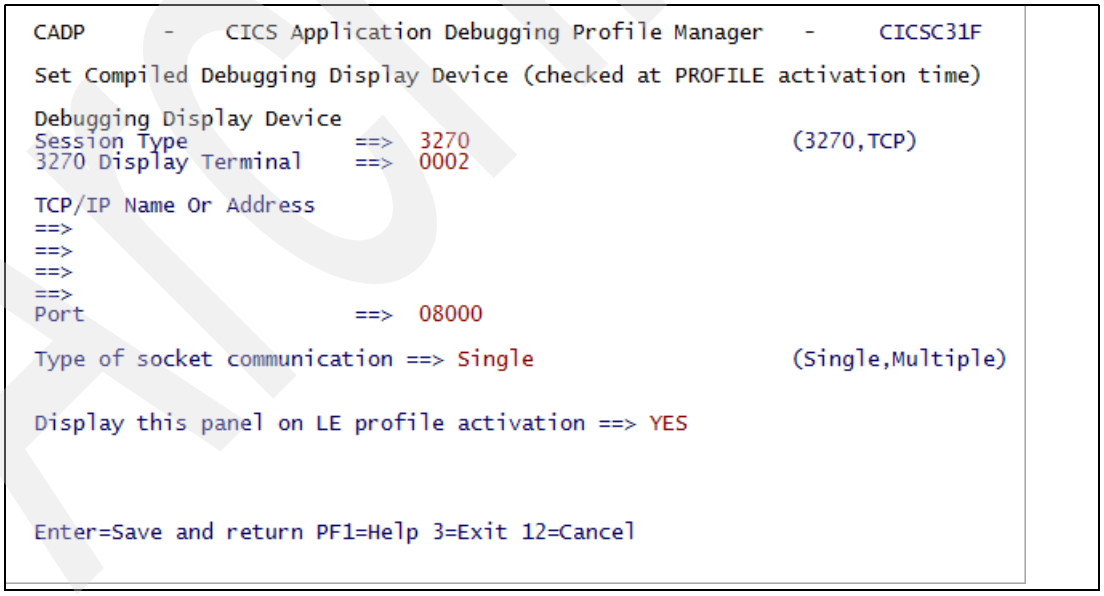


Figure 15-22 CADP - 3270 - set Compiled Debugging display device

Complete the details of the display device that you want to associate with the profile. The fields on the Set LE debugging display device screen are:

- **Session Type**

  Specifies how you will interact with Debug Tool:

  3270 You will interact with Debug Tool using a 3270 type terminal. Specify the terminal ID in the Display ID field. This is the default value.

  TCP You will interact with Debug Tool using a debugging client on your workstation. The client will communicate with Debug Tool using TCP/IP. Specify the port number at which the client listens for a connection in the Port field.

- **3270 Display Terminal**

  When the session type is 3270, specify the terminal ID of the terminal with which you will interact with Debug Tool. The default value is the ID of the terminal at which you run CADP.

- **TCP/IP Name Or Address**

  When the session type is TCP, specify the IP address or name of the host where the debugging client is running.

- **Port**

  When the session type is TCP, specify the port number at which the debugging client listens for a connection. Specify a value in the range 0 to 65535. The default is 8000.

- **Type of socket communication**

  When the session type is TCP, specifies whether the debugging client and debugging server will communicate using a single socket or more than one socket.

  Single Use a single socket for communication. This is the default value, and is the preferred value when you use a WebSphere Studio product as your debugging client.

  Multiple Use more than one socket for communication. You must specify this value when you use a VisualAge product as your debugging client.

- **Display this panel on LE profile activation**

  Specifies whether you want to display the Set LE Debugging display device screen whenever you activate debugging profiles for compiled language programs:

  YES Display the Set LE Debugging display device screen whenever debugging profiles are activated. This is the default behavior.

  NO Do not display the Set LE debugging display device screen whenever debugging profiles are activated. The display device that you specify will be associated with all of the profiles that you activate.

Activating debugging profile with the Web interface

Starting with the List profiles page, follow these steps:

1. Select the profile and select **Activate** if it is a profile for a compiled language program. The Set LE display device page is displayed, as shown in Figure 15-23.

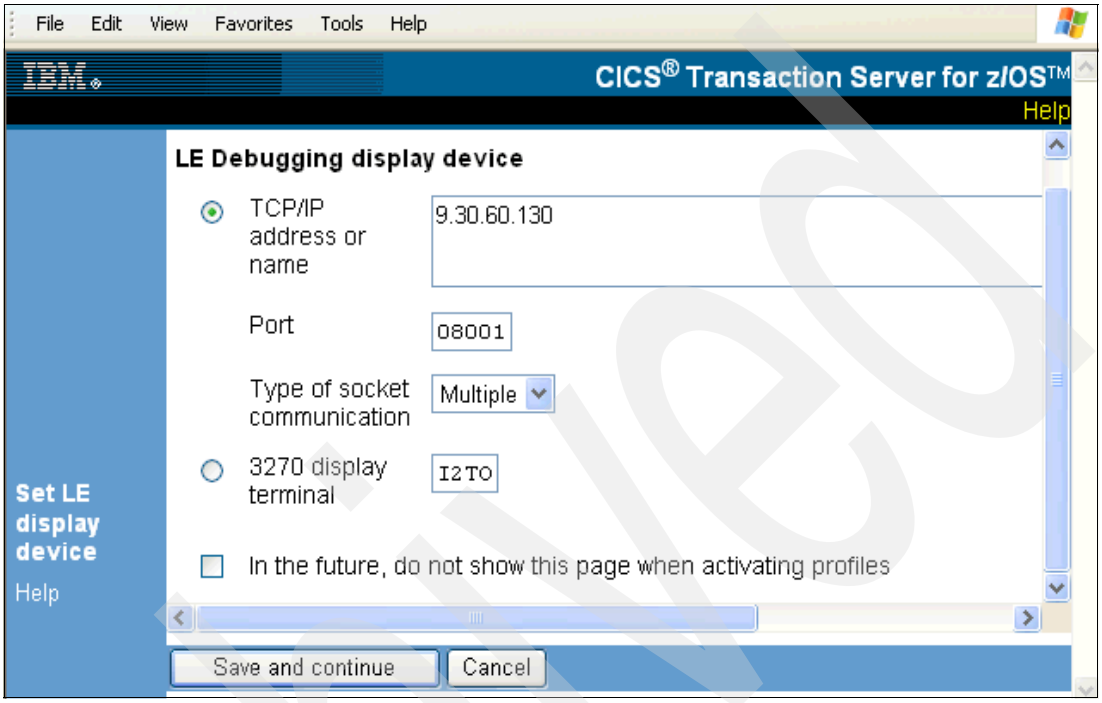


Figure 15-23 CADP - Web interface - Set LE display device page

2. Complete the details of the display device that you want to associate with the profile. You can use a 3270 terminal or a debugging tool on a workstation.

   Use the radio buttons to select how you will interact with the debugger. The fields on the Set LE display device page are:

   – TCP/IP address or name

     Specifies that you will interact with the debugger using a debugging client on your workstation. Supply the following information:

     - The IP address or name of the host where the debugging client is running. By default, CICS inserts the IP address of the client that is running the browser, or — if there is a firewall between the browser and CICS — the IP address of the firewall.
     - The port number at which the debugging client listens for a connection. Specify a value in the range 0 to 65535. The default is 8000.

   – Type of socket communication

     For a debugging client on your workstation, specifies whether the debugging client and debugging server will communicate using a single socket or more than one socket:

     - Single - Use a single socket for communication. This is the default value, and is the preferred value when you use a WebSphere Studio product as your debugging client.
     - Multiple - Use more than one socket for communication. You must specify this value when you use a VisualAge product as your debugging client.

– 3270 display terminal

  Specifies that you will interact with the debugger using a 3270 type terminal. Supply the terminal ID of the terminal at which you will interact with the debugger.

3. Click **Save and continue**. CICS saves the display device settings, and displays the List profiles page.

Starting Debug Tool using CEDF

No specific preparation is required to use CEDF to start Debug Tool other than compiling the application with the appropriate compiler options and saving the source/listing.

CEDF has the `,I` option that starts Debug Tool. This option starts both EDF and Debug Tool in Dual Terminal mode. In Dual Terminal mode, EDF and Debug Tool screens are displayed on the terminal where you issue the CEDF command. Application screens are displayed on the application terminal.

Note: You must know the ID of each terminal. One way to get this information is by using the CEOT transaction.

To start Debug Tool, enter the CEDF transaction as follows, where *xxxx* is the terminal on which you want to start the transaction to be debugged:

```
CEDF xxxx,ON,I
```

This terminal is where the application is started. CICS displays the following message:

```
TERMINAL C5TZ: OPTION I... REQUESTS Debug Tool : EDF MODE ON. DEBUG TOOL ON
```

Then, on the *xxxx* terminal, enter *TRAN*, where *TRAN* is the ID for the transaction being debugged.

Once the command is entered, Debug Tool is started for all Language Environment-enabled programs that are running on the terminal where Debug Tool is started. Debug Tool will continue to be active on this terminal, even if you turn off EDF.

Using this approach, you are able to debug CICS statements using standard CEDF functions, and language statements with Debug Tool. On the terminal where CEDF has been issued, you can switch between EDF and Debug Tool displays, as shown in Figure 15-24.

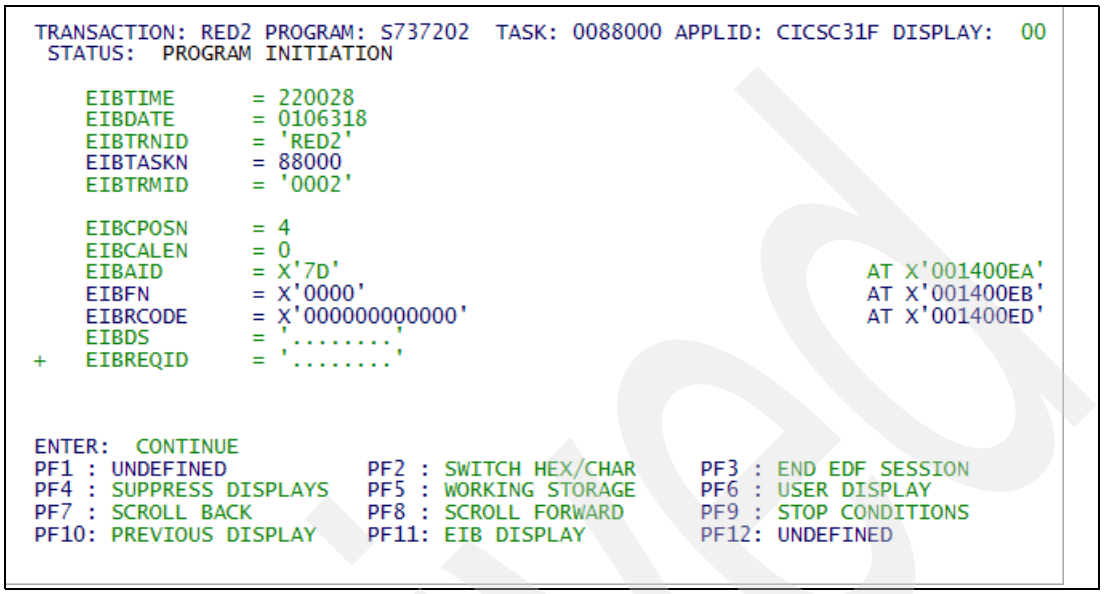


Figure 15-24 CEDF - Debug Tool display (1)

All CEDF functions are available by viewing or scrolling as required.

Starting Debug Tool using CEEUOPT

To request that Language Environment start Debug Tool every time the application is run, assemble a CEEUOPT module with an appropriate TEST runtime option. It is a good idea to link-edit the CEEUOPT module into a library and add an INCLUDE LibDD(CEEUOPT) statement to the link-edit options when you link your application. Whenever the application program runs, Debug Tool is started.

Debug Tool runs in the mode defined in the TEST runtime option you supplied, normally Single Terminal mode, although you could provide a primary commands file and a log file and not use a terminal at all.

Starting Debug Tool using compiler directives

When compile directives are processed by your program, Debug Tool is started in single terminal mode (this method supports only single terminal mode).

If you do not want to compile your program with hooks, you can use CEETEST calls to start Debug Tool at strategic points in your program as shown in Figure 15-25. If you decide to use this method, you still must compile your application so that symbolic information is created.

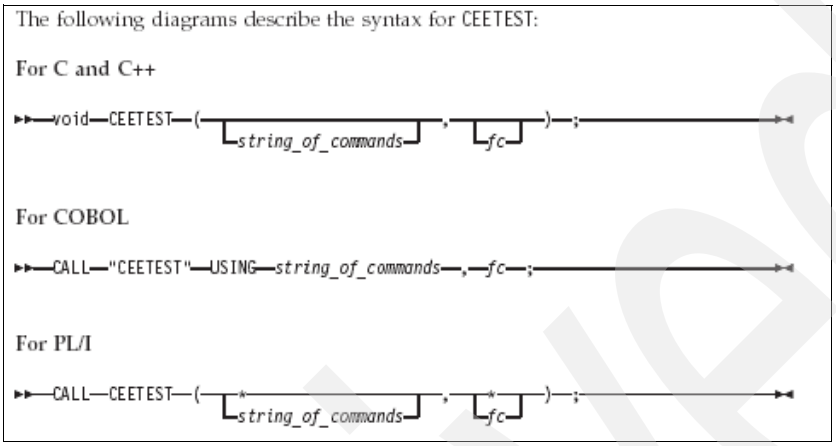


Figure 15-25 CEETEST syntax

The syntax for CEETEST is as shown in Figure 15-25.

- string_of_commands (input)

  Halfword-length prefixed string containing a Debug Tool command list. string_of_commands is optional. If Debug Tool is available, the commands in the list are passed to the debugger and carried out. If string_of_commands is omitted, Debug Tool prompts for commands in interactive mode. For Debug Tool, remember to use the continuation character if your command exceeds 72 characters.

- fc (output)

  A 12-byte feedback code, optional in some languages, that indicates the result of this service. Language Environment provides a callable service called CEEDCOD to help you decode the fields in the feedback code. We recommend that you request the return of the feedback code.

For C and C++ and COBOL, if Debug Tool was started through CALL CEETEST, the GOTO command is only allowed after Debug Tool has returned control to your program via STEP or GO.

15.1.4 How to debug CICS programs

The program source file (for C and C++ and Enterprise PL/I), the program listing (for COBOL and all other PL/I), or separate debug file (for COBOL) must be retained in a permanent data set for Debug Tool to read when you debug your program.

Note: For C, C++ and Enterprise PL/I, it is the input to the compiler that must be retained. To enhance performance when using Debug Tool, use a large block size when saving these files.

Debug modes under CICS

Debug Tool can run in several different modes, providing you with the flexibility to debug your applications in the way that suits you best. These modes include:

- Single terminal mode:

  A single 3270 session is used by both Debug Tool and the application, swapping displays on the terminal as required. As you step through your application, the terminal shows Debug Tool screens, but when an EXEC CICS SEND command is issued, that screen is displayed. Debug Tool holds that screen on the terminal for you to review. Simply press Enter to return to a Debug Tool screen. When your application issues EXEC CICS RECEIVE, the application screen appears again, so you can fill in the screen details.

- Dual terminal mode:

  This mode can be useful if you are debugging screen I/O applications. Debug Tool displays its screens on a 3270 session that is separate from the terminal displaying the application. Step through the application using Debug Tool terminal and, whenever the application issues an EXEC CICS SEND, the screen is sent to the application display terminal. When the application issues an EXEC CICS RECEIVE, Debug Tool terminal will wait until you respond to the application terminal.

- Interactive batch mode:

  Use this mode if you are debugging a transaction that does not have a terminal associated with it. The transaction continues to run without a CICS principal facility, but Debug Tool screens are displayed on a 3270 session that you name.

- Noninteractive batch mode:

  In this mode, Debug Tool does not have a terminal associated with it at all. It receives its commands from a command file and writes its results to a log file. This mode is useful if you want Debug Tool to debug a program automatically.

Preventing Debug Tool from stopping at EXEC CICS RETURN

Debug Tool stops at EXEC CICS RETURN and displays the following message:

```
CEE0199W The termination of a thread was signaled due to a STOP statement.
```

To prevent Debug Tool from stopping at every EXEC CICS RETURN statement in your application and suppress this message, set the TEST level to ERROR by using the SET TEST ERROR command.

Saving settings while debugging a pseudo-conversational program

If you change Debug Tool display settings while you debug a pseudo-conversational CICS program, Debug Tool might restore the default settings. To ensure that your changes remain in effect every time your program starts Debug Tool, store your display settings in the preferences file or the commands file.

Saving and restoring breakpoints

When breakpoints are set in a CICS transaction, Debug Tool saves these breakpoint settings and restores them the next time this transaction is started. However, saving and restoring of breakpoints in assembler compilation units is not currently supported.

Restrictions when debugging under CICS

The following restrictions apply when debugging programs with Debug Tool in a CICS environment:

- Debug Tool supports the use of CRTE terminals if both the application and Debug Tool share the terminal as a principal facility in single terminal mode. CICS does not permit the use of a CRTE terminal by Debug Tool if the terminal is not the application task's principal facility (which is the case in Dual terminal mode).
- The __ctest() function with CICS does nothing.
- The CDT# transaction is a Debug Tool service transaction used during Dual terminal mode debugging and is not intended for activation by direct terminal input. If CDT# is started via terminal entry, it will return to the caller (no function is performed).
- Applications that issue EXEC CICS POST cannot be debugged in Dual terminal mode.
- References to ddnames are not supported. All files, including the log file, USE files, and preferences file, must be referred to by their full data set names.
- The commands TSO, SET INTERCEPT, and SYSTEM cannot be used.
- CICS does not support an attention interrupt from the keyboard.
- The log file (INSPLOG) is not automatically started. You must use the SET LOG ON command.
- Ensure that you allocate a log file big enough to hold all the log output from a debug session because the log file is truncated after it becomes full. (A warning message is not issued before the log is truncated.)
- Save files (INSPSAFE) are not used under CICS.
- You can start Debug Tool when a Non-Language Environment assembler program under CICS starts by defining a debug profile using CADP or DTCN. However, Debug Tool will only start on a CICS Link Level boundary, such as when the first program of the task starts or for the first program to run at a new Link Level. For profiles defined in CADP or DTCN, which list a non-Language Environment assembler program name that is dynamically called using EXEC CICS LOAD/CALL, Debug Tool will not start. Non-Language Environment assembler programs that are called in this way are identified by Debug Tool in an already running debugging session and can be stopped by using a command like AT APPEARANCE or AT ENTRY. However, they cannot be used to trigger a Debug Tool session initially.

CALL %CEBR and CALL %CECI (full-screen 3270 only commands)

These:

Call %CEBR	Invokes the CICS Temporary Storage Browser Program. It browses CICS Temporary Storage Queues (and supports *importing* of Transient Data Queues as TS queues for TD browsing).
Call %CECI	Invokes the CICS Command Level Interpreter Program. It gives you the capability to dynamically execute EXEC CICS commands (such as READ of a VSAM file) while the transaction runs.

15.2 Debug Tool and DB2

In this section we discuss the following topics:

- Compiling source for both DB2 and Debug Tool
- Invoking the DB2 application specifying the TEST run-opt
- Using DTSU to invoke your DB2 application
- Considerations for DB2 stored procedures

15.2.1 Compiling for DB2 and Debug Tool

There is one basic issue when compiling DB2 programs for use with Debug Tool. This is the fact that the DB2 pre-compiler will expand your source code, and then the debug information is generated based on that expansion.

There are two solutions to this problem:

- As of DB2 V7, the DB2 product provides utilities for compilers to use to do the DB2 expansion on the fly. When the compiler front ends do this expansion, you do not have to keep track of any extra code, and the code you are looking at when debugging matches the source files you normally work with. Look for options named DB2 or SQL on your compilers.
- If you do not have a compiler that supports an integrated DB2 pre-compiler, you must prepare your code for debugging in a two-step process:
  a. Pre-compile the source and save the generated source in a file. This file is required when debugging C and PL/I (and might potentially be useful for COBOL).
  b. Compile the source as usual for debugging.

15.2.2 Invoking a DB2 application using the TEST runtime option

To use the TSO command interface to start executing your application program, issue the DSN command to invoke DB2, as shown in Example 15-8.

Example 15-8 Debug Tool and DB2 - using DSN command

```
DSN SYSTEM(DBA1) RUN PROGRAM(TRADERD2) PLAN(TRADERD2)
LIB('DAVINR2.DEMOS.PDPAK.LOAD')
PARMS('/NOTEST(ALL,*,PROMPT,MFI:INSPREF) ')
```

To use the TSO Call Access Facility (CAF), link-edit the CAF interface module, DSNALI, with your application program. Then issue the TSO CALL command for your application program, and include the TEST runtime option as a parameter, as shown in Example 15-9.

Example 15-9 Debug Tool and DB2 - using CAFF interface

```
CALL ‘change.mgmt.test.loadlib(progname)’ ‘/TEST’
```

15.2.3 Using DTSU to invoke your DB2 application

The panel shown in Figure 15-26 lets you select the Initialize New setup file for DB2. This is done by selecting option **2** from the Debug Tool main panel.

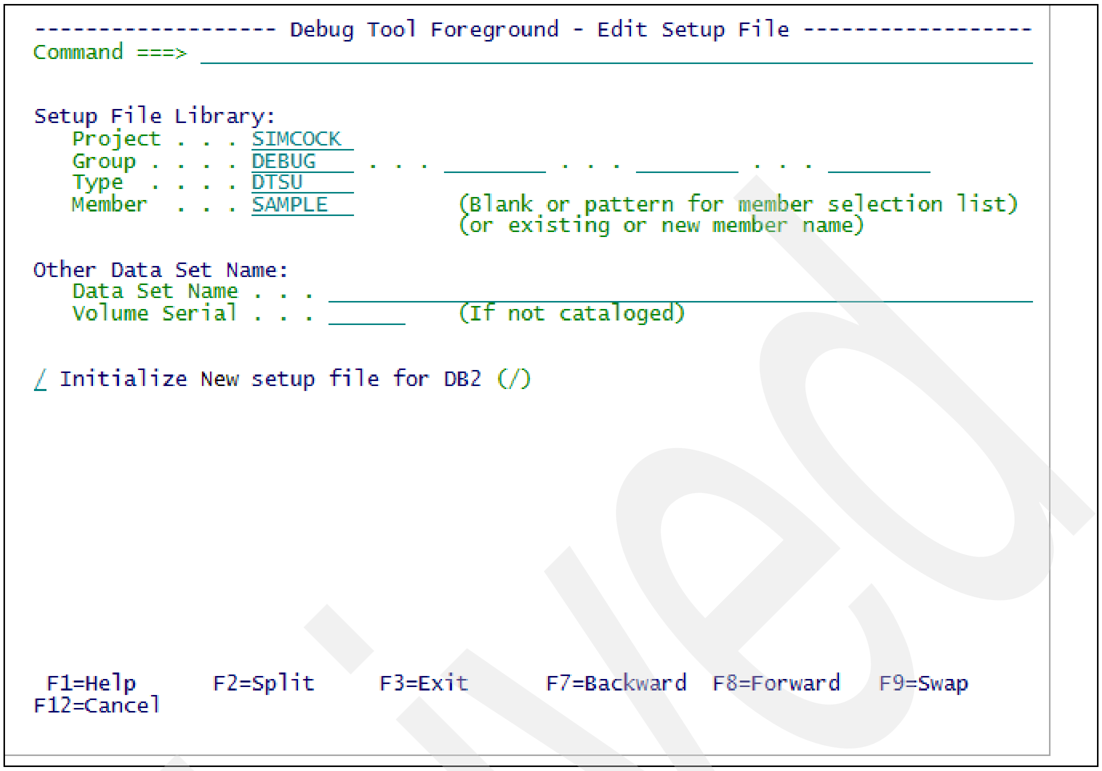

------------------ Debug Tool Foreground - Edit Setup File ------------------
Command ===>
Setup File Library:
Project . . . SIMCOCK
Group . . . . DEBUG . . . .
Type . . . . DTSU
Member . . . SAMPLE
(Blank or pattern for member selection list)
(or existing or new member name)
Other Data Set Name:
Data Set Name . . .
Volume Serial . . .
(If not cataloged)
/ Initialize New setup file for DB2 (/)
F1=Help F2=Split F3=Exit F7=Backward F8=Forward F9=Swap
F12=Cancel

Figure 15-26 Debug Tool Edit Setup File screen

Specify a file to allocate (if the given PDS does not exist) and you are presented with the Edit Setup File panel, as shown Figure 15-27.

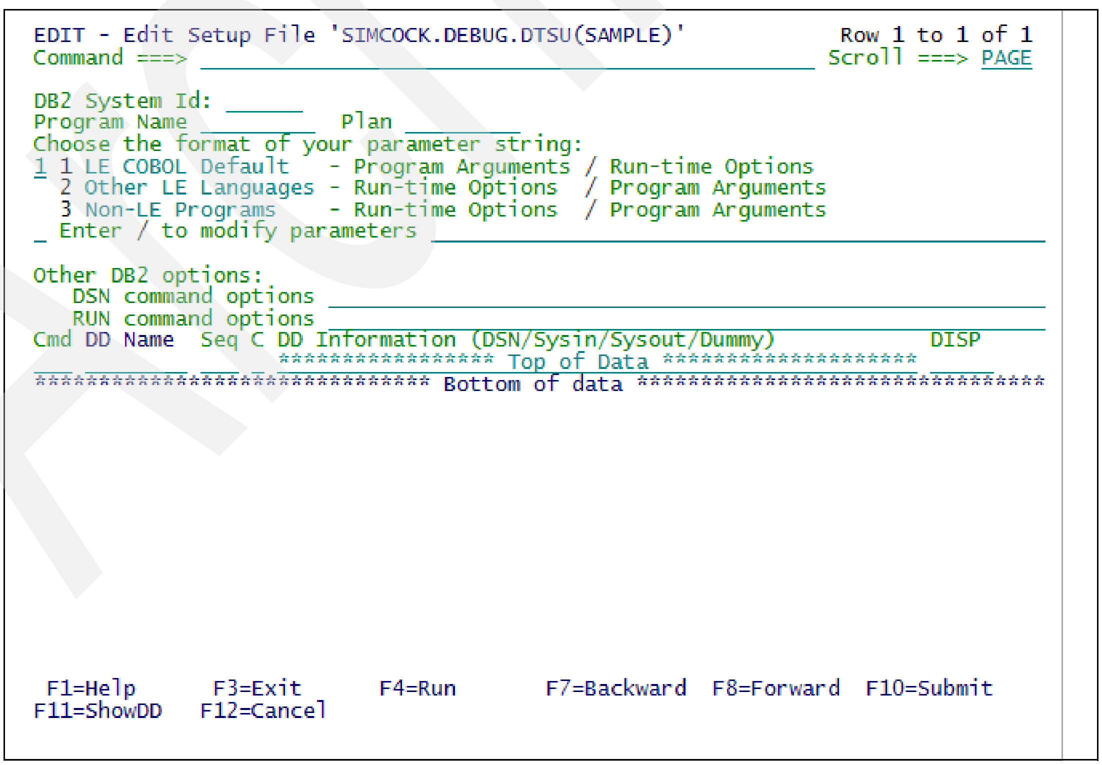

EDIT - Edit Setup File 'SIMCOCK.DEBUG.DTSU(SAMPLE)'
Row 1 to 1 of 1
Command ===>
Scroll ===> PAGE
DB2 System Id:
Program Name
Plan
Choose the format of your parameter string:
1 1 LE COBOL Default - Program Arguments / Run-time Options
2 Other LE Languages - Run-time Options / Program Arguments
3 Non-LE Programs - Run-time Options / Program Arguments
_ Enter / to modify parameters
Other DB2 options:
DSN command options
RUN command options
Cmd DD Name Seq C DD Information (DSN/Sysin/Sysout/Dummy) DISP
****************** Top of Data ********************
******************************** Bottom of data ********************************
F1=Help F3=Exit F4=Run F7=Backward F8=Forward F10=Submit
F11=ShowDD F12=Cancel

Figure 15-27 Creating a new setup file for DB2

On this panel, you can define your DB2 subsystem ID, the program, and the plan name. Eventually you can add DB2 DSN and DB2 RUN command options, and add the DD statements for the resources required. Then enter the DTSU run command (PF4) to run your DB2 application.

If you intend to run your DB2 program using the TSO CAF facilities, do not select for DB2. Instead, use DTSU for normal BATCH programs. (Remember to link DSNALI to your DB2 application.)

If you are using DTSU, you do not require your own CLISTs for the file allocations. We recommend that you use DTSU.

15.2.4 DB2 stored procedures considerations

Debug Tool supports the debugging of DB2 stored procedures. To define the runtime option TEST, you could use a CEEUOPT containing only this TEST option. However, a CEEUOPT will probably not work in most situations because most DB2 stored procedures can be defined as Type=SUB, and CEEUOPT is ignored by Language Environment for non-MAIN programs. We recommend that you use the DB2-provided RUNOPTS field of the DB2 catalog by using the appropriate DB2 SQL commands.

For DB2 Version 6 and later, the stored procedure is defined as follows:

- Create procedure: `traderb`
- Language: `cobol`
- External name: `traderb`
- Parameter style: `general`
- WLM environment: `wlmenv1`
- Run options: '`TEST(,,,vaDTCPIP&9.28.194.155%8001:INSPPREF`'
- Program type: `sub`

For DB2 Version 6 and later, to verify that the stored procedure is defined correctly, use the SQL SELECT command on the appropriate DB2 table, as follows:

```
select * from sysibm.sysroutines;
```

If the definition is not correct, or if you want to remove the TEST option (we recommend that you do not remove it, but change it from TEST to NOTEST), use the appropriate SQL command to modify the stored procedure, as follows:

```
alter procedure traderb run options
'TEST(,,,vaDTCPIP&9.28.194.155%8000:INSPPREF*);'
```

The TEST runtime option used in these examples assumes that you will use remote debugger. If you want to use full-screen 3270, your TEST runtime option should look as follows:

```
'TEST(,,,MFI%LUOTCP08:INSPPREF*);'
```

You can then call your stored procedure.

15.3 Debug Tool and IMS

This section describes how to compile, link, and debug a program under IMS.

The first part describes the facilities available prior to IMS V8. The second part describes some new facilities available to purchasers of Debug Tool Utilities and Advanced Features and IMS Version 8.

15.3.1 Traditional methods of debugging under IMS

Traditionally, the hardest part of preparing for debugging of an IMS program or transaction is setting the runtime options.

For compilation, the instructions given in the individual language chapters should be used.

For runtime options, there has traditionally never been a way to specify command-line runtime options under IMS. In releases prior to IMS V8, you must use one of the following ways to set the runtime options for a transaction or program:

- An LE CEEUOPT
- The #pragma runopts (for C/C++)
- PLIXOPT for PL/I

Full-screen 3270 single terminal mode is not available under IMS. Dual terminal or a remote debugger should be used.

15.3.2 Debugging with IMS V8 and later

If you have Debug Tool Utilities and Advanced Functions installed on your system and are using IMS Version 8, there are some new DTU facilities that allow you to easily change your LE runtime options on the fly. The Debug Tool interface uses the IMS Single Point of Control facilities that are described in the IMS books, but provides an easy-to-use full featured interface tailored towards Debug Tool users.

In the past, the only way to provide the TEST runtime option that is required to bring up Debug Tool was to compile it into the program (with a CEEUOPT or using #pragma runopts with C/C++). This caused problems because:

- All users of the program were affected by this change, not simply the person debugging the program.
- Two people could not debug the same program simultaneously because only one VTAM LUNAME or TCP/IP address could be given.
- The CEEUOPT or program had to be recompiled and relinked every time your TCP/IP address or available VTAM terminal changed.

A second set of features is usable without IMS Version 8 and allows you to create or edit setup files for your IMS program using Debug Tool Utilities.

Linking an application to allow the managing of runtime options

IMS uses a customized version of the LE CEEBXITA to allow LE runopts to be specified. This exit is called DFSBXITA and is found in SDFSSMPL. This exit can be used in one of two ways:

- You can assemble and link it with your application in order to allow dynamic changing of the LE runtime options for that application only.
- Your system programmer can assemble and then link it into the SCEERUN(CEEBINIT) module. When this SCEERUN is used with your IMS region, it will allow the new functionality to be used with *all* applications running in the region. The JCL shown in Example 15-10 is the assembled DFSBXITA that was in HLQ.MY.OBJECT(CEEBXITA).

Example 15-10 JCL

```
//LKED EXEC PGM=IEWL,REGION=4M,
// PARM='CALL,LIST,XREF,LET,RENT,REFR,MAP,AMODE=24,RMODE=24'
//SYSLIB DD DSN=CEE.SCEELKED,DISP=SHR
// DD DSN=CEE.SCEERUN,DISP=SHR
//MYOBJ DD DSN=HLQ.MY.OBJECT,DISP=SHR
//SYSLMOD DD DSN=<>,DISP=SHR
//SYSUT1 DD DSN=&&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(3,1)),
// DCB=BLKSIZE=1024
//SYSPRINT DD SYSOUT=*
//SYSLIN DD DDNAME=SYSIN
//SYSIN DD *
INCLUDE MYOBJ(CEEBXITA)
REPLACE CEEBXITA
INCLUDE SYSLIB(CEEBINIT)
ORDER CEEBINIT
MODE AMODE(24),RMODE(24)
ENTRY CEEBINIT
NAME CEEBINIT(R)
/*
```

Debug Tool IMS support

Option 4 on Debug Tool Utilities panel, as shown in Figure 15-28, provides the IMS debug support facilities, as shown in Figure 15-29.

```
------------------------- Debug Tool Utilities -------------------------
Option ===> _____________________________________________________________

                                                                More:     +
0  Manage Job Card
   For Program Preparation and Setup File Management

1  Program Preparation
   Compile old or new COBOL programs with newer compilers, convert old COBOL
   source into new COBOL source, use other compilers, and link edit.

2  Manage and Use Debug Tool Setup Files
   You can manage setup files and use them to run your program interactively
   with Debug Tool in TSO Foreground or submit your program to run in
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs
   You can dynamically update Language Environment run-time options
   and create private message region.

5  Load Module Analyzer
   Analyze load modules to determine the language translator used to
   compile or assemble each CSECT in the load module.

6  Manage TEST Run-time Option Data Set

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 15-28 Main Debug Tool Utilities menu

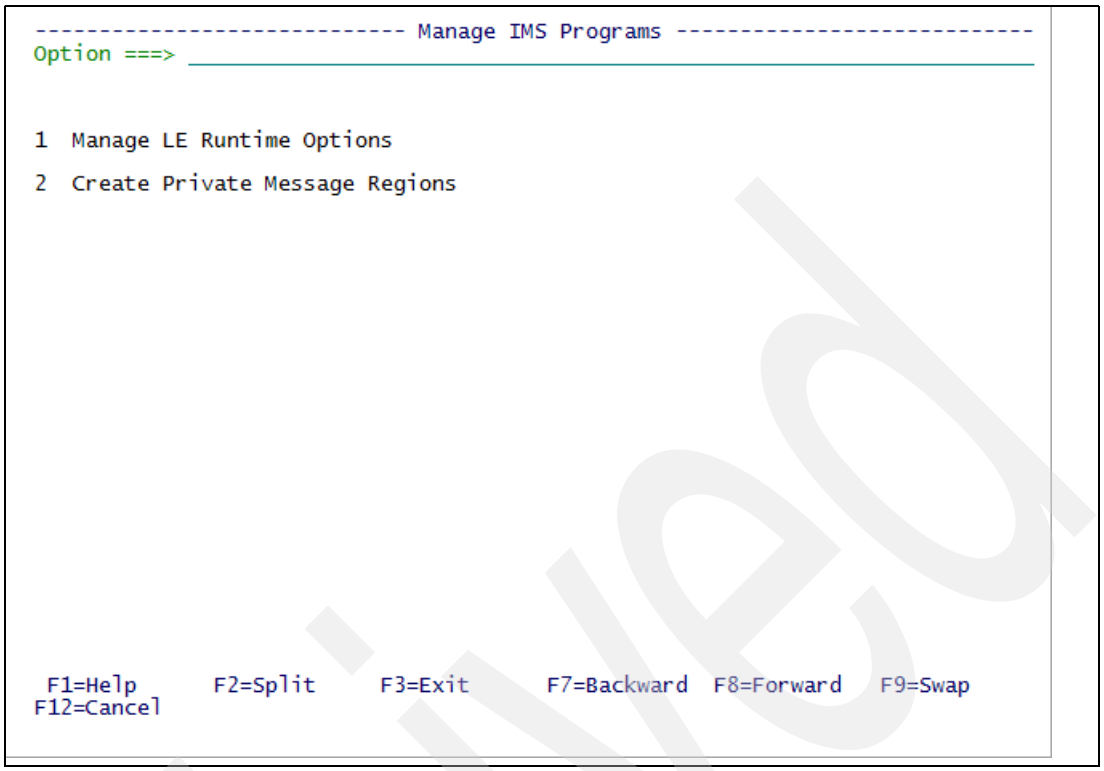


Figure 15-29 Option 4 - Manage IMS Programs

Two options are available for customization. The panel shown in Figure 15-30, arrived at by selecting option 1 in the previous panel, requires you to enter the IMS Plex ID. All other fields are optional and can be used to reduce the number of entries on the following panel. Unless you have a large number of entries or know specifically which entry you are looking for, it might not be a good idea to specify any query qualifiers.

This facility allows the specification of runtime options to be as global as the whole IMS plex, or as specific as a given program in a given transaction from a given terminal submitted by a specific user ID. Simply stating the transaction you are looking for might not find all profiles that apply to that transaction.

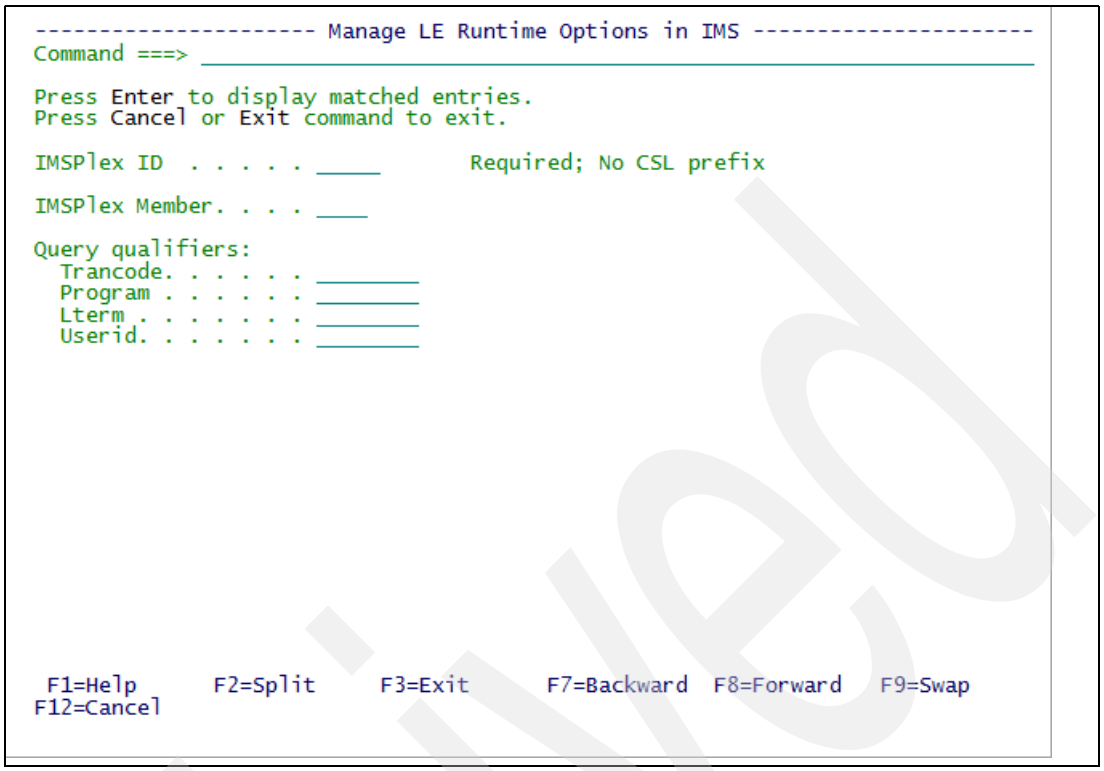


Figure 15-30 IMS PLEX ID

Figure 15-31 shows the LE runtime options profiles that match the given query parameters. In this example there is only one.

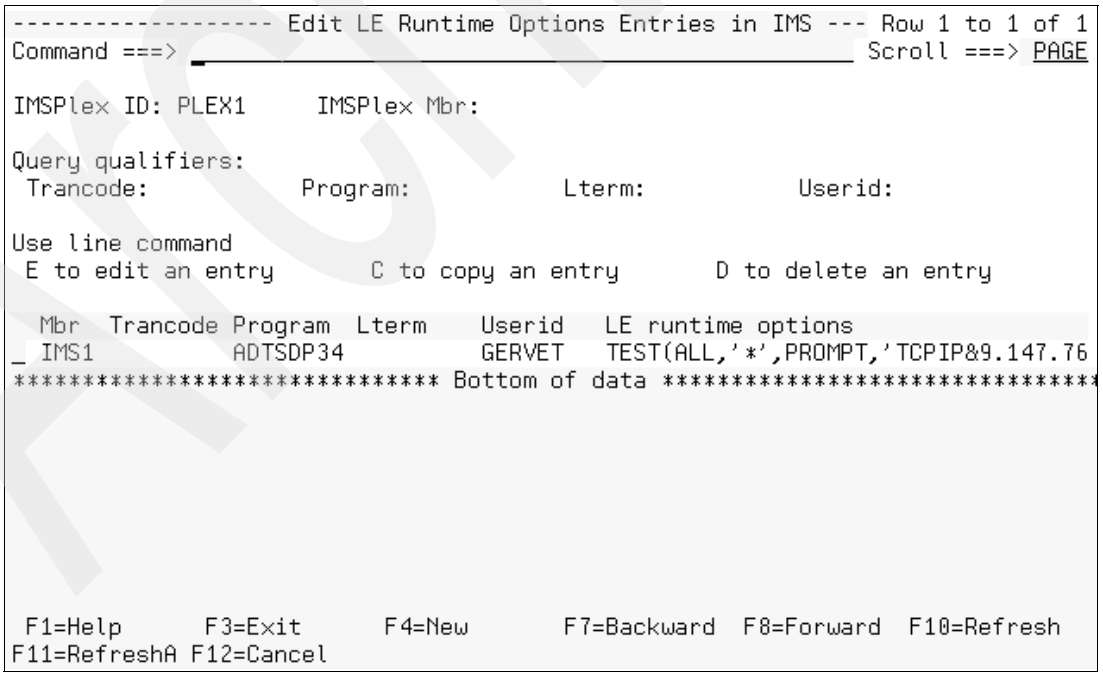


Figure 15-31 LE runtime options profiles

PF4 allows us to add a new profile. We will create one specifically for transaction TRXX01, as shown in Figure 15-32. This allows us to provide qualifiers to identify the transaction these runtime options should apply to. It is a long page. Press F8 to move forward through the fields. Do not attempt to put VTAM or remote debugger parameters in the top part of the page. They are collected in the bottom part of the panel.

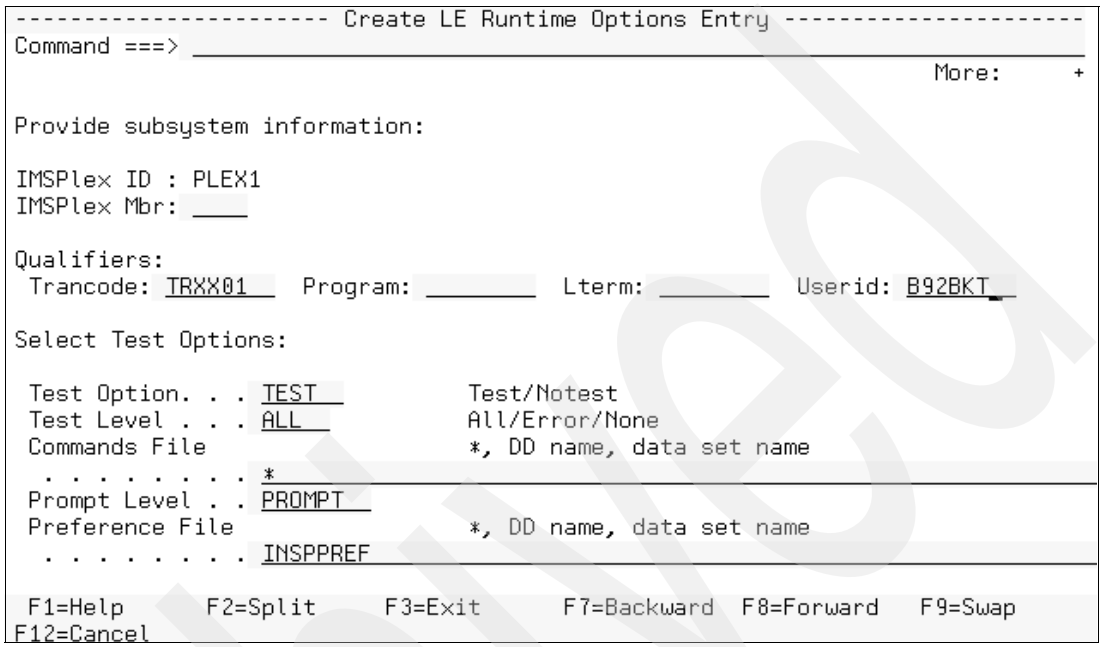


Figure 15-32 LE runtime option entry (part 1 of 2)

Press F8 to move forward and view the information shown in Figure 15-33.

The bottom half of the panel allows you to fill in the parameters to specify where debugging will occur:

- Select **Full-screen mode** if you are going debug using either full-screen mode through a VTAM terminal or the Terminal Interface Manager.
- Select **Remote debug mode** if you do not have an available VTAM terminal and you have installed one of the remote debuggers.

When all the data has been entered, press F3 (Exit) to create the new profile.

```
---------------------- Create LE Runtime Options Entry ----------------------
Command ===> ________________________________________________________________
                                                              More:    -
  . . . . . . . . *
 Prompt Level . . PROMPT__
 Preference File              *, DD name, data set name
  . . . . . . . . INSPPREF

Select (/) a session type and provide parameters:

 / Full-screen mode
   Terminal LU. . ________       MFI VTAM Terminal LU

 _ Remote debug mode
   Connection type. . ________   SINGLE/MULTIPLE socket
   Address. . . . . . __________________________________________________
   Port . . . . . . . __________

Other Language Environment run-time options: ______________________________


 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 15-33 Create LE runtime options (part 2 of 2)

Figure 15-34 shows the old and the new profile.

```
------------------- Edit LE Runtime Options Entries in IMS --- Row 1 to 2 of 2
Command ===> _______________________________________________ Scroll ===> PAGE

IMSPlex ID: PLEX1     IMSPlex Mbr:

Query qualifiers:
 Trancode:           Program:            Lterm:            Userid:

Use line command
 E to edit an entry       C to copy an entry       D to delete an entry

  Mbr  Trancode Program  Lterm    Userid   LE runtime options
_ IMS1          ADTSDP34          GERVET   TEST(ALL,'*',PROMPT,'TCPIP&9.147.76
_ IMS1 TRXX01                     B92BKT   TEST(ALL,'*',PROMPT,'MFI:INSPPREF')
******************************* Bottom of data *******************************







 F1=Help      F3=Exit      F4=New       F7=Backward  F8=Forward  F10=Refresh
F11=RefreshA F12=Cancel
```

Figure 15-34 Edit LE runtime options

The Edit LE runtime options entries in the IMS menu provide three activity choices for each created entry.

- E - Edit an entry to make some modification.

  Editing will bring you to the same screens shown in Figure 15-33 on page 389 and Figure 15-34. You can then make the required modifications.

- C - Copy an entry.

  Copying an entry will bring you to the same screens as Figure 15-33 on page 389 and Figure 15-34, where you can make modifications. An additional entry that incorporates your modifications is displayed in the next screen.

- D - Delete an entry that is no longer required.

  As shown in Figure 15-35, deleting an entry will produce an additional menu on top of the one shown in Figure 15-34 on page 389.

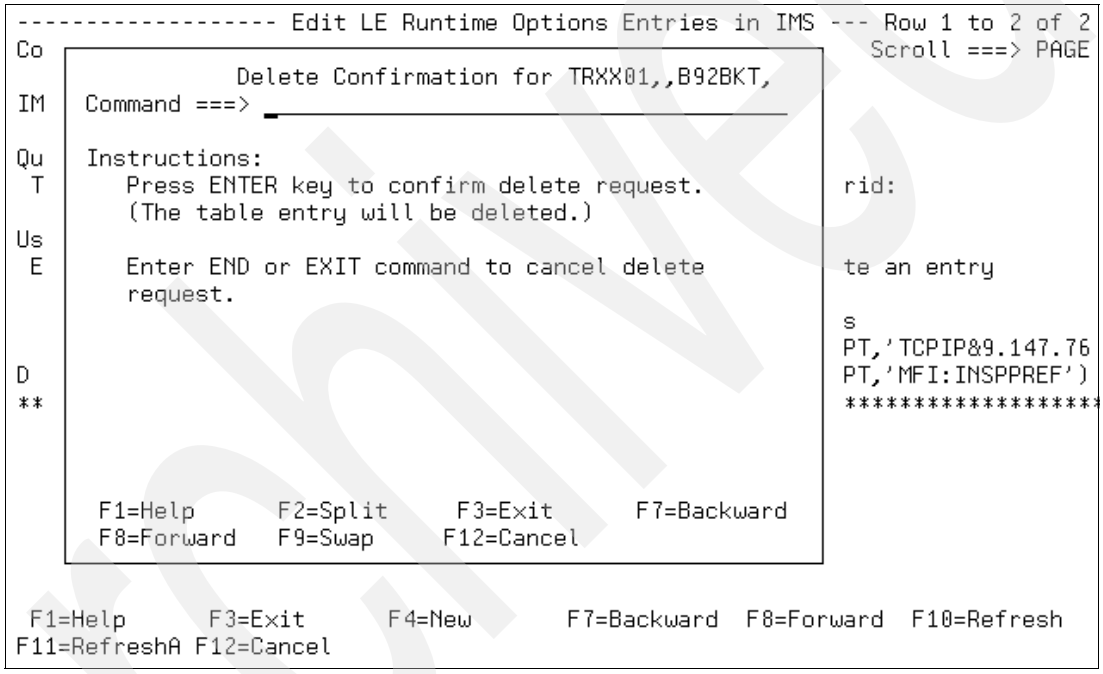


Figure 15-35 Deleting a profile entry

Press F3 twice to go back to the Manage IMS Programs menu, as shown in Figure 15-36. At this point, choose option **2**: Create Private Message Regions.

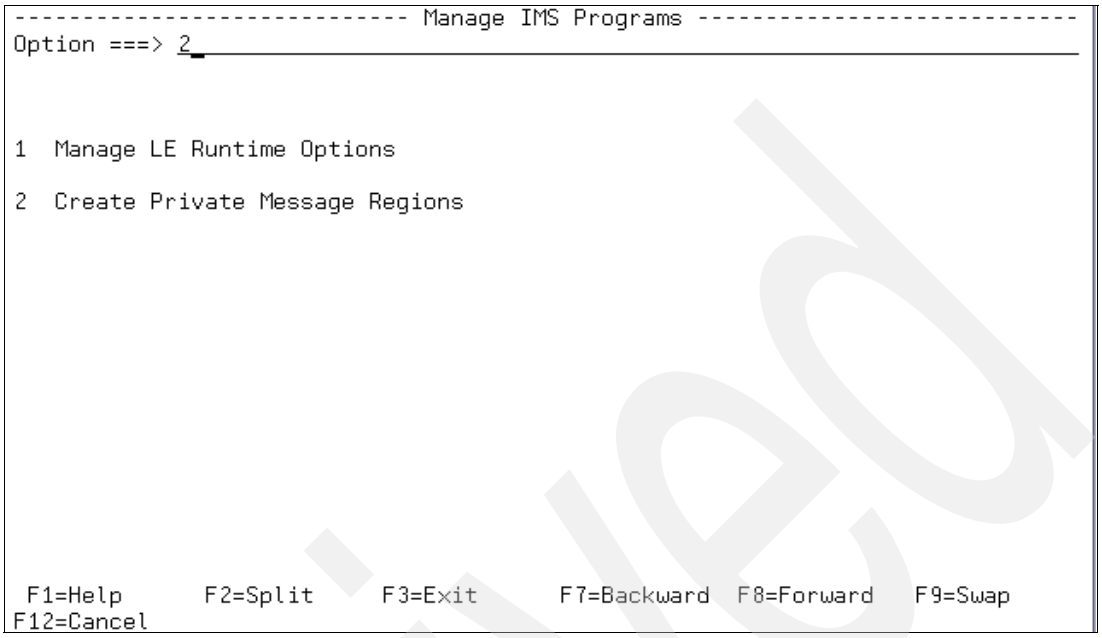


Figure 15-36 Manage IMS programs

We now create a new setup file, called IMSREG1, for our private message region. Specify your setup file as shown in Figure 15-37 and press Enter. You could optionally edit an existing member here by leaving the member name blank or by entering a selection pattern in it.

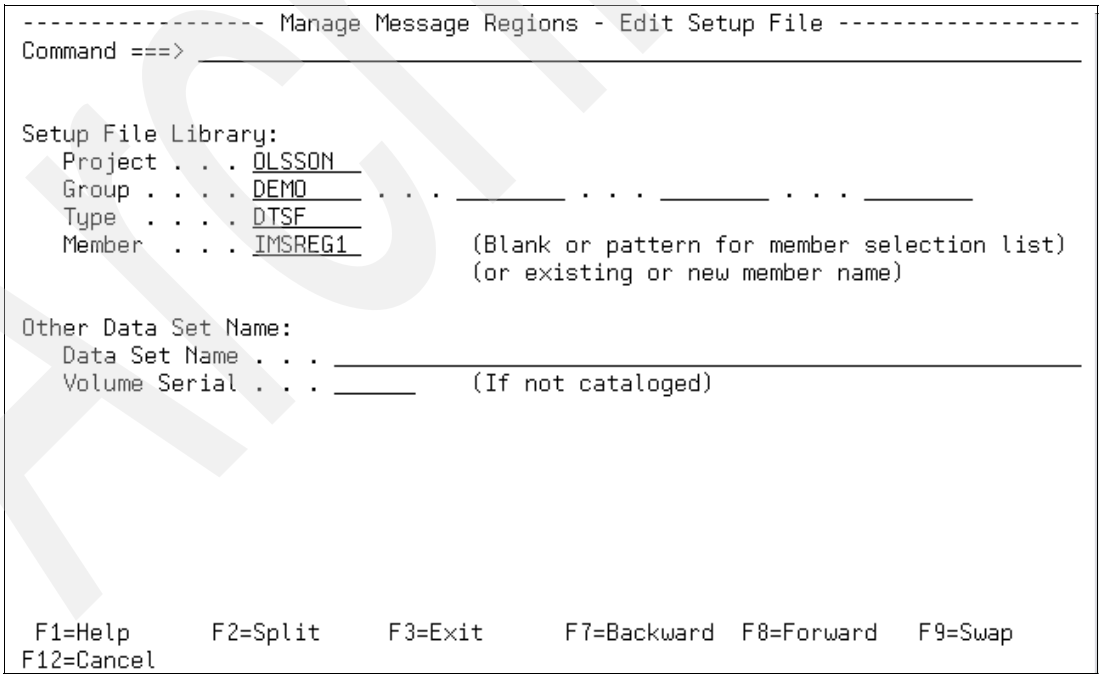


Figure 15-37 Manage message region - Edit Setup File

Figure 15-38 shows the Edit Setup panel. We enter COPY on the command line and press Enter.

```
Edit - Edit Setup File 'OLSSON.DEMO.DTSF(IMSREG1)'              Row 1 to 1 of 1
Command ===> COPY_____________________________________________ Scroll ===> PAGE

Modify information and use the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Batch Job Name. . . ________

Load Module Name. . ________

IMS Subsystem ID. .

_ Enter / to modify parameters


Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)           DISP
___ ________ ___ _ ****************** Top of Data ********************* _____
******************************** Bottom of data ********************************




 F1=Help      F3=Exit      F7=Backward  F8=Forward  F10=Submit   F12=Cancel
```

Figure 15-38 Edit Setup panel with COPY command

After entering the COPY command, Debug Tool Foreground Copy panel is displayed. Knowing the member we wish to copy, we enter the details, as shown in Figure 15-39. We could have left the member field blank or entered a wild card selection into it to get a list of members from which a specific selection could be made.

```
------------ Debug Tool Foreground - Copy from Setup File or JCL ------------
Command ===> ____________________________________________________________

Select data to copy into 'OLSSON.DEMO.DTSF(IMSREG1)'

Setup File or JCL Library:
   Project . . . OLSSON__
   Group . . . . DEMO___ . . . ________ . . . ________ . . . ________
   Type  . . . . CNTL___
   Member  . . . IMSSTRT        (Blank or pattern for member selection list)
                                (or existing or new member name)

Other Data Set Name:
   Data Set Name . . . ____________________________________________________
   Volume Serial . . . ______    (If not cataloged)


Note: When you copy from another setup file the entire contents are copied.
      When copying from JCL you can select the information you want to copy.



 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 15-39 Debug Tool Foreground Copy panel

The JCL for the selected member is shown in Figure 15-40. On the command line we enter S* to select all statements, then press Enter.

```
---------------- Debug Tool Foreground - Copy from JCL Datas Row 1 to 14 of 17
Command ===> S*_____________________________________________ Scroll ===> PAGE

Enter S* on the command line or on a Sel line to select all JCL statements.
Enter S on a Sel line to select that JCL statement.
Enter RESET to deselect all JCL statements.

Sel   JCL Image
____ //IMSMPP1 JOB
____ //* from 'IMS8ADT.PROCLIB (DFSMPR)'
____ //IMS8M11 EXEC PGM=DFSRRC00,
____ //  PARM=(MSG,001000000000,N00010000,,,,,6,5,IMS1,IVP,,,,DC,,Y,'',,)
____ //STEPLIB  DD  DSN=IMS8ADT.PGMLIB,DISP=SHR
____ //         DD  DSN=IMS8ADT.USERRESL,DISP=SHR
____ //         DD  DSN=IMS800A.SDFSRESL,DISP=SHR
____ //PROCLIB  DD  DSN=IMS8ADT.PROCLIB,DISP=SHR
____ //SYSUDUMP DD  SYSOUT=*,DCB=(LRECL=121,BLKSIZE=3129,RECFM=VBA),SPACE=(125,
____ //DFSCTL   DD  DISP=SHR,DSN=IMS8ADT.PROCLIB(DFSSBPRM)
____ //DFSSTAT  DD  SYSOUT=*
____ //SYSEXEC  DD  DISP=SHR,DSN=IMS8ADT.INSTALIB
____ //         DD  DISP=SHR,DSN=IMS800A.SDFSEXEC
____ //SYSTSIN  DD  *
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 15-40 JCL for the selected member

All JCL statements are selected, as shown in Figure 15-41. We press F3 to continue.

```
---------------- Debug Tool Foreground - Copy from JCL Datas Row 1 to 14 of 17
Command ===> _____________________________________________ Scroll ===> PAGE

Enter S* on the command line or on a Sel line to select all JCL statements.
Enter S on a Sel line to select that JCL statement.
Enter RESET to deselect all JCL statements.

Sel   JCL Image
S___ //IMSMPP1 JOB
____ //* from 'IMS8ADT.PROCLIB (DFSMPR)'
S___ //IMS8M11 EXEC PGM=DFSRRC00,
____ //  PARM=(MSG,001000000000,N00010000,,,,,6,5,IMS1,IVP,,,,DC,,Y,'',,)
S___ //STEPLIB  DD  DSN=IMS8ADT.PGMLIB,DISP=SHR
S___ //         DD  DSN=IMS8ADT.USERRESL,DISP=SHR
S___ //         DD  DSN=IMS800A.SDFSRESL,DISP=SHR
S___ //PROCLIB  DD  DSN=IMS8ADT.PROCLIB,DISP=SHR
S___ //SYSUDUMP DD  SYSOUT=*,DCB=(LRECL=121,BLKSIZE=3129,RECFM=VBA),SPACE=(125,
S___ //DFSCTL   DD  DISP=SHR,DSN=IMS8ADT.PROCLIB(DFSSBPRM)
S___ //DFSSTAT  DD  SYSOUT=*
S___ //SYSEXEC  DD  DISP=SHR,DSN=IMS8ADT.INSTALIB
S___ //         DD  DISP=SHR,DSN=IMS800A.SDFSEXEC
S___ //SYSTSIN  DD  *
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 15-41 JCL for the selected member

Figure 15-42 shows the newly created Setup File. We can now scroll forward and backward through the file, making changes as required. An asterisk (*) in the C column will comment out that line.

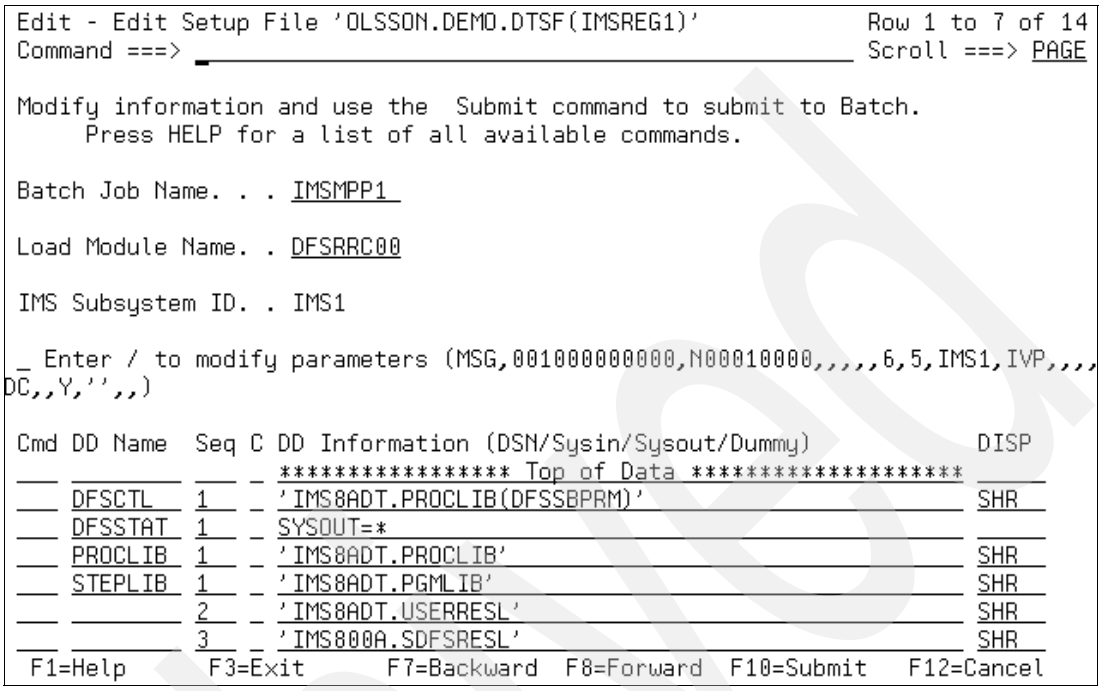


Figure 15-42 Newly created setup file

We now issue the SUBmit (PF10) command, as shown in Figure 15-43.

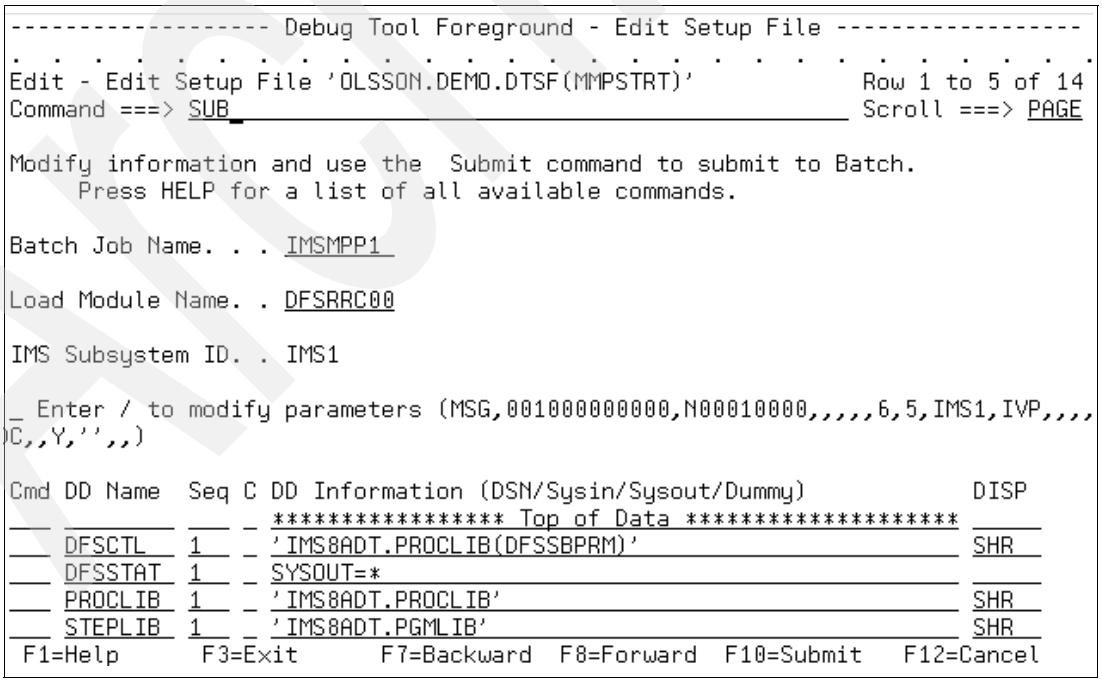


Figure 15-43 Newly created setup file

Figure 15-44 shows the JCL, ready for submission, that will create the new setup file for our private message region.

```
------------------- Debug Tool Foreground - Edit Setup File ------------------
.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
VIEW       OLSSON.DTSUTEMP.XJC2D022                      Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
=NOTE= Enter Submit to submit the JCL.
=NOTE= Press PF3 to exit.
=NOTE= -Warning- Changes will NOT be saved.
=NOTE=
000001 //IMSMPP1    JOB 'OLSSON     ',
000002 //    CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1),
000003 // NOTIFY=OLSSON,
000004 // REGION=32M
000005 //* Generated by DTSU
000006 //*
000007 //REGION EXEC PGM=DFSRRC00,
000008 // PARM='MSG,001000000000,N00010000,,,,,6,5,IMS1,IVP,,,,DC,,Y,'''',,'
000009 //DFSCTL DD DSN=IMS8ADT.PROCLIB(DFSSBPRM),DISP=(SHR,,)
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 15-44 JCL to create a new setup file

15.3.3 Precedence of the information about the LE Runtime Options panel

When using the DTU Language Environment (LE) runtime options support, two obvious questions arise.

If my program matches more than one profile, which one will be used?

The first match located is used. The list is searched in the order in which profiles are entered. Editing the entry does not change its position in the order.

Based on this, we recommend that this feature be used as little as possible for specifying runtime options other than TEST, and that all users on a given IMS plex have an agreed upon a way of setting up the profiles (perhaps requiring the user ID, or always stating the transaction). This facility is best used to simply specify the TEST runtime option. It should only be used to specify other runtime options as a limited time test of the given LE runtime options efficiency.

Where do these runopts fit in the LE runopt hierarchy?

This question concerns which runopts have precedence (panel, ceeuopt, ceeropt, ceedopt).

Language Environment has several ways of specifying the runtime options for a program. The following list gives them in ascending order of precedence (that is, things lower in the list can override things higher in the list):

- CEEDOPT: Language Environment options specified at installation
- CEEROPT: region-wide CICS or IMS default options
- CLER under CICS
- CEEUOPT: also #pragma runopts, or PLIXOPTS
- Command Line, or _CEE_RUNOPTS
- CEEBXITA
- Language Environment Storage Tuning User Exit
- Options defined at installation time that have the “non-overridable” attribute

The usual Language Environment runtime option merging occurs (that is, options or suboptions from a lower precedence entity are taken).

16

Debug Tool Utilities and Advanced Functions

In this chapter we describe the Coverage Utility, the Load Module Analyzer, and the Convert and Compile option of Debug Tool Utilities and Advanced Functions.

Debug Tool Coverage Utility (DTCU) enables you to determine which code statements have been executed and which statements have not been executed. You can use this information to decide whether to adapt the application's input data and parameters to get full test coverage. It supports all environments except UNIX System Services. In this chapter we emphasize measuring code coverage in Enterprise COBOL for z/OS and OS/390.

Debug Tool Load Module Analyzer (LMA) analyzes MVS load modules or program objects to determine the language translator (compiler or assembler) used to generate the object code for each CSECT. It can process all or selected load modules or program objects in a concatenation of PDS and PDSE data sets.

The Convert and Compile function included with Debug Tool Utilities and Advanced Functions allows you to convert and compile older COBOL programs such as:

- Programs written for the OS/VS COBOL compiler (68/74 Standard COBOL).
- Programs written for the VS COBOL II (or later) compiler that was last compiled with the CMPR2 compiler option (74/85 Standard).

The conversion creates code that is suitable for compiling with ENTERPRISE COBOL for z/OS (5648-G53). After compilation, you can debug your converted COBOL source code with Debug Tool using the object module created from the new compiler.

16.1 Coverage Utility

To effectively use DTCU (Debug Tool Coverage Utility), you must customize it first.

16.1.1 DTCU customization

This section describes:

- Where to place certain Coverage Utility load modules so that the correct people have access to them
- How to install the monitor SVCs
- How to customize the product

Make the load modules in hlq.SEQAMOD accessible to all users

Certain Coverage Utility load modules must be placed in an APF-authorized data set that is accessible to all your users. The APF-authorized data set does not have to be in the link list. Perform the following steps:

1. Mark the hlq.SEQAMOD data set as APF-authorized and make it accessible to all users or copy the following load modules to an APF-authorized data set that all users can access:
   - EQACUOCM (monitor interface)
   - EQACU9M0 (monitor messages)
   - EQACU9M1 (monitor messages)
   - EQACU9M2 (monitor messages, and only if the JPN feature is installed)
2. Add the EQACUOCM program to the AUTHPGM entry in the member IKJTSOxx of the SYS1.PARMLIB data set.
3. Issue the PARMLIB UPDATE(xx) command from TSO or IPL your system.
4. Edit the EQASTART member of the hlq.SEQAEXEC data set and set the INST_Auth_SEQAMOD variable to the name of the data set from step 1 that contains EQACUOCM.

Placing load modules in an APF data set not accessible to general users

Certain Coverage Utility load modules must be placed in an APF-authorized data set that is accessible only to system programmers. The APF-authorized data set must not be in the link list. Do one of the following:

- Mark the hlq.SEQAAUTH data set as APF-authorized and limit access to these load modules.
- Do not mark the hlq.SEQAAUTH data set as APF-authorized. Copy the following load modules into an APF-authorized data set that only system programmers can access:
  - EQACUOIN (SVC installer)
  - EQACUOSV (SVCs)

Installing and enabling the monitor SVCs

The EQACUOIN module installs and enables the monitor SVCs. The monitor SVCs must be installed and enabled before a user starts a monitor session. The EQACUOIN module must be run:

- When the SVCs are initially installed
- After service is applied
- Any time you IPL your system

The monitor SVCs use a common system storage, as described later. In addition, each user session uses ECSA storage:

- CSA: 13248 bytes
- SQA: 25496 bytes

To perform the initial installation:

1. Reserve two free user SVC numbers. IEASVCxx does not have to be updated, because these user SVCs are installed dynamically.
2. Customize the EQACUOIS member of the hlq.SEQASAMP data set.
3. Submit EQACUOIS on the system on which you intend to run the monitor.
4. To verify that the monitor was installed properly, run the following command from ISPF 6:

```
ex 'hlq.SEQAEXEC(EQACUOSE)' 'LEVEL'
```

Verify that the monitor release is V7R1M0 and the date is 2002.245 or later, as shown in Example 16-1.

Example 16-1 EQACUOSE reply

```
******************************** Top of Data ********************************
Monitor Release: V7R1M0 Date: 2002.245
MAST: 00FC0620 PSA: 00FA9000 CPU: 00000000 SEST: 00FC0308 UNID: 00000000
******************************** Bottom of Data *****************************
```

To have the monitor SVCs installed automatically after a system IPL, make the following changes:

1. Add COM='S EQACUOIN' to the COMMNDxx member of the SYS1.PARMLIB data set.
2. Copy and customize hlq.SEQASAMP(EQACUOPS) to your SYS1.PROCLIB data set as member EQACUOIN.
3. The process started by EQACUOIN must have access to the EQACUOIN and EQACUOSV load modules.

Customizing the product defaults

Complete the following steps to edit hlq.SEQAEXEC(EQACUDFT):

1. Change all occurrences of EQAW to hlq.
2. In the execute step data entry for EXEJOBLB, change the name hlq.SEQAMOD to the name of the APF-authorized data set you used for the EQACUOCM program.
3. Enter the Coverage Utility Monitor SVC numbers (in hexadecimal notation) in the CUSVC2B and CUSVC4B entries.
4. When you create JCL, the *JOBLn lines become the first three lines of the JOB card for each respective job. Customize these lines.
5. If your site requires a specification for allocation parameters such as STORCLAS or UNIT on new or temporary data set allocations, look for the word SPACE in this EXEC and the 'hlq.SEQAS*' data sets and update the allocation specifications.
6. If you want Coverage Utility to generate or build each data set as sequential or partitioned, set the USEPRGNM variable to Y. To generate a data set as sequential, set the DSORG variable to SEQ. To generate a data set as partitioned, set the DSORG variable to PDS.
7. If you do not want Coverage Utility to generate or build any data set names automatically, set the USEPRGNM variable to N.

16.1.2 DTCU requirements

For coverage analysis, DTCU requires:

- The listings created by the compilers and the assemblers. These products offer options that enable you to include assembler statements in the listings. These assembler statements in the listing are required by DTCU.
- The object module or load modules. DTCU creates copies with inserted breakpoints.

16.1.3 Running DTCU

To run DTCU, perform the following steps:

1. Setup:
   a. Compile the source code using required compile or assembler options.
   b. Generate DTCU JCL using the DTCU ISPF dialog:
      i. Edit the DTCU control file.
      ii. Create the setup JCL.
      iii. Create the start monitor JCL.
      iv. Create the report or summary JCL.
   c. Edit the link-edit JCL to include the modified object.
   d. Edit the program invocation JCL to point to the modified load module.
2. Execution:
   a. Run the setup JCL.
   b. Run the link-edit JCL.
   c. Run the JCL to start a monitor session.
   d. Run your application using the modified load modules.
   e. Stop the monitor session.
3. Report: Run the report or summary JCL.

Figure 16-1 shows a diagram of the entire process.

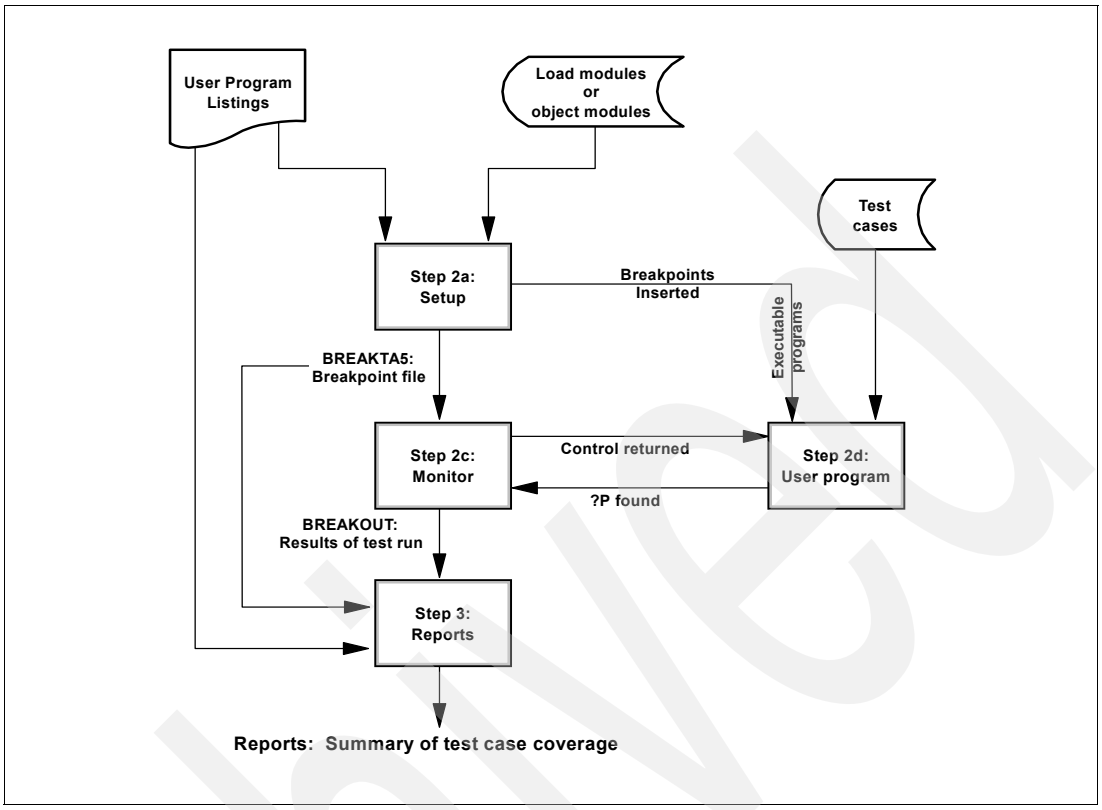


Figure 16-1 DTCU flow

16.1.4 Using DTCU

DTCU comes with an ISPF user interface where you can start all preparation steps required, as follows:

1. Set up the DTCU defaults.
2. Set up your control file.
3. Create the JCL for the setup.
4. Create the JCL to start the monitor.
5. Create JCL for a report.

Starting the DTCU ISPF dialog

If an option was installed to access Debug Tool Utilities from an existing ISPF panel, then select that option.

If Debug Tool Utilities were not installed in your ISPF environment, enter the following command from ISPF option 6:

```
EXEC 'hlq.SEQAEXEC(EQASTART)'
```

Next, select option **3** for DTCU. The next panel you see is Debug Tool Coverage Utility, as shown in Figure 16-2.

16.1.5 Modifying your DTCU defaults

Select option **0** (Defaults) from Debug Tool Coverage Utility panel to specify your Coverage Utility user default. The Manipulate Defaults panel is displayed, as shown in Figure 16-3.

```
------------------------ Debug Tool Coverage Utility -------------------------
Option ===> ____________________________________________________________

0  Defaults      Manipulate defaults
1  CntlFile      Work with the Control File
2  Setup         Create JCL for Setup
3  StartMon      Create JCL to Start the Monitor
4  Reports       Create Reports
5  Monitor       Control the Monitor
6  FastPath      FastPath

Enter X to Terminate
```

Figure 16-2 DTCU - initial panel

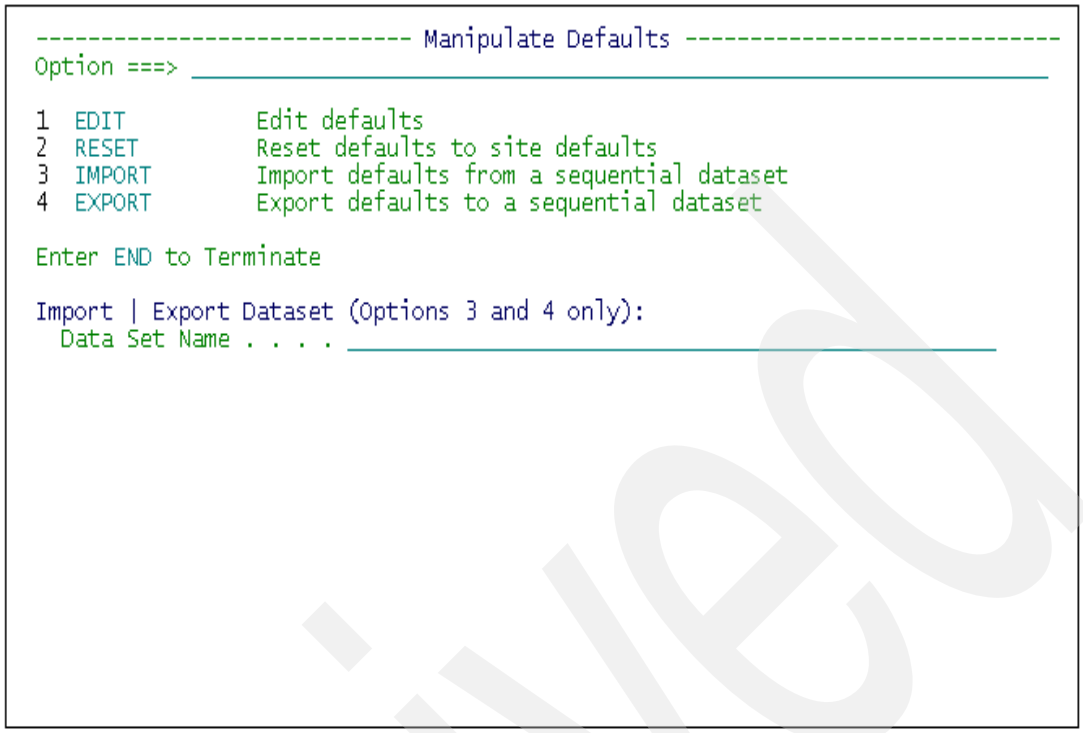


Figure 16-3 DTCU - Manipulate Defaults

To edit your user defaults, perform these steps:

1. Select option **1** on the Manipulate Defaults panel. The scrollable Edit Defaults panel is displayed, as shown in Figure 16-4 (use F7 and F8 to navigate).

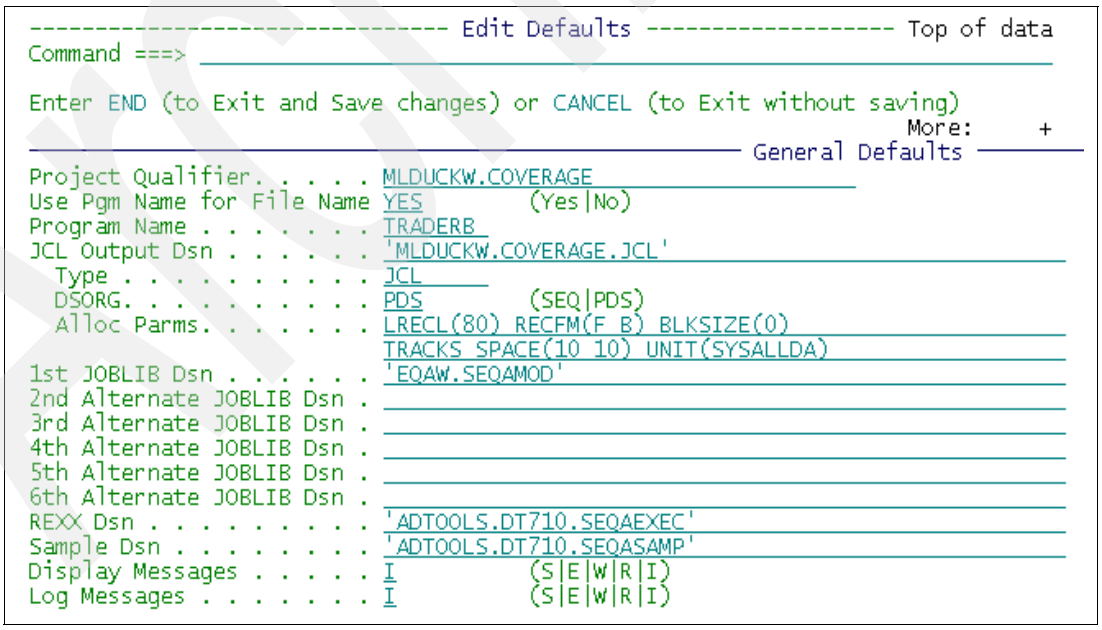


Figure 16-4 DTCU - Edit Defaults

2. Change the Project Qualifier value to the high-level qualifier that you want the Coverage Utility to use to construct names for user and project data sets.
3. To generate or build any data set names automatically, ensure that the Use Pgm Name For File Name is set to yes. The Coverage Utility uses the project qualifier, the program name, and the specified values for type and DSORG for each data set to build names of the following forms:
   - Sequential data sets:

     ```
     'proj_qual.program_name.file_type'
     ```

   - Partitioned data sets:

     ```
     'proj_qual.file_type(program_name)'
     ```

   If you specify no in the Use Pgm Name For File Name field, the Coverage Utility does not automatically build or change any data set names.

Once done, you can save (export) your defaults. Later, you can either reset the defaults to your site defaults, or you can import the defaults that you have exported.

16.1.6 Compiling the program to be measured

To compile your source code, do these steps:

1. Edit your compile JCL.
2. Run the JCL.
3. Make sure that all steps run with a return code of zero and that the listings and object modules are created.

We used our program TRADERB, and made sure that the compile time option LIST was on. This option is available for all compilers. For the required compile time options, refer to *Coverage Utility User's Guide and Messages V6R1*, SC18-9535. The SYSLIN output from the compiler, as well as the compiler listings, are available as members of partitioned data sets.

Example 16-2 is a sample for the compile JCL we used.

Example 16-2 Sample JCL for code coverage

```
000007 //COMPILE  EXEC PGM=IGYCRCTL,
000008 //            PARM='NOTEST,SOURCE,LIST,OBJECT,NOOPT,NONUMBER,LIB'
000009 //STEPLIB   DD DSN=IGY.V3R4M0.SIGYCOMP,DISP=SHR
000010 //SYSLIN    DD DSN=CHABERT.BOOK2005.COBOBJ(&MBR),DISP=SHR
000011 //SYSLIB    DD DSN=CHABERT.TRADER.COPYLIB,DISP=SHR
000012 //SYSDEBUG  DD DSN=CHABERT.BOOK2005.SIDEFILE(&MBR),DISP=SHR
000013 //SYSPRINT  DD DSN=CHABERT.BOOK2005.COBLIST(&MBR),DISP=SHR
000014 //SYSUT1    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
000015 //SYSUT2    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
000016 //SYSUT3    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
000017 //SYSUT4    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
000018 //SYSUT5    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
000019 //SYSUT6    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
000020 //SYSUT7    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
000021 //SYSIN     DD DSN=CHABERT.BOOK2005.SOURCE(&MBR),DISP=SHR
```

16.1.7 Setting up your control file

The Coverage Utility uses assembler statements from the compiler listings to determine where to insert breakpoints. You supply the names of the listing files in the Coverage Utility control file.

Select **1** (CntlFile) from the main DTCU panel. The control file data set name is based on the defaults you have defined. In the control file, describe the compile units that you want to be analyzed. In our test case, the control file name is 'userid.COVERAGE.DTCU(TRADERB)'. Select 1 (Edit).

The data in the control file consists of the following information:

- The type of listing file (COBOL)
- The names of the listing files for the programs that you want to test
- The names of the load modules that contain the code of each listing
- The copy to or from information for making copies of the object modules into which the breakpoints are inserted

We wanted to make this discussion easy to follow, so our TRADERB is very simple, as shown in Figure 16-5.

```
  Menu  Utilities  Compilers  Help
 BROWSE    ABAPHEN.COVERAGE.DTCU(TRADERB) - 01.02     Line 00000000 Col 001 080
 Command ===>                                                Scroll ===> CSR
****************************** Top of Data ********************************
*
* This is an example DTCU control file that was copied from
* your system's SEQASAMP library. Make appropriate edits for your testing.
*
* COBOL Example
*
* Statements required for coverage
*
       Defaults ListDSN=CHABERT.book2005.COBLIST(*),
                LoadMod=TRADERB,
                FromObjDSN=ABAPHEN.book2005.COBOBJ,
                ToObjDSN=ABAPHEN.book2005.DTCUOBJ

       COBOL ListMember=TRADERB
****************************** Bottom of Data ******************************
```

Figure 16-5 DTCU control file

The last line, `COBOL ListMember=TRADERB`, defines what COBOL program is prepared and measured. If you have an application with several compile units, add a line for each of those compile units that you want to be analyzed. For example:

```
COBOL ListMember=TRADERB
PLI Listmember=TRADPLI
```

16.1.8 Creating the setup JCL

When you run the setup JCL, the Coverage Utility setup program analyzes the assembler statements in the compiler listings and creates a table that contains breakpoint data (address, op code, and so on). Breakpoints are inserted into the instrumented object modules or load modules. If you instrumented object modules, you then link these modified object modules into a modified sample load module for the Coverage Utility to use.

Select option **2** (Setup) from the main DTCU panel and Figure 16-6 is displayed.

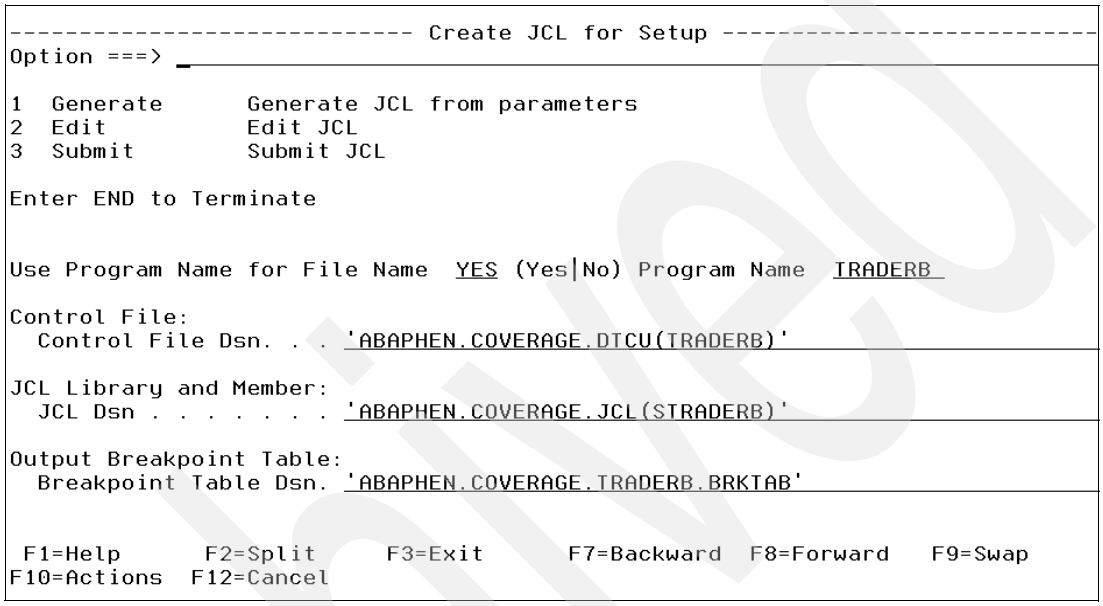


Figure 16-6 Create JCL for Setup

Select option **1** (Generate). Informational messages are written to your screen as the JCL is created. The created JCL is put into the JCL library that is identified on the panel by the member name *Sprogname.*

The generated JCL is used to create a new object, which has breakpoints inserted based on the assembler code (because of the LIST compile time option) found in the compiler listings.

16.1.9 Creating JCL to start a monitor session

JCL is required to start a Coverage Utility monitor session.

To create the JCL to start a monitor session, do these steps:

Select option **3** (StartMon) from Debug Tool Coverage Utility panel and the Create JCL to Start the Monitor panel is displayed, as shown in Figure 16-7.

If necessary, change the program name.

Select option **1** (Generate). Informational messages are written to your screen as the JCL is created. The created JCL is put into the JCL library that is identified on the panel by the member name X*progname.*

```
----------------------- Create JCL to Start the Monitor -----------------------
Option ===> _____________________________________________________________________

1  Generate      Generate JCL from parameters
2  Edit          Edit JCL
3  Submit        Submit JCL

Enter END to Terminate


Use Program Name for File Name  YES (Yes|No) Program Name  TRADERB

Session ID  . . . . . . ABAPHEN

Input File:
  Breakpoint Table Dsn. 'ABAPHEN.COVERAGE.TRADERB.BRKTAB'

JCL Library and Member:
  JCL Dsn . . . . . . . 'ABAPHEN.COVERAGE.JCL(XTRADERB)'

Output File:
  Breakout Dsn. . . . . 'ABAPHEN.COVERAGE.TRADERB.BRKOUT'
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F10=Actions  F12=Cancel
```

Figure 16-7 Create JCL to start the monitor

16.1.10 Creating JCL for a report

JCL is required to generate a summary report. To create the summary report JCL:

1. Select option **4** (Reports) from Debug Tool Coverage Utility panel. The Create JCL for Reports panel is displayed.
2. Select option **1** (Summary). The Create JCL for Summary Report panel in displayed, as shown in Figure 16-8. Create the JCL for generating the sample summary report from this panel.

   You can also use option **2** (Annotation) to create JCL for summary and annotation reports.

```
------------------------ Create JCL for Summary Report ------------------------
Option ===> _____________________________________________________________________

1  Generate      Generate JCL from parameters
2  Edit          Edit JCL
3  Submit        Submit JCL

Enter END to Terminate
                                                                More:     +
Use Program Name for File Name  YES (Yes|No) Program Name  TRADERB

Input Files:
  Breakpoint Table Dsn. 'ABAPHEN.COVERAGE.TRADERB.BRKTAB'
  Breakout Dsn. . . . . 'ABAPHEN.COVERAGE.TRADERB.BRKOUT'

JCL Library and Member:
  JCL Dsn . . . . . . . 'ABAPHEN.COVERAGE.JCL(TTRADERB)'

Output Summary Type and File:
  Type. . . . . INTERNAL (Internal|External)
  Inline  . . . . . N        (I|N)
  Report Dsn  . . . . . 'ABAPHEN.COVERAGE.TRADERB.SUMMARY'
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F10=Actions  F12=Cancel
```

Figure 16-8 Create JCL for summary report

3. Select option **1** (Generate). Informational messages, as shown in Example 16-3, are written to your screen as the JCL is created. The created JCL is put into the JCL library that is identified on the panel by the member name T*progname*.

Example 16-3 Sample information messages from create JCL for summary report

```
EQACU137I Summary is starting
 EQACU138I Summary is verifying your parameters
 EQACU162I Performing File Tailoring EQACUSUM 'ABAPHEN.COVERAGE.JCL(TTRADERB)'
 EQACU139I Summary is done
 ***
```

16.1.11 Jobs generated so far

After you have generated the JCL, there are three jobs in your JCL data set, as shown in Figure 16-9.

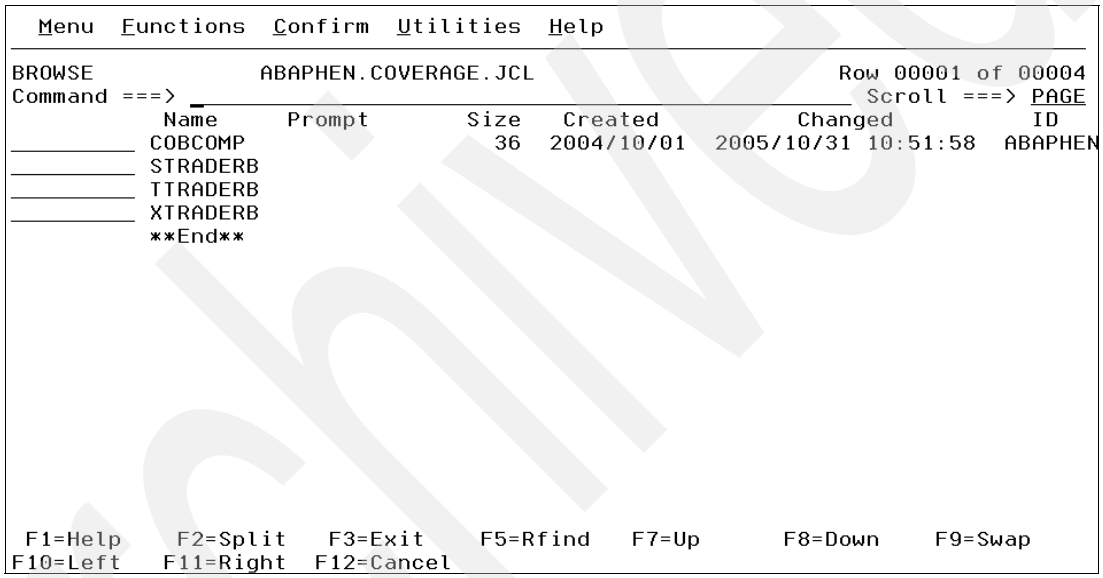


Figure 16-9 Jobs after successful JCL creation

The three jobs are named with your program name, prefixed by:

- S: JCL for the setup step
- T: JCL for the summary report
- X: JCL to start up the monitor session

Of course, you could add the JCL for the compile of your modules, and a job to link-edit the modified object. For both, there are samples available in the hlq.SEQASAMP.

Next we start the coverage measurement process.

16.1.12 Performing the setup step

Submit the JCL (S*progname*) generated by the Create JCL for Setup step.

The job should end with RC=0. After that, you can find two new data sets. In our sample they were called:

- CHABERT.COVERAGE.TRADERB.BRKTAB (a file of breakpoint-related information that can be reused later when the monitor is started and during the report printing)
- CHABERT.TRADER.DTCUOBJ (the modified object, which is linked as a regular application)

16.1.13 Performing the link step

This is a regular link-edit job, as shown in Example 16-4.

Example 16-4 Sample link JCL

```
//LINK     EXEC PGM=HEWL,PARM='MAP'
//SYSLIN    DD DSN=ABAPHEN.BOOK2005.DTCUOBJ(TRADERB),DISP=SHR
//SYSLIB    DD DSN=CEE.SCEELKED,DISP=SHR
//SYSPRINT  DD DSN=ABAPHEN.BOOK2005.COBLIST(TRADERB),DISP=SHR
//SYSUT1    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSLMOD   DD DSN=ABAPHEN.BOOK2005.LOAD(TRADERB),DISP=SHR
//*
```

After the link, you have a regular load module ready to be executed. Before you do that, you must start the monitor session.

16.1.14 Starting the monitor session

For this, submit the JCL (Xprogname) generated by the Create JCL to start a monitor session step.

16.1.15 Starting your application

Although there is some JCL provided in the SEQASAMP (member EQACU1G9), you can use your original JCL. Eventually, the program ends with RC, so now you can look for the output.

16.1.16 Displaying statistics online

This is an optional step.

From ISPF option 6, run the command EXEC hlq.SEQAEXEC(EQACUOSA).

You get some output depending on the number of source lines. This is only recommended for small test cases. An example is shown in Figure 16-10.

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
 BROWSE    ABAPHEN.MSGS.FILE                          Line 00000000 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
******************************** Top of Data ********************************
Num Listing                                  Date   Time     PAs   BPs
------------------------------------------------------------------------------
001 CHABERT.BOOK2005.COBLIST(TRADERB)        05.304 11:22.30 00039 000422
PA    ADR      BPS    EVNTS      ACTVE
------------------------------------------------------------------------------
00001 00000000 000019 0000000000 000019
00002 00000000 000012 0000000000 000012
00003 00000000 000001 0000000000 000001
00004 00000000 000023 0000000000 000023
00005 00000000 000030 0000000000 000030
00006 00000000 000001 0000000000 000001
00007 00000000 000028 0000000000 000028
00008 00000000 000001 0000000000 000001
00009 00000000 000034 0000000000 000034
00010 00000000 000001 0000000000 000001
00011 00000000 000007 0000000000 000007
00012 00000000 000001 0000000000 000001
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 16-10 Output from the EQACUOSA command

For each program area, the following fields are displayed:

Num The sequential number of the listing

Listing The name of the listing data set

Date The date of the compile

Time The time of the compile

PAs The number of program areas in the listing

BPs The number of break points in the listing

PA The sequential number of the program area in the listing

ADR When the program area has run, the storage address of the program area

BPS The number of break points that have been executed for the program area

EVNTS (events) The number of break points that have executed for this program area

ACTVE (active) The number of break points that are still in storage in the program area

16.1.17 Stopping the monitor

From ISPF option 6, run the command EXEC hlq.SEQAEXEC(EQACUOSP).

Example 16-5 is a sample reply.

Example 16-5 Output from the EQACUOSP command

```
******************************** Top of Data ********************************
  Monitor session ABAPHEN  stopped - session data written to disk
******************************* Bottom of Data ******************************
```

16.1.18 Creating the summary report

For this example, we submit the JCL (Tprogname) generated by the Create JCL for Reports step.

First part of the summary report

Example 16-6 is the first part of the summary report.

Example 16-6 DTCU - summary report (part 1 of 2)

```
*********  DTCU SUMMARY:                                          PROGRAM AREA DATA                    ********
              DATE: 11/14/2005
              TIME: 14:21:11
       TEST CASE ID:
|<--                    PROGRAM IDENTIFICATION                  -->|
|                          |                                       |  STATEMENTS:          | BRANCHES:          |
| PA LOAD MOD PROCEDURE    | LISTING NAME                          |   TOTAL     EXEC   %  |  CPATH   TAKEN   %  |
-------------------------------------------------------------------------------------------------------------------------
   1 TRADERB  MAINLINE       CHABERT.TRADER.COBLIST(TRADERB)          12        0   0.0
   2          SETUP-FILES                                           10        0   0.0
   3          SETUP-FILES-EXIT                                       1        0   0.0
   4          CLOSEDOWN-FILES                                       19        0   0.0
   5          READ-TRANSACT-FILE                                    14        0   0.0
   6          READ-TRANSACT-FILE-EXIT                                1        0   0.0
   7          BUY-SELL                                              10        0   0.0
   8          BUY-SELL-EXIT                                          1        0   0.0
   9          BUY-SELL-BUY-FUNCTION                                 12        0   0.0
  10          BUY-SELL-BUY-FUNCTION-EXIT                             1        0   0.0
  11          CALCULATE-SHARES-BOUGHT                                5        0   0.0
  12          CALCULATE-SHARES-BOUGHT-EXIT                           1        0   0.0
  13          CALCULATE-SHARES-SOLD                                  1        0   0.0
  14          CALCULATE-SHARES-SOLD-EXIT                             1        0   0.0
  15          BUY-SELL-SELL-FUNCTION                                14        0   0.0
  16          BUY-SELL-SELL-FUNCTION-EXIT                            1        0   0.0
  17          VALIDATE-COMPANY-EXISTS                                2        0   0.0
  18          VALIDATE-COMPANY-EXISTS-EXIT                           1        0   0.0
  19          GET-SHARE-VALUE                                        9        0   0.0
  20          READ-CUSTFILE                                         15        0   0.0
  21          READ-CUSTFILE-EXIT                                     1        0   0.0
  22          WRITE-CUSTFILE                                        15        0   0.0
  23          WRITE-CUSTFILE-EXIT                                    1        0   0.0
  24          REWRITE-CUSTFILE                                      12        0   0.0
  25          REWRITE-CUSTFILE-EXIT                                  1        0   0.0
  26          READ-COMPFILE                                         13        0   0.0
  27          READ-COMPFILE-EXIT                                     1        0   0.0
  28          BUILD-NEW-CUSTOMER                                    13        0   0.0
  29          BUILD-NEW-CUSTOMER-EXIT                                1        0   0.0
  30          SET-DUMMY-CUST-RECORD                                  7        0   0.0
  31          GENERATE-CUSTOMER-REPORT                               9        0   0.0
  32          GENERATE-CUSTOMER-REPORT-EXIT                          1        0   0.0
  33          CALCULATE-SHARE-VALUE                                  8        0   0.0
  34          START-CUSTFILE                                         8        0   0.0
  35          READ-CUSTFILE-NEXT                                     8        0   0.0
  36          WRITE-HEADER                                           7        0   0.0
  37          WRITE-DETAILS                                          5        0   0.0
  38          WRITE-TRANSACTION-REPORT                              15        0   0.0
  39          WRITE-TRANSACTION-REPORT-EXIT                          1        0   0.0
-------------------------------------------------------------------------------------------------------------------------
Summary for all PAs:                                              258        0   0.0       0       0 100.0
```

The summary report tells you how much of the procedures (blocks) has been executed. If a procedure shows 100%, it means that all the statements have been executed at least one time. If a block or procedure shows less then 100%, it tells you how many are in that block, and how many statements have been executed.

Each section of the report includes a program identification area. This information gives the load module name, procedure name, and listing name for the program.

The columns in the PROGRAM IDENTIFICATION area are:

PA The number of the program area.

LOAD MOD The name of the load module.

PROCEDURE COBOL: Paragraph name.

PL/I: Procedure, ON-unit, or Begin-block name.

C/C++: Function name.

ASM: CSECT name.

LISTING NAME The name of the listing. If the listing name is longer than 40 characters, only the right-most 40 characters are shown.

The section of the report called PROGRAM AREA DATA contains coverage statistics in addition to the program identification information. The columns in the coverage statistics area are:

STATEMENTS: TOTAL The statements of code for this test case run
STATEMENTS: EXEC The statements of code that executed
STATEMENTS: % The percentage of statements that executed
BRANCHES: CPATH The number of conditional branch paths
BRANCHES: TAKEN The number of conditional branch paths that executed
BRANCHES: % The percentage of conditional branch paths that executed

Second part of the summary report

This shows exactly what statements in their correspondence blocks have not been executed, as shown in Example 16-7.

Example 16-7 DTCU - summary report (part 2 of 2)

```
 *********  DTCU SUMMARY:                  UNEXECUTED CODE                  ********
             DATE: 11/14/2005
             TIME: 14:21:11
       TEST CASE ID:
|<--                    PROGRAM IDENTIFICATION                      -->|           UNEXECUTED CODE
|                          |                                           |
| PA LOAD MOD PROCEDURE    | LISTING NAME                              | start   end     start   end     start   end
-------------------------------------------------------------------------------------------------------------------------
   1 TRADERB  MAINLINE         CHABERT.TRADER.COBLIST(TRADERB)            328    346
   2          SETUP-FILES                                                 352    377
   3          SETUP-FILES-EXIT                                            381    381
   4          CLOSEDOWN-FILES                                             384    409
   5          READ-TRANSACT-FILE                                          417    439
   6          READ-TRANSACT-FILE-EXIT                                     444    444
   7          BUY-SELL                                                    448    460
   8          BUY-SELL-EXIT                                               464    464
   9          BUY-SELL-BUY-FUNCTION                                       472    489
  10          BUY-SELL-BUY-FUNCTION-EXIT                                  493    493
  11          CALCULATE-SHARES-BOUGHT                                     499    506
  12          CALCULATE-SHARES-BOUGHT-EXIT                                510    510
  13          CALCULATE-SHARES-SOLD                                       515    515
  14          CALCULATE-SHARES-SOLD-EXIT                                  519    519
  15          BUY-SELL-SELL-FUNCTION                                      526    547
  16          BUY-SELL-SELL-FUNCTION-EXIT                                 551    551
  17          VALIDATE-COMPANY-EXISTS                                     554    555
  18          VALIDATE-COMPANY-EXISTS-EXIT                                558    558
  19          GET-SHARE-VALUE                                             561    573
  20          READ-CUSTFILE                                               580    598
```

```
21          READ-CUSTFILE-EXIT                                  602    602
22          WRITE-CUSTFILE                                      606    623
23          WRITE-CUSTFILE-EXIT                                 627    627
24          REWRITE-CUSTFILE                                    631    644
25          REWRITE-CUSTFILE-EXIT                               648    648
26          READ-COMPFILE                                       651    666
27          READ-COMPFILE-EXIT                                  670    670
28          BUILD-NEW-CUSTOMER                                  675    687
29          BUILD-NEW-CUSTOMER-EXIT                             690    690
30          SET-DUMMY-CUST-RECORD                               695    701
31          GENERATE-CUSTOMER-REPORT                            706    718
32          GENERATE-CUSTOMER-REPORT-EXIT                       725    725
33          CALCULATE-SHARE-VALUE                               728    737
34          START-CUSTFILE                                      742    752
35          READ-CUSTFILE-NEXT                                  757    766
36          WRITE-HEADER                                        771    777
37          WRITE-DETAILS                                       781    785
38          WRITE-TRANSACTION-REPORT                            789    809
39          WRITE-TRANSACTION-REPORT-EXIT                       811    811
-------------------------------------------------------------------------------------------------------------------
```

The section of the report called UNEXECUTED CODE contains information for unexecuted code segments in addition to the program identification information. The columns for this area are:

start The line or statement number of the first unexecuted instruction in this unexecuted segment.

end The line or statement number of the last unexecuted instruction in this unexecuted segment.

The number that appears for start and end is the number that is used to identify each line or statement in the compiler listing.

16.1.19 Creating the summary and annotated listing report

The first part of this report is the summary report described in the previous section. Example 16-8 is an abstract of the annotated listing.

Example 16-8 DTCU - annotated listing

```
000520              ****************************************************************
000521               BUY-SELL-SELL-FUNCTION SECTION.
000522              * CHECK WE HAVE A RECORD FOR THIS CUSTOMER.COMPANY, IF NOT EXIT
000523              * CHECK THAT WE CAN MEET THE SELL REQUEST, IF NOT EXIT
000524              * CALCUATE NEW NUMBER OF SHARES AND UPDATE CUSTFILE
000525              * CALCULATE NEW SHARE TOTAL SHARE VALUE
000526 ^                 MOVE 'ENTRY FOR SELL' TO COMMENT-FIELD                    82
000527              * CHECK WHETHER WE HAVE ANY SHARES TO SELL
000528 ^                 PERFORM READ-CUSTFILE                                     579
000529                  EVALUATE RETURN-VALUE                                     178
000530 ^                  WHEN CLEAN-RETURN                                        180
000531 ^    1                  IF TR-NO-OF-SHARES IS GREATER THAN DEC-NO-SHARES    154 118
000532      1                  THEN
000533 ^    2                    MOVE INVALID-SALE TO RETURN-VALUE                 190 178
000534 ^    2                    MOVE TOO-MANY-SHARES-MSG TO COMMENT-FIELD         101 82
000535      1                  ELSE
000536 ^    2                    PERFORM CALCULATE-SHARES-SOLD                     512
000537 ^    2                    PERFORM REWRITE-CUSTFILE                          629
000538 ^    2                    IF RETURN-VALUE = CLEAN-RETURN                    178 180
000539 ^    3                       PERFORM WRITE-TRANSACTION-REPORT               788
000540      2                    END-IF
000541              * @TEST 2 LINES
000542      1                 END-IF
000543 ^                   WHEN CUSTOMER-NOT-FOUND                                 194
000544 ^    1                  MOVE INVALID-SALE TO RETURN-VALUE                  190 178
000545 ^    1                  MOVE NO-SHARES-MSG TO COMMENT-FIELD                103 82
000546                    WHEN OTHER
000547 ^    1                  MOVE BAD-CUST-READ TO RETURN-VALUE                 184 178
000548                  END-EVALUATE
000549                  .
000550               BUY-SELL-SELL-FUNCTION-EXIT.
000551 ^                 EXIT.
000552              ****************************************************************
```

DTCU annotation symbols

Each instruction line of the listing has a character to the right of the statement number to indicate what happened during the test run:

&	A conditional branch instruction that has executed both ways
>	A conditional branch instruction that has branched but not fallen through
V	A conditional branch instruction that has fallen through but not branched
:	Non-branch instruction that has executed
^	Instruction that has not executed
@	Data area in the assembler listing
%	An unconditional branch instruction that has executed in the assembler listing

16.1.20 Using the monitor panel

Some of the monitor-related functions can also been done from a DTU panel.

Select option **5** (Monitor) from the Coverage Utility panel, and the panel shown in Figure 16-11 is displayed.

```
------------------------------ Control the Monitor ---------------------------
Option ===> _

1  Start         Create JCL to Start the Monitor

2  Stop          Stop monitor execution normally                    (EQACUOSP)

3  SessDisplay   Display all active sessions                        (EQACUOSE)

4  Listings      Display listings                                   (EQACUOSL)
5  Statistics    Display statistics                                 (EQACUOSA)
6  BPDisplay     Display Breakpoint status                          (EQACUOBP)

7  AddId         Specify a unique testcase id                       (EQACUOID)
8  Snapshot      Take snapshot of data                              (EQACUOSN)
9  Reset         Reset all data in monitor                          (EQACUORE)

10 Quit          Terminate monitor without saving breakpoint data   (EQACUOQT)

Enter END to Terminate


 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F10=Actions  F12=Cancel
```

Figure 16-11 DTCU - Control the Monitor

The options are:

- Option 1, Start: This generates the JCL to start the monitor (similar to 16.1.9, “Creating JCL to start a monitor session” on page 406).
- Option 2, Stop the monitor: This is the ISPF version of the command-line version of the function shown in 16.1.17, “Stopping the monitor” on page 410.
- Option 3, Display all active sessions: This shows the currently active monitor sessions.
- Option 4, Listings: This shows the listing data sets that were actually used (compiler listings).

- Option 5, Statistics: These are the ISPF versions of the command-line versions shown in 16.1.16, “Displaying statistics online” on page 409.
- Option 6, BPDisplay: This shows the breakpoint status, the corresponding assembler hooks, and their offsets.
- Option 7, Specify a unique test case ID: This lets you specify an ID for this test case.
- Option 8, Snapshot: This shows the breakpoint ID, if any, and the name of the saved statistical data.
- Option 9, Reset: This is used to reset all monitor data.
- Option 10, Quit: This is used to quit.

16.2 Load Module Analyzer

Debug Tool Utilities and Advanced Functions provides functionality that allows you to analyze your load modules to determine the language translator used to generate the object code. This can be particularly useful when you must identify candidate OS/VS COBOL programs for conversion to Enterprise COBOL.

16.2.1 Starting the Load Module Analyzer

You can start the Load Module Analyzer (LMA) in one of two ways:

- Member EQAZLMA of data set *hlq*.SEQASAMP can be edited and then submitted to run as a batch job.
- Option 5 of Debug Tool Utility panel can be selected.

Only the second method is described here. Start by selecting option 5 from Debug Tool Utilities main panel, as shown in Figure 16-12.

```
---------------------------- Debug Tool Utilities ----------------------------
Option ===> 5_

                                                                        More:   -
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs
   You can dynamically update LE Runtime options and create private
   message region.

5  Load Module Analyzer
   Analyze load modules to determine the language translator used to
   compile or assemble each CSECT in the load module.

G  Getting Started

Enter X to Terminate

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 16-12 Selecting the Load Module Analyzer

You are then presented with the Load Module Analyzer input screen. Figure 16-13 shows this screen with data entered into the input fields.

```
---------------------- Debug Tool - Load Module Analyzer ----------------------
Command ===> _____________________________________________________________________

                                                                        More:     +
Load Module Library:
   Project . . . CHABERT
   Group . . . . BOOK2005 . . . ________ . . . ________ . . . ________
   Type  . . . . LOAD
   Member  . . . TRADERD         (Asterisk for all members)


Other Data Set Name:
   Data Set Name . . . _____________________________________________________________
   Volume Serial . . . ______    (If not cataloged)


Listing Data Set . . . DTU.LIST_____________________________________________________


_ OS/VS COBOL only (/)
_ Display prefix and program data (/)
/ Show information for all compiler / system library routines (/)
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 16-13 Load Module Analyzer input screen

The Member field can contain either an asterisk (*) or a specific member name, or you can leave it blank.

- If you enter a specific member name to be analyzed, LMA runs in the TSO foreground and the output is both returned to your screen and written to the sequential data set EQAPRINT.
- If you enter an asterisk (*), the entire PDS or PDSE is analyzed. For this, LMA generates JCL for a batch job that you must then run or save. If you submit the job, the output report is written to the sequential data set EQAPRINT.
- If you leave the field blank, you are presented with a list of all the members of the specified PDS or PDSE. From here you can select a specific member to be analyzed.

Figure 16-14 shows a portion of the output that might be seen from the analysis of an Enterprise COBOL load module.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Iest  Help
VIEW       ABAPHEN.DTU.LIST                                Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
000005  Parm:
000006  NATLANG=ENU,DATEFMT=YYYYMMDD,SORTBY=OFFSET
000007
000008  EQAIN:
000009
000010   SELECT MEMBER=TRADERD
000011
000012 1 5655-P15   Debug Tool Version 6 Release 1 Load Module Analyzer   2005/
000013             Load Module  CHABERT.BOOK2005.LOAD(TRADERD)
000014
000015  CSECT      Offset Len/Ent  Program-ID  Trn-Date    Program-Description
000016  TRADERD        20    3F20   5655G5300  2005/10/21  Enterprise COBOL for
000017  CEE*                        5696234                High Level Assembler
000018  DFH*                        5696234                High Level Assembler
000019  DSN*                        5696234                High Level Assembler
000020  IGZ*                        5696234                High Level Assembler
****** **************************** Bottom of Data ****************************

 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 16-14 Portion of the output report from Load Module Analyzer

For additional information about the Load Module Analyzer, refer to Appendix G of Debug Tool *for z/OS Debug Tool Utilities and Advanced Functions User's Guide,* SC18-9534.

16.3 Conversion of old COBOL

The ability to compile old COBOL programs using the Convert and Compile option of Debug Tool Utilities and Advanced Functions provides you with an easy way of debugging old COBOL programs, while giving you the option of using the converted source and new compiler, or continuing to use the old source and old compiler. This section describes what the Convert and Compile option requires, and what steps are required to set it up.

16.3.1 Requirements for the conversion tools

To successfully convert an old program to ANSI85 standard, you require:

- Your application source code
- Any copy books associated with the application

16.3.2 Setting up for convert and compile

In this section we show the steps required to convert an OS/VS COBOL Program to the ANSI85 standard. We start from DTU, either by selecting DTU directly from ISPF or by selecting from ISPF option 6 (TSO commands) and issuing the following command:

```
EXEC 'D/T_high_lvl_qualifier.SEQAEXEC(EQASTART)'
```

From the main DTU panel, select **1** (Program Preparation), as shown in Figure 16-15.

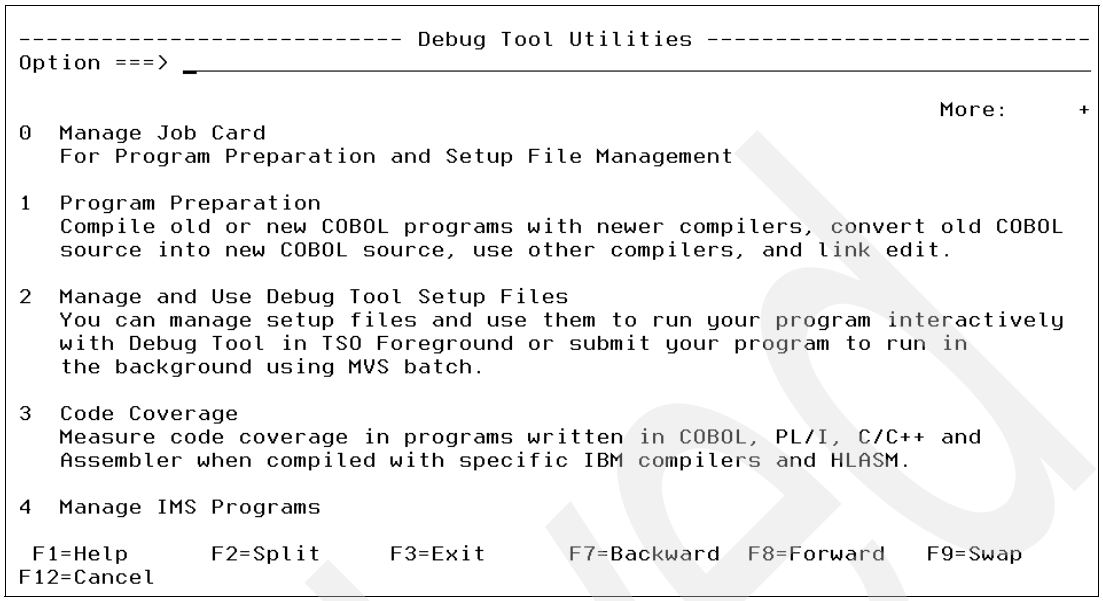


Figure 16-15 Getting to the COBOL Convert and Compile interface

Note: If you intend to do your conversion and compilation in batch, choose option 0 and verify your job card before selecting option 1 from the main menu.

The Debug Tool Program Preparation menu shown in Figure 16-16 is displayed.

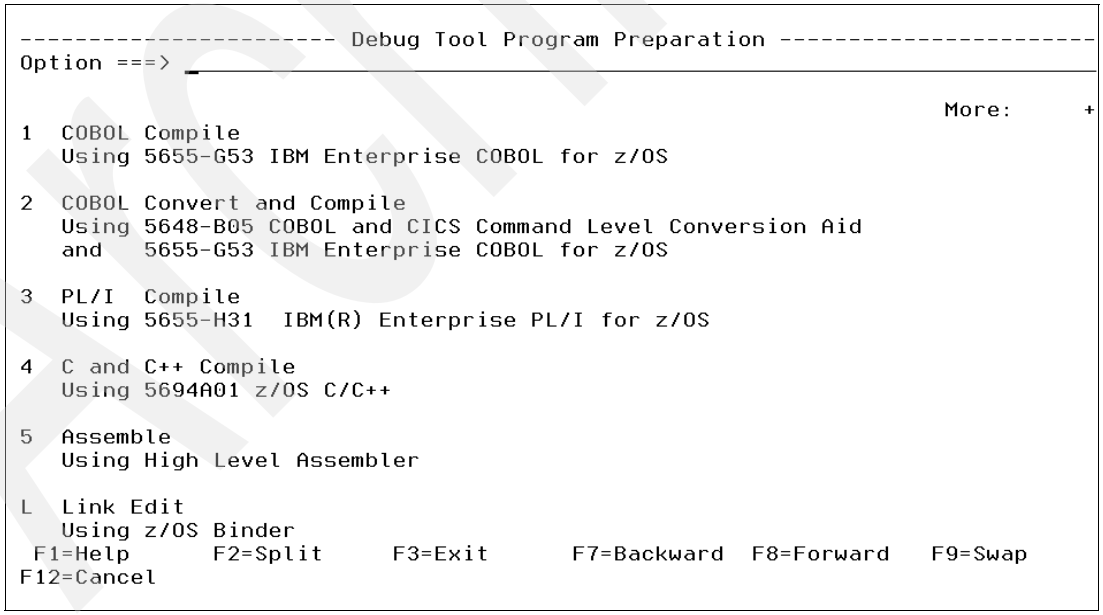


Figure 16-16 Debug Tool Program Preparation

Ensure that the compiler and converter settings are correct

To do this:

1. Verify the settings for the Enterprise COBOL Compiler and the COBOL and CICS Command Level Conversion Aid. To do this simply page forward (F8) and select option **S**, as shown in Figure 16-17.

```
----------------------- Debug Tool Program Preparation -----------------------
Option ===> s_____________________________________________________________________

                                                                        More:    -
4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder

F  Fault Analyzer Side file Create
   Using 5655-G74 IBM Fault Analyzer for z/OS and OS/390

C  Convert old 68/74 Std COBOL to 85 Std COBOL
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid

S  Manage System and User Settings


 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 16-17 Manage system and user settings selection

2. On the panel shown in Figure 17-17, set up the COBOL and Conversion Utilities appropriately for your system using selection **C** and selections **2** and **9** from the screen shown in Figure 16-18.

```
-------------- Debug Tool Program Preparation - Manage Settings --------------
Command ===> _____________________________________________________________________

                                                                      More:     +
System Settings for Program Preparation (compiler product datasets, etc):
These settings are refreshed at the start of every session.
  1  COBOL Compile
  2  COBOL Conversion (CCCA)
  3  PL/I  Compile
  4  C and C++ Compile
  5  HLASM
  6  Link Edit
  7  Fault Analyzer IDILANGX

User Settings for Program Preparation (input and output datasets, etc):
These settings are saved between sessions until RESET.
  8  COBOL Compile
  9  COBOL Convert and Compile
 10  PL/I  Compile
 11  C and C++ Compile
 12  HLASM
 13  Link Edit
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 16-18 Debug Tool program preparation - Manage Settings

3. Select 2 (COBOL Conversion) to define the system settings as follows:
   a. Define the library where the conversion code exists, or LINKLIST if the library is in the MVS linklist.
   b. Define the shared and private VSAM and non-VSAM data set qualifiers.
   c. We recommend that you do not change the allocation sizes for the different data sets.
   d. Press PF3 to save.
4. Select 9 (COBOL Convert and Compile) to define the user settings.
5. Change the data set qualifiers and their allocation sizes and DB2 precompiler and CICS translator option.

Note: Remember the DTU's help function. Select the fields using the cursor and press F1.

Once you have finished configuring the settings, you can use F3 until you get to Debug Tool Program Preparation menu.

16.3.3 Starting the conversion process

To do the conversion, select option **2** from Debug Tool Program Preparation menu, as shown in Figure 16-19.

```
------------------------ Debug Tool Program Preparation -----------------------
Option ===> 2_

                                                                        More:     +
1  COBOL Compile
   Using 5655-G53 IBM Enterprise COBOL for z/OS

2  COBOL Convert and Compile
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid
   and   5655-G53 IBM Enterprise COBOL for z/OS

3  PL/I  Compile
   Using 5655-H31  IBM(R) Enterprise PL/I for z/OS

4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 16-19 COBOL Convert and Compile

This brings up Debug Tool Program Preparation - Convert and Compile page, as shown in Figure 16-20.

```
-----------Debug Tool Program Preparation - COBOL Convert and Compile---------
Command ===> _____________________________________________________________________

Convert using 5648-B05 COBOL and CICS Command Level Conversion Aid
Compile using 5655-G53 IBM Enterprise COBOL for z/OS

F Foreground or Batch Processing  (F B)
                                                                        More:     +
Specify primary input data set for conversion and compilation.
Source Library:
   Project . . . ________
   Group . . . . ________ . . . ________ . . . ________ . . . ________
   Type  . . . . ________
   Member  . . . ________        (Blank or pattern for member selection list)

Other Partitioned or Sequential Data Set:
   Data Set Name . . . _____________________________________________________
   Volume Serial . . . ______    (If not cataloged)

Test options, data set name patterns, CICS and SQL processing options:
_ Enter / to edit options and data set name patterns
N CICS    I Integrated CICS translator, S Separate CICS translator, N None
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 16-20 The top part of the Convert and Compile panel

Figure 16-21 looks very similar to the other program preparation panel for COBOL compile. The steps are:

1. Select a foreground or background convert/compile.
2. Enter the name of the OS/VS COBOL (or other COBOL) source data set.
3. Page forward to see the second half of the panel, as shown in Figure 16-21.

```
-----------Debug Tool Program Preparation - COBOL Convert and Compile---------
Command ===> _____________________________________________________________________

Convert using 5648-B05 COBOL and CICS Command Level Conversion Aid
Compile using 5655-G53 IBM Enterprise COBOL for z/OS

F Foreground or Batch Processing  (F B)
                                                                    More:   - +
N CICS    I Integrated CICS translator, S Separate CICS translator, N None
N DB2/SQL I Integrated SQL coprocessor, S Separate DB2 precompiler, N None

Compiler Options (TEST compiler option is automatically added):
_____________________________________________________________________________


Specify secondary input data sets for Copy processing:
DDname          Syslib data set Names
  SYSLIB   . . . _____________________________________________________
  ________ . . . _____________________________________________________
  ________ . . . _____________________________________________________
  ________ . . . _____________________________________________________
  ________ . . . _____________________________________________________
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 16-21 The bottom of the Convert and Compile panel

4. You must:
   - Mark for DB2 and/or CICS if EXEC SQL or EXEC CICS statements are in source.
   - Add or change compiler options.
   - Add SYSLIB data sets for your copybooks.
5. Press Enter and select the member to be converted/compiled.

The next panel shows the input data set, compiler options, and all data sets for the convert, compile, and precompile/translate step. You can modify any of these data set names and options.

We used the sample OS/VS COBOL program, which includes the EXHIBIT statement, as shown in Example 16-9.

Example 16-9 Original OS/VS COBOL source

```
IDENTIFICATION DIVISION.
       PROGRAM-ID. MINI.
       ENVIRONMENT DIVISION.
       DATA DIVISION.
       WORKING-STORAGE SECTION.
         77 X PIC 99.
       PROCEDURE DIVISION.

           DISPLAY "MINI!".

           EXHIBIT X
           DISPLAY "MINI!".

           GOBACK.
```

These are the steps:

1. Press Enter to start the foreground convert/compile. If you selected for background compile check the upcoming JCL and press Enter.
2. If any of the steps did not work, browse the corresponding conversion listing files directly from the next panel. Edit the appropriate input data sets. Example 16-10 shows the converted source.

Example 16-10 OS/VS program after conversion

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. MINI.
000030*             PROGRAM CONVERTED BY
000040*             CCCA FOR OS/390 & MVS & VM 5648-B05
000050*             CONVERSION DATE 10/03/02 15:48:24.
000060 ENVIRONMENT DIVISION.
000070 DATA DIVISION.
000080 WORKING-STORAGE SECTION.
000090   77 X PIC 99.
000100 PROCEDURE DIVISION.
000110
000120     DISPLAY "MINI!".
000130
000140     DISPLAY
000150             "X = " X " "
000160     DISPLAY "MINI!".
000170
000180     GOBACK.
```

Now, how does convert/compile know the source and target language level and other options? It inherits these options from the settings you provide in the Convert old 68/74 Standard COBOL to 85 Standard COBOL panel. Select option **1** from the DTU panel. In this sample we used:

- Source language level: 3 OS/VS COBOL LANGLVL(1)
- Target language level: 4 COBOL for OS/390

You can also overwrite the source language level in the Edit Data Set Name Patterns and Other Options panel. Use the help function to learn what the numbers mean, for example, 3. You get to the Edit Data Set Name Patterns and Other Options panel by inserting a slash (/) just in front of Enter / to edit data set name patterns, and DB2 or CICS translation options line.

Now exit out to the Program Preparation panel and select L link to link your program.

Finally, run the program and use Debug Tool to debug your converted application.

16.3.4 Converting old 68/74 COBOL to 85 COBOL

The conversion tool can be selected from the Program Preparation panel option C (Convert old 68/74 Std COBOL to 85 Std COBOL), as shown in Figure 16-22.

```
----------------------- Debug Tool Program Preparation ----------------------
Option ===> C_

                                                                        More:   -
4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder

F  Fault Analyzer Side file Create
   Using 5655-G74 IBM Fault Analyzer for z/OS and OS/390

C  Convert old 68/74 Std COBOL to 85 Std COBOL
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid

S  Manage System and User Settings


 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 16-22 Choosing the option to do conversion only

Figure 16-23 shows the conversion main panel.

```
----------------------------- CCCA Master Menu -------------------------------
Option ===> _
                                                          Userid    - ABAPHEN
                                                          Terminal  - 3278A
   1  CONVERT    - Convert COBOL source programs          Time      - 10:37
                                                          PF Keys   - 24
   2  CUSTOMIZE  - LCP Development Aid                    Applid    - ISR

   0  OPTIONS    - Set environment and conversion options









   COBOL and CICS Command Level Conversion Aid for OS/390 & MVS & VM
                  5648-B05  Version 2  Release 1
        Copyright (C) IBM Corp 1982, 1998 - All rights reserved


 F1=Help      F2=Split     F3=Exit      F5=Refresh   F7=Up        F8=Down
F12=Cancel
```

Figure 16-23 Master Menu

Before you start converting an OS/VS COBOL program, select option **1.1.1** for Environment, giving the panel in Figure 16-24. You can find the option (such as jobcard and data set prefixes) preset with what you already entered in the system and user settings.

```
-------------------------- CCCA Environment Options ---------- OPTIONS UPDATED
COMMAND ===> _

   High level qualifiers:
     Non-VSAM Shared Data Sets .  ===> ADTOOLS.CCCA210
     NON-VSAM Private Data Sets   ===> ABAPHEN
     VSAM Shared Data Sets .....  ===> ADTOOLS.CCCA210
     VSAM Private Data Sets ....  ===> ABAPHEN

   UNIT for Work Files .......... ===> SYSDA
   CLIST debugging .............. ===> N   Y/N

Job statement information:      (Verify before proceeding)
  ===> //ABAPHEN  JOB (ACCOUNT),'NAME'
  ===> //*
  ===> //*
  ===> //*


SYSOUT CLASS ===> *


 F1=Help      F2=Split     F3=Exit      F5=Refresh   F7=Up        F8=Down
F12=Cancel
```

Figure 16-24 Environment options

Now go back to the Master Menu and choose O.2. This brings you to the menu shown in the Figure 16-25. You can define the source language level and the target language level in the page shown in Figure 16-25.

```
---------------------------- CCCA Language Level -------------------------------
Command ===> _

   Source language level ===> 3   1. DOS/VS COBOL LANGLVL(1)
                                  2. DOS/VS COBOL LANGLVL(2)
                                  3. OS/VS COBOL LANGLVL(1)
                                  4. OS/VS COBOL LANGLVL(2)
                                  5. VS COBOL II Release 1.0  1.1  2.0, or
                                     any COBOL with the CMPR2 option
                                  6. VS COBOL II NOCMPR2 Release 3.0  3.1  3.2
                                  7. VS COBOL II NOCMPR2 Release 4.0
                                  8. COBOL/370 NOCMPR2
                                  9. COBOL/VSE NOCMPR2
                                 10. COBOL for MVS and VM NOCMPR2
                                 11. COBOL for OS/390 and VM NOCMPR2


   Target language level ===> 4   1. VS COBOL II
                                  2. COBOL/VSE
                                  3. IBM COBOL
                                  4. Enterprise COBOL for z/OS & OS/390

 F1=Help      F2=Split     F3=Exit      F5=Refresh   F7=Up        F8=Down
F12=Cancel
```

Figure 16-25 Language level options

The Conversion Options 1 panel (option **1.1.3** from the main options menu) displays options that are output-related, like list formatting and program generation options. Consider customizing the following options for your installation, as shown in Figure 16-26.

- Re-sequence source lines.
- Reserved word suffix.
- Flag report writer statements (affects Report Writer programs).
- Remove value clauses in file/linkage sections (affects Report Writer programs).

```
-------------------------- CCCA Conversion Options 1 ----------------------
Command ===> _

   Lines per report page . . . . ===> 60      01 to 99

   Resequence source lines . . . ===> Y       Y/N
   Sequence number increment . . ===> 0010    0001 to 9999

   Reserved word suffix. . . . . ===> 74      Default value 74
   Generate new program. . . . . ===> Y       Y/N
   Generate new copy members . . ===> Y       Y/N
   Replace like-named copy members ===> Y     Y/N
   Print old source lines. . . . ===> Y       Y/N
   Print copy members. . . . . ===> Y         Y/N
   Print diagnostics of level >= . ===> 00    00 to 99
   Report Heading. . . . . ===> SAMPLE REPORT
   Generate tokenization listing . ===> Y     Y/N




 F1=Help      F2=Split     F3=Exit      F5=Refresh   F7=Up        F8=Down
F12=Cancel
```

Figure 16-26 Conversion options 1

The options included on the Conversion Options 2 panel are shown in Figure 16-27.

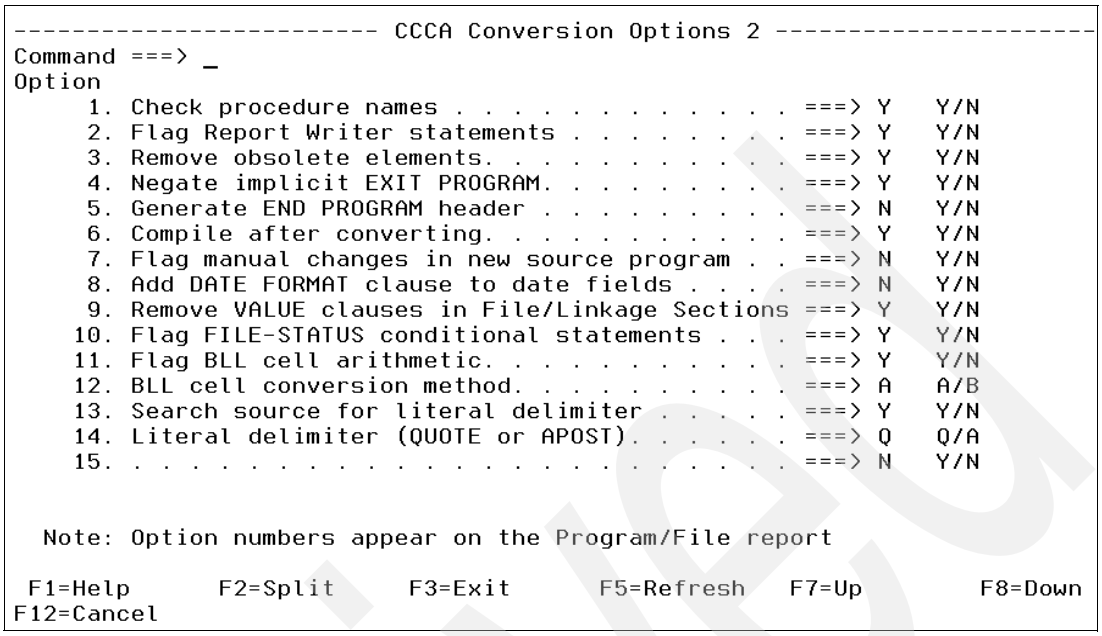


Figure 16-27 Conversion options 2

Note: Option **6** (Compile after converting) means that the program is compiled, but there is no output saved.

Returning to the Master Menu, choose selection 2, CUSTOMIZE - LCP Development Aid (Language Conversion Program). This selection allows you to:

- Change the reserved word table.
- Compile the LCP source.
- Delete LCP or activate/deactivate debugging for an LCP.
- Generate a directory of the LCP library.
- Update the message file.
- Set environment and conversion options.
- Convert COBOL source programs.

Setting environment and conversion options selections from this menu is identical to selecting options from the Master Menu.

Selecting Convert COBOL Source Programs from this menu is identical to selecting Conversion from the main conversion panel.

Now select **7** (CONVERT PROGRAM - Convert COBOL source programs). The first panel presents you with the job statement information. Verify the data and press Enter.

The next panel lets you enter the source input library, the program and copy library, and the options (like SQL, CICS, DLI). After pressing Enter, you can select one or more members from the source input library. Return from this panel with F3 Exit.

The conversion submission panel lets you either select more members to convert (press Enter), submit the JCL with or without return, or cancel the conversion.

Press PF3 to submit the job (or jobs) and to return to the conversion menu.

To know the results and statistics of your conversions, select from the conversion menu options 3 to 8, and L. Option E lets you delete the conversion statistics.

16.3.5 Summary

Debug Tool Utilities and Advanced Functions provides you with an easy way of dealing with old COBOL programs:

- You can continue to use the old source and just use the convert and compile options for debugging. (This would allow you to continue to use the old compiler for the production version of the program if you wish.)
- You can convert your old COBOL source to 85 COBOL source on the fly and debug or just use the facility to create your new production drivers.
- You can convert your old COBOL source to 85 COBOL and keep the converted source (recommended) for use in your normal production and debugging.

Part 5

Fault Analyzer for z/OS

In this part of the book, we discuss Fault Analyzer for z/OS, including:

- Settings and customizations
- Interactive analysis
- CICS system dump analysis
- Interactive analysis under CICS
- Subsystems

17

Fault Analyzer settings and customizations

This chapter is written for systems programmers and application programmers. It is meant to provide a better comprehension about the setup and customization of Fault Analyzer (FA).

17.1 Invocation exits

For FA to analyze an abend, it must be set up to be invoked through the appropriate abend processing exit. Table 17-1 shows the exits that are required for FA to function effectively.

Table 17-1 Invocation exits

Description	Name	FA program
CICS global user exits	XPCANBD XDUREQ	**IDIXCX52 or IDIXCX53**
MVS pre-dump exit	IEAVTABX	**IDIXDCAP**
CICS LE abnormal termination exit	CEECXTAN CSECT	**IDIXCCEE**
BATCH LE abnormal termination exit	CEEEXTAN CSECT	**IDIXCEE**

17.2 Fault Analyzer Subsystem

One of the optional components of Fault Analyzer is called the *Subsystem*. Although optional, the recommendation from this book would be that it should not be considered as such, since the tasks it performs add significant functional value to Fault Analyzer.

These are the main functions of the Fault Analyzer subsystem:

- Connect to DB2 subsystems to read the catalog if the connection failed from the abending address space.
- Perform SVC dump registration when the IDIXTSEL post-dump exit is installed. This is primarily intended for CICS system dumps.
- Manage history file $$INDEX members for PDSE history file data sets for improved performance.
- Enable IMS fast-path duplicate fault suppression specified using the NoDup(ImageFast(IMS(...))) option.
- Provide recovery fault recording support see Recovery Fault Recording.

Once the subsystem is started (see 17.2.1, “Starting and stopping the subsystem” on page 432), the above functions are performed seamlessly to any application abend analysis.

17.2.1 Starting and stopping the subsystem

In its basic form, to start the Fault Analyzer subsystem, you should submit a job similar to the JCL shown in Example 17-1.

Example 17-1 Fault Analyzer Subsystem JCL and PROC

```
//IDISS    JOB MSGLEVEL=1
//STARTING EXEC IDISS
.
.
//IDISS    PROC
//IDISBSYS EXEC PGM=IDISAMAN,TIME=NOLIMIT,PARM=UPDINDEX,REGION=0M
//DB2DB1E  DD DISP=SHR,DSN=SYS1.DB1E.SDSNLOAD
//DB2D80H  DD DISP=SHR,DSN=SYS1.D81H.SDSNLOAD
//DB2D70F  DD DISP=SHR,DSN=SYS1.D71F.SDSNLOAD
//DB2D80G  DD DISP=SHR,DSN=SYS1.D81G.SDSNLOAD
//DB2D90F  DD DISP=SHR,DSN=SYS1.D91F.SDSNLOAD
```

In this example, the UPDINDEX parameter and six different DB2 subsystems have been specified.

If the foregoing JCL had been used to start the subsystem, then the MODIFY command shown in Example 17-2 could be used to stop it.

Example 17-2 Modify command to stop subsystem

```
/F IDISS,STOP
```

17.2.2 Why use the subsystem?

For normal application abend analysis, the existence of the FA subsystem is likely to have no visible effect on the abend analysis report produced by Fault Analyzer. The exception to this is a situation where the subsystem has been called to extract DB2 catalog information when such information could not be determined directly from the abending address space.

So why use the subsystem? Possibly the most significant non-visible advantage of the subsystem is its handling of $$INDEX and the performance advantages associated with this. The $$INDEX member of a history file data set contains an index of all fault entries in the history file and is used for quick access to basic information for each fault. It is also the sole repository for all duplicate information against any fault in the history file.

If the history file data set is a PDSE and the userid associated with the subsystem has UPDATE access to the data set, then all updates to the $$INDEX member are managed by the subsystem (within the same MVS image). This therefore means that I/O times for $$INDEX are eliminated from the abending address space as the information in the $$INDEX member is cached and subsequently managed by the subsystem. This can be of real significance, for example, in CICS, where overall analysis time has to be made as short as possible.

Another significant performance advantage of using the subsystem is the duplicate fault detection and subsequent analysis suppression for IMS. This feature, which is analogous to the fast duplicate detection under CICS, allows an IMS fault to be quickly determined as a duplicate of an existing fault and hence eliminate unnecessary abend analysis.

17.3 Duplicate abend processing

In situations where multiple identical abends occur it might be more useful and manageable to have the duplicate abends recorded as duplicate statistics against the initial abend/fault entry rather than have individual fault entries for each abend occurrence. For example, in a CICS environment, if the connection to DB2 stopped it might lead to all users of the *BILL* transaction receiving a *NODB* abend. In this instance, the initial abend analysis would be sufficient to identify the problem, and therefore, having all duplicates record against this fault entry rather than individual fault entries would be desirable.

There are two main types of duplicate abend detection in Fault Analyzer. The first is termed NORMAL duplicate detection. This type of duplicate detection occurs after the attached Fault analyzer analysis TCB has completed analysis. It is then that the abend details are compared to existing fault entries in the current history file data set. If there is a matching duplicate, which occurred with in the system defined time, the duplicate statistics are updated for the existing fault entry.

The implication of this type of duplicate detection is that the overhead of full abend analysis has had to occur before the duplicate can be detected. Although this overhead might be less important for a batch type of abend, it might be very significant in a CICS or IMS environment where, for example, the abending task might be holding locks on recoverable resources. For this reason, the second duplicate abend detection method exists in Fault Analyzer.

The second type of duplicate detection is FAST detection, and this occurs before the Fault Analyzer analysis TCB has been attached. In this case, if the abend “signature,” as determined by the invocation, exists, it is compared to recent abend signatures. If a match is found, then no abend analysis is performed and the DUP statistics are updated accordingly.

The duplicate detection process can be seen in Figure 17-1.

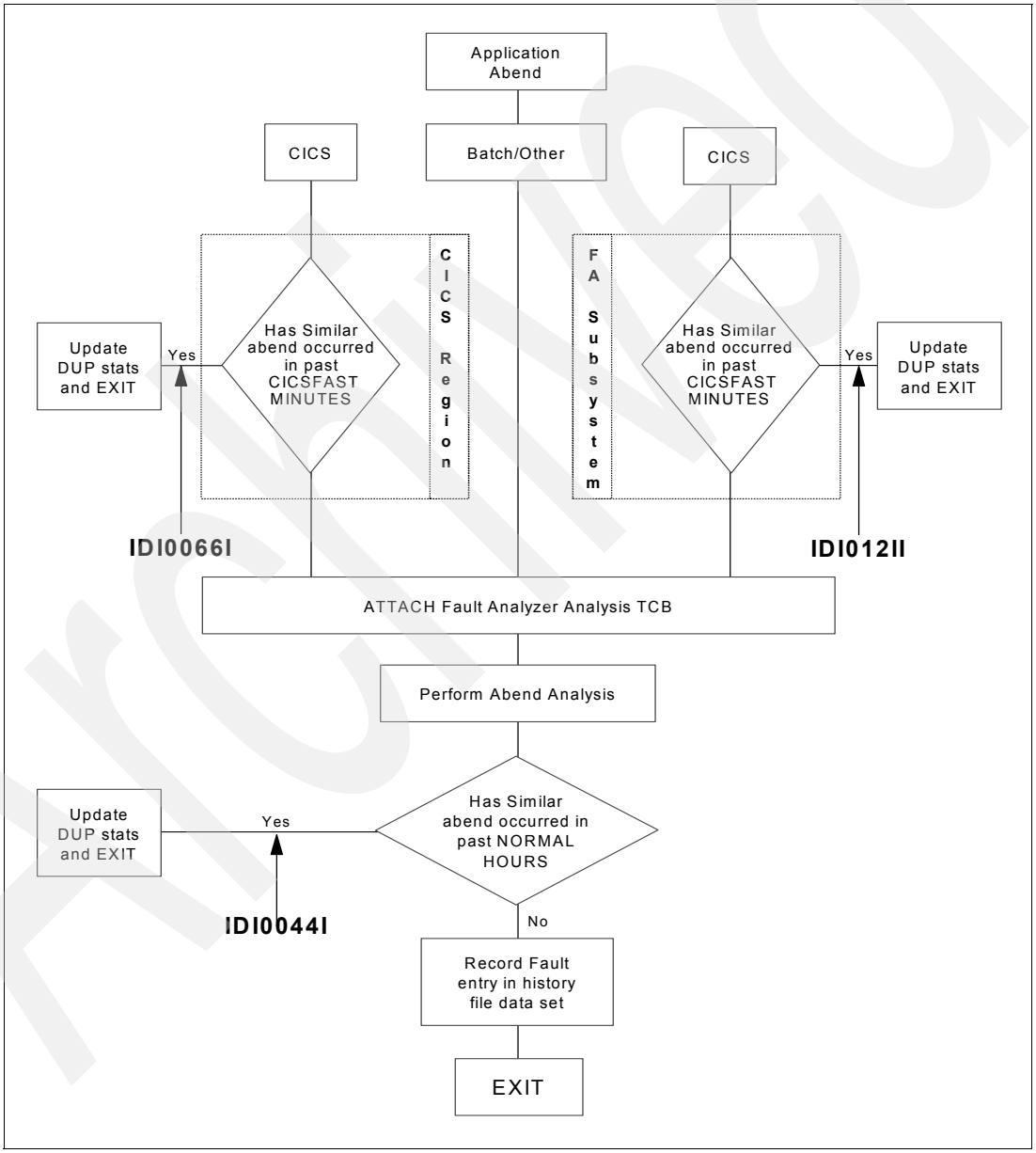


Figure 17-1 Duplicate abend detection process

An important point to note is that the FAST detection method under CICS occurs in the CICS region itself, whereas the FAST detection for IMS occurs in the FA subsystem. The implication of this is that the history file data set must be a PDSE, because this is the only data set type that the FA subsystem manages.

Any duplicate statistics can be seen in the main fault entry display in the DUPS column and the specific details can be seen either by entering an **H** next to the fault entry or by placing the cursor under the DUPS count and pressing Enter.

Figure 17-2 shows the fault entry display, having issued the MATCH command shown in Example 17-3. Notice that **DUPS *** displays only those faults which have some duplicate statistics, such as DUPS > 0.

Example 17-3 MATCH command to restrict fault entry display

```
match user_id simcock dups *
```

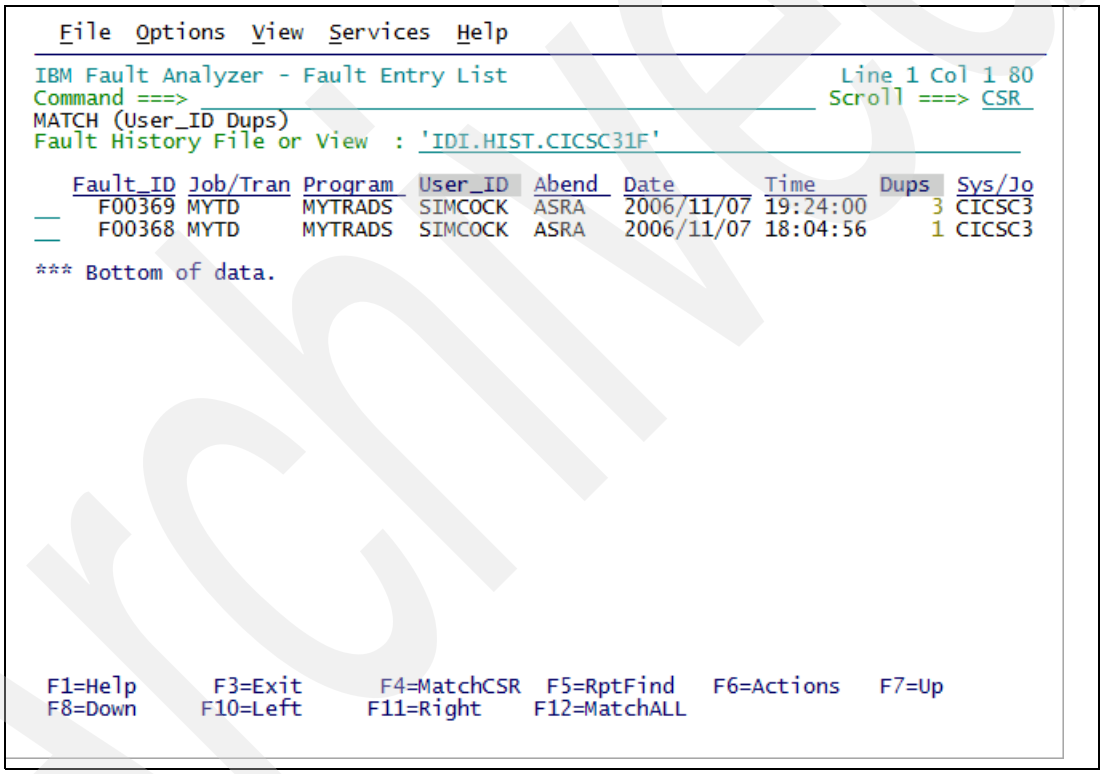


Figure 17-2 MATCH on Userid and DUPS > 0

From the foregoing display, first we placed the cursor under the 3 of fault entry F00369 and pressed Enter. This displayed the panel shown in Figure 17-3.

```
  File  View  Services  Help
Fault Entry Duplicate History                                  Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR

Most recent duplicate
  occurred. . . . . . . . . : 2006/11/07 19:27:16
Initial abend occurred. . . : 2006/11/07 19:24:00
Total duplicate count . . . : 3

Duplicate details in reverse chronological order: IDI.HIST.CICSC31F(F00369)

Date       Time     Jobname  Job ID   System   Dup Type
2006/11/07 19:27:16 CICSC31F STC03560 STLABF6  Fast
  User ID  Term ID Count
  SIMCOCK  0017        2

Date       Time     Jobname  Job ID   System   Dup Type
2006/11/07 19:24:49 CICSC31F STC03560 STLABF6  Fast
  User ID  Term ID Count
  SIMCOCK  0017        1

*** Bottom of data.




 F1=Help      F3=Exit      F5=RptFind   F7=Up        F8=Down     F10=Left
F11=Right
```

Figure 17-3 Display showing FAST duplicate statistics

Next we entered an `H` next to fault ID F00368, which displays the panel shown in Figure 17-4.

```
  File  View  Services  Help
Fault Entry Duplicate History                                  Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR

Most recent duplicate
  occurred. . . . . . . . . : 2006/11/07 18:20:48
Initial abend occurred. . . : 2006/11/07 18:04:56
Total duplicate count . . . : 1

Duplicate details in reverse chronological order: IDI.HIST.CICSC31F(F00368)

Date       Time     Jobname  Job ID   System   Dup Type User ID  Stepname
2006/11/07 18:20:48 CICSC31F STC03560 STLABF6  Normal   SIMCOCK  CICSC31F

*** Bottom of data.




 F1=Help      F3=Exit      F5=RptFind   F7=Up        F8=Down     F10=Left
F11=Right
```

Figure 17-4 Display showing NORMAL duplicate statistics

17.4 The Fault Analyzer ISPF interface

At any time after an abend you can, as a TSO user, start the Fault Analyzer ISPF interface to review the fault.

Using this interface you can:

- View the stored real-time analysis report (unless suppressed, for example, by the *DeferredReport* option).
- Start a batch reanalysis.
- Start an interactive reanalysis.
- View information about the fault.
- Delete the fault entry.

The ISPF interface also permits you to review the Fault Analyzer User's Guide and Reference online by using BookManager® Read against the current softcopy version of the book.

Reanalysis: You can only perform reanalysis of a fault if either a minidump or a SYSMDUMP was written.

Compiler listing or side file data sets that were allocated or specified via the DataSets option when the real-time analysis took place are automatically reused if performing reanalysis (if they are available in the reanalysis environment).

To make the reanalysis different from the initial real-time analysis, you must do one (or more) of the following steps:

1. Supply compiler listings (or side files) for the programs involved in the abend (if they were not available for the initial real-time analysis).
2. Change analysis options.
3. Use the interactive reanalysis to review dump storage.

The main difference between the batch and interactive reanalysis steps is:

- Interactive reanalysis always provides full detail and lets you look at storage locations that might not be included in the analysis report, while batch reanalysis provides the level of detail you ask for through the Detail option, and does not display all storage locations.

17.4.1 Invoking the interface

Invoking the Fault Analyzer ISPF interface depends on how it was customized. One way it is generally done is by adding an option to one of your ISPF selection panels. A systems programmer or the person who customized Fault Analyzer would be responsible of setting this up.

An alternative way of invoking Fault Analyzer is to enable a line command that can then be subsequently entered against an MVS dump or a fault history data set name, for example, next to the data set name displayed in ISPF option 3.4.

The z/OS system on which this book was conducted had Fault Analyzer as an option on one of the ISPF panels; however, it invoked an EXEC which did dynamic allocation of the required ISPF data sets. As such, the line command code was modified as shown in Example 17-4.

Example 17-4 REXX code for the FA line command

```
/* Rexx */
Parse Arg dsn .

outl. = ''
x = Outtrap('OUTL.',,"NOCONCAT")
Address TSO "LISTDS " || dsn
x  = Outtrap('OFF')
svcdump = 0
If outl.0 > 2 Then
  Do
    lrecl = Word(outl.3,2)
    If lrecl = 4160 Then
      svcdump = 1
  End

Address ISPEXEC
"LIBDEF ISPPLIB DATASET ID('ADTOOLS.FA710.SIDIPLIB') STACK"
"LIBDEF ISPMLIB DATASET ID('ADTOOLS.FA710.SIDIMLIB') STACK"
"LIBDEF ISPSLIB DATASET ID('ADTOOLS.FA710.SIDISLIB') STACK"
"LIBDEF ISPTLIB DATASET ID('ADTOOLS.FA710.SIDITLIB') STACK"
Address TSO "ALTLIB ACT APPLICATION(EXEC) DSNAME('ADTOOLS.FA710.SIDIEXEC')"
If svcdump = 1 Then
  'SELECT PGM(IDIPDDIR) NEWAPPL(IDI) PARM(DSN('dsn')) PASSLIB'
Else
  'SELECT PGM(IDIPDDIR) NEWAPPL(IDI) PARM(ISPFHISTDSN('dsn')) PASSLIB'
"LIBDEF ISPPLIB"
"LIBDEF ISPMLIB"
"LIBDEF ISPSLIB"
"LIBDEF ISPTLIB"
Address TSO "ALTLIB DEACT APPLICATION(EXEC)"

Exit
```

In the remainder of this section we discuss the ISPF interface as invoked for a Fault Entry data set, rather than for MVS dump analysis. The MVS dump analysis (specifically, CICS system dumps) is discussed in *Fault Analyzer CICS System Dump Analysis*.

The Fault Entry List display is shown when the Fault Analyzer ISPF interface is started. Figure 17-5 shows an example of a Fault Entry List display.

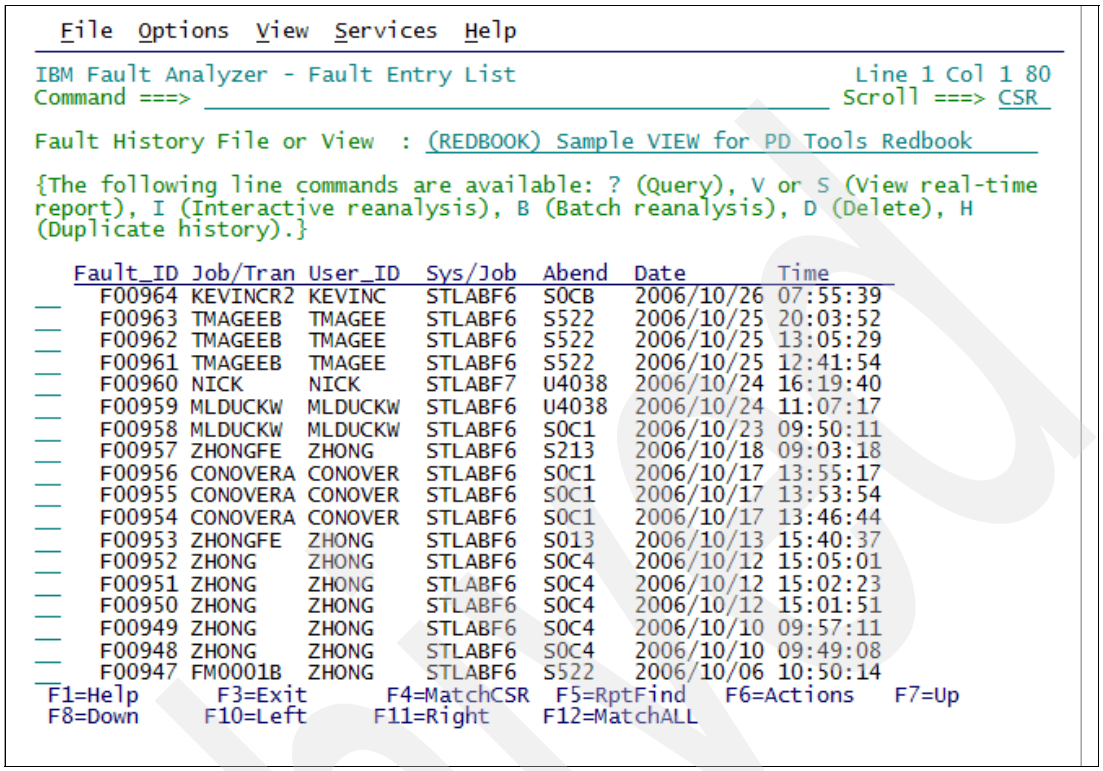


Figure 17-5 Fault entry list display

Note: If your Fault Entry List display does not show the PF keys, and you would like to see them, then enter the ISPF command:

```
FKA ON
```

The history file or view that was last selected while using the Fault Analyzer ISPF interface is shown by default. The first time the interface is used, the initial history file name is obtained using the IDIHIST suboption of the DataSets option in effect.

If a view was selected the last time the ISPF interface was used, and the view contains errors, then it is possible that an error display is presented prior to the Fault Entry List display. An example of an error display is shown in Figure 17-6.

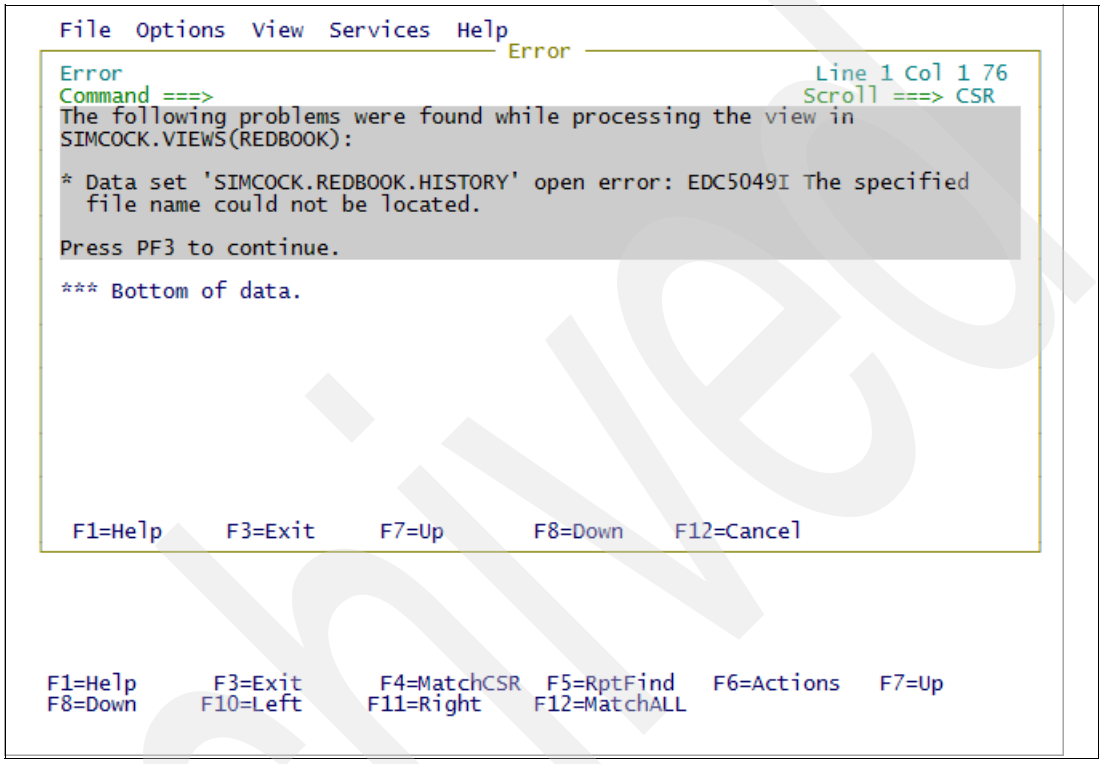


Figure 17-6 Error display prior to the Fault Entry List display

Note: To exit from the error display, press PF3.

The error display is shown each time the incorrect view member is read.

The identified errors in the view should be corrected to avoid this display. Entries in the Fault Entry List display are listed in reverse chronological order with the most recent fault entry (based on abend date and time) shown at the top.

Each fault entry in the list occupies a single line and is identified by a fault ID on the left side of the display.

The default information is displayed if no HistCols option has been specified and no customization has been made by the user. You can use the displayed fields to identify the faults you are interested in, or reduce the display to only a subset of the faults using the MATCH command.

Shown at the top of the display, if help text is enabled, are the available line commands; see Figure 17-5 on page 439.

You can show or hide help text by using the VIEW selection on the Action bar, as shown in Figure 17-7.

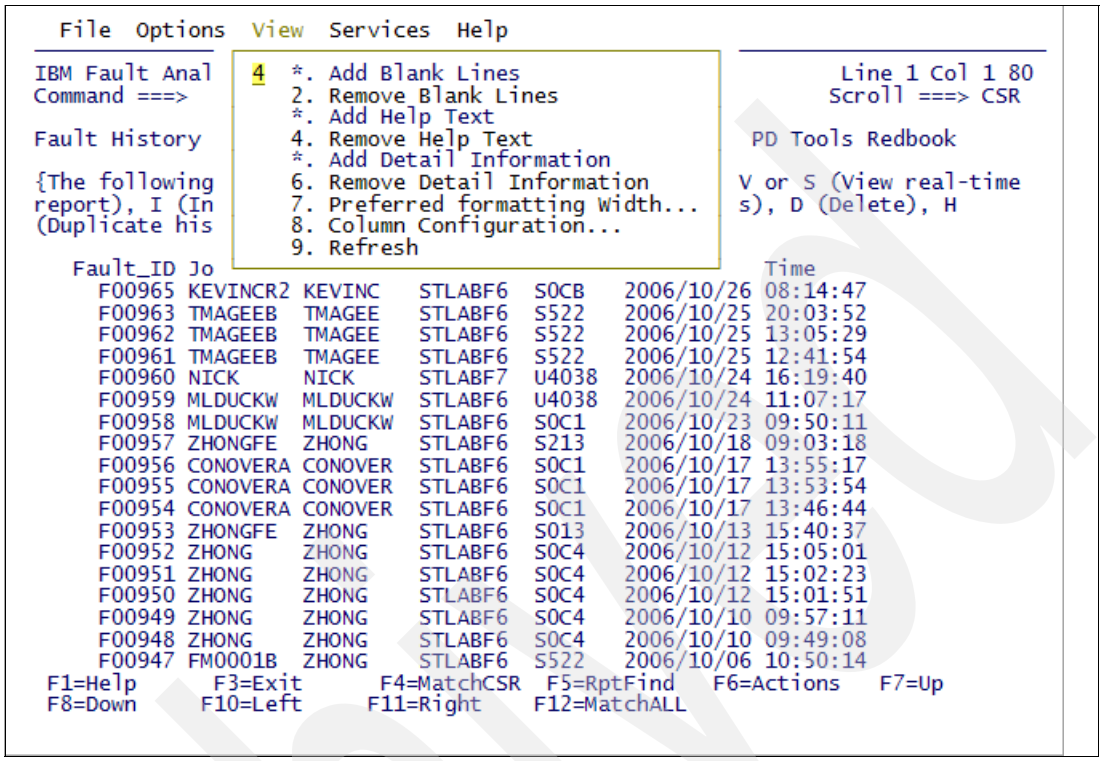


Figure 17-7 VIEW selections to show or hide help text

This panel responds to the standard UP, DOWN, LEFT, and RIGHT commands, which by default are assigned to the PF7, PF8, PF10, and PF11 function keys, respectively.

These can be used to scroll the display horizontally or vertically as required to see all of the information available.

In the top right corner of the panel is the current top-most line number and indication of the left-most and right-most columns currently displayed.

The end of the fault entry list is indicated by the line:

```
*** BOTTOM OF DATA.
```

This line is used to indicate the bottom of all Fault Analyzer ISPF interface scrollable displays. You exit from the Fault Analyzer ISPF interface by issuing the Exit command (PF3) from the Fault Entry List display, or by selecting the Exit Fault Analyzer option from the Fault Entry List display File menu.

17.4.2 Using views

Depending on how Fault history data sets are used, you might find it useful to concurrently view fault entries from more than a single data set. In order to do this, a *VIEW* can be defined. For example, one possible setup for Fault history data sets might be to have a separate data set for production CICS faults and another for test CICS faults. In this scenario it might also be useful to be able to view the production and test faults concurrently, and in this case, a *VIEW* could be set up.

Each VIEW definition corresponds to a member of a PDS(E). The one or many PDS(E) view data sets are pointed to by the IDIVIEWS suboption of the DataSets option in the IDICNF00 PARMLIB CONFIG member.

As well as being able to identify multiple history data sets, views can also be used to provide a specific column layout for the Fault Entry List display, or to provide a selection criteria for the initially displayed list of fault entries. For example, as well as the VIEW for all test CICS region faults, it might also be useful for a developer to have a similar view that filtered the faults on their own userid.

17.4.3 Defining a view

As an example of setting up a view, this section gives step-by-step instructions on how we defined a view to list test CICS fault entries for transaction RED1and how to have a modified column layout:

1. We added a &SYSUID..VIEWS statement to our SYS1.PARMLIB configuration member as shown in Figure 17-8. This allowed each user to have and use a <userid>.VIEWS data set without further customization.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       SYS1.PARMLIB(IDICNF00) - 01.39            CHARS '&SYSUID..VIEWS' f
Command ===>                                          Scroll ===> PAGE
****** ***************************** Top of Data ******************************
000001 INCLUDE(TYPE(TSU))              /* EXCLUDE TSO USERS                    */
000002 INCLUDE(TYPE(STC))              /* Exclude started tasks                */
000003 INCLUDE(CICSABEND(*))           /* Include all CICS abends - even if    */
000004                                 /* CICS is a started task               */
000005 DATASETS(
000006   IDIDOC  (ADTOOLS.FA710.SIDIDOC1)   /* BOOKS AND INDEXES     */
000007   IDIBOOKS(ADTOOLS.FA710.SIDIBOOK)   /* Softcopy books        */
000008   IDILCOB (CHABERT.TRADER.COBLIST)          /* Softcopy books         */
000009   IDICACHE(IDI.CACHE)           /* Softcopy book cache data set name    */
000010   IDIHIST (IDI.HIST)            /* Fault History file data set name     */
000011   IDIMAPS (ADTOOLS.FA710.SIDIMAPS)          /*                      */
000012   IDIVIEWS(ADTOOLS.FAULT.ANAL51.VIEWS,&SYSUID..VIEWS) /* views displ */
000013   IDILANGX(ADTOOLS.MNA.U6F6.LANGX.PLI,    /* side files             */
000014            ADTOOLS.MNA.S2U1F6.LANGX.PLI)
000015 )                                        /*                      */
000016 NODUP(NORMAL(01))              /* Suppress duplicate faults if less    */
000017                                /* than 24 hours apart                  */
000018 MAXMINIDUMPPAGES(999)          /* Permit 512 page minidump size        */
000019 NODEFERREDREPORT
****** ***************************** Bottom of Data ***************************


 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 17-8 SYSUID addition to SYS1.PARMLIB(IDICNF00)

2. In the <userid>.VIEWS data set, we included a member called MYCICS as shown in Figure 17-9 .

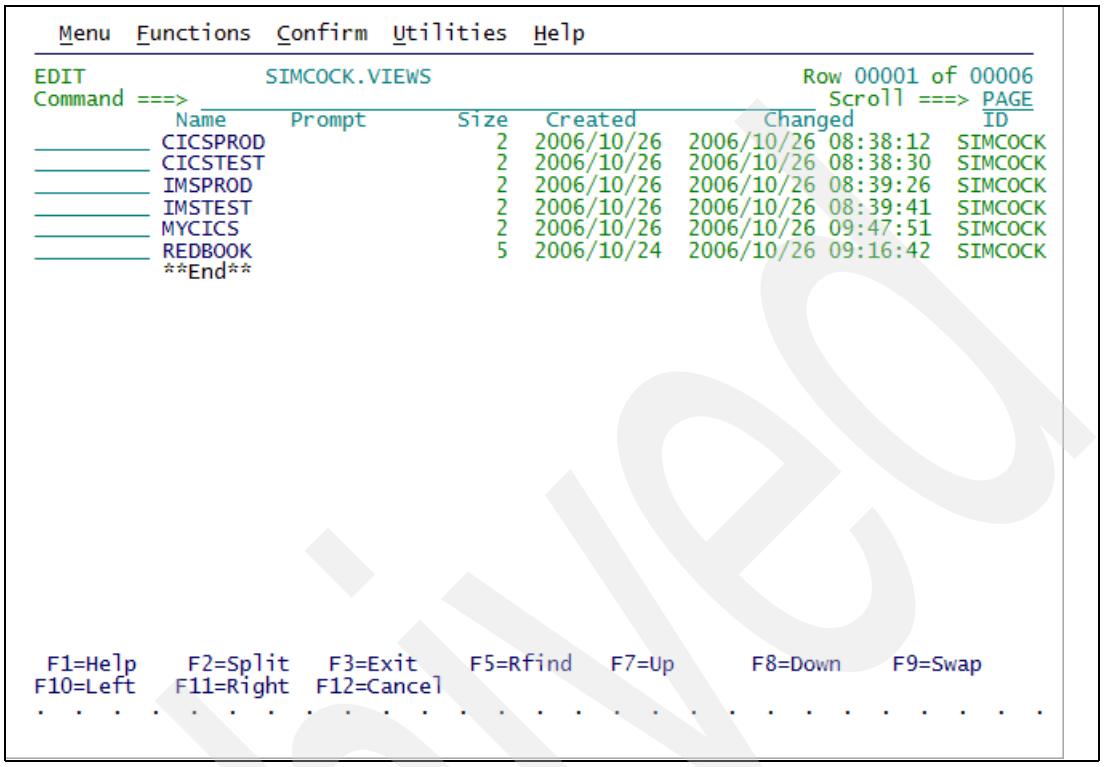


Figure 17-9 MYCICS member in VIEWS data set

3. We then EDITed the MYCICS member as shown in Figure 17-10. As can been seen, we included a MATCH command for CICS transaction and userid plus an alternative order for the columns.
4. We then invoked the Fault Analyzer ISPF interface and specified (MYCICS) as the view name. The resulting Fault Entry list as shown in Figure 17-11.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       SIMCOCK.VIEWS(MYCICS) - 01.03                Columns 00001 00072
Command ===>                                              Scroll ===> PAGE
****** ***************************** Top of Data ********************************
000001 * My TEST CICS Fault Entries
000002 IDI.HIST
000003 IDI.CICS.TEST.AOR2
000004 -MATCH(CICS_TRN RED1,USER_ID SIMCOCK)
000005 -HISTCOLS(FAULT_ID USER_ID CICS_TRN ABEND PROGRAM DATE TIME)
****** **************************** Bottom of Data ******************************

 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 17-10 MYCICS Member

```
  File  Options  View  Services  Help
IBM Fault Analyzer - Fault Entry List                    Line 1 Col 1 80
Command ===>                                             Scroll ===> CSR
MATCH (User_ID CICS_Trn)
Fault History File or View  : (MYCICS) My TEST CICS Fault Entries

{The following line commands are available: ? (Query), V or S (View real-time
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete), H
(Duplicate history).}

   Fault_ID User_ID  CICS_Trn Abend  Program  Date       Time
__   F00347 SIMCOCK  RED1     AEI0   PROGA    2006/10/26 10:56:23
__   F00346 SIMCOCK  RED1     ASRA   PROGA    2006/10/26 10:55:20

*** Bottom of data.

 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=MatchALL
```

Figure 17-11 MYCICS VIEW display.

17.4.4 Changing the history file or the view displayed

When the Fault Analyzer ISPF interface is started initially, the history file or view last used is displayed.

To select a different history file or view, do one of the following actions:

- Type a different history file or view name. After typing the history file or view name, press Enter to show the fault entries.

  The following explanation defines the rules for naming history files and views:

  – For history file names, the standard TSO naming convention applies (that is, the name typed is automatically prefixed by the TSO prefix if not enclosed in single quotation marks, as shown in Example 17-5).

Example 17-5 How a history file is typed

```
If TSO prefix is set to IDI and the specified history file name is IDI.HIST, 2 choices to
enter the history file name ON THE 'FAULT HISTORY FILE OR VIEW' LINE:
   – Type HIST
   – Type 'IDI.HIST'
```

  – If missing, the ending quote is automatically added.

  – View names are member names in one of the data sets associated with the IDIVIEWS DDname. These are specified by enclosing them in parenthesis.

  – To specify that the view member ABC is to be displayed, type (ABC) on the Fault history file or view line. If missing, the closing parenthesis is automatically added.

- Select a previously used history file or view.

  A record is maintained of the last 10 history files or views displayed.

  To select a previously displayed history file or view:

  – Select the File menu Last Accessed Fault History Files or Views option. This brings up the Last Accessed Fault History Files or Views display, as shown in Figure 17-12.

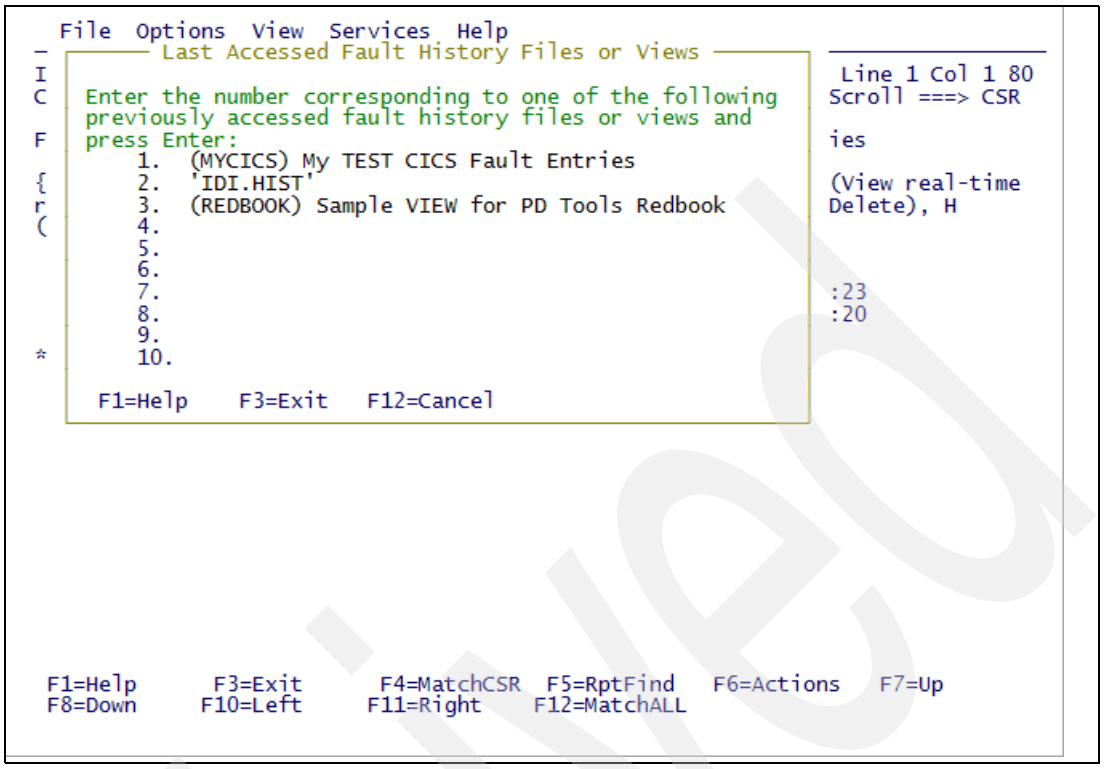


Figure 17-12 Sample last accessed Fault history file or Views display

- From the Last Accessed Fault History Files or Views display shown in Figure 17-12, type the number corresponding to the desired history file or view name. Press Enter to display the entries for the selected history file or view.
- To return to the Fault Entry List display without making any changes, press either PF3 or PF12.

- You might also want to display the last accessed Fault History File entries. From the dropped down File option, selecting option **2** performs this, as shown in Figure 17-13.

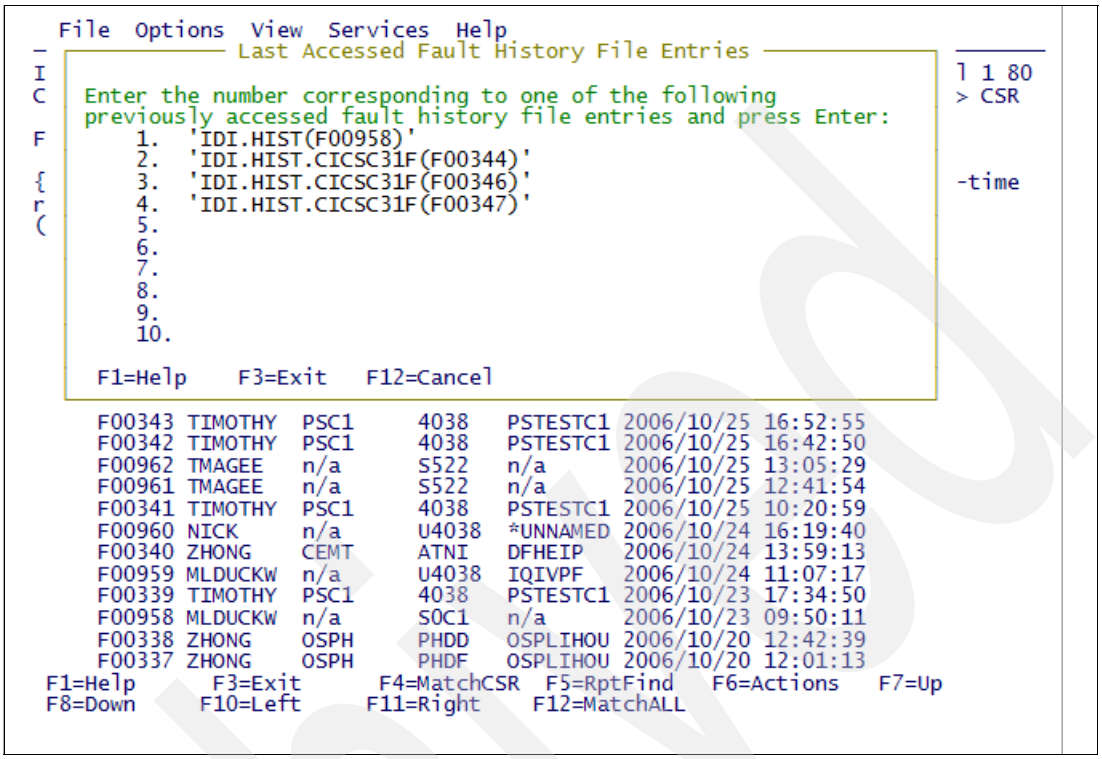


Figure 17-13 Last Accessed Fault History File Entries selected

  - Up to the last 10 accessed Fault entries can be re-accessed. To select your choice, type its number on the line. In our case we selected number 4.
  - You can then perform your regular command line against the FaultID displayed, as shown in Figure 17-14.

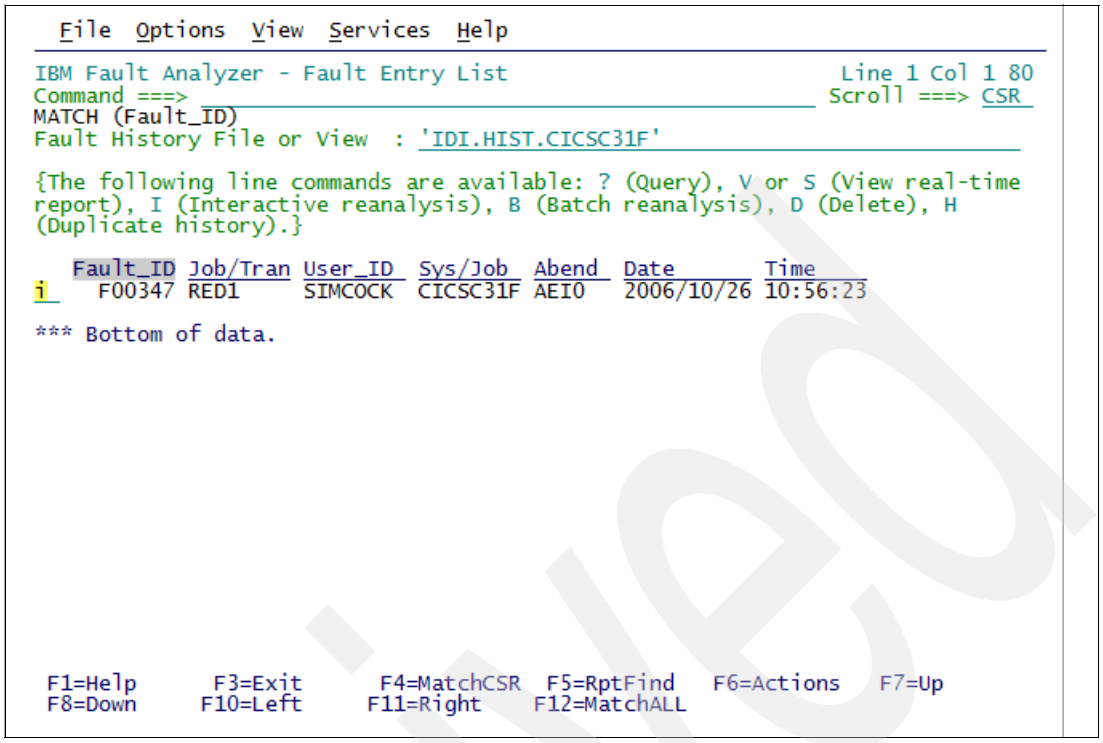


Figure 17-14 Sample of last FaultID selected from last accessed Fault entries history list

17.4.5 Fault entry list column configuration

The fault information shown on the Fault Entry List display is determined by the HistCols option in effect.

Individual users are able to alter the Fault Entry List display information by either entering the **COLS** command or by selecting the View menu Column Configuration option.

The Fault Entry List Column Configuration display is divided into sections, as shown in Figure 17-15. The first section is the Current Fault Entry List Column Configuration section, which shows the current column configuration with headings and sample data. This permits you to see which of the selected columns are visible on the Fault Entry List display without first having to scroll the display horizontally. The second section is the Column Configuration Settings section, which permits you to modify the columns used in the Fault Entry List display. This section is a list of all available columns, and PF8 can be used to scroll through them.

To make a column visible, or to change its relative display position, enter a non-zero positive value in the Order column. To hide a column, enter 0. After pressing Enter, the resulting column configuration is shown in the current fault entry list column configuration section.

The Fault_ID column cannot be hidden. If it is not given a specific display position, then it defaults to being the first column.

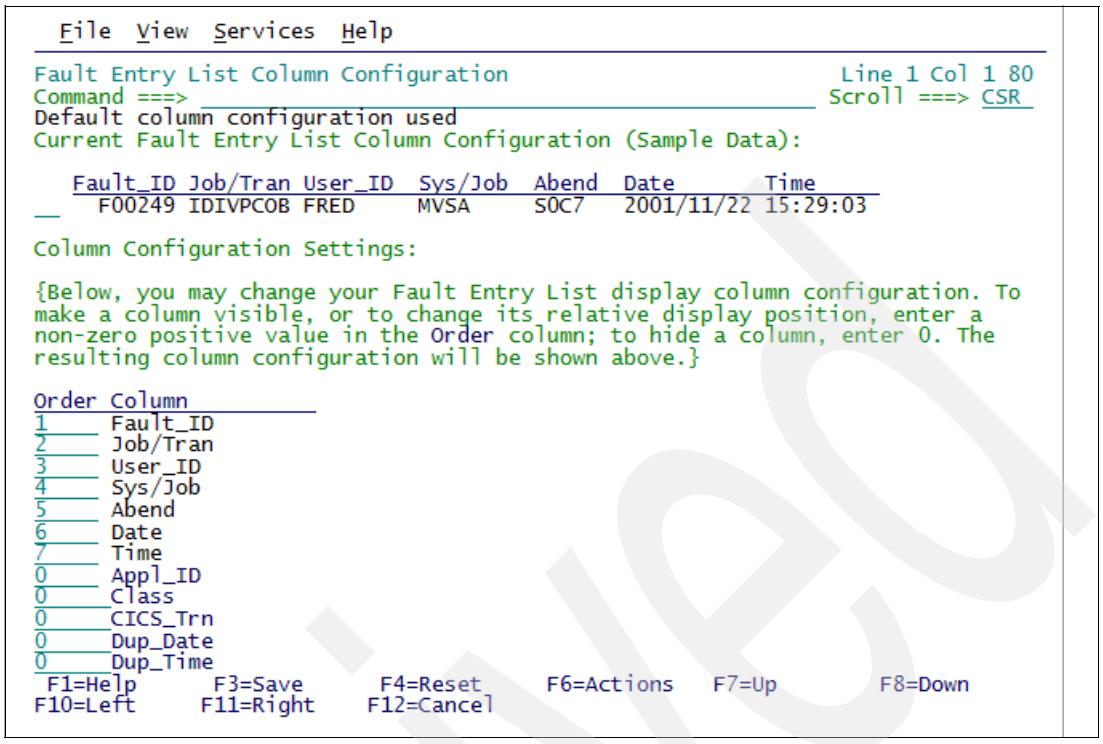


Figure 17-15 Fault Entry List Column Configuration panel

As an example, we used the COLS command to insert the PROGRAM column after the Job/Tran column and added the DUPS column to the end of the display.

- First we pressed PF8 to scroll the list of available columns. We entered a **3** next to Program and **99** next to Dups as shown in Figure 17-16, and pressed Enter.

Tip: Entering **99** is a quick way of automatically adding the selected column to the end of the display rather than having to enter its exact column number.

- We then pressed PF3 to go back to the Fault Entry list display, which was then modified as shown in Figure 17-17.

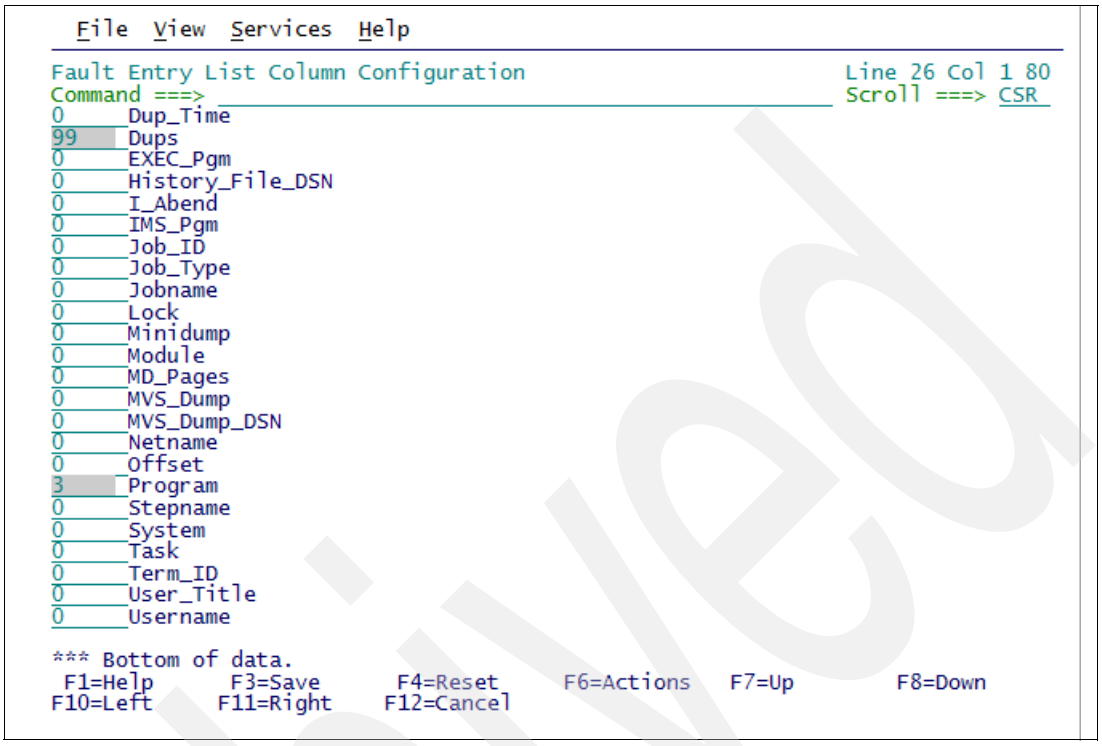


Figure 17-16 List of available columns

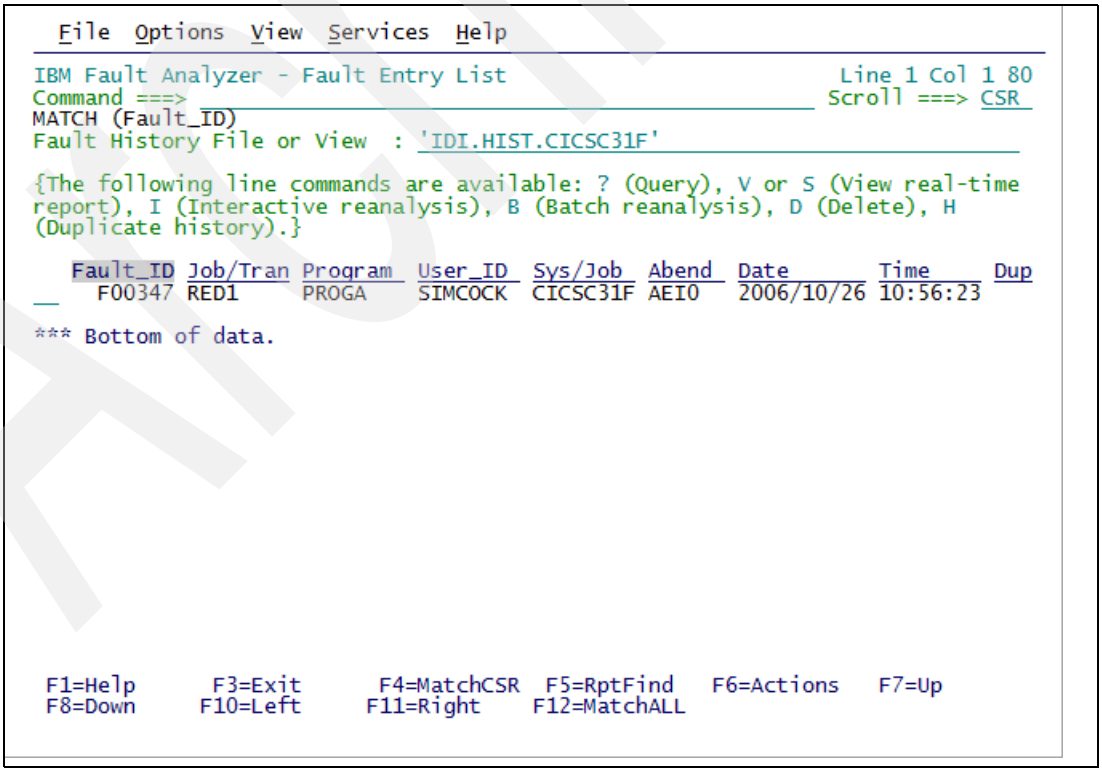


Figure 17-17 Modified column display

17.4.6 Filtering fault entries

Fault Analyzer allows you to match fault entries in the fault entry list display in order to limit or filter the list of fault entries. This can be especially useful if you are using a VIEW where there could be a large number of fault entries. There are three ways of matching fault entries as follows:

- Cursor selecting a match value
- Over-typing an existing value
- Using the MATCH command

The *Fault Analyzer Users Guide* has a comprehensive discussion on matching faults. In this section we give examples and typical panels showing how we used this feature.

We started with the Fault Entry list as shown in Figure 17-18.

```
  File  Options  View  Services  Help
IBM Fault Analyzer - Fault Entry List                    Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR

Fault History File or View  : 'IDI.HIST.CICSC31F'

{The following line commands are available: ? (Query), V or S (View real-time
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete), H
(Duplicate history).}

   Fault_ID Job/Tran Program  User_ID  Sys/Job  Abend  Date       Time     Dup
__   F00347 RED1     PROGA    SIMCOCK  CICSC31F AEI0   2006/10/26 10:56:23
__   F00346 RED1     PROGA    SIMCOCK  CICSC31F ASRA   2006/10/26 10:55:20
__   F00345 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/26 10:51:39
__   F00344 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/25 17:14:40
__   F00343 PSC1     PSTESTC1 TIMOTHY  CICSC31F 4038   2006/10/25 16:52:55
__   F00342 PSC1     PSTESTC1 TIMOTHY  CICSC31F 4038   2006/10/25 16:42:50
__   F00341 PSC1     PSTESTC1 TIMOTHY  CICSC31F 4038   2006/10/25 10:20:59
__   F00340 CEMT     DFHEIP   ZHONG    CICSC31F ATNI   2006/10/24 13:59:13
__   F00339 PSC1     PSTESTC1 TIMOTHY  CICSC31F 4038   2006/10/23 17:34:50
__   F00338 OSPH     OSPLIHOU ZHONG    CICSC31F PHDD   2006/10/20 12:42:39
__   F00337 OSPH     OSPLIHOU ZHONG    CICSC31F PHDF   2006/10/20 12:01:13
__   F00336 PSC1     PSTESTC1 TIMOTHY  CICSC31F 4038   2006/10/17 18:24:45
__   F00335 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/17 17:50:12
__   F00334 CEMT     DFHEIP   TIMOTHY  CICSC31F ATNI   2006/10/17 16:07:04
__   F00333 PSC1     n/a      TIMOTHY  CICSC31F 4038   2006/10/17 14:27:51
__   F00332 PSPM     PSAPOL01 TIMOTHY  CICSC31F PA08   2006/10/17 11:01:33
__   F00331 MNAP     PSMAIN   TMROSS   CICSC31F PMN1   2006/10/16 15:43:05
__   F00330 PSC1     PSTESTC1 TIMOTHY  CICSC31F ABM3   2006/10/16 15:29:11
 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=MatchALL
```

Figure 17-18 Initial Fault entry list prior to any MATCH command

- First we wanted to match on CICS abend codes, that is, those starting with an **A**. To do this, we over-typed the entry in the Abend column with **A*** as shown in Figure 17-19. Pressing Enter displayed the panel shown in Figure 17-20.

```
 File  Options  View  Services  Help
IBM Fault Analyzer - Fault Entry List                          Line 1 Col 1 80
Command ===>                                                   Scroll ===> CSR

Fault History File or View  : 'IDI.HIST.CICSC31F'

{The following line commands are available: ? (Query), V or S (View real-time
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete), H
(Duplicate history).}

   Fault_ID Job/Tran Program  User_ID  Sys/Job  Abend  Date       Time     Dup
__   F00347 RED1     PROGA    SIMCOCK  CICSC31F A*     2006/10/26 10:56:23
__   F00346 RED1     PROGA    SIMCOCK  CICSC31F ASRA   2006/10/26 10:55:20
__   F00345 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/26 10:51:39
__   F00344 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/25 17:14:40
__   F00343 PSC1     PSTESTC1 TIMOTHY  CICSC31F 4038   2006/10/25 16:52:55
__   F00342 PSC1     PSTESTC1 TIMOTHY  CICSC31F 4038   2006/10/25 16:42:50
__   F00341 PSC1     PSTESTC1 TIMOTHY  CICSC31F 4038   2006/10/25 10:20:59
__   F00340 CEMT     DFHEIP   ZHONG    CICSC31F ATNI   2006/10/24 13:59:13
__   F00339 PSC1     PSTESTC1 TIMOTHY  CICSC31F 4038   2006/10/23 17:34:50
__   F00338 OSPH     OSPLIHOU ZHONG    CICSC31F PHDD   2006/10/20 12:42:39
__   F00337 OSPH     OSPLIHOU ZHONG    CICSC31F PHDF   2006/10/20 12:01:13
__   F00336 PSC1     PSTESTC1 TIMOTHY  CICSC31F 4038   2006/10/17 18:24:45
__   F00335 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/17 17:50:12
__   F00334 CEMT     DFHEIP   TIMOTHY  CICSC31F ATNI   2006/10/17 16:07:04
__   F00333 PSC1     n/a      TIMOTHY  CICSC31F 4038   2006/10/17 14:27:51
__   F00332 PSPM     PSAPOL01 TIMOTHY  CICSC31F PA08   2006/10/17 11:01:33
__   F00331 MNAP     PSMAIN   TMROSS   CICSC31F PMN1   2006/10/16 15:43:05
__   F00330 PSC1     PSTESTC1 TIMOTHY  CICSC31F ABM3   2006/10/16 15:29:11
 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

*Figure 17-19 Over-typing Abend field with A**

```
 File  Options  View  Services  Help
IBM Fault Analyzer - Fault Entry List                          Line 1 Col 1 80
Command ===>                                                   Scroll ===> CSR
MATCH (Abend)
Fault History File or View  : 'IDI.HIST.CICSC31F'

{The following line commands are available: ? (Query), V or S (View real-time
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete), H
(Duplicate history).}

   Fault_ID Job/Tran Program  User_ID  Sys/Job  Abend  Date       Time     Dup
__   F00347 RED1     PROGA    SIMCOCK  CICSC31F AEI0   2006/10/26 10:56:23
__   F00346 RED1     PROGA    SIMCOCK  CICSC31F ASRA   2006/10/26 10:55:20
__   F00345 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/26 10:51:39
__   F00344 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/25 17:14:40
__   F00340 CEMT     DFHEIP   ZHONG    CICSC31F ATNI   2006/10/24 13:59:13
__   F00335 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/17 17:50:12
__   F00334 CEMT     DFHEIP   TIMOTHY  CICSC31F ATNI   2006/10/17 16:07:04
__   F00330 PSC1     PSTESTC1 TIMOTHY  CICSC31F ABM3   2006/10/16 15:29:11
__   F00329 MNAP     n/a      TIMOTHY  CICSC31F ASRA   2006/10/16 14:59:30
__   F00328 MNAP     n/a      TIMOTHY  CICSC31F ASRA   2006/10/16 14:33:21
__   F00327 MNAP     n/a      TIMOTHY  CICSC31F ASRA   2006/10/16 14:15:46
__   F00326 MNAP     n/a      TIMOTHY  CICSC31F ASRA   2006/10/16 14:02:35
__   F00325 MNAP     n/a      TIMOTHY  CICSC31F ASRA   2006/10/16 12:53:29
__   F00317 CDBT     DFHEIP   ZHONG    CICSC31F AEY9   2006/10/12 10:27:59
__   F00315 OSAD     OSADMIN  ZHONG    CICSC31F ADCE   2006/10/11 10:44:38
__   F00309 TIMJ     DFHSJAS  TIMOTHY  CICSC31F AJ04   2006/10/11 09:34:16
__   F00308 TIMJ     DFHSJAS  TIMOTHY  CICSC31F AJ07   2006/10/11 09:16:56
__   F00307 TIMJ     DFHSJAS  TIMOTHY  CICSC31F AJ07   2006/10/10 16:53:06
 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

*Figure 17-20 Result of matching on abend A**

- Next we wanted to match on only those Faults created by User_ID TIMOTHY. In this case we positioned the cursor under one of the TIMOTHY entries and pressed PF4. This resulted in the panel shown in Figure 17-21.

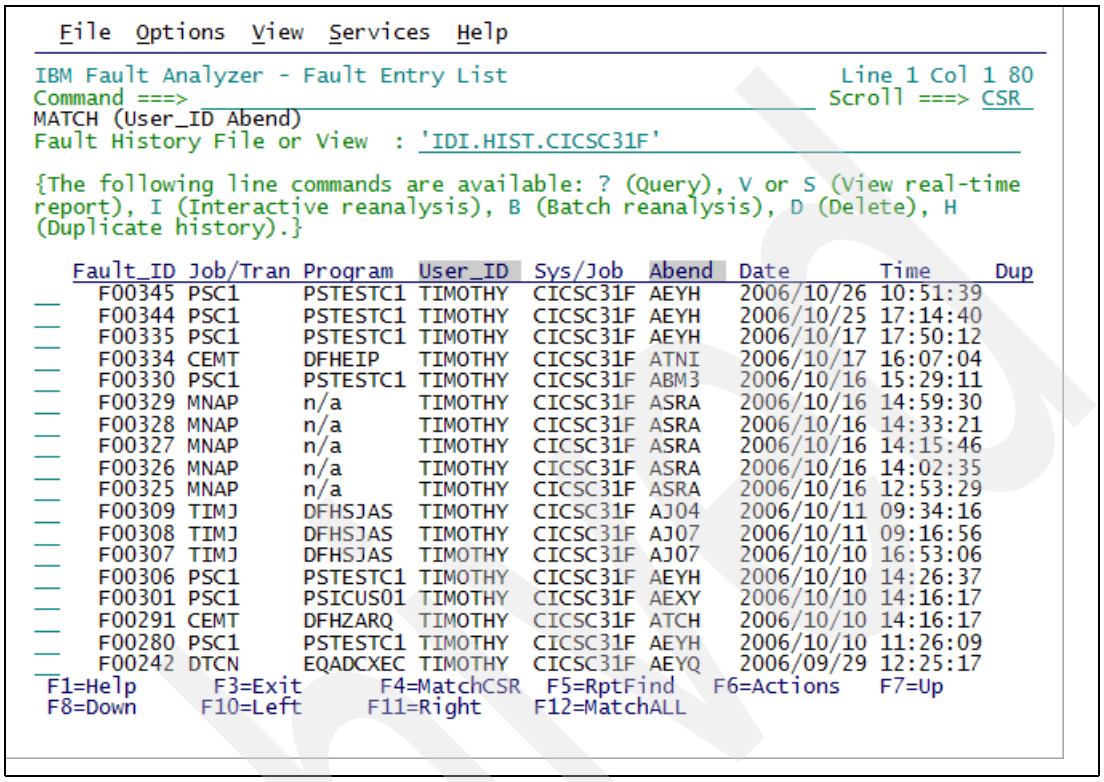


Figure 17-21 Match on Abend A and User_ID TIMOTHY*

- Finally, we wanted to match where Job/Tran was equal to PSC1. In this case we entered the MATCH command as shown inFigure 17-22. The result of pressing Enter is shown in Figure 17-23.
- Pressing PF12 MatchAll reverts to the initial display as shown in Figure 17-18 on page 451.

```
 File  Options  View  Services  Help
IBM Fault Analyzer - Fault Entry List                          Line 1 Col 1 80
Command ===> MATCH job/tran PSC1                               Scroll ===> CSR
MATCH (User_ID Abend)
Fault History File or View  : 'IDI.HIST.CICSC31F'

{The following line commands are available: ? (Query), V or S (View real-time
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete), H
(Duplicate history).}

   Fault_ID Job/Tran Program  User_ID  Sys/Job  Abend  Date       Time     Dup
__   F00345 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/26 10:51:39
__   F00344 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/25 17:14:40
__   F00335 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/17 17:50:12
__   F00334 CEMT     DFHEIP   TIMOTHY  CICSC31F ATNI   2006/10/17 16:07:04
__   F00330 PSC1     PSTESTC1 TIMOTHY  CICSC31F ABM3   2006/10/16 15:29:11
__   F00329 MNAP     n/a      TIMOTHY  CICSC31F ASRA   2006/10/16 14:59:30
__   F00328 MNAP     n/a      TIMOTHY  CICSC31F ASRA   2006/10/16 14:33:21
__   F00327 MNAP     n/a      TIMOTHY  CICSC31F ASRA   2006/10/16 14:15:46
__   F00326 MNAP     n/a      TIMOTHY  CICSC31F ASRA   2006/10/16 14:02:35
__   F00325 MNAP     n/a      TIMOTHY  CICSC31F ASRA   2006/10/16 12:53:29
__   F00309 TIMJ     DFHSJAS  TIMOTHY  CICSC31F AJ04   2006/10/11 09:34:16
__   F00308 TIMJ     DFHSJAS  TIMOTHY  CICSC31F AJ07   2006/10/11 09:16:56
__   F00307 TIMJ     DFHSJAS  TIMOTHY  CICSC31F AJ07   2006/10/10 16:53:06
__   F00306 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/10 14:26:37
__   F00301 PSC1     PSICUS01 TIMOTHY  CICSC31F AEXY   2006/10/10 14:16:17
__   F00291 CEMT     DFHZARQ  TIMOTHY  CICSC31F ATCH   2006/10/10 14:16:17
__   F00280 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/10 11:26:09
__   F00242 DTCN     EQADCXEC TIMOTHY  CICSC31F AEYQ   2006/09/29 12:25:17
 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-22 Match on Abend A , User_ID TIMOTHY and Job/Tran PSC1*

```
 File  Options  View  Services  Help
IBM Fault Analyzer - Fault Entry List                          Line 1 Col 1 80
Command ===>                                                   Scroll ===> CSR
MATCH (Job/Tran User_ID Abend)
Fault History File or View  : 'IDI.HIST.CICSC31F'

{The following line commands are available: ? (Query), V or S (View real-time
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete), H
(Duplicate history).}

   Fault_ID Job/Tran Program  User_ID  Sys/Job  Abend  Date       Time     Dup
__   F00345 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/26 10:51:39
__   F00344 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/25 17:14:40
__   F00335 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/17 17:50:12
__   F00330 PSC1     PSTESTC1 TIMOTHY  CICSC31F ABM3   2006/10/16 15:29:11
__   F00306 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/10 14:26:37
__   F00301 PSC1     PSICUS01 TIMOTHY  CICSC31F AEXY   2006/10/10 14:16:17
__   F00280 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/10/10 11:26:09
__   F00232 PSC1     PSTESTC1 TIMOTHY  CICSC31F AICA   2006/09/27 16:37:19
__   F00230 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEIZ   2006/09/27 16:16:42
__   F00229 PSC1     PSICUS01 TIMOTHY  CICSC31F ADCI   2006/09/27 16:14:37
__   F00225 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/09/27 15:15:38
__   F00221 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEIZ   2006/09/27 14:44:20
__   F00220 PSC1     PSTESTC1 TIMOTHY  CICSC31F ASRA   2006/09/27 14:40:25
__   F00219 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEI2   2006/09/27 14:14:16
__   F00214 PSC1     DFHMCE   TIMOTHY  CICSC31F APCT   2006/09/27 12:08:38
__   F00213 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEYH   2006/09/27 11:08:37
__   F00212 PSC1     EQACSEXE TIMOTHY  CICSC31F ASRA   2006/09/27 10:51:48
__   F00211 PSC1     PSTESTC1 TIMOTHY  CICSC31F AEIZ   2006/09/27 10:25:34
 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-23 Final display

17.4.7 Applying an action against a particular fault

You can apply an action to a particular fault by entering a line command against the entry. The available actions are as listed in Figure 17-5 on page 439.

The choices are to select:

- **B**: Batch reanalysis

  Submit a batch job to perform reanalysis against the selected fault entry. The analysis report is written to SYSPRINT.

- **D**: Delete

  Delete the fault entry from the fault history file. After you delete an entry, it is immediately removed from the Fault Entry List display, and is not displayed by any subsequent refresh.

- **H**: Duplicate history

  When available, shows details about faults which have occurred, that were deemed duplicates of the selected fault entry.

- **I**: Interactive reanalysis

  Run interactive reanalysis against the selected fault. After a little while, the interactive report is displayed. The interactive report does not replace the real-time analysis report.

- **V** (or **S**): View report

  View the dump analysis report about the fault entry that was generated at real-time.

- **?**: View fault entry information

  View the fault entry information. In particular, this shows the associated MVS dump data set name, if there is one.

Entering a line command against an entry, and Fault Analyzer is unable to complete the command (for example, of you attempt to run a batch dump reanalysis against a fault that has no associated dump data set, or the dump data set is unavailable), then the line command is not cleared from the line.

You can type line commands against many entries before you press Enter. In this case, Fault Analyzer attempts to honor each command, starting with the entry at the top. If Fault Analyzer is unable to honor a command, then it stops processing. It clears the line commands from each entry it was able to process, but leaves the line commands for each entry it failed to process or the entry at which it could not honor the command.

17.4.8 History file properties

To display attributes and statistics for the currently selected fault history file or combined information for a **VIEW**, first select the File menu **Fault History File Properties** option. This brings up the Fault History File Properties display, as shown in Figure 17-24.

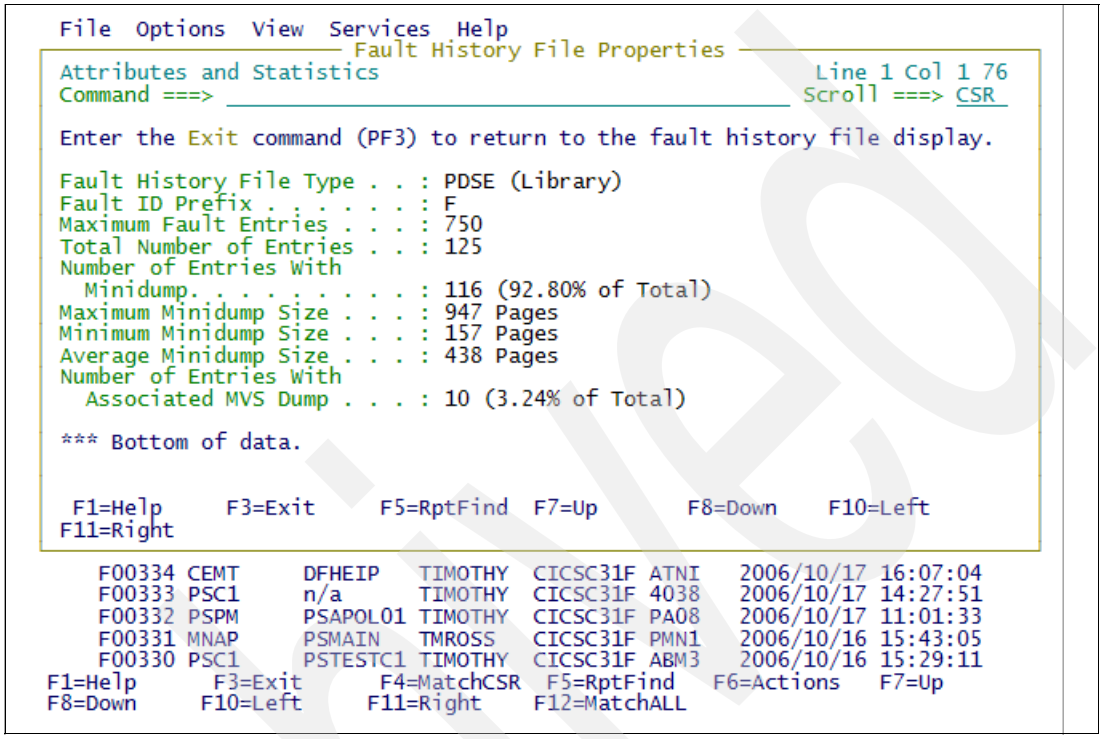


Figure 17-24 Fault history File Properties option selection

17.4.9 Resetting history file access information

To reset all information about previously accessed fault history files or views, and previously accessed fault history file entries, select the File menu Clear Last Accessed Information option.

Immediately after selecting this option, no entries are available when selecting the File menu Last Accessed Fault History File Entries option. However, the File menu Last Accessed Fault History Files or Views option shows a single entry for the currently active history file or view.

17.4.10 Action bar pull-down menus

Most of the displays used by the Fault Analyzer ISPF interface include an action bar located at the top of the panel.

The ACTIONS ISPF command (by default mapped to PF6) can be used to place the cursor at the left-most action available.

Depending on ISPF settings, you might then be able to move the cursor to other actions by pressing the Tab key. Alternatively, you can simply use the up/down/left/right arrow keys to place the cursor on the action of your choice.

Using a PF key to issue the ACTIONS command is advantageous, as, when you press the PF key again, the cursor is automatically repositioned to its starting position. Once the cursor is placed on an action bar item, press Enter to show the associated pull-down menu.

17.4.11 Fault Entry List display

Samples of the various task bar menus are shown below. Options in each menu can be selected by either entering the associated option number at the initial cursor position, or by placing the cursor (using the up/down/left/right arrow keys) anywhere on the line of the option and pressing Enter. Any options not available for selection are indicated by an asterisk (*) instead of a numerical option number.

- **File menu:** This is shown in Figure 17-25.

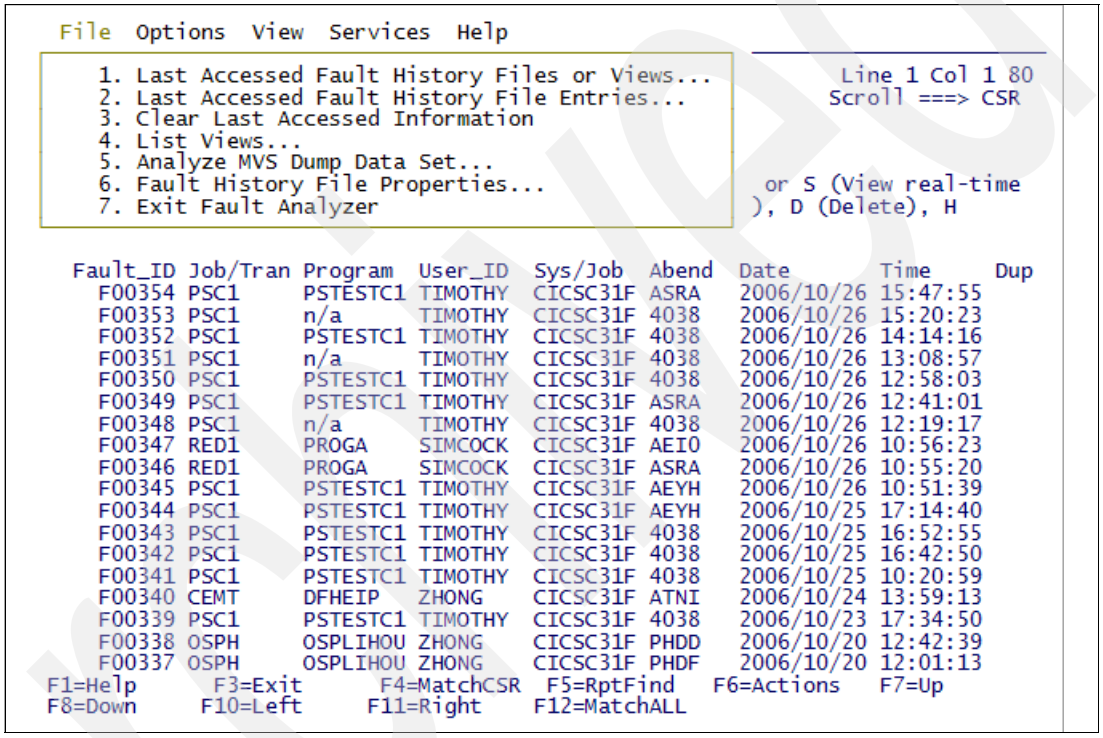


Figure 17-25 File display menu options

- **Options menu:** This is shown in Figure 17-26.

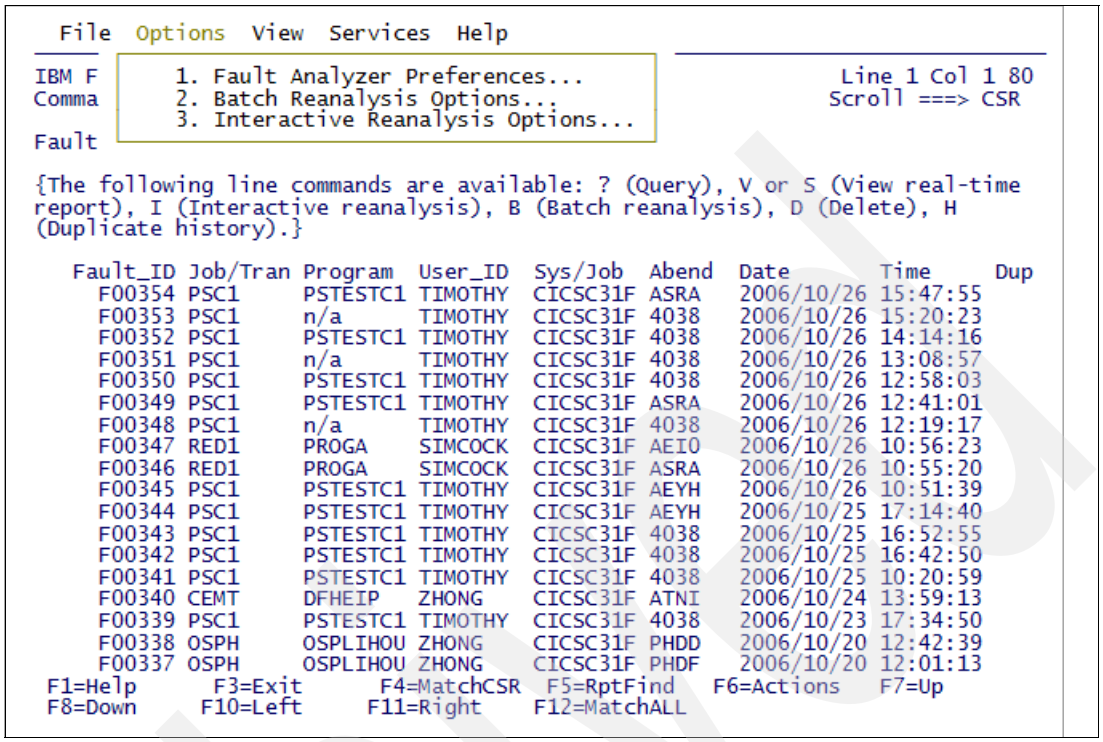


Figure 17-26 Fault entry list Options menu

- **View menu:** This is shown in Figure 17-27.

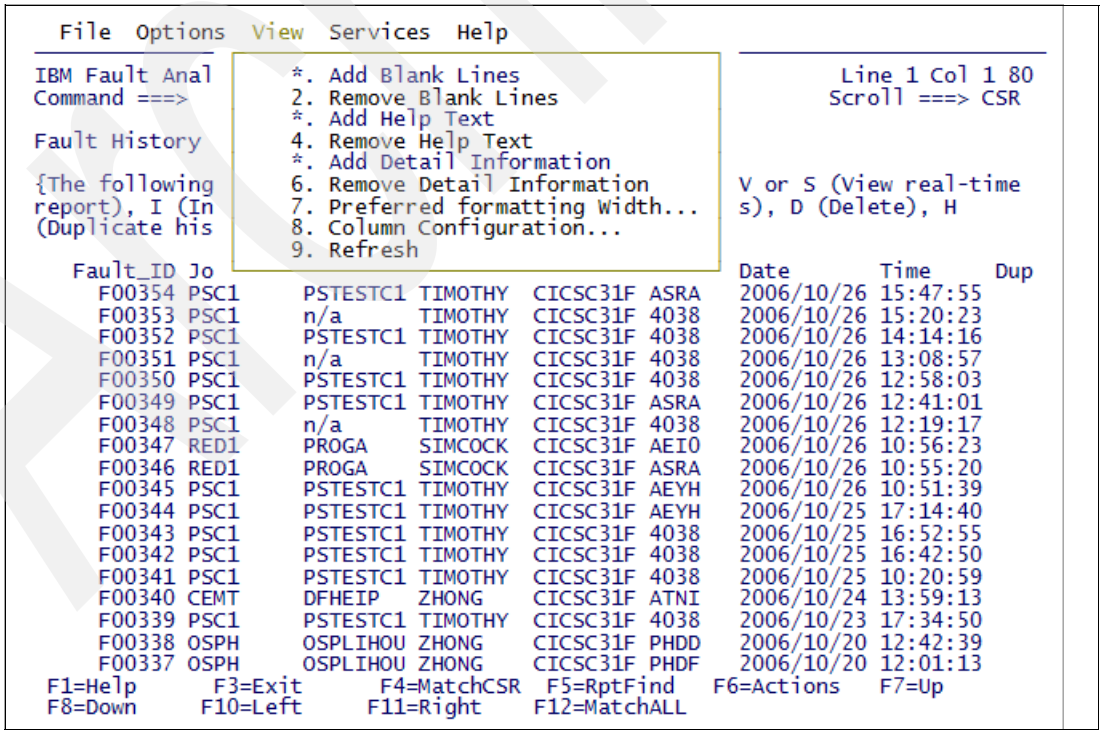


Figure 17-27 Fault entry list View menu

- **Services menu:** This is shown in Figure 17-28.

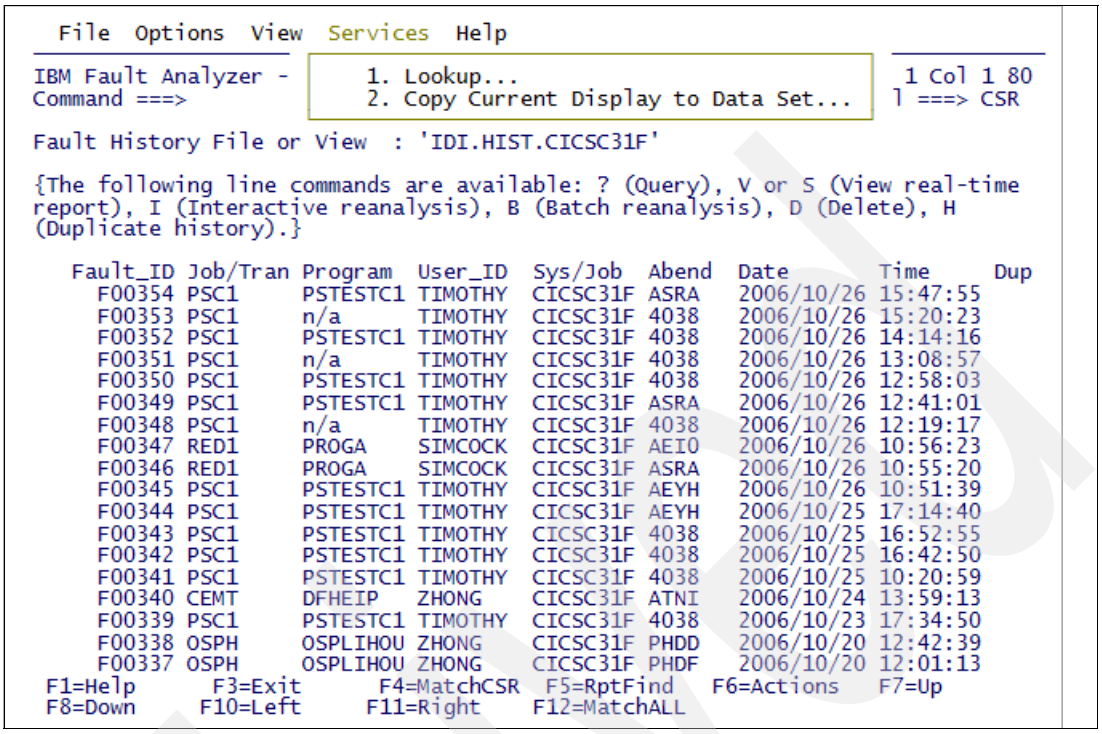


Figure 17-28 Fault entry list Services menu

- **Help menu:** This is shown in Figure 17-29.

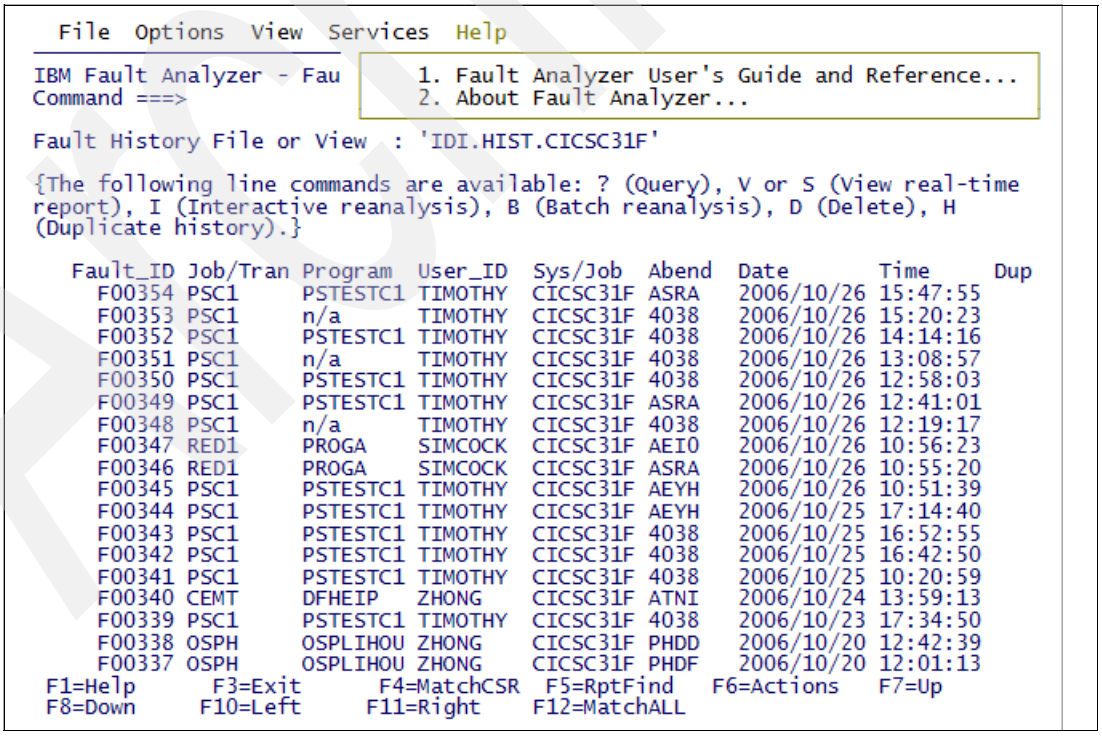


Figure 17-29 Fault entry list Help menu

18

Fault Analyzer interactive analysis

Interactive reanalysis provides several advantages over batch reanalysis:

- The sections of the report that are of interest can be selected and examined separately.
- Any storage area that is included in the associated minidump or SYSMDUMP can be displayed, regardless of whether it is included in the Fault Analyzer report.
- Source code information (if provided via compiler listing or side file) can be viewed in its entirety.
- This is the only way to analyze CICS system dumps.

18.1 Interactive reanalysis options

To specify general interactive reanalysis options that apply to your interactive reanalysis sessions only, first select **Interactive Reanalysis Options** from the Fault Entry List display Options menu (see Figure 18-1). This then shows the Interactive Reanalysis Options display, as shown in Figure 18-2.

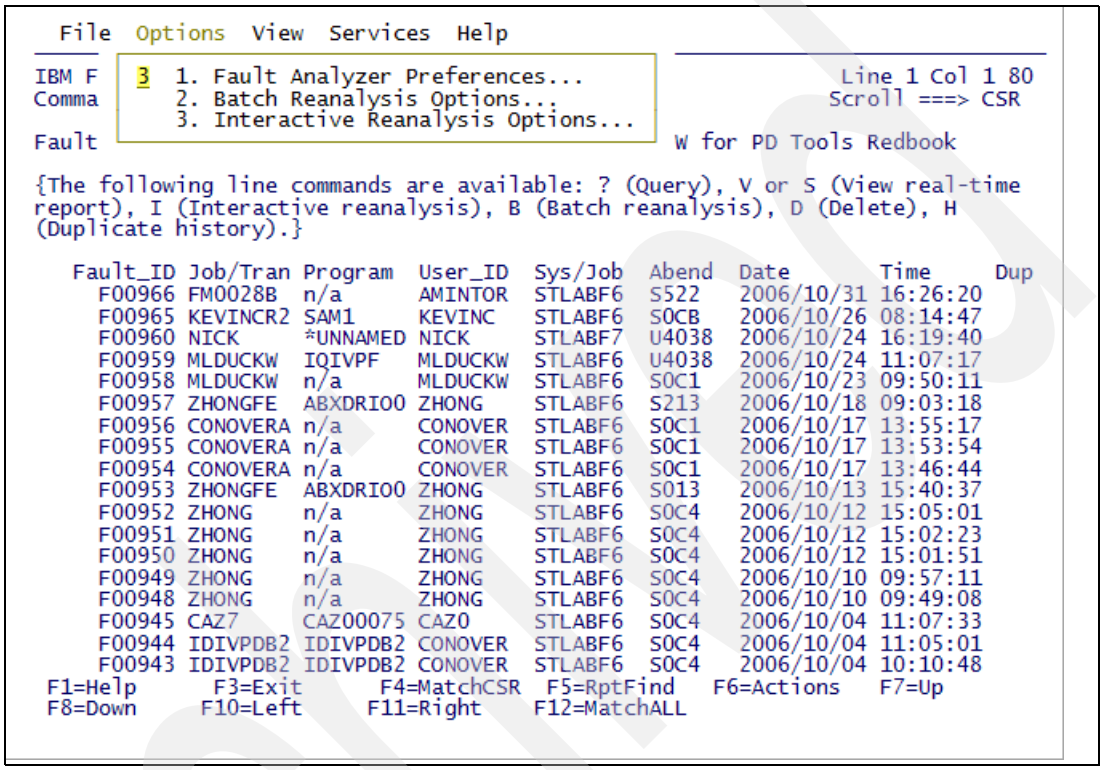


Figure 18-1 Selecting Interactive reanalysis options.

The following selections can be specified using this display:

- Options line for interactive reanalysis:
  - Options that apply to all interactive reanalysis sessions that you initiate can be specified here.
  - These options, which are the equivalent of the PARM field options used by batch reanalysis jobs, take precedence over any options specified through an options file.
- Redisplay this panel before each reanalysis:

  If this option is set to **Y**, then the Interactive Reanalysis Options display is shown each time an interactive reanalysis is requested.
- Display panel to alter allocated data sets:
  - If this option is set to **Y**, then you are presented with an ISPF EDIT panel of the pseudo JCL stream generated by Fault Analyzer, as shown in Figure 18-2.

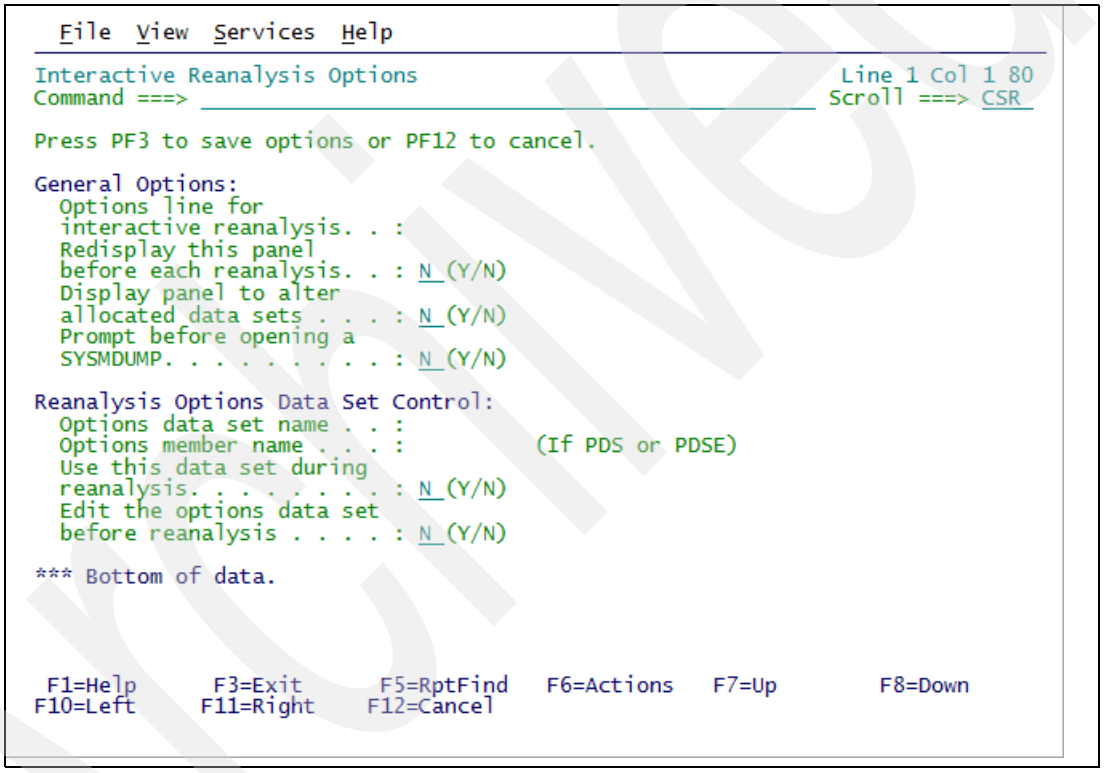


Figure 18-2 Interactive reanalysis options display

– Make any changes you wish to make in accordance with the instructions displayed, then enter the EXIT (PF3) or CANCEL (PF12) command as appropriate to initiate the interactive reanalysis. If this option is set to **N**, then the interactive reanalysis commences without first displaying the pseudo JCL EDIT screen, as shown in Figure 18-3.

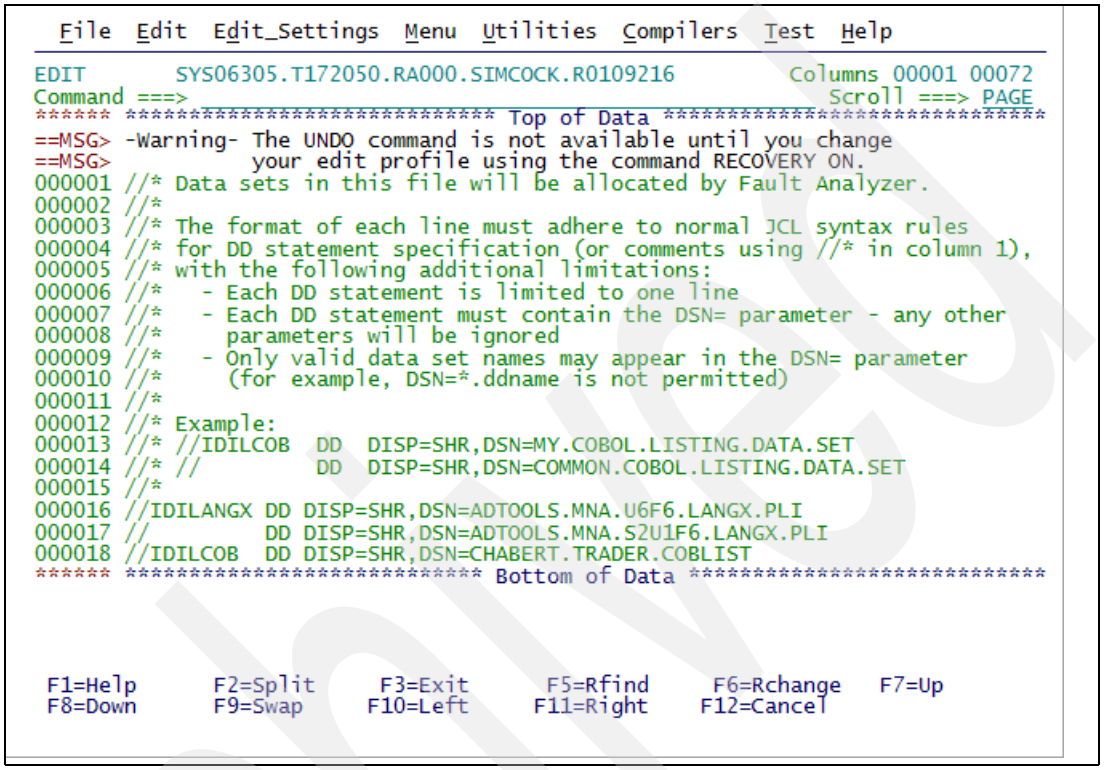


Figure 18-3 Alter allocated data sets

- Prompt before opening a SYSMDUMP:
  - If this field is set to **Y** and, during the interactive reanalysis or as a result of displaying storage locations from within the interactive report, access is required to a storage location that is not contained in the saved minidump, a display is shown before opening an associated SYSMDUMP or SVC dump data set to look for the missing storage. See Figure 18-4.
  - You are only prompted, at most, once during any interactive reanalysis session. If the open is cancelled by entering CANCEL or EXIT, no further attempts are made to open the SYSMDUMP data set. Likewise, if the open is confirmed, Fault Analyzer checks the SYSMDUMP for all references to storage locations not contained in the minidump. If this field is set to **N**, then the associated dump data set is opened if required.

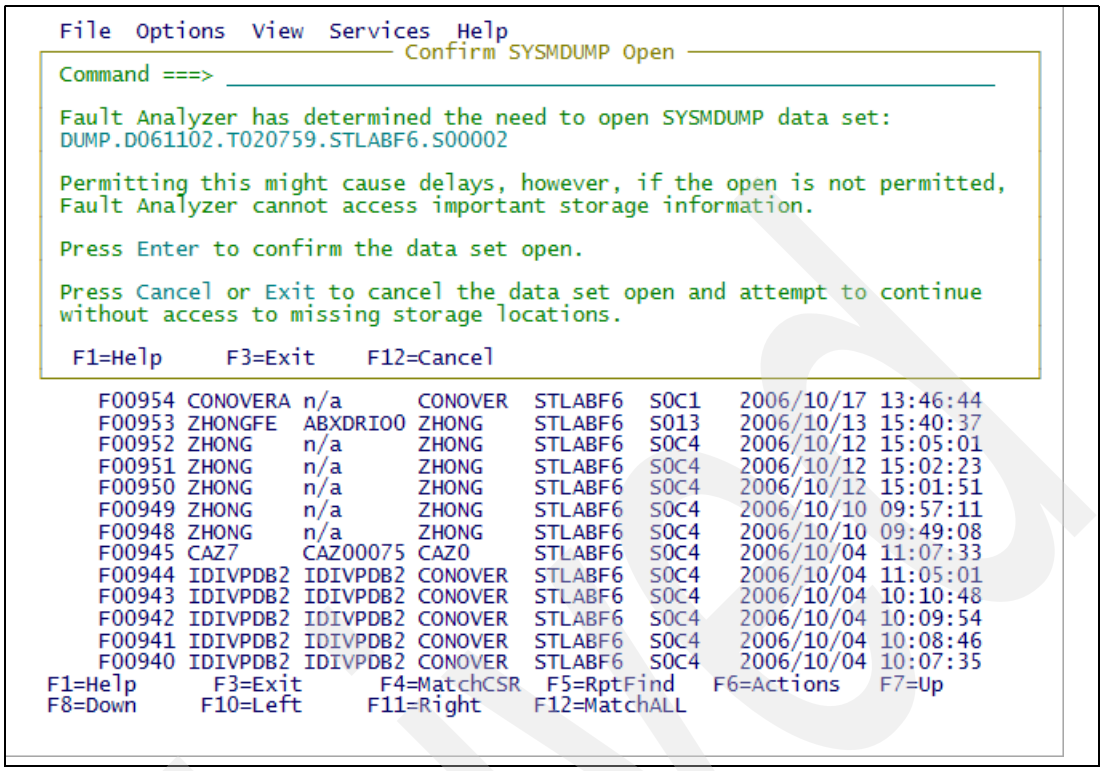


Figure 18-4 Confirm SYSMDUMP Open

- Options data set name:
  - This field can optionally specify the name of a PDS(E) data set in which a member contains Fault Analyzer options. The data set and member name are used as the IDIOPTS user options file. This data set can be used, for example, if more options are required than can fit on the options line at the top of this display.

    Note: The options data set is only used if the option, Use this data set during reanalysis, is set to **Y**.

    Options specified on the options line take precedence over options specified in this data set.

- Options member name:

  This is the member name of the data set specified in Options data set name.

- Use this data set during reanalysis:

  If this option is set to **Y**, then the data set and member name specified above are used by Fault Analyzer during the interactive reanalysis. If it is set to **N**, then the data set and member name are not used.

- Edit the options data set before reanalysis:
  - If this field is set to **Y**, then an ISPF EDIT panel of the member in the options data set specified above is presented prior to commencing the interactive reanalysis.
  - Having made any changes to the options data set (if any), enter the EXIT command (usually mapped to PF3).

18.2 Initiating interactive reanalysis of a Fault Entry

To initiate interactive reanalysis, enter the letter **I** against the fault history entry. After analysis has completed, you see a display similar to the one in Figure 18-5.

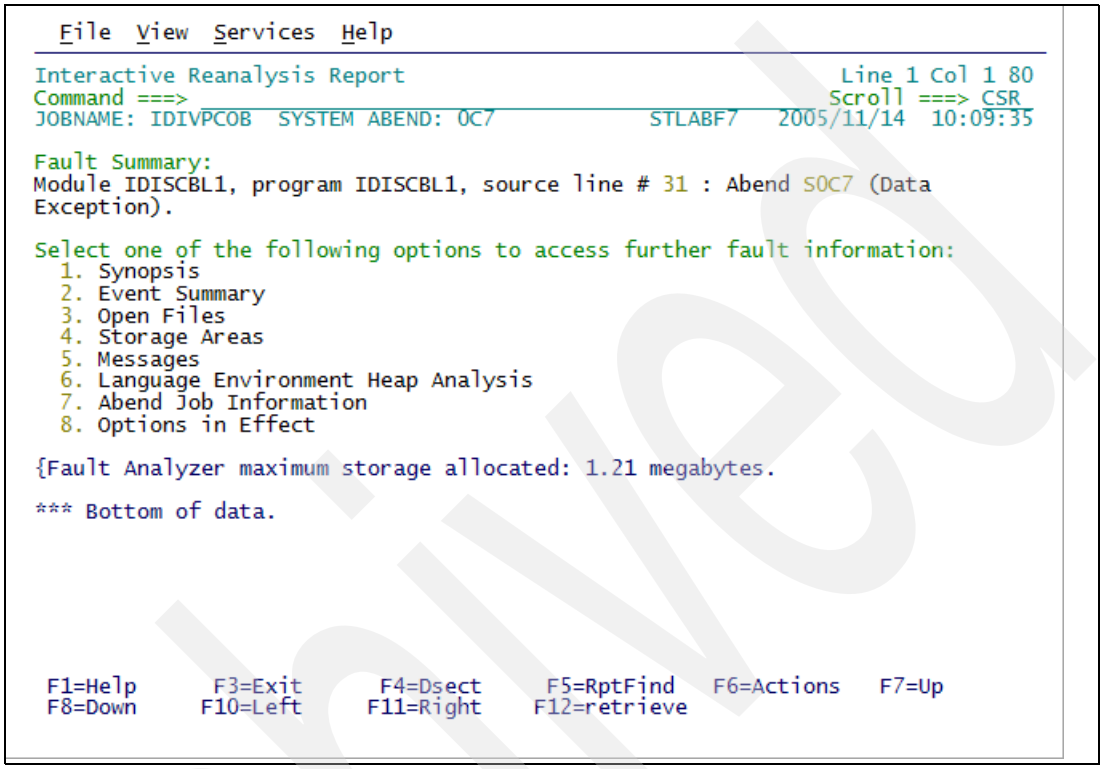


Figure 18-5 Main panel for Interactive reanalysis.

A fault summary is provided at the top of the initial display, which is equivalent to the summary provided in message IDI0002I that is issued during the real-time analysis of any fault.

The option numbers displayed can change between analyses of different faults, for example, if the fault was CICS related or if there was DB2 information. Options that are available can be entered on the command line, or the cursor can be placed on the option number and the Enter key pressed. Explained in the sections that follow are descriptions of the options shown in Figure 18-5.

18.2.1 Synopsis

Selecting option **1** from the initial interactive report display results in the display of the Synopsis section of the report, as shown in Figure 18-6.

```
  File  View  Services  Help
Synopsis                                                    Line 1 Col 1 80
Command ===>                                               Scroll ===> CSR
JOBNAME: IDIVPCOB  SYSTEM ABEND: 0C7          STLABF7   2005/11/14  10:09:35

A system abend 0C7 occurred in module IDISCBL1 program IDISCBL1 at offset
X'422'.

A program-interruption code 0007 (Data Exception) is associated with this
abend and indicates that:

  A decimal digit or sign was invalid.

The cause of the failure was program IDISCBL1 in module IDISCBL1.  The COBOL
source code that immediately preceded the failure was:

  Source
  Line #
  000029          CLEAR SECTION.
  000030          START001.
  000031              DIVIDE NUMBERX BY ERROR-COUNT GIVING BAD-RESULT.

The COBOL source code for data fields involved in the failure:

  Source
  Line #
  000011          01  NUMBERX PIC 999999 COMP-3.
  000013              05  ERROR-COUNT PIC 999999 COMP-3.
  000016          01  BAD-RESULT PIC 99 COMP-3.
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 18-6 Synopsis

Using your mouse or cursor tab, bring your cursor to source line # **000013** and press Enter. This displays the source code using the appropriate listing or sidefile as shown in Figure 18-7.

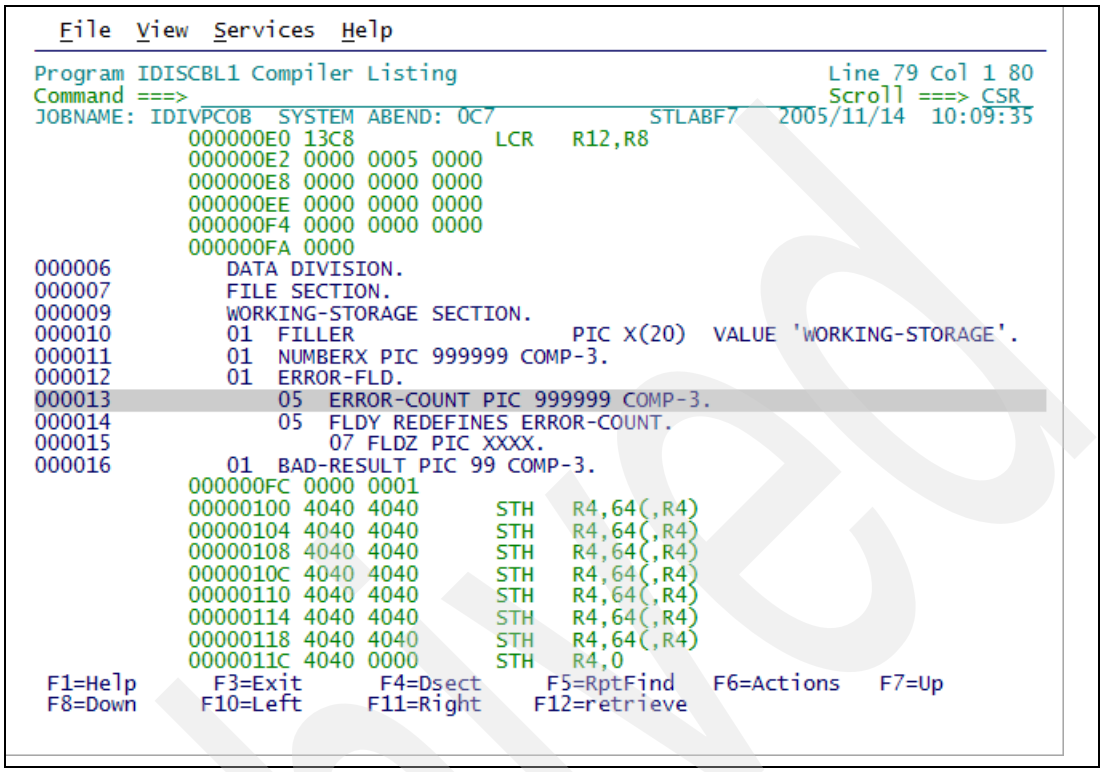


Figure 18-7 Source listing

18.2.2 Event summary

Selecting option **2** from the initial interactive report display results in the display of the Event Summary section of the report, as shown in Figure 18-8.

```
  File  View  Services  Help
Event Summary                                                Line 1 Col 1 80
Command ===>                                                Scroll ===> CSR
JOBNAME: IDIVPCOB  SYSTEM ABEND: 0C7          STLABF7   2005/11/14  10:09:35

{The following events are presented in chronological order.}

Event           Fail  Module   Program  EP
#  Type         Point Name     Name     Name     Event Location (*)     Loaded
 1 Abend S0C7   ***** IDISCBL1 IDISCBL1 IDISCBL1 L#31 P+422              SYS053
 2 Call               CEEPLPKA n/a      CEEHDSP  E+3C44                 LPA
 3 Abend U4039        CEEPLPKA n/a      CEEHSDMP E+D4                   LPA


(*) One or more of the following abbreviations might appear in the "Event
    Location" column:

    F#n  Source file number (refer to detailed event information for file
         identification)
    L#n  Source file line number
    S#n  Listing file statement number (refer to detailed event information
         for file identification)
    M+x  Offset from start of load module
    P+x  Offset from start of program
    E+x  Offset from start of entry point

*** Bottom of data.


 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 18-8 Event Summary

Individual events can be selected from this summary by placing the cursor on the event number and pressing Enter or by entering the event number on the command line. For example, selecting event **1** gave the display shown in Figure 18-9.

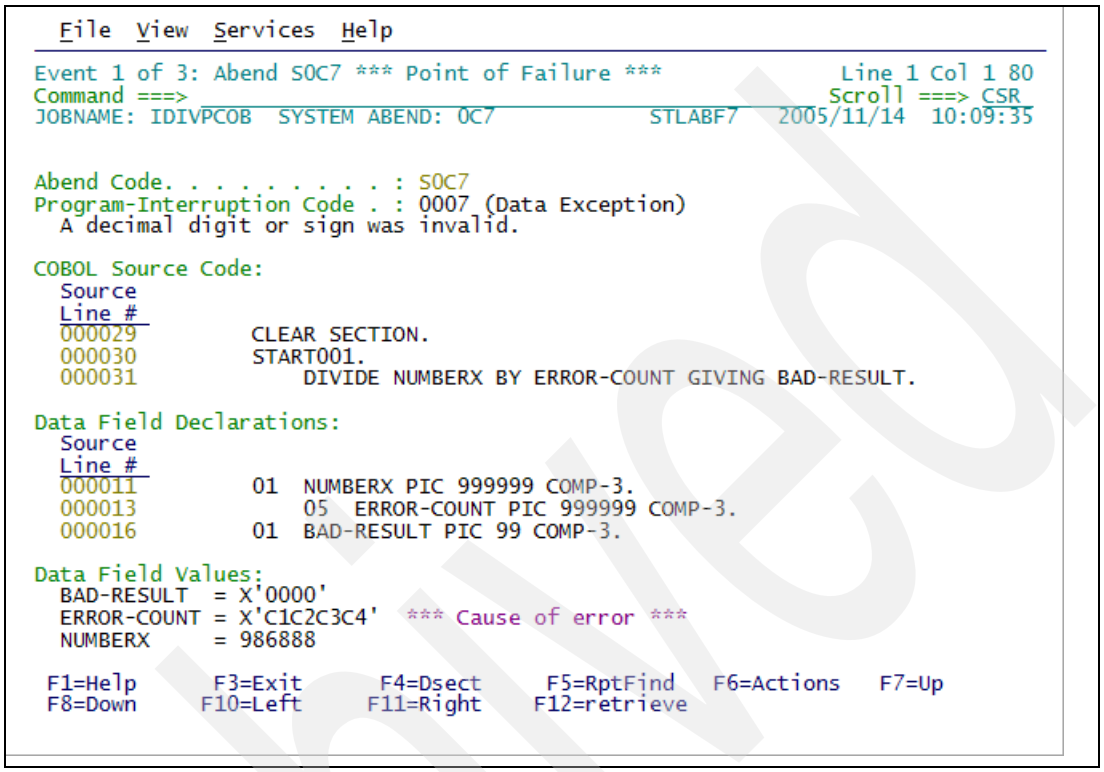


Figure 18-9 Event 1 of 3

Point-and-shoot fields are provided for most of the information in the Event Location column.

- If selecting offset-type information (M+x, P+x, or E+x), the Dump Storage display is presented for the corresponding address.
- If selecting source or listing information (L#n or S#n), the Compiler Listing display is presented for the appropriate line or statement, as shown in Figure 18-7 on page 468.

18.2.3 Open files

The Open Files display shows files that were open at the time of abend. This includes files that could not be related to any specific event, as well as files that are also shown in the detailed display for individual events. See Figure 18-10.

To see the detailed information for a file, place the cursor on the file name point-and-shoot field and press Enter.

Use the scroll commands UP (PF7), DOWN (PF8), LEFT (PF10), and RIGHT (PF11) as necessary to view the entire display.

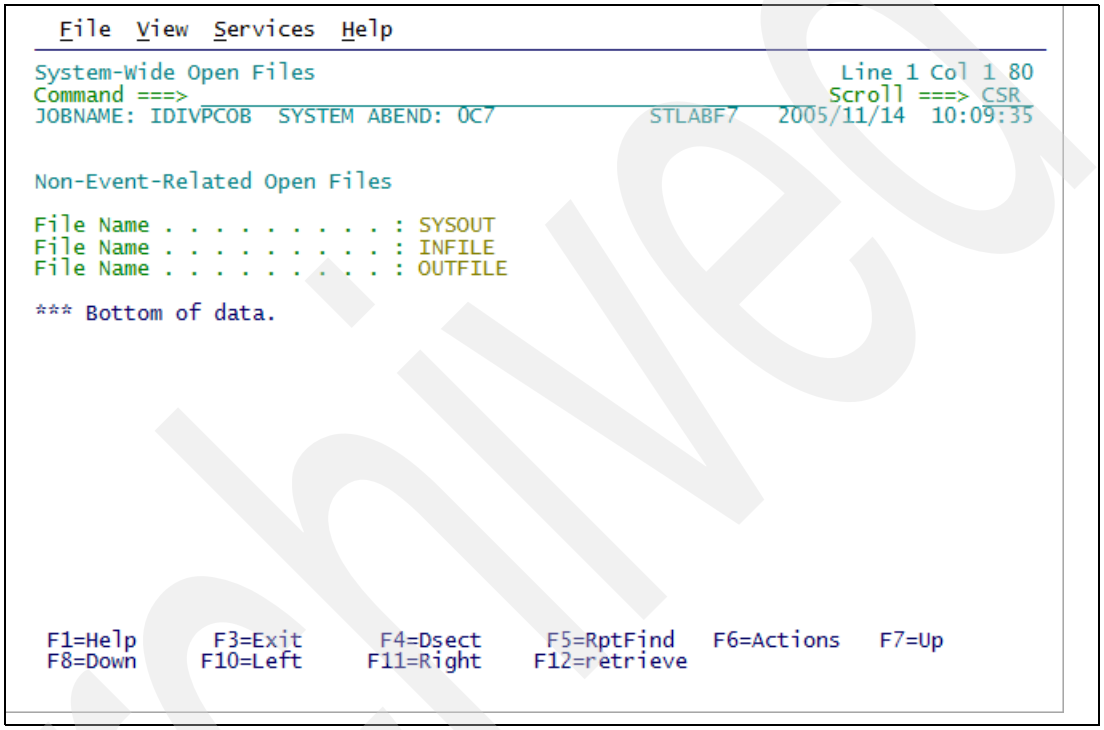


Figure 18-10 Open Files display

18.2.4 Storage Areas

The display shown in Figure 18-11 provides information about storage areas, whether formatted for a specific event or simply hex-dumped. Storage areas formatted for specific events are shown first, followed by any hex-dumped storage. To select one of the storage areas, place the cursor on the point-and-shoot field or enter the appropriate number on the command line and press Enter.

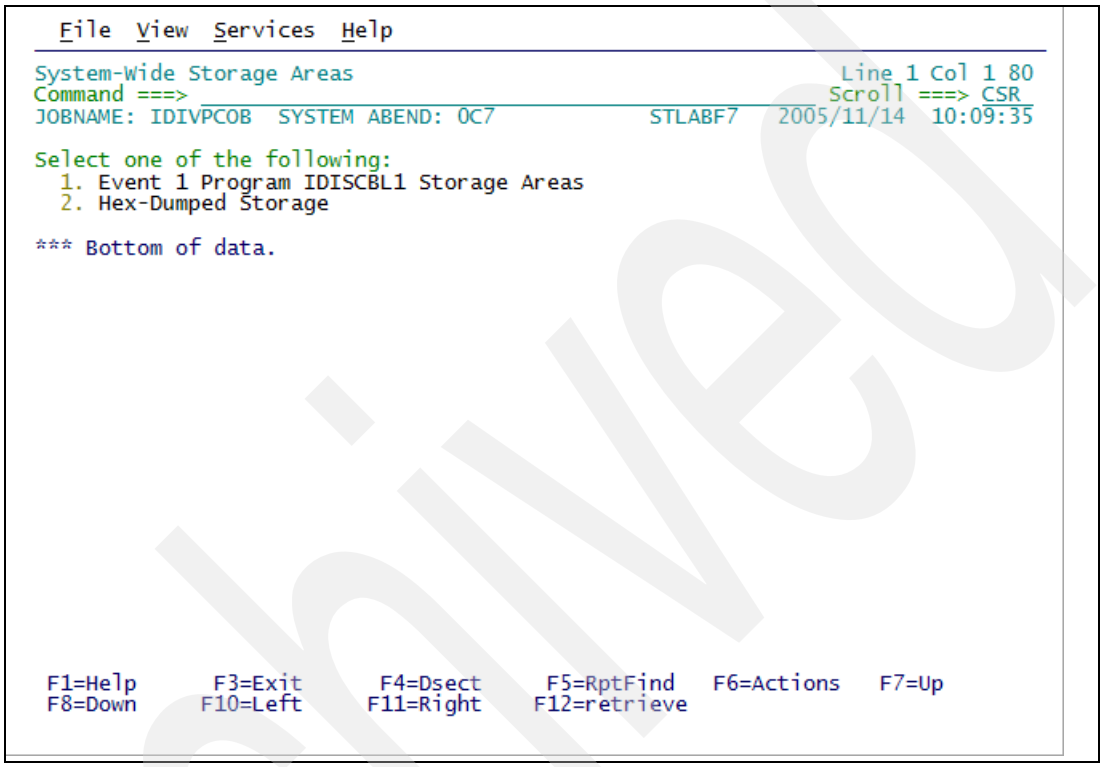


Figure 18-11 System-Wide Storage Areas

Figure 18-12 shows formatted storage areas that are associated with even 1.

```
  File  View  Services  Help
Event 1 Program IDISCBL1 Storage Areas                          Line 1 Col 1 80
Command ===>                                                   Scroll ===> CSR
JOBNAME: IDIVPCOB  SYSTEM ABEND: 0C7            STLABF7   2005/11/14  10:09:35

Task Global Table (TGT) at address 18289448 for length 328

WORKING-STORAGE SECTION
     - Collapse hex
 Off Hex Value                          Data Value          Source (Starting a
BLW=0000 at address 182AD0D0
   0 E6D6D9D2 C9D5C760 E2E3D6D9 C1C7C540 *WORKING-STORAGE *  01  FILLER
  10 40404040                           *                 *
  18 0986888F                           986888               01  NUMBERX PIC 9
                                                             01  ERROR-FLD.
  20 C1C2C3C4                           *ABCD             *      05  ERROR-COU
                                                                 05  FLDY REDE
  20 C1C2C3C4                           *ABCD             *          07 FLDZ P
  28 0000                               *..               *  01  BAD-RESULT PI


LINKAGE SECTION
BLL=0000 has not been assigned an address

*** Bottom of data.



 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 18-12 Event 1 storage areas

18.2.5 Messages

Figure 18-13 provides information about all messages, whether identified as belonging to a specific event or not. Messages for specific events are shown first, followed by messages that are not event related. To see the explanation for a message, place the cursor on the message ID point-and-shoot field, and press Enter.

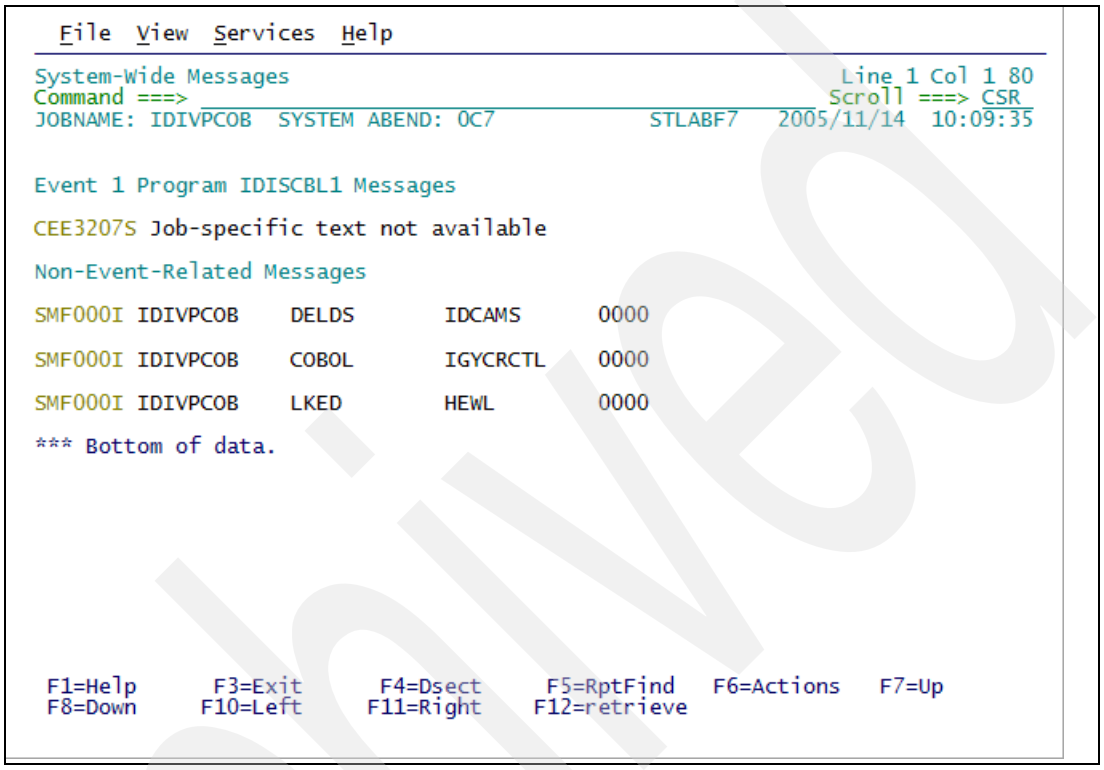


Figure 18-13 System-Wide Messages

18.2.6 Language Environment Heap Analysis

Figure 18-14 shows information about the Language Environment heap analysis.

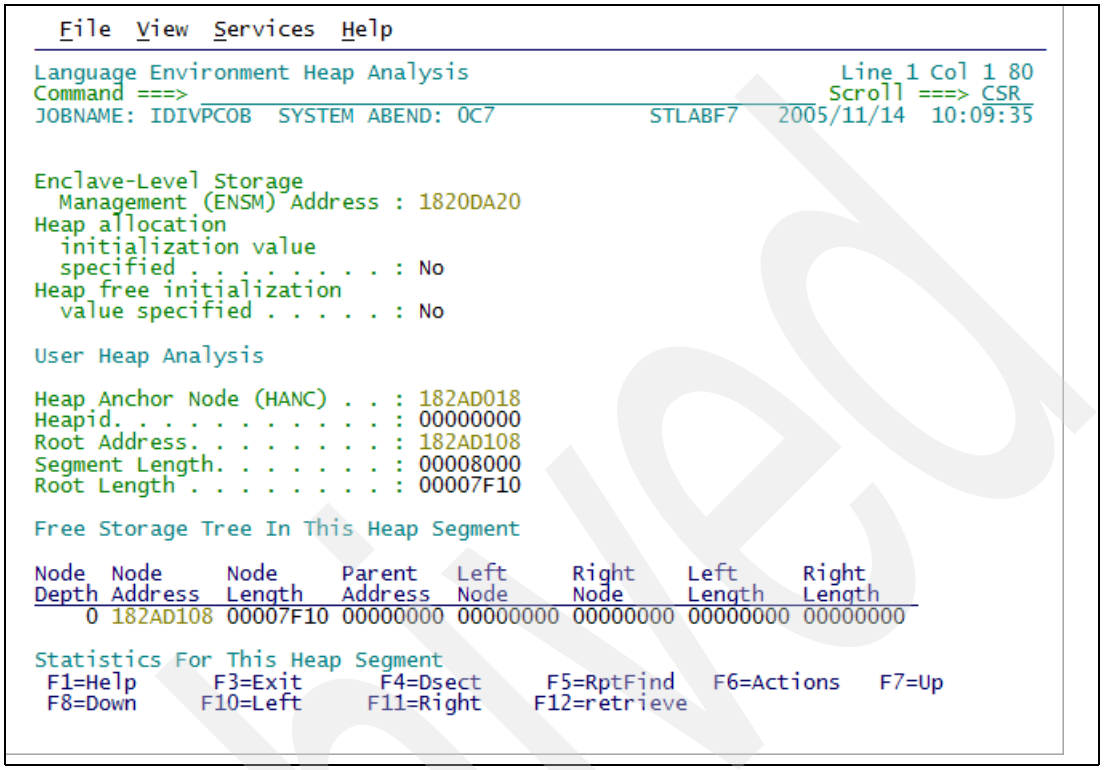


Figure 18-14 Language Environment Heap Analysis

18.2.7 Job abend information

Figure 18-15 to Figure 18-19 show information from the interactive reanalysis report after selecting the Abend Job Information option.

```
File  View  Services  Help
Abend Job Information                                          Top of data
Command ===>                                                 Scroll ===> CSR
JOBNAME: IDIVPCOB  SYSTEM ABEND: 0C7          STLABF7   2005/11/14  10:09:35

IBM Fault Analyzer Abend Job Information:

  Abend Date. . . . . . . . : 2005/11/14
  Abend Time. . . . . . . . : 10:09:35
  System Name . . . . . . . : STLABF7
  Job Type. . . . . . . . . : Batch
  Job ID. . . . . . . . . . : JOB03772
  Job Name. . . . . . . . . : IDIVPCOB
  Job Step Name . . . . . . : GO
  ASID. . . . . . . . . . . : 38
  Abend TCB Address . . . . : 008CBE88
  Job Execution Class . . . : A
  Region Size . . . . . . . : 4M
  EXEC Program Name . . . . : IDISCBL1
  User ID . . . . . . . . . : GRACINE
  Accounting Information. . : 1

Event-Related Application Programs:

  {The following list of event-related application programs is sorted by
  module link-edit date/time and program compilation date/time in reverse
  chronological order.}

  Module   Link-Edit          Program  Compilation
F1=Help      F3=Exit       F4=Dsect     F5=RptFind   F6=Actions   F7=Up
F8=Down     F10=Left      F11=Right    F12=retrieve
```

Figure 18-15 Abend Job information part 1 of 4

```
File  View  Services  Help
Abend Job Information                                        Line 26 Col 1 80
Command ===>                                                 Scroll ===> CSR
JOBNAME: IDIVPCOB  SYSTEM ABEND: 0C7          STLABF7   2005/11/14  10:09:35
  Name     Date       Time     Name     Date       Time
  IDISCBL1 2005/11/14 10:09:35 IDISCBL1 2005/11/14 10:09:34

Point Of Failure LINKEDIT Map:

  Address   Offset    Length    Type   Date       Time     Rmode  Amode  Lan
  18200D28         0      12D8  MODULE 2005/11/14 10:09:35
  18200D28         0         0  EP
  18200D28         0       688  CSECT  2005/11/14 10:09:34 ANY    MIN    COB
  182013B0       688        18  CSECT  2005/03/22          ANY    MIN    ASM
  182013C8       6A0        28  CSECT  2005/03/21          ANY    MIN    ASM
  182013F0       6C8        B0  CSECT  2005/03/21          ANY    MIN    ASM
  182014A0       778       570  CSECT  2005/03/22          ANY    31     ASM
  18201A10       CE8        B0  CSECT  2005/03/21          ANY    MIN    ASM
  18201AC0       D98       2D8  CSECT  2005/03/21          ANY    31     ASM
  18201D98      1070        E2  CSECT  2005/03/21          ANY    MIN    ASM
  18201E80      1158        70  CSECT  2005/03/21          ANY    MIN    ASM
  18201EF0      11C8        A4  CSECT  2005/03/21          ANY    MIN    ASM
  18201F98      1270        5C  CSECT  2005/03/21          ANY    MIN    ASM
  18201FF8      12D0         8  CSECT  2005/03/21          ANY    MIN    ASM

Execution Environment:

  Operating System. . . . . : z/OS V1R7M0
  Data Facility Product . . : DFSMS z/OS V1R7M0
F1=Help      F3=Exit       F4=Dsect     F5=RptFind   F6=Actions   F7=Up
F8=Down     F10=Left      F11=Right    F12=retrieve
```

Figure 18-16 Abend Job information part 2 of 5

```
File  View  Services  Help
Abend Job Information                                          Line 51 Col 1 80
Command ===>                                                   Scroll ===> CSR
JOBNAME: IDIVPCOB  SYSTEM ABEND: 0C7          STLABF7   2005/11/14  10:09:35
  Job Entry Subsystem . . . : JES2
  Language Environment. . . : V1 R7.0
  CPU Model . . . . . . . . : 2064

Language Environment Run-Time Options:

  Last Where Set                Option
  PARMLIB(CEEPRM00)               ABPERC(NONE)
  Invocation command              ABTERMENC(RETCODE)
  PARMLIB(CEEPRM00)             NOAIXBLD
  PARMLIB(CEEPRM00)               ALL31(ON)
  PARMLIB(CEEPRM00)               ANYHEAP(16384,8192,ANYWHERE,FREE)
  PARMLIB(CEEPRM00)             NOAUTOTASK
  PARMLIB(CEEPRM00)               BELOWHEAP(8192,4096,FREE)
  PARMLIB(CEEPRM00)               CBLOPTS(ON)
  PARMLIB(CEEPRM00)               CBLPSHPOP(ON)
  PARMLIB(CEEPRM00)               CBLQDA(OFF)
  PARMLIB(CEEPRM00)               CHECK(ON)
  PARMLIB(CEEPRM00)               COUNTRY(US)
  PARMLIB(CEEPRM00)             NODEBUG
  PARMLIB(CEEPRM00)               DEPTHCONDLMT(10)
  Installation default            ENVAR("")
  PARMLIB(CEEPRM00)               ERRCOUNT(0)
  PARMLIB(CEEPRM00)               ERRUNIT(6)
  PARMLIB(CEEPRM00)               FILEHIST
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 18-17 Abend Job information part 3 of 5

```
File  View  Services  Help
Abend Job Information                                          Line 76 Col 1 80
Command ===>                                                   Scroll ===> CSR
JOBNAME: IDIVPCOB  SYSTEM ABEND: 0C7          STLABF7   2005/11/14  10:09:35
  PARMLIB(CEEPRM00)               FILETAG(NOAUTOCVT,NOAUTOTAG)
  Default setting               NOFLOW
  PARMLIB(CEEPRM00)               HEAP(32768,32768,ANYWHERE,KEEP,8192,4096)
  PARMLIB(CEEPRM00)               HEAPCHK(OFF,1,0,0,0)
  PARMLIB(CEEPRM00)               HEAPPOOLS(OFF,8,10,32,10,128,10,256,10,1024
  PARMLIB(CEEPRM00)               INFOMSGFILTER(OFF,,,,)
  PARMLIB(CEEPRM00)               INQPCOPN
  PARMLIB(CEEPRM00)               INTERRUPT(OFF)
  PARMLIB(CEEPRM00)               LIBSTACK(4096,4096,FREE)
  PARMLIB(CEEPRM00)               MSGFILE(SYSOUT,FBA,121,0,NOENQ)
  PARMLIB(CEEPRM00)               MSGQ(15)
  PARMLIB(CEEPRM00)               NATLANG(ENU)
  Mapped                        NONONIPTSTACK(See THREADSTACK)
  PARMLIB(CEEPRM00)               OCSTATUS
  PARMLIB(CEEPRM00)             NOPC
  PARMLIB(CEEPRM00)               PLITASKCOUNT(20)
  PARMLIB(CEEPRM00)               POSIX(OFF)
  PARMLIB(CEEPRM00)               PROFILE(OFF,"")
  PARMLIB(CEEPRM00)               PRTUNIT(6)
  PARMLIB(CEEPRM00)               PUNUNIT(7)
  PARMLIB(CEEPRM00)               RDRUNIT(5)
  PARMLIB(CEEPRM00)               RECPAD(OFF)
  PARMLIB(CEEPRM00)               RPTOPTS(ON)
  PARMLIB(CEEPRM00)               RPTSTG(ON)
  PARMLIB(CEEPRM00)             NORTEREUS
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 18-18 Abend Job information part 4 of 5

```
  File  View  Services  Help
Abend Job Information                                          Line 92 Col 1 80
Command ===>                                                   Scroll ===> CSR
JOBNAME: IDIVPCOB  SYSTEM ABEND: 0C7           STLABF7   2005/11/14  10:09:35
  PARMLIB(CEEPRM00)                POSIX(OFF)
  PARMLIB(CEEPRM00)                PROFILE(OFF,"")
  PARMLIB(CEEPRM00)                PRTUNIT(6)
  PARMLIB(CEEPRM00)                PUNUNIT(7)
  PARMLIB(CEEPRM00)                RDRUNIT(5)
  PARMLIB(CEEPRM00)                RECPAD(OFF)
  PARMLIB(CEEPRM00)                RPTOPTS(ON)
  PARMLIB(CEEPRM00)                RPTSTG(ON)
  PARMLIB(CEEPRM00)              NORTEREUS
  PARMLIB(CEEPRM00)              NOSIMVRD
  PARMLIB(CEEPRM00)                STACK(131072,131072,ANYWHERE,KEEP,524288,13
  PARMLIB(CEEPRM00)                STORAGE(NONE,NONE,NONE,0)
  Invocation command               TERMTHDACT(UADUMP,,96)
  PARMLIB(CEEPRM00)              NOTEST(ALL,"*","PROMPT","INSPPREF")
  PARMLIB(CEEPRM00)                THREADHEAP(4096,4096,ANYWHERE,KEEP)
  Ignored                          THREADSTACK(OFF,4096,4096,ANYWHERE,KEEP,131
  PARMLIB(CEEPRM00)                TRACE(OFF,4096,DUMP,LE=0)
  Invocation command               TRAP(ON,SPIE)
  PARMLIB(CEEPRM00)                UPSI(00000000)
  PARMLIB(CEEPRM00)              NOUSRHDLR(,)
  PARMLIB(CEEPRM00)                VCTRSAVE(OFF)
  Installation default             XPLINK(OFF)
  PARMLIB(CEEPRM00)                XUFLOW(AUTO)

*** Bottom of data.
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 18-19 Abend Job information part 5 of 5

18.2.8 Options in effect

The Options in Effect display shows information about the Fault Analyzer options that were in effect during the reanalysis of the fault; see Figure 18-20.

```
  File  View  Services  Help
Options in Effect                                          Line 1 Col 1 80
Command ===>                                              Scroll ===> CSR
JOBNAME: IDIVPCOB  SYSTEM ABEND: 0C7          STLABF7   2005/11/14  10:09:35

IBM Fault Analyzer Options in Effect:

  NoErrorHandler
  FaultID(F00677)
  Language(ENU)
  NoLocale
  NoPermitLangx

  Data Sets:

    {The following Fault Analyzer data set or path names were either
    preallocated, specified via DataSets options, or provided as defaults.}

    DDname   Data Set or Path Name
    IDIBOOKS ADTOOLS.FA710.SIDIBOOK
    IDICACHE IDI.CACHE
    IDIDOC   ADTOOLS.FA710.SIDIDOC1
    IDIHIST  IDI.HIST
    IDILANGX ADTOOLS.MNA.U6F6.LANGX.PLI
             ADTOOLS.MNA.S2U1F6.LANGX.PLI
    IDILCOB  GRACINE.BOOK2005.IVPCB.LISTINGS
             CHABERT.TRADER.COBLIST
    IDIMAPS  ADTOOLS.FA710.SIDIMAPS

 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 18-20 Options in Effect

18.2.9 Other possible options

As well as the options described in the preceding sections, the following options can also be displayed:

- **CICS Information (including)**
  - CICS Control blocks
  - CICS Transaction Storage
  - Last 3270 Screen
  - CICS Trace
  - CICS Recovery information
  - CICS Channels and Containers (see 18.2.10, “CICS Webservices Awareness” on page 480)
- **DB2 Information (including)**
  - DB2 Version
  - Plan Name
  - Plan Owner
  - Package information
  - Last executed SQL statement
- **IMS Information (including)**
  - IMS Version
  - Region type
  - Subsystem name
  - PSB name
- **Java**
- **WebSphere**
- **MTRACE**

18.2.10 CICS Webservices Awareness

Fault Analyzers support for CICS channels and containers is also aware of those containers that directly relate to CICS Webservices functions. As such, Fault Analyzer provides additional information for CICS containers that have recognized CICS names.

Shown in Figure 18-21 and Figure 18-22 are the CICS levels, Commareas, and Channels for a Webservices transaction that issued an EXEC CICS DUMP command. As can be seen, Fault Analyzer displays all the container names and indicates if the data stored in each container is ASCII or EBCDIC.

```
  File  View  Services  Help
CICS Levels, Commareas, and Channels                          Line 1 Col 1 80
Command ===> ____________________________________________________ Scroll ===> CSR
TRANID: CPIH      DUMP CODE: CVER              FAE1      2006/09/14  15:44:13

Number of Link Levels . . . : 5

Level 1 of 5 : Load Module DFHPIDSH

Fault Analyzer Event #. . . : n/a

Load Module Address . . . . : 178E7F90
Entry Point Address . . . . : 178E7FB8
Addressing Mode . . . . . : AMODE 31
Current Channel Name. . . . : DFHAHC-V1
Count of Containers . . . . : 18

Container ADRIAN2 at address 16A4A210 has a length of X'B'
   Data Offset X'000000' HEX     C8C1D5C3 17706868 177068

Container ADRIAN at address 16A4A440 has a length of X'2800'
   Data Offset X'000000' HEX     C8C1D5C3 17706868 17706868 00000000  177118B8
   Data Offset X'000FD8' HEX     00000000 00000000 00000000 00000000  00000000
   Data Offset X'001FB0' HEX     00000000 00000000 00000000 00000000  00000000

Container DFHWS-DATA at address 16A4A3A0 has a length of X'233'
   Data Offset X'000000' EBCDIC  T 0000.urrent TCB 008E3188.

Container DFHWS-OPERATION at address 16A4A260 has a length of X'10'
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 18-21 CICS levels, Commareas, and channels (1 of 2)

```
  File  View  Services  Help
CICS Levels, Commareas, and Channels                          Line 26 Col 1 80
Command ===>                                                  Scroll ===> CSR
TRANID: CPIH      DUMP CODE: CVER              FAE1      2006/09/14  15:44:13
   Data Offset X'000000' EBCDIC  CWSTESTOperation

Container DFHWS-BODY at address 16A4A580 has a length of X'3FC'
   Data Offset X'000000' ASCII   <SOAP-ENV:Body>

Container DFHWS-XMLNS at address 16A4A530 has a length of X'3B'
   Data Offset X'000000' ASCII   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/so

Container DFHWS-SOAPLEVEL at address 16A4A4E0 has a length of X'4'
   Data Offset X'000000' HEX     00000001

Container DFH-HANDLERPLIST at address 16A4A3F0 has a length of X'0'

Container DFHRESPONSE at address 16A4A490 has a length of X'0'

Container DFHFUNCTION at address 16A4A350 has a length of X'10'
   Data Offset X'000000' EBCDIC  PROCESS-REQUEST

Container DFHWS-URI at address 16A4A300 has a length of X'B'
   Data Offset X'000000' EBCDIC  /fa/cwstest

Container DFHREQUEST at address 16A4A2B0 has a length of X'502'
   Data Offset X'000000' ASCII   <?xml version="1.0" encoding="UTF-8" ?>

Container DFH-SERVICEPLIST at address 16A4A1C0 has a length of X'0'
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 18-22 CICS levels, Commareas, and channels (2 of 2)

Selecting the DFHWS-BODY container **DATA** point-and-shoot field results in the display shown in Figure 18-23. Notice that Fault Analyzer detects that the data in this container is ASCII and hence performs the appropriate translation of the hex values into ASCII rather than EBCDIC.

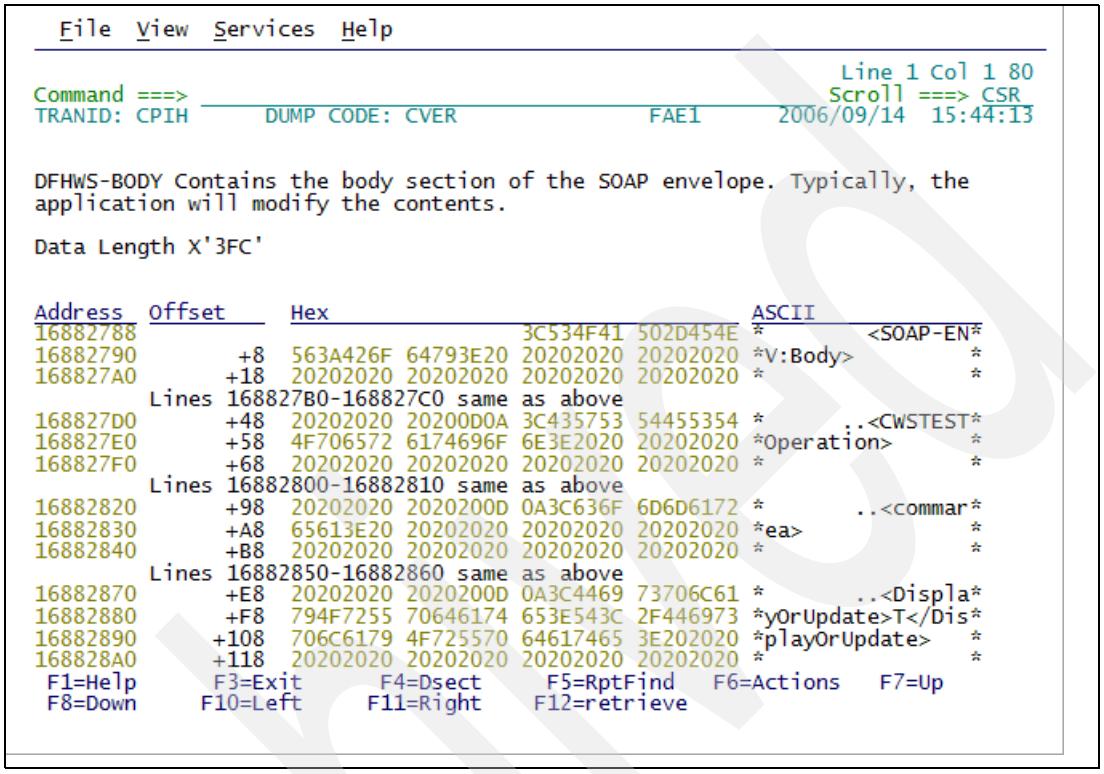


Figure 18-23 DFHWS-BODY container description and ASCII translated data

18.3 Expanding messages and abend codes

Messages or abend codes are initially not expanded when using the interactive dump reanalysis feature of Fault Analyzer. This is to prevent the necessity of scrolling through potentially long explanations to see report items that might follow. Instead, to view the explanation for messages or abend codes in the interactive report, you can place the cursor on the message identifier or abend code and press Enter. This brings up a display similar to what you see in the batch report. For example pressing ENTER under the S0C7 shown in Figure 18-5 would result in the message explanation shown in Figure 18-24.

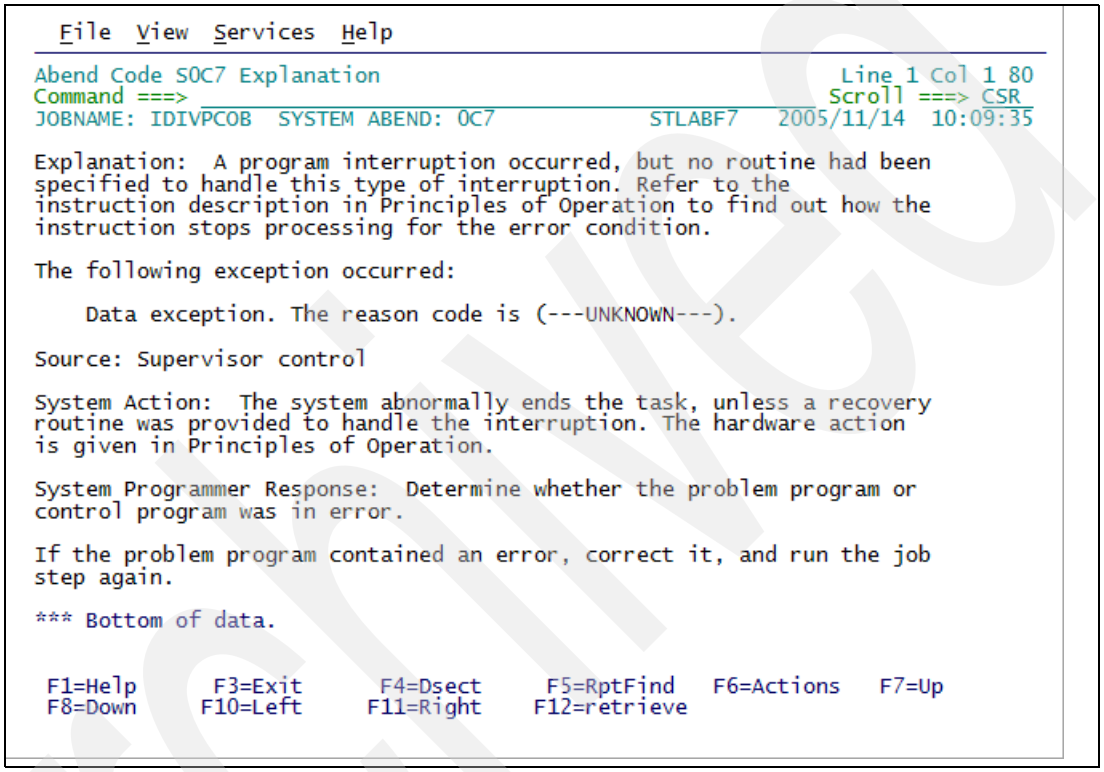


Figure 18-24 S0C7 Message explanation

18.4 Message LOOKUP command

The LOOKUP command, which can be entered on any command line in an interactive Fault Analyzer session, can be used to get an explanation of an abend code or message ID. Entering LOOKUP without any parameters displays the screen shown in Figure 18-25.

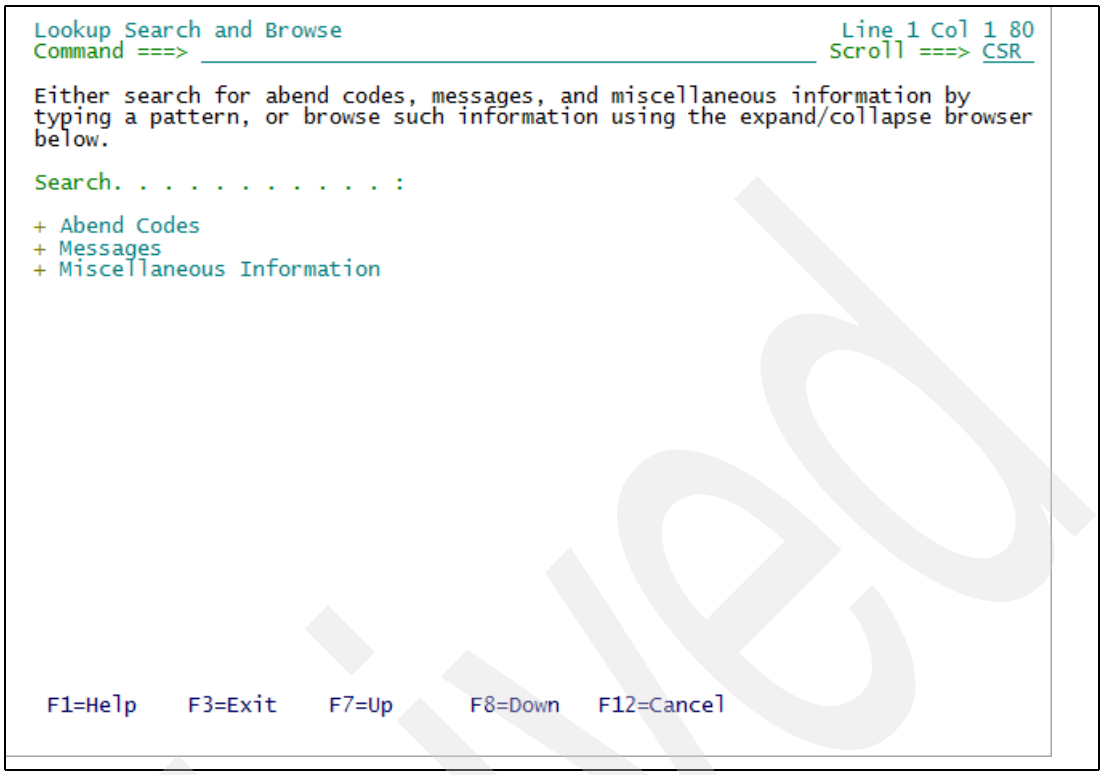


Figure 18-25 LOOKUP command main panel

From this display, the various categories can be expanded or collapsed by placing the cursor under the **+** or **-** and pressing ENTER. For example, see Figure 18-26.

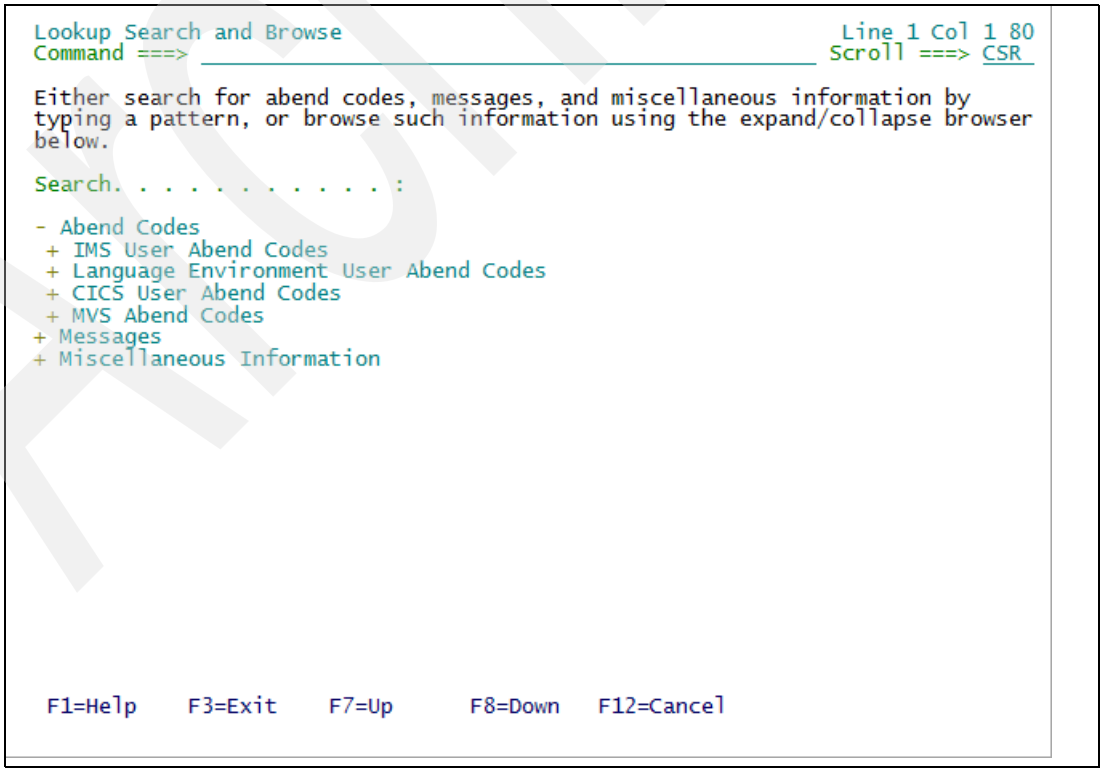


Figure 18-26 LOOKUP - expanded Abend Codes

You can continue to use the **+** and **-** fields to drill down to the desired abend or message code or alternatively you can enter its code directly into the Search field - wild cards are supported. For example, entering **`S0C4`** displays the explanation shown in Figure 18-27.

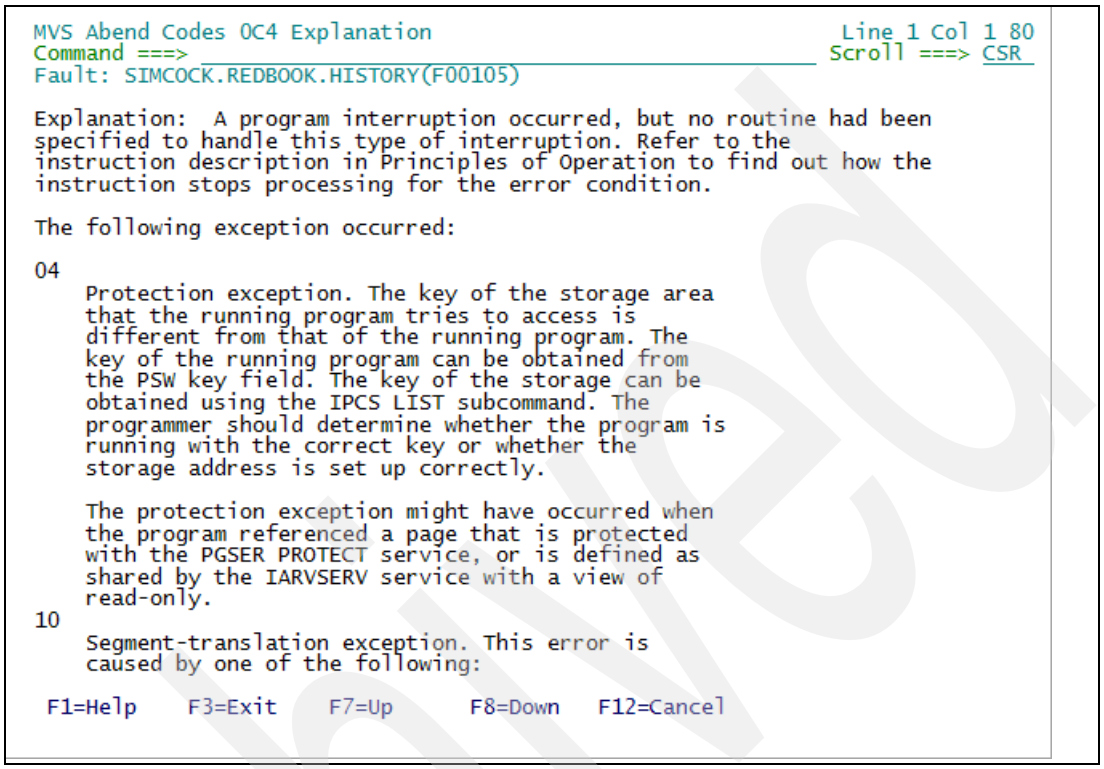


Figure 18-27 S0C4 explanation

18.5 Creating and managing user notes

User notes are comments that the interactive user can add against any storage location. They are saved in the history file fault entry and are available to all users. User notes are created from the Dump Storage display by placing the cursor on the area of storage to which the note applies, typing one or more characters that are distinguishable from hexadecimal digits, and pressing Enter.

For example, given the Dump Storage display shown in Figure 18-28 placing the cursor at the address and typing `Note to A Developer I suspect`, the display in Figure 18-29 would be expected.

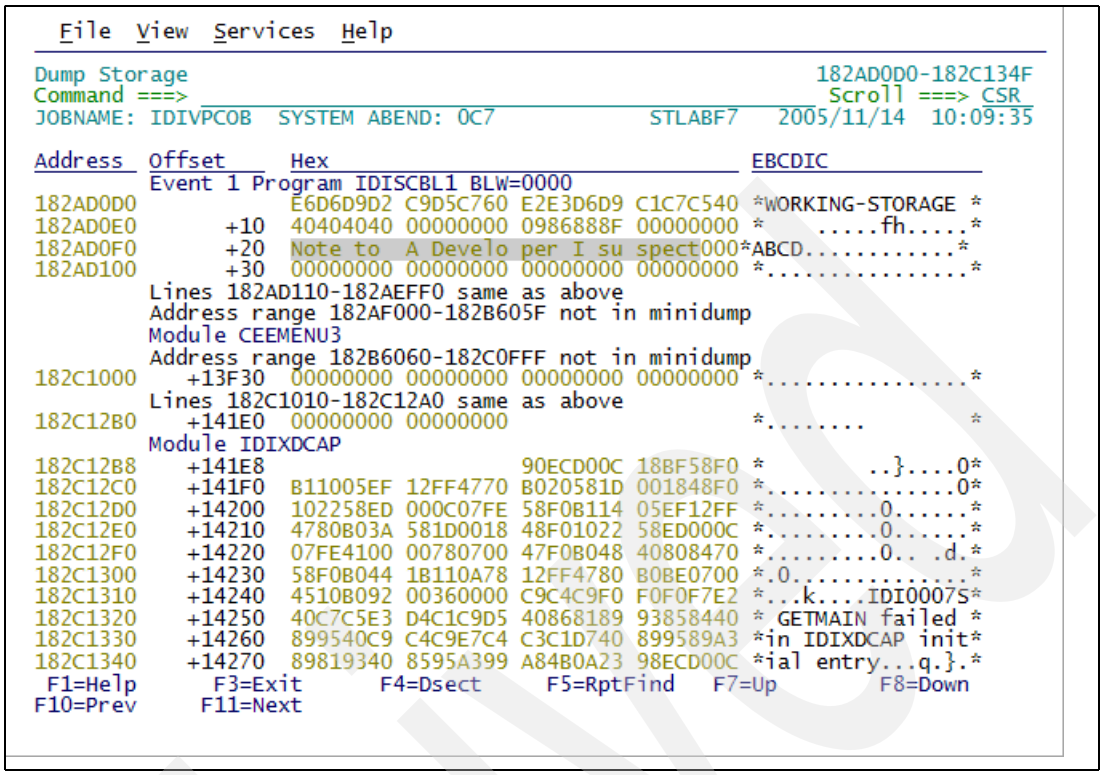


Figure 18-28 Typing a user note

Pressing ENTER would then display an ISPF EDIT session where addition data can be entered. As shown in Figure 18-29.

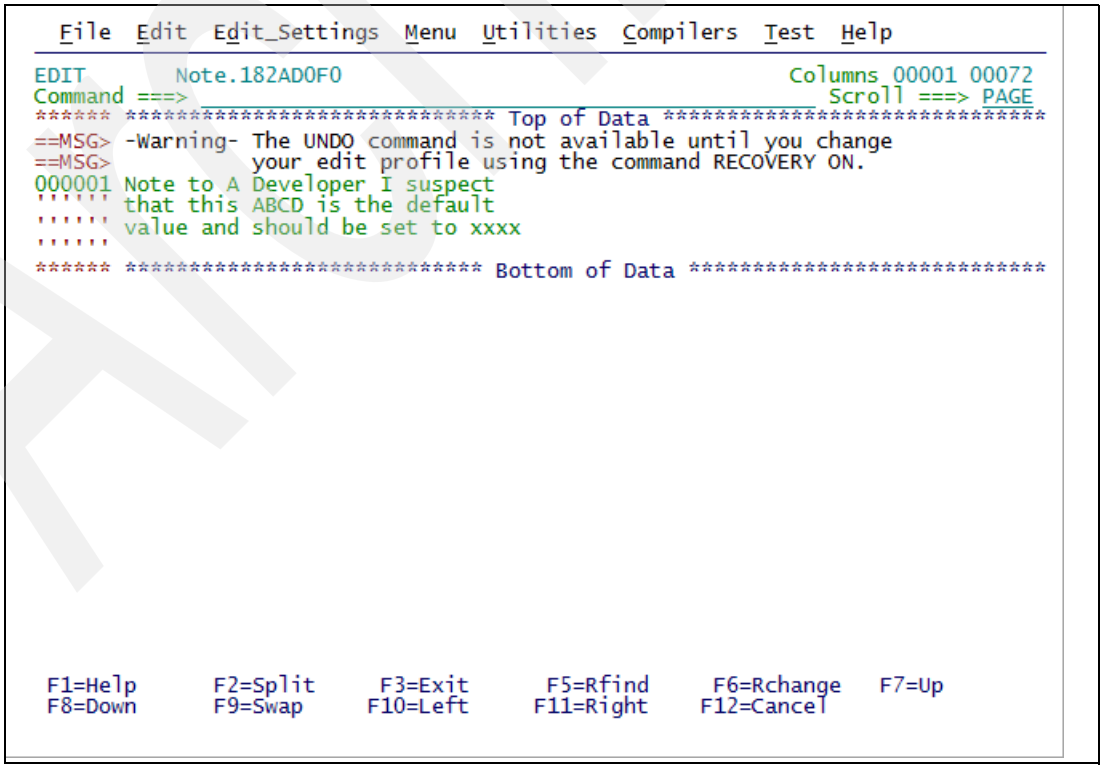


Figure 18-29 Edit Session to allow further typing

After the required data has been entered, pressing PF3 returns you to the storage display with the user note inserted accordingly as shown in Figure 18-30.

```
  File  View  Services  Help
Dump Storage                                                  182AD0D0-182C131F
Command ===>                                                   Scroll ===> CSR
JOBNAME: IDIVPCOB  SYSTEM ABEND: 0C7           STLABF7   2005/11/14  10:09:35

Address  Offset    Hex                                     EBCDIC
         Event 1 Program IDISCBL1 BLW=0000
182AD0D0           E6D6D9D2 C9D5C760 E2E3D6D9 C1C7C540 *WORKING-STORAGE *
182AD0E0      +10  40404040 00000000 0986888F 00000000 *    .....fh.....*
- Note to A Developer I suspect
  that this ABCD is the default
  value and should be set to xxxx
182AD0F0      +20  C1C2C3C4 00000000 00000000 00000000 *ABCD............*
182AD100      +30  00000000 00000000 00000000 00000000 *................*
         Lines 182AD110-182AEFF0 same as above
         Address range 182AF000-182B605F not in minidump
         Module CEEMENU3
         Address range 182B6060-182C0FFF not in minidump
182C1000   +13F30  00000000 00000000 00000000 00000000 *................*
         Lines 182C1010-182C12A0 same as above
182C12B0   +141E0  00000000 00000000                    *........       *
         Module IDIXDCAP
182C12B8   +141E8                   90ECD00C 18BF58F0 *        ..}....0*
182C12C0   +141F0  B11005EF 12FF4770 B020581D 001848F0 *..............0*
182C12D0   +14200  102258ED 000C07FE 58F0B114 05EF12FF *.........0......*
182C12E0   +14210  4780B03A 581D0018 48F01022 58ED000C *.........0......*
182C12F0   +14220  07FE4100 00780700 47F0B048 40808470 *.........0.. .d.*
182C1300   +14230  58F0B044 1B110A78 12FF4780 B0BE0700 *.0.............*
182C1310   +14240  4510B092 00360000 C9C4C9F0 F0F0F7E2 *...k....IDI0007S*
 F1=Help     F3=Exit     F4=Dsect     F5=RptFind   F7=Up        F8=Down
F10=Prev     F11=Next
```

Figure 18-30 Resulting display after additional typing

By default, all user notes are shown expanded, as indicated by the minus sign point-and-shoot field preceding the note.

By placing the cursor on this field and pressing Enter, the note is collapsed. The preceding point-and-shoot field now indicates collapsed by a plus sign instead. By simply placing the cursor on this point-and-shoot field, you can toggle between the collapsed and expanded views.

It is also possible to overtype the point-and-shoot field with two additional action characters (case insensitive):

- D: Used to delete the user note.
- E: Used to edit the user note.

To see all user notes that exist for the current fault entry, enter the NOTELIST command from the command line of any display within the interactive report, or select the List User Notes option from the View action bar pull-down menu.

As indicated in the optional help text on this display, the point-and-shoot field preceding each user note can be typed over to request a specific action in the same way as in the Dump Storage display.

Additionally, the User Note List display permits all user notes to be expanded or collapsed simultaneously by selecting the expand all or collapse all point-and-shoot fields at the top of the display. The expand/collapse state of any note is common between the User Note List display and the Dump Storage display, so that any changes made in one display is reflected in the other.

To display the storage associated with a user note, place the cursor on the address point-and-shoot field, and press Enter. User notes are saved in the history file fault entry when the user exits from the interactive report. At this time, if user notes have been added or modified, the user is prompted to acknowledge the update of the fault entry.

18.6 Mapping storage areas using DSECT information

The DSECT command can be used to provide a Dsect name to be used to map a specified storage address. For this book, the following steps were performed to illustrate the use of the DSECT command.

- COPY the CICS SDFHMAC data set
- Run the IDIPDSCU utility
- Update the Interactive reanalysis options to reference the new data set
- Create a CICS Fault Entry using the CFA transaction
- Perform Interactive Reanalysis against the fault entry
- Get the TCA address from the CICS system information, control block section
- Map the TCA storage against the TCA DSECT

Each of these steps is described in detail below.

18.6.1 Copying the CICS SDFHMAC data set

We used ISPF to create a new data set called SIMCOCK.DSECTS and copied the contents of the CICS SDFHMAC data set to this new data set.

18.6.2 Running the IDIPDSCU utility

Next to data set SIMCOCK.DSECTS in option 3.4 of ISPF, we entered IDIPDSCU to execute the DSECT indexing utility as shown in Example 18-1. In our case the IDIPDSCU utility reported:

Example 18-1 IDIPDSCU output

```
IDIPDSCU: 446 Members processed: 514 Dsects found.
```

18.6.3 Updating the Interactive reanalysis options

To reference the new DSECTS data set, we updated our interactive reanalysis options as shown in Figure 18-31.

```
  File  View  Services  Help
Interactive Reanalysis Options                          Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR

Press PF3 to save options or PF12 to cancel.

General Options:
  Options line for
  interactive reanalysis. . : DATASETS(IDIDSECT(SIMCOCK.DSECTS))
  Redisplay this panel
  before each reanalysis. . : N (Y/N)
  Display panel to alter
  allocated data sets . . . : N (Y/N)
  Prompt before opening a
  SYSMDUMP. . . . . . . . . : Y (Y/N)

Reanalysis Options Data Set Control:
  Options data set name . . :
  Options member name . . . :          (If PDS or PDSE)
  Use this data set during
  reanalysis. . . . . . . . : N (Y/N)
  Edit the options data set
  before reanalysis . . . . : N (Y/N)

*** Bottom of data.



 F1=Help      F3=Exit      F5=RptFind   F6=Actions   F7=Up        F8=Down
F10=Left     F11=Right    F12=Cancel
```

Figure 18-31 Updated interactive reanalysis options

18.6.4 Creating a CICS Fault Entry using the CFA transaction

The Fault Analyzer supplied transaction CFA provides 3 IVP tests. We selected the EXEC CICS ABEND test as shown in Figure 18-32.

```
                    Fault Analyzer IVP Testing

Options: S=Select

                    IVP Description
_ 0C1 in program IDIXFA
_ EXEC CICS DUMP DUMPCODE(FAD1) - XDUREQ exit must be installed
s EXEC CICS ABEND ABCODE(FLT1)









DFHAC2206 21:26:25 CICSC31F Transaction CFA failed with abend FLT1. Updates to
 local recoverable resources backed out.
```

Figure 18-32 CFA transaction IVP

18.6.5 Performing Interactive Reanalysis

We then entered an **I** next to the Fault entry as shown in Figure 18-33.

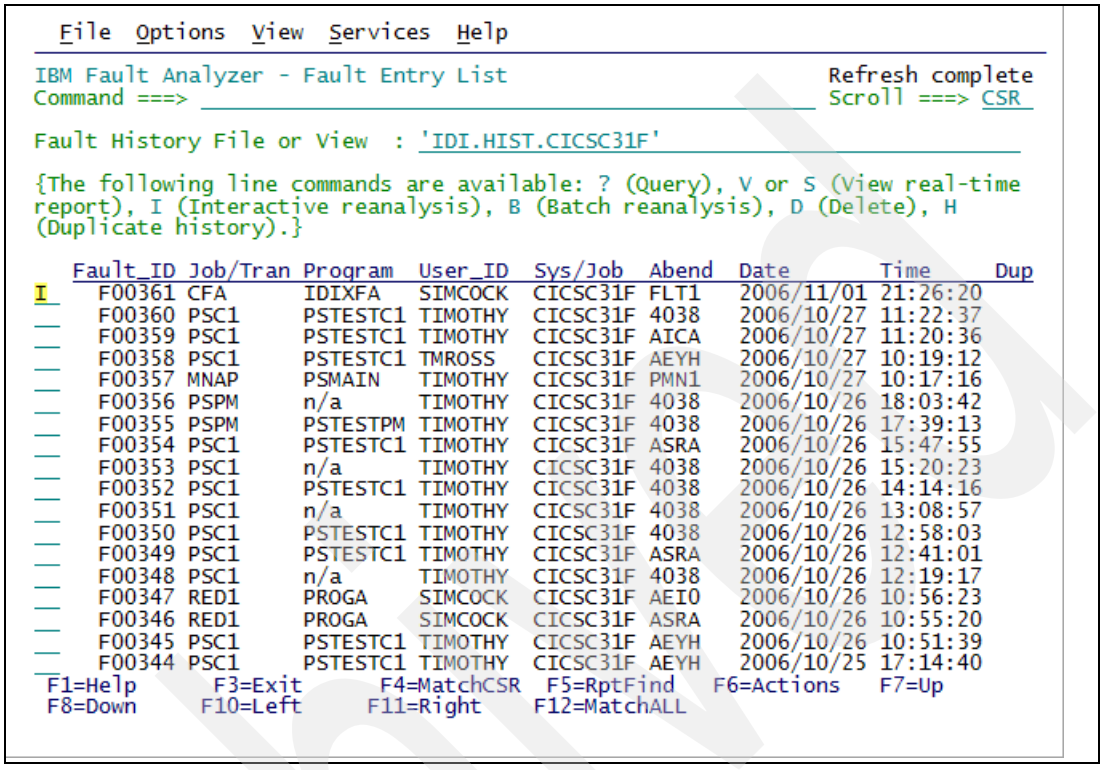


Figure 18-33 Selecting fault entry for interactive reanalysis

18.6.6 Determining the TCA address

Once analysis was complete, we selected **CICS Information** followed by **CICS Control Blocks** and located the TCA as shown in Figure 18-34.

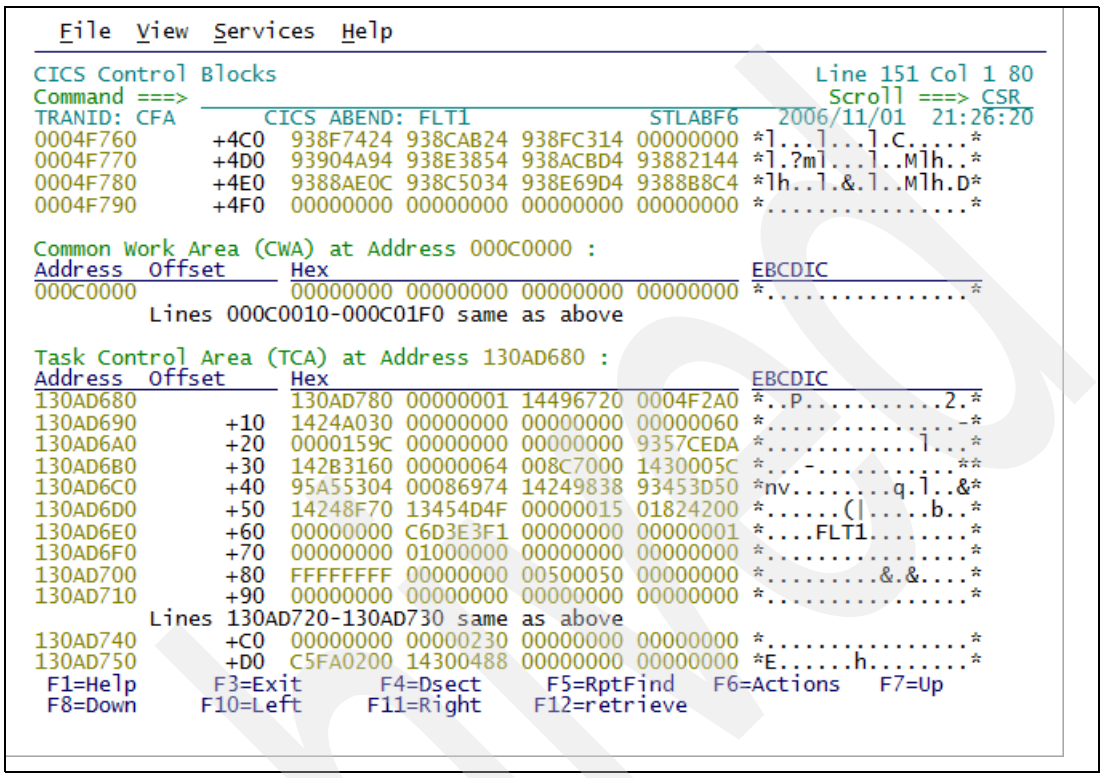


Figure 18-34 Locating the TCA

18.6.7 Mapping the TCA DSECT

We then pressed PF4 to display the DSECT pop-up menu as shown in Figure 18-35.

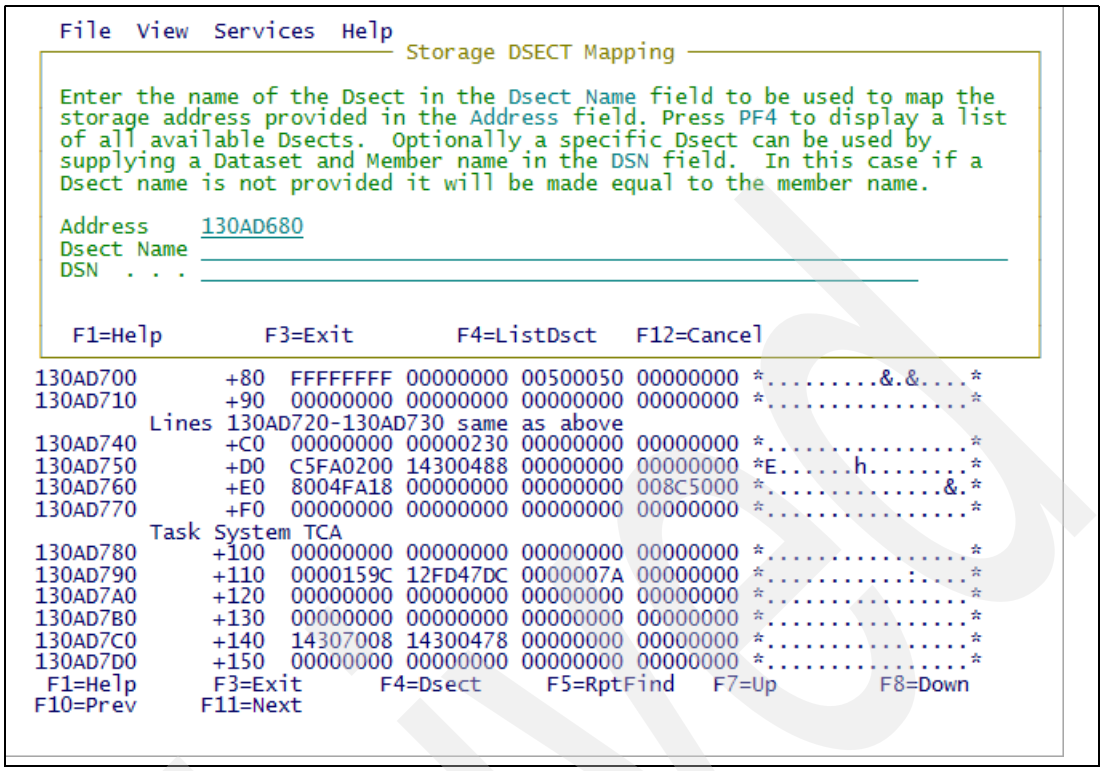


Figure 18-35 DSECTS pop-up

From here we pressed PF4 again to display a list of all available DSECTS as shown in Figure 18-36.

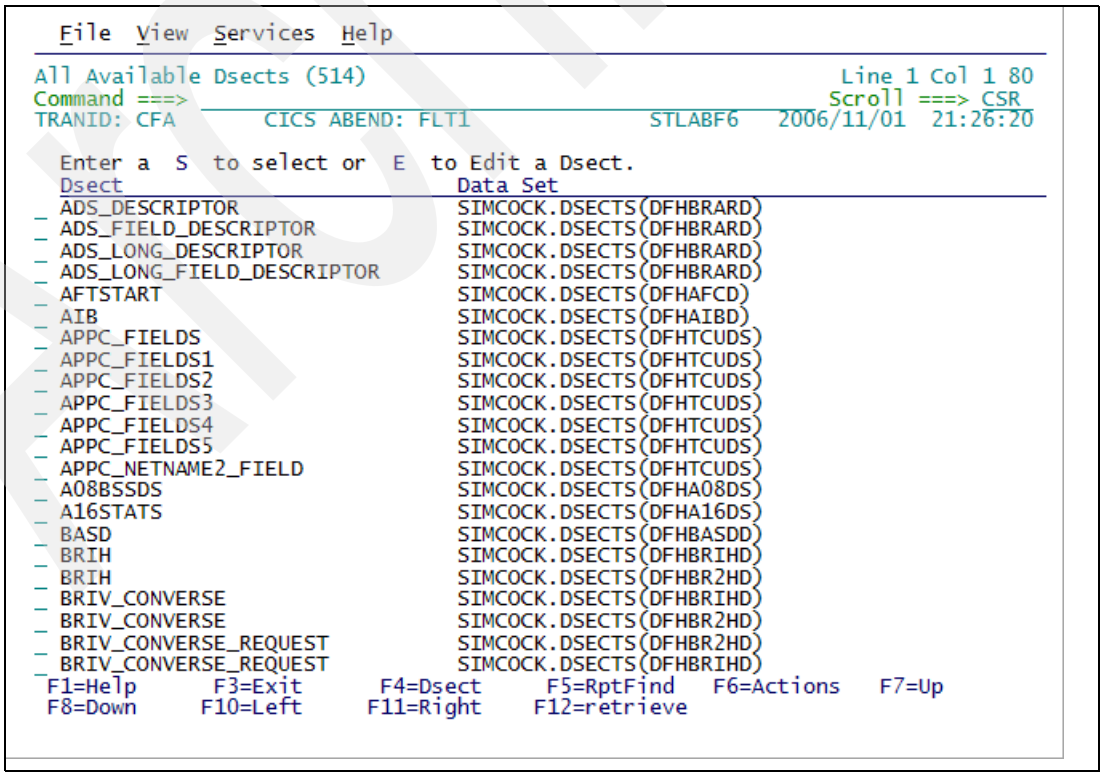


Figure 18-36 All available dsects

We then located and selected the **DFHTCADS DSECT** as shown in Figure 18-37.

```
  File  View  Services  Help
All Available Dsects (514)                                  Line 229 Col 1 80
Command ===>                                                 Scroll ===> CSR
TRANID: CFA       CICS ABEND: FLT1              STLABF6   2006/11/01  21:26:20
  DFHSZSDS                      SIMCOCK.DSECTS(DFHSZSDS)
  DFHTCADS                      SIMCOCK.DSECTS(DFHTRAP)
s DFHTCADS                      SIMCOCK.DSECTS(DFHTCADS)
  DFHTCADS                      SIMCOCK.DSECTS(DFHTCA)
  DFHTCADY                      SIMCOCK.DSECTS(DFHTRAP)
  DFHTCTFX                      SIMCOCK.DSECTS(DFHTCTFX)
  DFHTCTLE                      SIMCOCK.DSECTS(DFHTCTLE)
  DFHTCTLE                      SIMCOCK.DSECTS(DFHTACLE)
  DFHTCTRS                      SIMCOCK.DSECTS(DFHTCTFN)
  DFHTCTRS                      SIMCOCK.DSECTS(DFHTCTTE)
  DFHTCTSK                      SIMCOCK.DSECTS(DFHTCTPX)
  DFHTCTTE                      SIMCOCK.DSECTS(DFHTCTFN)
  DFHTCTTE                      SIMCOCK.DSECTS(DFHTCTTE)
  DFHTCTWA                      SIMCOCK.DSECTS(DFHTCTWA)
  DFHTCXDS                      SIMCOCK.DSECTS(DFHTCX)
  DFHTDGDS                      SIMCOCK.DSECTS(DFHTDGDS)
  DFHTDOA                       SIMCOCK.DSECTS(DFHTDOA)
  DFHTDRDS                      SIMCOCK.DSECTS(DFHTDRDS)
  DFHTERID                      SIMCOCK.DSECTS(DFHTERID)
  DFHTIOA                       SIMCOCK.DSECTS(DFHTIOA)
  DFHTQGDS                      SIMCOCK.DSECTS(DFHTQGDS)
  DFHTQRDS                      SIMCOCK.DSECTS(DFHTQRDS)
  DFHTRADS                      SIMCOCK.DSECTS(DFHTRAP)
  DFHTRADS                      SIMCOCK.DSECTS(DFHTRADS)
  DFHTREN                       SIMCOCK.DSECTS(DFHTREND)
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 18-37 Selecting DFHTCADS DSECT

Having selected the DSECT, the resulting Fault Analyzer display is shown in Figure 18-38.

```
  File  View  Services  Help
Dsect mapping for DFHTCADS at addrees 130AD680               Line 1 Col 1 80
Command ===>                                                 Scroll ===> CSR
TRANID: CFA       CICS ABEND: FLT1              STLABF6   2006/11/01  21:26:20

130AD680 +0000                                  DSECT DFHTCADS
                                                TCACBAR  EQU   12
                                                DFHUSTCA DS    0D
130AD680 +0000 130AD780                         TCASYAA  DS    A
130AD684 +0004 00                               TCAXMSRF DS    X
                                                TCAENQ31 EQU   X'80'

                                                TCAENQTA EQU   X'40'

                                                TCATCQLN DS    0X
130AD685 +0005 00                               TCATCQL4 DS    X

130AD686 +0006 00                               TCAGFLG1 DS    X
                                                TCAACPAC EQU   X'80'
130AD687 +0007 01                               TCAFCI   DS    X
                                                TCAFCTDM EQU   X''00'
                                                TCAFCTRM EQU   X'01'
                                                TCAFCMCM EQU   X'02'
                                                TCAFCICM EQU   X'04'
                                                TCAFCDCM EQU   X'08'
                                                TCAFCAID EQU   X'10'
130AD688 +0008 14496720                         TCAFCAAA DS    A
130AD68C +000C 0004F2A0                         TCACSOAD DS    A
130AD690 +0010 1424A030                         TCALCDSA DS    A
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 18-38 Final DSCET mapping

18.7 Displaying chained data areas

Storage can be scanned for a chain of linked control blocks. The **RUNCHAIN** command can be invoked either by entering RUNCHAIN on any interactive report command line or by assigning **RUNCHAIN** to a PF key. When invoked, you are shown a pop-up panel similar to the one in Figure 18-39.

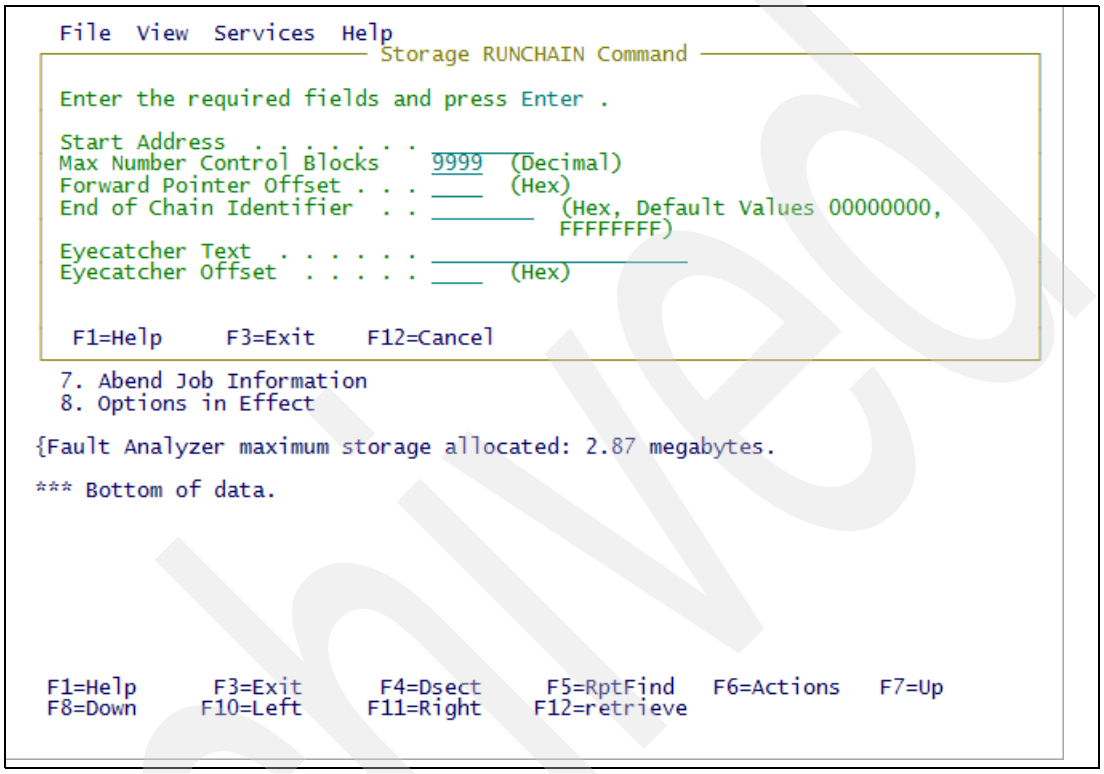


Figure 18-39 RUNCHAIN command pop-up

For a given Start Address and Forward Pointer Offset, the **RUNCHAIN** command follows the chain of control blocks until one of the following end conditions is met:

- The number of control blocks scanned has exceeded the maximum number set by the user (the default value is 9999).
- The forward pointer of the current control block contains one of the End of Chain values. These values are:
  - X'00000000'
  - X'FFFFFFFF'
  - The initial start address implying that the chain has looped
  - A user-supplied End Of Chain Identifier
- The forward pointer of the current control block points to invalid or unavailable storage.

For each control block, its address and the first 32 bytes of data are shown. Optionally, you can provide an eyecatcher and its offset in the control block, in which case, for each control block, the text at the specified offset is compared against the supplied text, and if they do not match, then a warning message is issued.

To exit from the **RUNCHAIN** command, enter EXIT (PF3).

18.8 Disassembling object code

The **DISASM** command disassembles object code at a given address.

The **DISASM** command is invoked either by entering **DISASM** on any interactive report command line or by assigning DISASM to a PF key. When invoked, you are shown a pop-up panel similar to Figure 18-40.

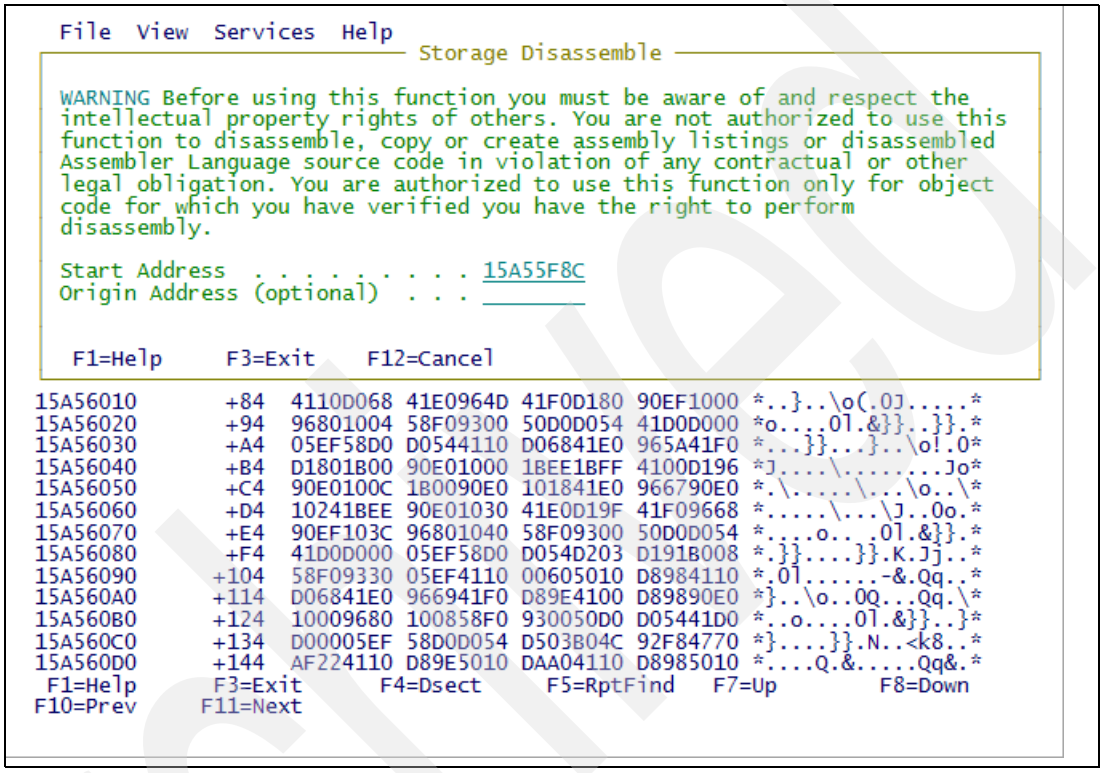


Figure 18-40 DISASM pop-up

Optionally, an origin address can be provided, in which case the offset of each disassembled instruction is calculated relative to the origin address, rather than the start address. If an origin address is not provided, then it defaults to the same as the start address.

The PF7 and PF8 can be used to scroll backwards and forwards. To exit from the **DISASM** command, enter EXIT (PF3).

18.9 Converting STORE CLOCK values

The **STCK** command converts binary STORE CLOCK values to human-readable date and time format. The **STCK** command is invoked either by entering **STCK** on any interactive report command line, or by assigning **STCK** to a PF key. When invoked you can enter the **STCK** value and when enter is pressed the human-readable format is displayed as shown in Figure 18-41.

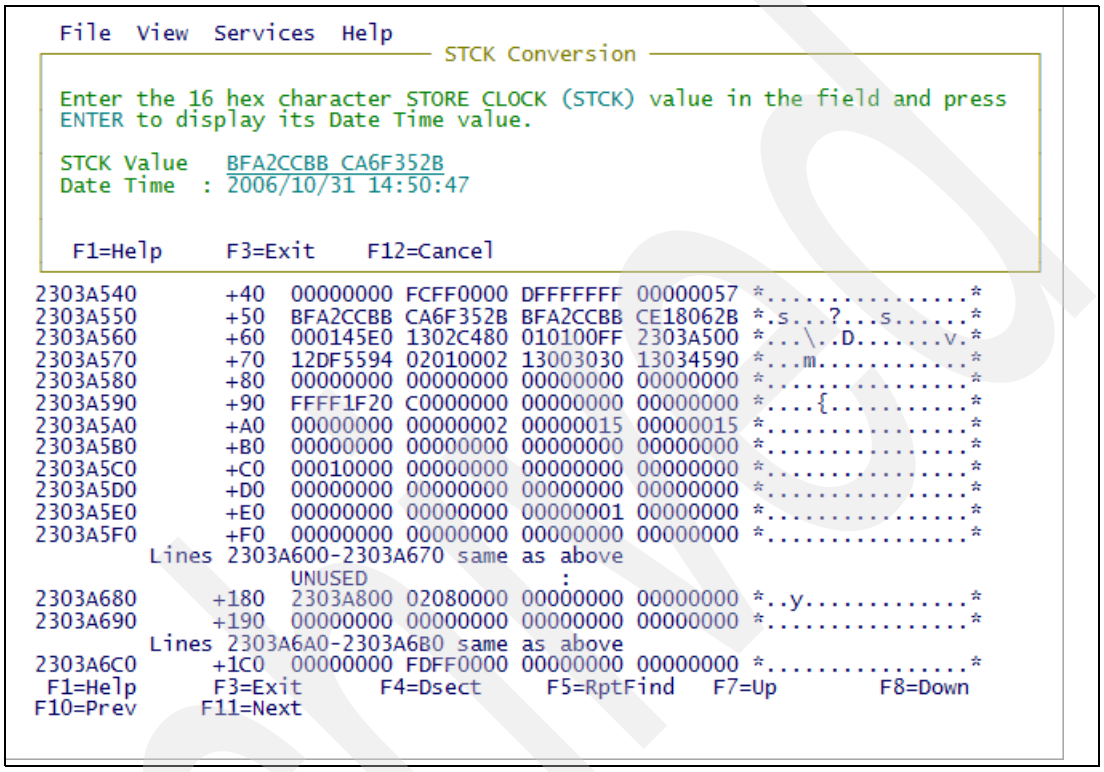


Figure 18-41 STCK display

The **STCK** value must be entered as 16 hexadecimal characters. Any imbedded blanks are ignored.

To exit from the **STCK** command, enter `EXIT (PF3)`.

18.10 User-specific report formatting

REXX Formatting user exits can be used to generate a display of user-specific information, such as formatting of data areas which are unique to the analyzed application environment. For this book, the following steps were performed to illustrate the use of REXX Formatting user exits.

- COPY the Fault Analyzer SIDISAM1data set
- Created a $$UFMTX member in the new data set
- Update the Interactive reanalysis options.
- Perform Interactive Reanalysis against the fault entry

Each of these steps is described in detail below.

18.10.1 Copying the SIDISAM1 data set

We used ISPF to create a SIMCOCK.EXEC data set and copied members IDISUFM* from the Fault Analyzer SIDISAM1 data set.

18.10.2 Creating the $$UFMTX member

We then created member $$UFMTX in data set SIMCOCK.EXEC as shown in Figure 18-42.

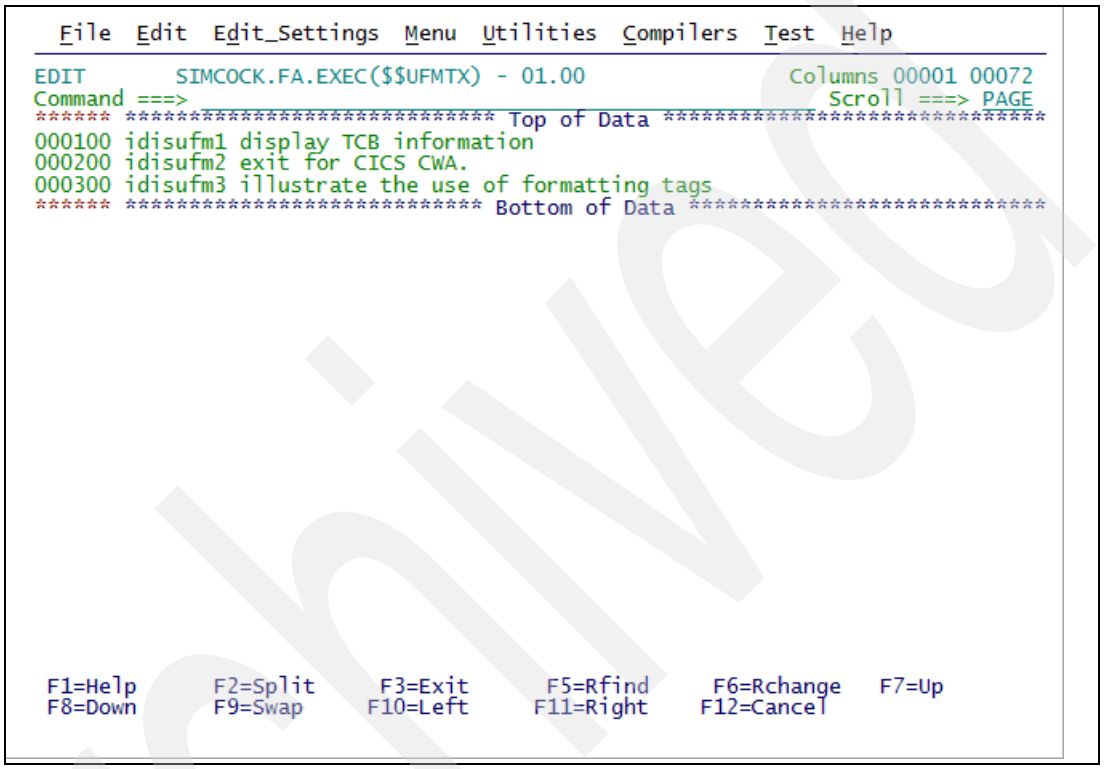


Figure 18-42 $$UFMTX member

18.10.3 Updating the Interactive reanalysis options

Since we now had more options than would fit on the Interactive options line, we created an options data set and member. We updated the Fault Analyzer interactive options to reference this new data set and member, as shown in Figure 18-43 and Figure 18-44.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       SIMCOCK.FA.OPTIONS(MYOPT) - 01.00               Columns 00001 00072
Command ===>                                                Scroll ===> PAGE
****** ***************************** Top of Data ********************************
000100 DATASETS(
000200   IDIEXEC(SIMCOCK.FA.EXEC)
000201   IDIDSECT(SIMCOCK.DSECTS)
000210  )
000300 EXITS(FORMAT(REXX(IDISUFM1)))
****** **************************** Bottom of Data ******************************


 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 18-43 New Fault Analyzer options data set

```
  File  View  Services  Help
Interactive Reanalysis Options                              Line 1 Col 1 80
Command ===>                                               Scroll ===> CSR

Press PF3 to save options or PF12 to cancel.

General Options:
  Options line for
  interactive reanalysis. . :
  Redisplay this panel
  before each reanalysis. . : N (Y/N)
  Display panel to alter
  allocated data sets . . . : N (Y/N)
  Prompt before opening a
  SYSMDUMP. . . . . . . . . : Y (Y/N)

Reanalysis Options Data Set Control:
  Options data set name . . : 'SIMCOCK.FA.OPTIONS'
  Options member name . . . : MYOPT     (If PDS or PDSE)
  Use this data set during
  reanalysis. . . . . . . . : Y (Y/N)
  Edit the options data set
  before reanalysis . . . . : N (Y/N)

*** Bottom of data.


 F1=Help      F3=Exit      F5=RptFind   F6=Actions   F7=Up        F8=Down
F10=Left     F11=Right    F12=Cancel
```

Figure 18-44 Updated Fault Analyzer interactive options

18.10.4 Performing Interactive Reanalysis

We then performed Interactive reanalysis of a Fault entry. Since we had added the *EXITS(FORMAT(REXX(IDISUFM1)))* option to our options data set, the main Fault Analyzer interactive display now had a *USER* option as shown in Figure 18-45.

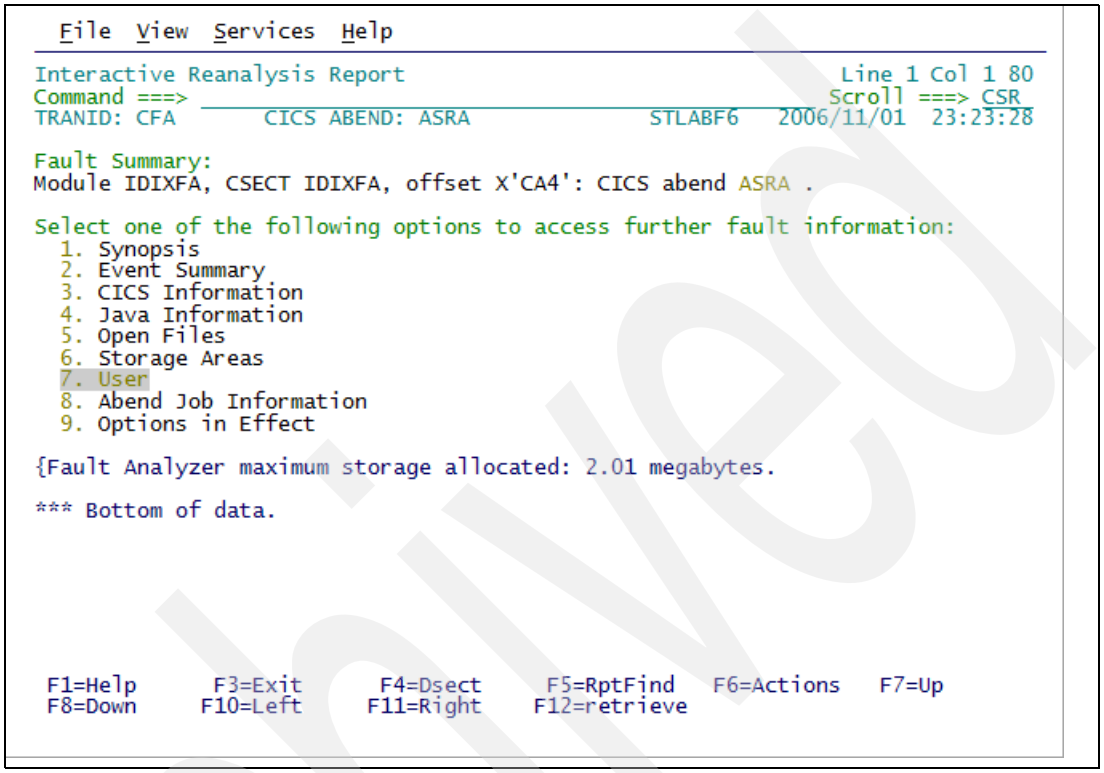


Figure 18-45 Additional USER option

Selecting this option displayed the panel shown in Figure 18-46, which was generated by the IDISUFM1 EXEC.

```
  File  View  Services  Help
                                                              Line 1 Col 1 80
Command ===> ____________________________________________________ Scroll ===> CSR
TRANID: CFA       CICS ABEND: ASRA              STLABF6   2006/11/01  23:23:28


Fault Analyzer Formatting User Exit Example
Display TCB Information

Jobstep TCB = 008E4E88

 001   008E4E88 RB count 0.063
 002       008E4B68 DFHTRTCB 0
 003       008E3D10 DFHKETCB 0.232
 004           008E39F0 DFHKETCB 0.632
 005               008E36D0 DFHKETCB 24.324
 006                   008E33B0 DFHKETCB 0.001
 007                   008E4410 DFHKETCB 0.009
 008                   008E41D8 DFHKETCB 0.015
 009                   008C6E88 DFHKETCB 0
 010                       008C6CD8 DFHKEATT 0.001
 011                   008BAAD0 DFSPRRA0 0.002
 012                       008C1988 DFSPAT00 0
 013                   008BA618 DFHKETCB 0.089
 014                   008BA468 RB count 0.097
 015                   008BA140 EZACIC03 0.009
 016                   008B6588 EZACIC03 0
 017                   008B63F0 EZACIC03 0
 018                   008B61D0 EZACIC03 0
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 18-46 USER option output

Using the EXEC command displayed a list of the formatting user exits as listed in the $$UFMTX member, as shown in Figure 18-47.

```
  File  View  Services  Help
Formatting User Exit Selection List                           Line 1 Col 1 80
Command ===> ____________________________________________________ Scroll ===> CSR
TRANID: CFA       CICS ABEND: ASRA              STLABF6   2006/11/01  23:23:28

{The following line commands are available: S (Select), B (Browse), E (Edit).}

   Name     Comment/Arguments
__ IDISUFM1 display TCB information                                        000
__ IDISUFM2 exit for CICS CWA.                                             000
__ IDISUFM3 illustrate the use of formatting tags                          000

*** Bottom of data.




 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 18-47 Output of EXEC command

Selecting IDISUFM3 with an **S** resulted in the display shown in Figure 18-48.

```
  File  View  Services  Help
                                                          Line 1 Col 1 80
Command ===>                                             Scroll ===> CSR
TRANID: CFA       CICS ABEND: ASRA              STLABF6   2006/11/01  23:23:28

First paragraph.

     Second paragraph, indented 5 characters from the first. This <P> tag is
     treated as text only.
     Third paragraph. Note that this paragraph is not preceded by a blank line

Fourth paragraph - now we are back at the left margin.
**************************************** This line will not wrap at the prefer

Previous area and 00625F22 are both point-and-shoot fields to the Dump Storage
display for address 00625F22 in the interactive reanalysis report.

This is a long definition
term. . . . . : This is the matching definition description
                which might wrap depending on the preferred
                formatting width.

A shorter definition term . : The definition description of the second term.

Address 0 storage for a length of 32 bytes:
Address  Offset    Hex                                      EBCDIC
00000000           000A0000 000130E1 00000000 00000000 *................*
00000010       +10 00FDE800 00000000 7FFFF000 7FFFF000 *..Y.....".0.".0.*
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 18-48 IDISUFM3 output

18.11 Prompting for compiler listing or side file

If no satisfactory compiler listing or side file was found for a COBOL or PL/I program, then a prompt is displayed similar to the one shown in Figure 18-49.

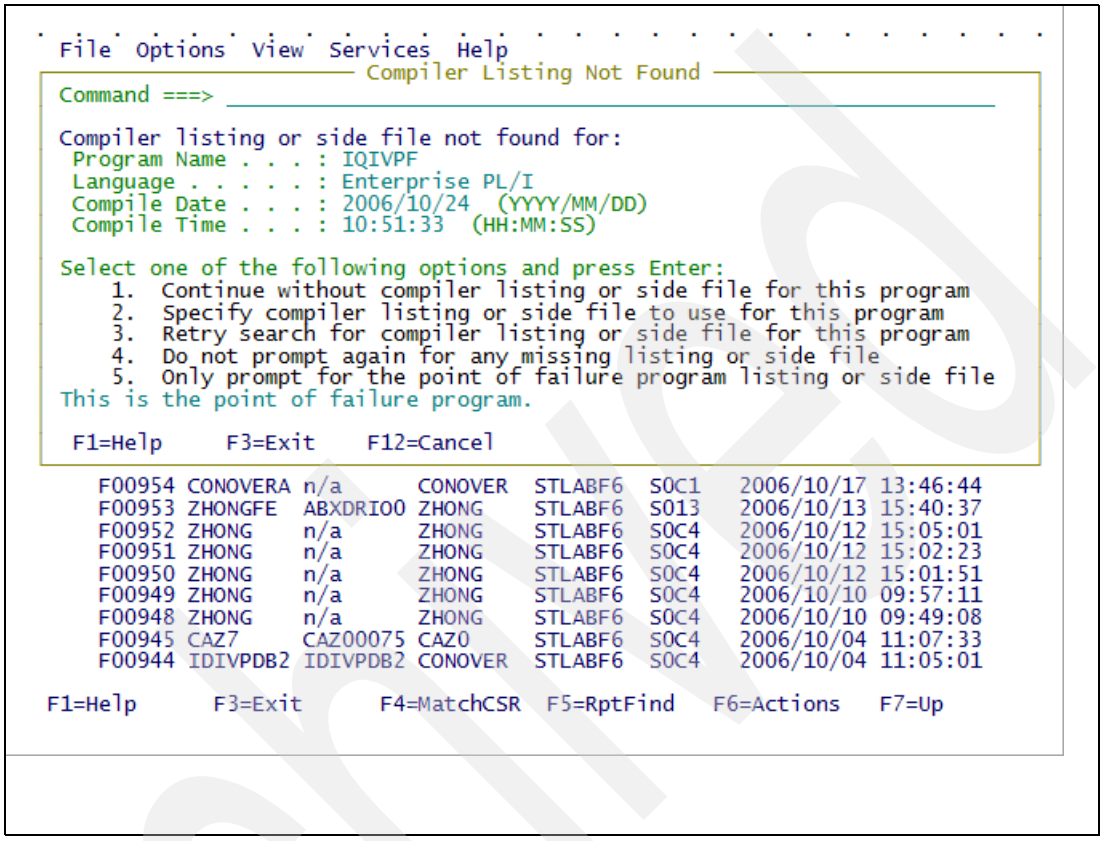


Figure 18-49 Missing listing prompt

The prompt provides you with these choices:

- Continue without compiler listing or side file for this program.

  If a compiler listing or side file cannot be supplied, select this option to continue without program source code information. Alternatively, enter the EXIT (PF3) or CANCEL (PF12) command.

- Specify compiler listing or side file to use for this program.

  This option displays a pop-up panel in which you can provide the data set and member name (if a PDS(E) data set) of a compiler listing or side file that should be used for the current program as shown in Figure 18-50.

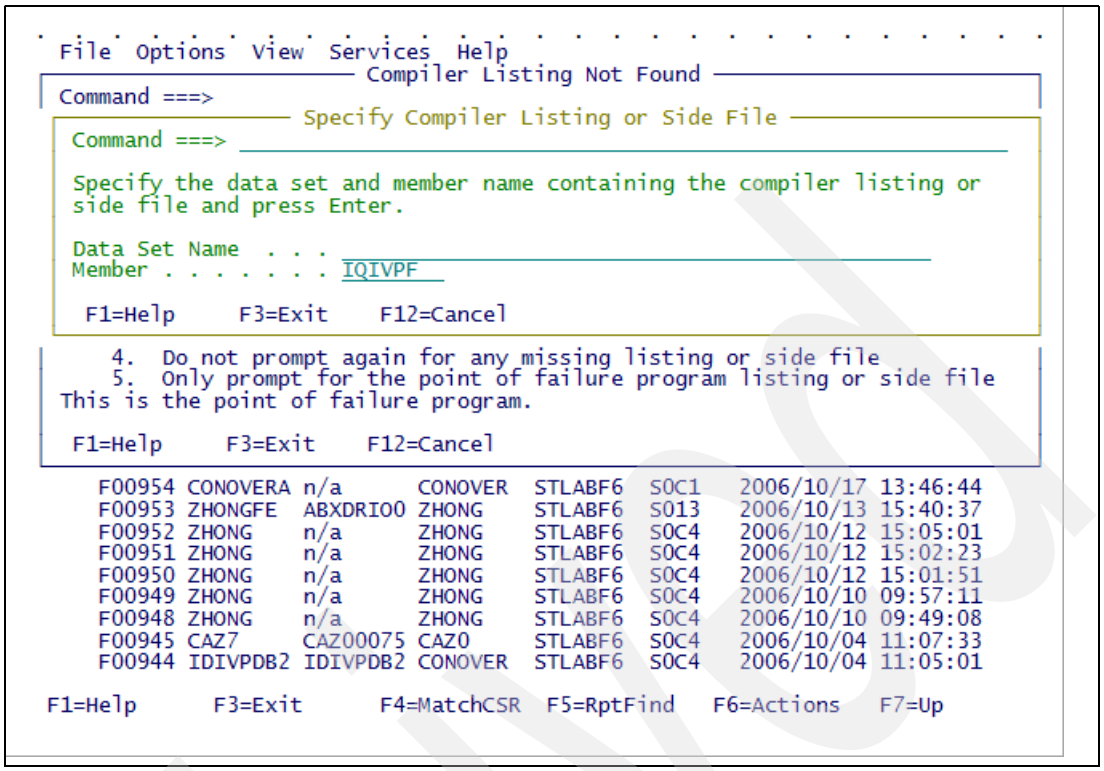


Figure 18-50 Specifying listing data set and member name

The data set name is specified in accordance with the ISPF convention of prefixing with the current TSO prefix, unless enclosed in single quotation marks. The member name defaults to the program name for which the listing or side file is required. If the actual member name for your listing or side file differs from the program name, you must change this field. If a sequential data set is specified, then the member name is ignored. Having specified the desired data set and member name, press Enter.

- Retry search for compiler listing or side file for this program.

  Selecting this option causes Fault Analyzer to repeat the search for the compiler listing or side file via the standard search path. This option can be selected after, for example, having recompiled the current program via a split-screen ISPF session and provided the compiler listing or side file to Fault Analyzer in, for example, the IDILCOB data set concatenation. This repeated search is only performed once. The user is not prompted a second time for the same program, even if the listing or side file is still not found.

- Do not prompt again for any missing listing or side file.

  If you select this option, then Fault Analyzer does not prompt you again for a missing compiler listing or side file for any program for the duration of the current interactive reanalysis session.

- Only prompt for the point of failure program listing or side file.

  If you select this option, then Fault Analyzer only prompts you again for a missing compiler listing or side file for a program if that program has been determined as belonging to the point-of-failure event. If the initial prompt is already for the point-of-failure program, then a message is added to the display to indicate this.

18.11.1 Controlling prompting

In order for the compiler listing or side file prompting to occur during interactive reanalysis, at least one IDILANGX or compiler listing data set (for example, IDILCOB) must have been provided.

It follows that, if source analysis is never to be performed, and therefore no prompting during interactive reanalysis is desired, that no IDILANGX or compiler listing data set definitions should be specified through DataSets options in the IDICNF00 config member.

On the other hand, to ensure that prompting always occurs during interactive reanalysis if no satisfactory compiler listing or side file is found, regardless of whether any compiler listing or side file data sets were specified for the real-time analysis, add at least one IDILANGX or compiler listing data set definition through the DataSets option in the IDICNF00 config member.

18.12 Data sets used for interactive reanalysis

When performing interactive reanalysis through the ISPF interface, pre-allocation is performed as required for any Fault Analyzer compiler listing or side file data sets that were used in real-time. Allocations are performed for Fault Analyzer data sets if they were explicitly included in the real-time JCL or supplied through the DataSets option or an Analysis Control user exit. These data sets are used in the reanalysis in an attempt to recreate the same execution environment as was used in real-time.

DataSets options specified via the IDIOPTS user options file or the PARM field cause those data sets to be logically concatenated to the data sets from the real-time execution.

If the Display panel to alter allocated data sets option on the Interactive options display is set to **Y** (see 18.1, “Interactive reanalysis options” on page 462), then it is possible to make changes to the real-time data set specifications before initiating the reanalysis. Also, any data sets that were used in real-time but do not exist in the reanalysis environment are identified by a comment, as shown in the following example for IDILCOB:

```
//* The following IDILCOB data set is unavailable:
//* DD DISP=SHR,DSN=DO1.COBOL.LISTINGS
```

19

Fault Analyzer CICS system dump analysis

In this chapter we discuss the Fault Analyzer CICS system dump analysis. As an example, we take you through the analysis of a typical system dump.

19.1 Introduction

As well as reanalysis of Fault Entries, the Fault Analyzer ISPF interface also has support for interactive analysis of system dumps. This feature has particular significance when the dump in question is a CICS system dump. In this case, Fault Analyzer performs extensive CICS specific analysis of the dump and does not only display information similar to that obtained from the CICS supplied IPCS VERBEXITs, but also analysis unique to Fault Analyzer — for example, “deadly embrace” detection.

19.2 Invoking the analysis

To analyze a CICS system dump, the name of the dump data set can be supplied in one of three ways.

1. Using the **File**, **Analyze MVS Dump Data Set...** options from the main Fault Analyzer ISPF panel; see Figure 19-1 and Figure 19-2.
2. If the **FA** command has been activated, as discussed in “Invoking the interface”, simply enter **FA** next to the dump data set.
3. If the Fault Analyzer Dump Registration exit IDIXTSEL has been installed, then perform interactive reanalysis against the recorded Fault Entry.

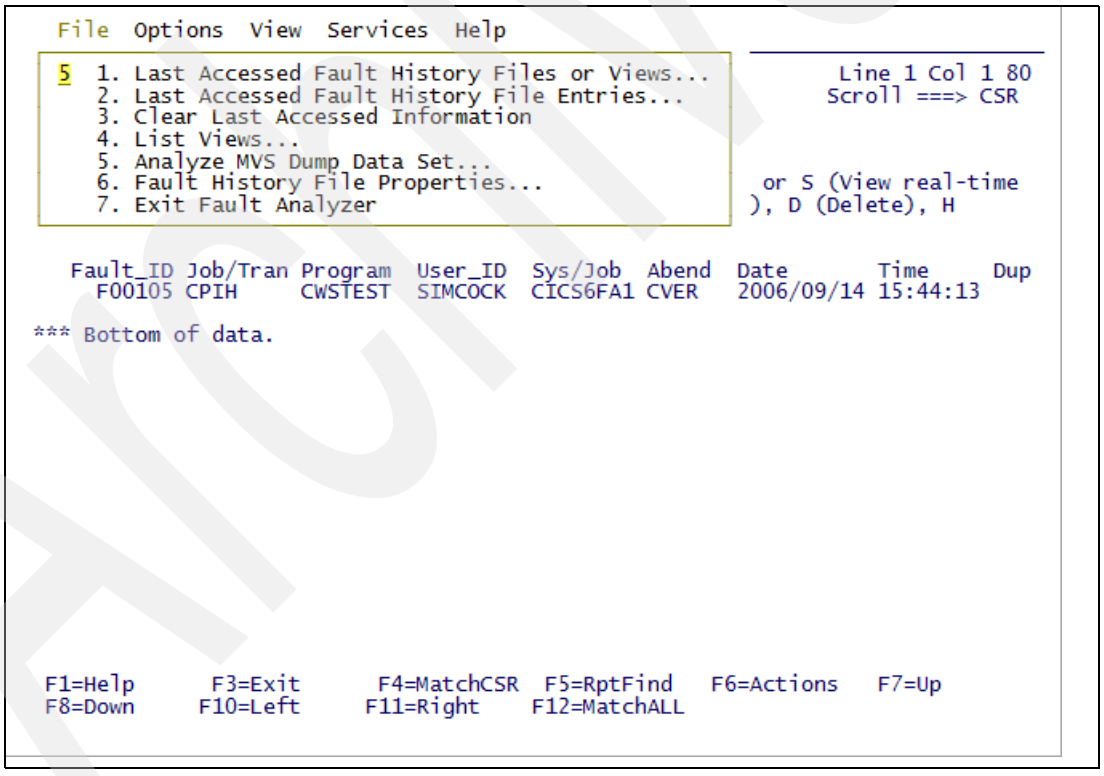


Figure 19-1 Fault analyzer File Menu

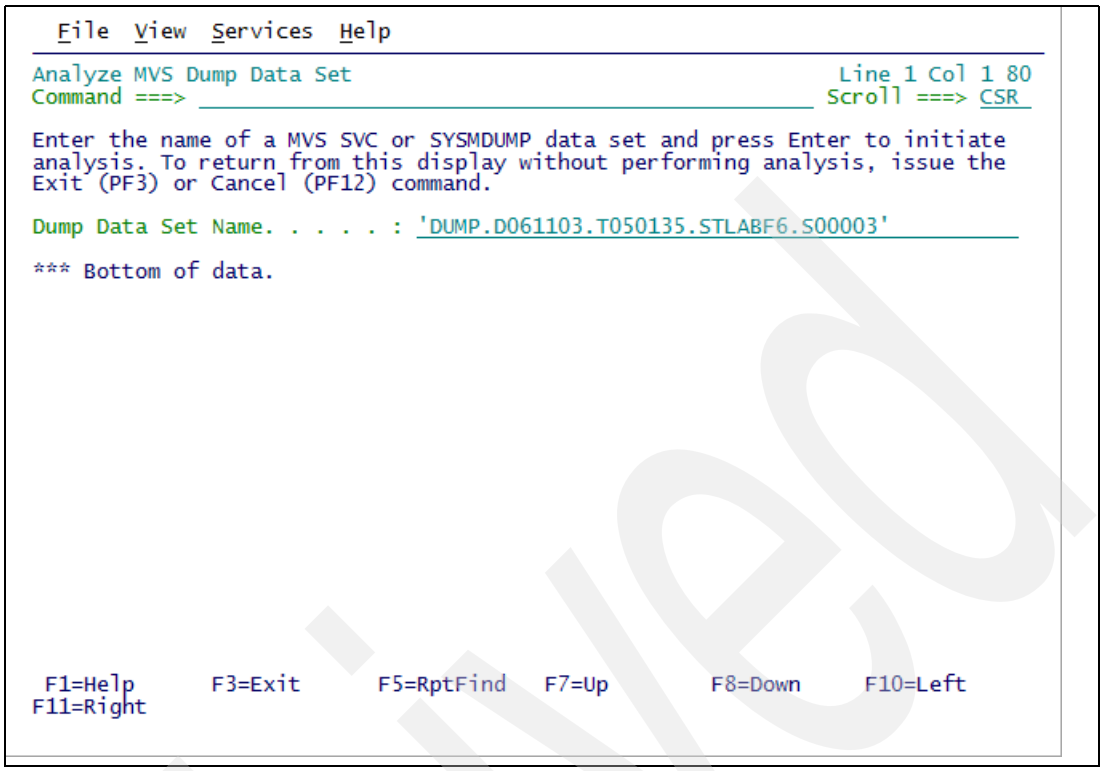


Figure 19-2 Supplying the dump data set name

19.3 Example analysis

For this book, we analyzed a system dump taken as a consequence of the Fault Analyzer 0C1 IVP (invoked by pressing PF9 in the CFA transaction). In this case the dump data set name was DUMP.D061103.T050135.STLABF6.S00003, and we entered **FA** next to this in ISPF option 3.4, as shown in Figure 19-3.

Figure 19-3 FA command against dump data set name in ISPF 3.4

After we pressed Enter, Fault Analyzer performed the analysis, and once complete, the main panel shown in Figure 19-4 was displayed.

```
  File  View  Services  Help
Interactive Reanalysis Report                                  Line 1 Col 1 80
Command ===>                                                  Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004          STLABF6   2006/11/02  21:01:37


Select one of the following options and press Enter to access further fault
info:

   1.  Synopsis
   2.  Abend Job Information
   3.  CICS System Information
   4.  Options in Effect


DFHAP0001 CICSC31F An abend (code 0C1/AKEA) has occurred at offset X'00011F8E'
in module IDIXFA.

Severity 3 Observations

o   Transaction CSKL number 0000031 is waiting for an unspecified resource

o   Transaction OSPH number 0000477 is waiting for resource type EKCWAIT
    resource SINGLE




 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 19-4 Initial screen after dump analysis.

From this panel, option **2** Abend Job Information was selected, by entering **2** on the command line. This displayed information about the job; see Figure 19-5 to Figure 19-7.

```
  File  View  Services  Help
Abend Job Information                                          Line 1 Col 1 80
Command ===>                                                  Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004          STLABF6   2006/11/02  21:01:37

Job information:

 Abend Date. . . . . . . . . : 2006/11/02
 Abend Time. . . . . . . . . : 21:01:37
 System Name . . . . . . . . : STLABF6
 Subsystem Info. . . . . . . : CICS V6 R4 M0
 Job Name. . . . . . . . . . : CICSC31F
 Job Step Name . . . . . . . : CICSC31F
 Exec Program Name . . . . . : DFHSIP
 Requested Region Size . . . : 768M
 User id . . . . . . . . . . : CICSUSER

Execution Environment:

 Operating System. . . . . : z/OS     V01R08M00
 Data Facility Product . . . : DFSMS z/OS V1R8M0
 CPU Model . . . . . . . . . : 2064

SDUMP Parameter List:

 +0000  FLAG0.... 12        FLAG1.... A1        SDATA.... BFE0      DCBAD....
 +0010  ECBAD.... 00000000  SRBAD.... 00000000  CASID.... 0035      TASID....
 +0020  SDDAT.... 00000000  FLAG2.... 00        CNTL1.... C0        TYP1.....
 +002C  EXIT..... E540      SDAT3.... 40        SDAT4.... 00        SPLST....
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 19-5 Abend Job Information (1 of 3)

```
File  View  Services  Help
Abend Job Information                                          Line 57 Col 1 80
Command ===>                                                   Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004          STLABF6   2006/11/02  21:01:37
CICS Specific:

 Region Applid . . . . . . . : CICSC31F
 SIT Name / Address. . . . . : DFHSIT   / 92DF2F98
 Start-up Type . . . . . . . : Cold
 Start-up Overrides. . . . . : 12DE9000
 CICS Status . . . . . . . . : Executing; Initialization Complete;
 Currently MXT?. . . . . . . : N
 Currently SOS?. . . . . . . : N

Data Sets:

DDname   Data Set or Path Name
DFHRPL   IBM.TS310.CICS.SDFHLOAD
         CHABERT.FBI.TEST.LOAD
         COBTST1.CICSTEST.COBMVV33.TS22.LOAD
         COBTST1.CICSTEST.PLIMVV33.TS22.LOAD
         CICSVS.C31F.TABLES
         CICSVS.C31F.USRLOAD
         TCPIP.ZOSV1R8.SEZATCP
         SYS1.LINKLIB
         CEE.SCEECICS
         CEE.SCEERUN
         SYS1.DB1E.SDSNLOAD
         ADTOOLS.FA710.SIDIAUTH
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 19-6 Abend Job Information (2 of 3)

```
File  View  Services  Help
Abend Job Information                                          Line 68 Col 1 80
Command ===>                                                   Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004          STLABF6   2006/11/02  21:01:37

DDname   Data Set or Path Name
DFHRPL   IBM.TS310.CICS.SDFHLOAD
         CHABERT.FBI.TEST.LOAD
         COBTST1.CICSTEST.COBMVV33.TS22.LOAD
         COBTST1.CICSTEST.PLIMVV33.TS22.LOAD
         CICSVS.C31F.TABLES
         CICSVS.C31F.USRLOAD
         TCPIP.ZOSV1R8.SEZATCP
         SYS1.LINKLIB
         CEE.SCEECICS
         CEE.SCEERUN
         SYS1.DB1E.SDSNLOAD
         ADTOOLS.FA710.SIDIAUTH
         ADTOOLS.DT710.SEQAMOD
         ADTOOLS.FM710.SFMNMOD2

STEPLIB  IBM.TS310.CICS.SDFHAUTH
         IBM.TS310.CICS.SDFJAUTH
         IMSVS.IM9F.EXITLIB
         IMSVS.IM9F.SDFSRESL
         SYS1.DB1E.SDSNLOAD
         ADTOOLS.DT710.SEQAMOD

*** Bottom of data.
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 19-7 Abend Job Information (3 of 3)

Next we returned to the main panel and selected option **3** CICS system information. This displayed the panel shown in Figure 19-8. As can been seen in this figure, more information about each CICS domain can be obtained by either placing the cursor under the relevant domain name point-and-shoot field or my entering its ID on the command line;
for example, XM.

Tip: The domain ID can be entered from any command line. Doing this “stacks” the new display on top of the current display — that is, pressing PF3 from the new display takes you back to the display from which you entered the domain ID. If you want to unstack all current displays, that is, as if you entered the domain ID from the main screen, enter !<domain ID> for example, !XM

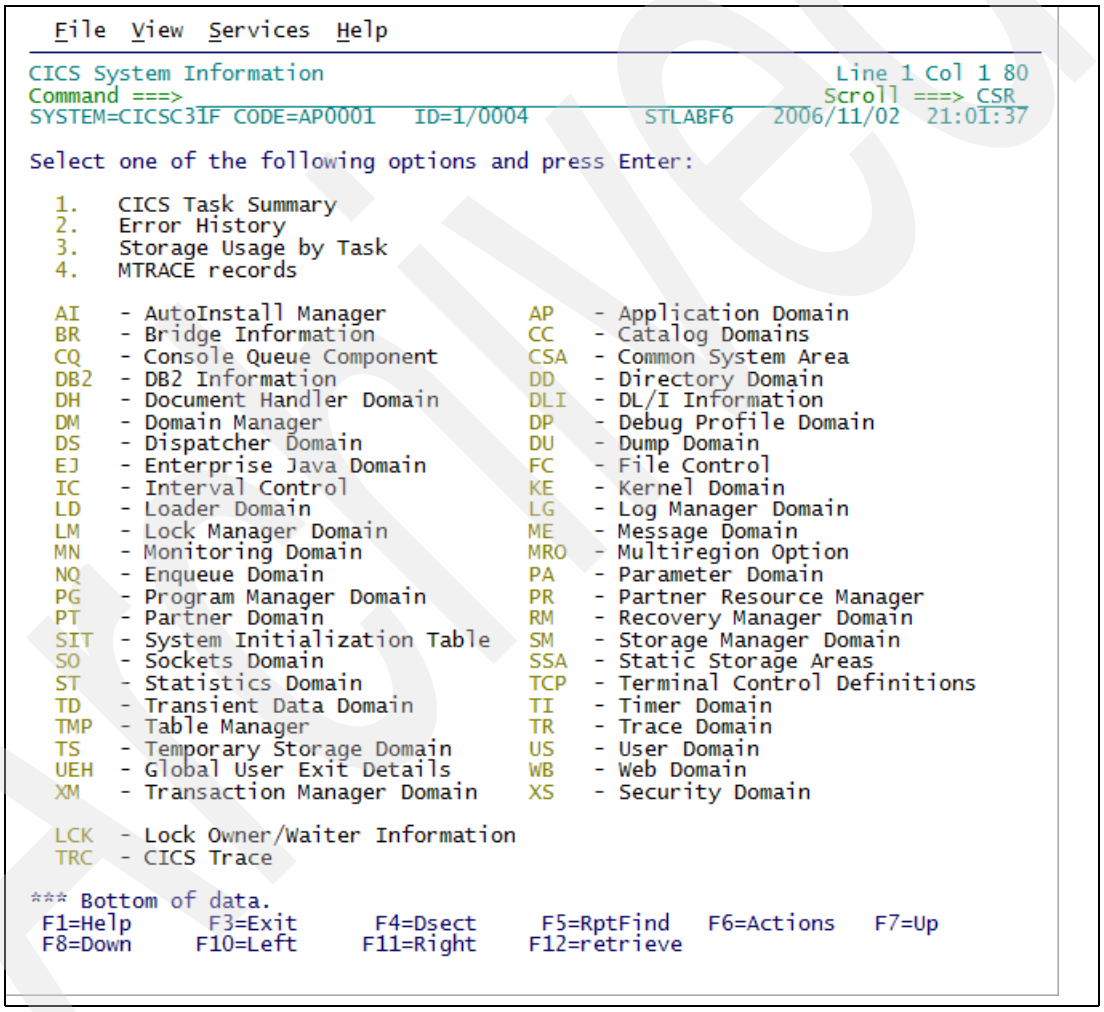


Figure 19-8 CICS system information

Most of the domains display summary information with the ability to select further information as required. Since there is more information than is practical to include in this book, we next provide a sample of the information we looked at.

- First we looked at the Error History as shown in Figure 19-9.

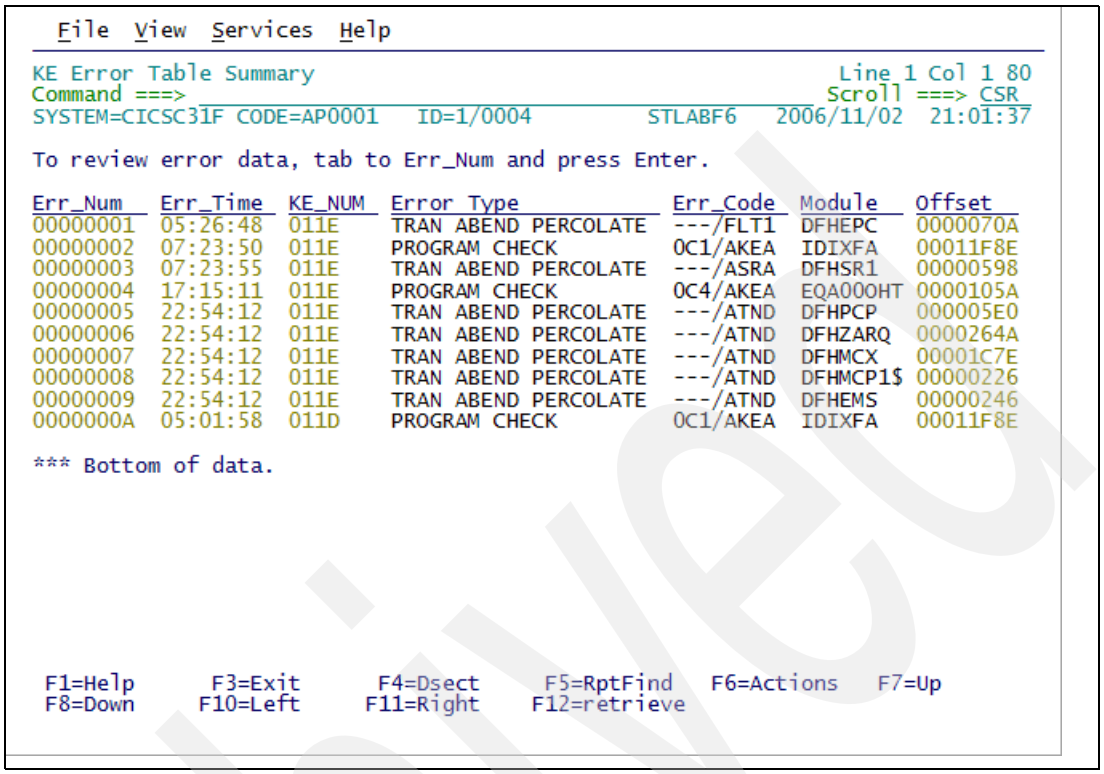


Figure 19-9 Error History display

- We then selected Err_Num **0000000A** which displayed the error information as shown in Figure 19-10.
- From here we selected the PSW address 95A55F8E. Note that Fault Analyzer automatically handles the high order bit. From the resulting hex storage display we used the **DISASM** command to disassemble the code at the PSW address as shown in Figure 19-11.

```
  File  View  Services  Help
KE Domain Error Data                                          Line 1 Col 1 80
Command ===>                                                  Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004          STLABF6   2006/11/02  21:01:37

Error Number: 0000000A   Error Code: 0C1/AKEA   Error Type: PROGRAM CHECK

Date (GMT): 2006/11/03   Time (GMT): 05:01:58

KE Error Data Address: 12D68798 KE_NUM: 011D   TCA Addr: 130AE680   DS_TASK: 2

Error occurred in program IDIXFA   at offset 00011F8E under the CICS RB in bas

Execution key at Program Check/Abend:  8

PSW:  078D0000 95A55F8E Module IDIXFA + X'11F8E'

CICS Registers:

R0:  143F75EA  141846 bytes of storage addressable
R1:  143F7070  143248 bytes of storage addressable
R2:  00000003  CICS CSB - Connection Status Block + X'3'
R3:  1590E008  237560 bytes of storage addressable
R4:  15A64018  Module IDIXFA + X'20018'
R5:  15A5574C  Module IDIXFA + X'1174C'
R6:  15A5574C  Module IDIXFA + X'1174C'
R7:  00001000  CICS CCB - Connection Control Block + X'20' - storage invalid
R8:  143F8008  139256 bytes of storage addressable
R9:  15A562E8  Module IDIXFA + X'122E8'
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 19-10 Err_Num 0000000A

```
  File  View  Services  Help
Disassemble command: MODULE IDIXFA.                           Line 1 Col 1 80
Command ===>                                                  Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004          STLABF6   2006/11/02  21:01:37
Address  Offset    Hex            Instruction
15A55F8E       +0 0000                                          Not a Valid I
15A55F90       +2 95E2 D5F9       CLI   1529(R13),226
15A55F94       +6 4770 ACE6       BC    7,3302(,R10)
15A55F98       +A 4110 D068       LA    R1,104(,R13)
15A55F9C       +E 41E0 963B       LA    R14,1595(,R9)
15A55FA0      +12 41F0 934C       LA    R15,844(,R9)
15A55FA4      +16 90EF 1000       STM   R14,R15,0(R1)
15A55FA8      +1A 9680 1004       OI    4(R1),128
15A55FAC      +1E 58F0 9300       L     R15,768(,R9)
15A55FB0      +22 50D0 D054       ST    R13,84(,R13)
15A55FB4      +26 41D0 D000       LA    R13,0(,R13)
15A55FB8      +2A 05EF            BALR  R14,R15
15A55FBA      +2C 58D0 D054       L     R13,84(,R13)
15A55FBE      +30 9240 D605       MVI   1541(R13),64
15A55FC2      +34 D244 D606 D605  MVC   1542(69,R13),1541(R13)
15A55FC8      +3A D245 D605 94E4  MVC   1541(70,R13),1252(R9)
15A55FCE      +40 95E2 D5FF       CLI   1535(R13),226
15A55FD2      +44 4770 AD14       BC    7,3348(,R10)
15A55FD6      +48 4110 D068       LA    R1,104(,R13)
15A55FDA      +4C 41E0 9644       LA    R14,1604(,R9)
15A55FDE      +50 41F0 9350       LA    R15,848(,R9)
15A55FE2      +54 90EF 1000       STM   R14,R15,0(R1)
15A55FE6      +58 9680 1004       OI    4(R1),128
15A55FEA      +5C 58F0 9300       L     R15,768(,R9)
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

*Figure 19-11 **DISASM** of PSW Address 15A55F8E*

- Next we returned to the CICS Information panel and selected **MTRACE.** This displayed the screen shown in Figure 19-12. Notice that the Job IDs are all point-and-shoot fields, and selecting one of these limits the MTRACE display to just those records for the selected job. For example, we selected job **JOB03126** as shown in Figure 19-13.

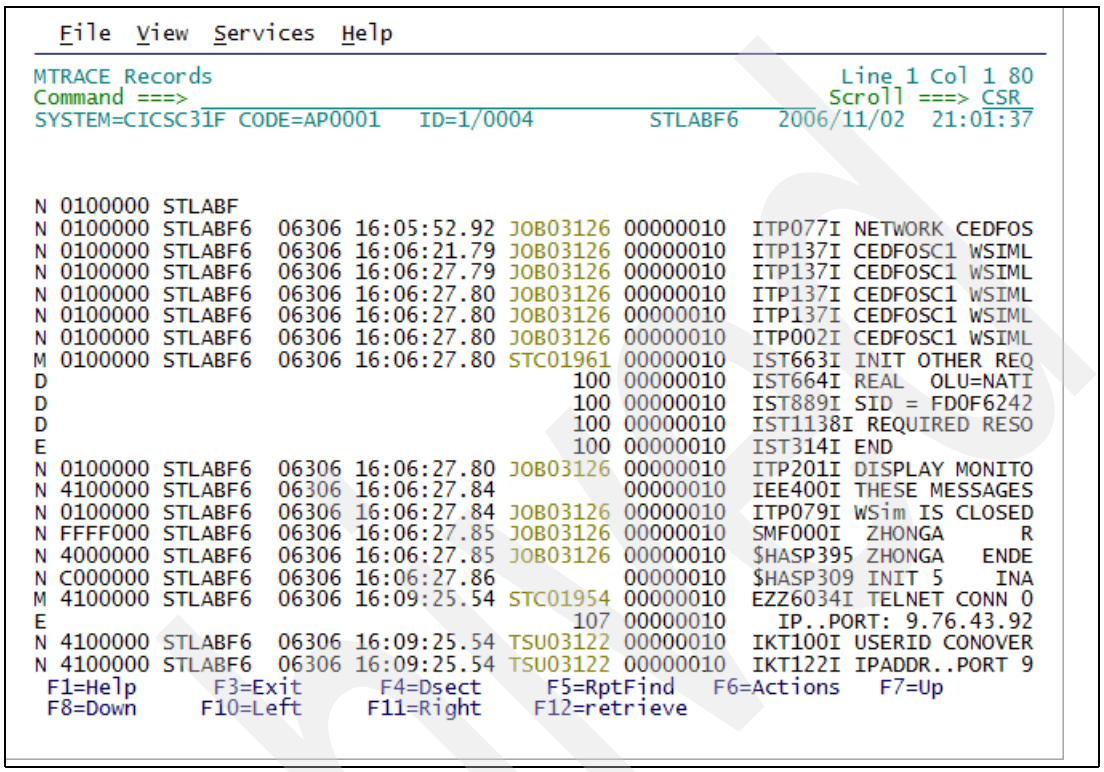


Figure 19-12 MTRACE display

```
  File  View  Services  Help
MTRACE Records filtered by JOB03126                          Line 1 Col 1 80
Command ===>                                                Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004         STLABF6   2006/11/02  21:01:37


N 0100000 STLABF6  06306 16:05:52.92 JOB03126 00000010  ITP077I NETWORK CEDFOS
N 0100000 STLABF6  06306 16:06:21.79 JOB03126 00000010  ITP137I CEDFOSC1 WSIML
N 0100000 STLABF6  06306 16:06:27.79 JOB03126 00000010  ITP137I CEDFOSC1 WSIML
N 0100000 STLABF6  06306 16:06:27.80 JOB03126 00000010  ITP137I CEDFOSC1 WSIML
N 0100000 STLABF6  06306 16:06:27.80 JOB03126 00000010  ITP137I CEDFOSC1 WSIML
N 0100000 STLABF6  06306 16:06:27.80 JOB03126 00000010  ITP002I CEDFOSC1 WSIML
N 0100000 STLABF6  06306 16:06:27.80 JOB03126 00000010  ITP201I DISPLAY MONITO
N 0100000 STLABF6  06306 16:06:27.84 JOB03126 00000010  ITP079I WSim IS CLOSED
N FFFF000 STLABF6  06306 16:06:27.85 JOB03126 00000010  SMF000I  ZHONGA      R
N 4000000 STLABF6  06306 16:06:27.85 JOB03126 00000010  $HASP395 ZHONGA    ENDE

*** Bottom of data.




 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 19-13 MTRACE records for job JOB03126

- Next we entered the command **!SIT** on the command line to display the **SIT** values, as shown in Figure 19-14 to Figure 19-16.

```
  File  View  Services  Help
SIT values                                                   Line 1 Col 1 80
Command ===>                                                Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004         STLABF6   2006/11/02  21:01:37

SIT Address: 12DF2F98


ADI . . . . . . . . . . . . : 30
AIBRIDGE. . . . . . . . . . : AUTO
AICONS. . . . . . . . . . . : NO
AIEXIT. . . . . . . . . . . : DFHZATDX
AILDELAY. . . . . . . . . . : 000000
AIQMAX. . . . . . . . . . . : 100
AIRDELAY. . . . . . . . . . : 000700
AKPFREQ . . . . . . . . . . : 4000
APPLID. . . . . . . . . . . : (CICSC31F,CICSC31F)
AUTCONN . . . . . . . . . . : 0
AUTODST . . . . . . . . . . : NO
AUXTR . . . . . . . . . . . : NO
AUXTRSW . . . . . . . . . . : NEXT
BMS . . . . . . . . . . . . : (FULL,,UNALIGN,DDS)
BRMAXKEEPTIME . . . . . . . : 86400
CDSASZE . . . . . . . . . . : 0K
CICSSVC . . . . . . . . . . : 216
CILOCK. . . . . . . . . . . : NO
CLINTCP . . . . . . . . . . : 437 (LATIN-1, LATIN-9)
CLSDSTP . . . . . . . . . . : NOTIFY
CLT . . . . . . . . . . . . :
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 19-14 Formatted SIT (1 of 3)

```
  File  View  Services  Help
SIT values                                                    Line 26 Col 1 80
Command ===>                                                  Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004          STLABF6   2006/11/02  21:01:37
CMDPROT . . . . . . . . . . : YES
CMDSEC. . . . . . . . . . . : ASIS
CONFDATA. . . . . . . . . . : HIDECT
CONFTXT . . . . . . . . . . : NO
CPSMCONN. . . . . . . . . . : NO
CRLSERVER . . . . . . . . . :
CSDACC. . . . . . . . . . . : READWRITE
CSDBKUP . . . . . . . . . . : STATIC
CSDBUFND. . . . . . . . . . : 0
CSDBUFNI. . . . . . . . . . : 0
CSDDISP . . . . . . . . . . :
CSDDSN. . . . . . . . . . . :
CSDFRLOG. . . . . . . . . . : NO
CSDINTEG. . . . . . . . . . : UNCOMMITTED
CSDJID. . . . . . . . . . . : NO
CSDLSRNO. . . . . . . . . . : 1
CSDRECOV. . . . . . . . . . : NONE
CSDRLS. . . . . . . . . . . : NO
CSDSTRNO. . . . . . . . . . : 6
CWAKEY. . . . . . . . . . . : USER
DAE . . . . . . . . . . . . : NO
DATFORM . . . . . . . . . . : MMDDYY
DB2CONN . . . . . . . . . . : YES
DBCTLCON. . . . . . . . . . : YES
DEBUGTOOL . . . . . . . . . : YES
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 19-15 Formatted SIT (2 of 3)

```
  File  View  Services  Help
SIT values                                                    Line 51 Col 1 80
Command ===>                                                  Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004          STLABF6   2006/11/02  21:01:37
DFLTUSER. . . . . . . . . . : CICSUSER
DIP . . . . . . . . . . . . : NO
DISMACP . . . . . . . . . . : YES
DOCCODEPAGE . . . . . . . . : 037
DSALIM. . . . . . . . . . . : 5120K
DSHIPIDL. . . . . . . . . . : 020000
DSHIPINT. . . . . . . . . . : 120000
DSRTPGM . . . . . . . . . . : NONE
DTRPGM. . . . . . . . . . . : DFHDYP
DTRTRAN . . . . . . . . . . : CRTX
DUMP. . . . . . . . . . . . : NO
DUMPDS. . . . . . . . . . . : AUTO
DUMPSW. . . . . . . . . . . : NO
DURETRY . . . . . . . . . . : 30
ECDSASZE. . . . . . . . . . : 0M
EDSALIM . . . . . . . . . . : 256M
EJBROLEPRFX . . . . . . . . :
ENCRYPTION. . . . . . . . . : STRONG
EODI. . . . . . . . . . . . : E0
ERDSASZE. . . . . . . . . . : 0M
ESDSASZE. . . . . . . . . . : 0K
ESMEXITS. . . . . . . . . . : NOINSTLN
EUDSASZE. . . . . . . . . . : 0M
FCT . . . . . . . . . . . . : NO
FEPI. . . . . . . . . . . . : NO
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 19-16 Formatted SIT (3 of 3)

- Next we entered **!XM** to display the Transaction Manager domain details as shown in Figure 19-17 and Figure 19-18.

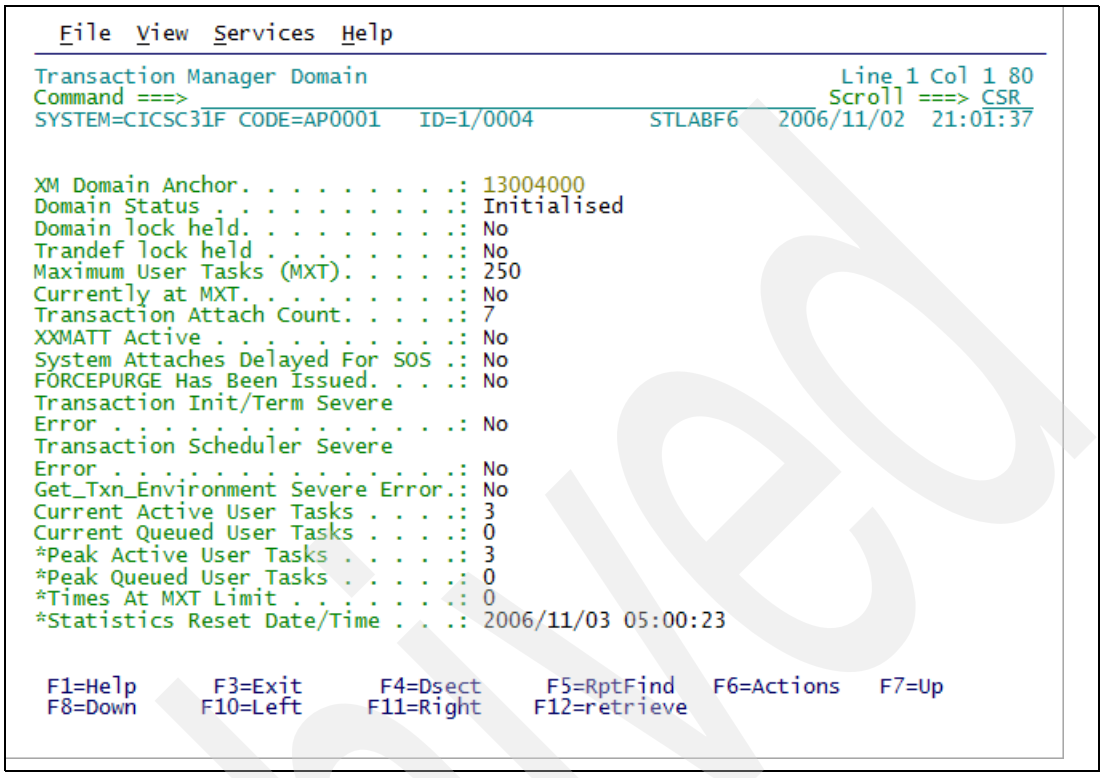


Figure 19-17 Transaction manager (XM) display (1 of 2)

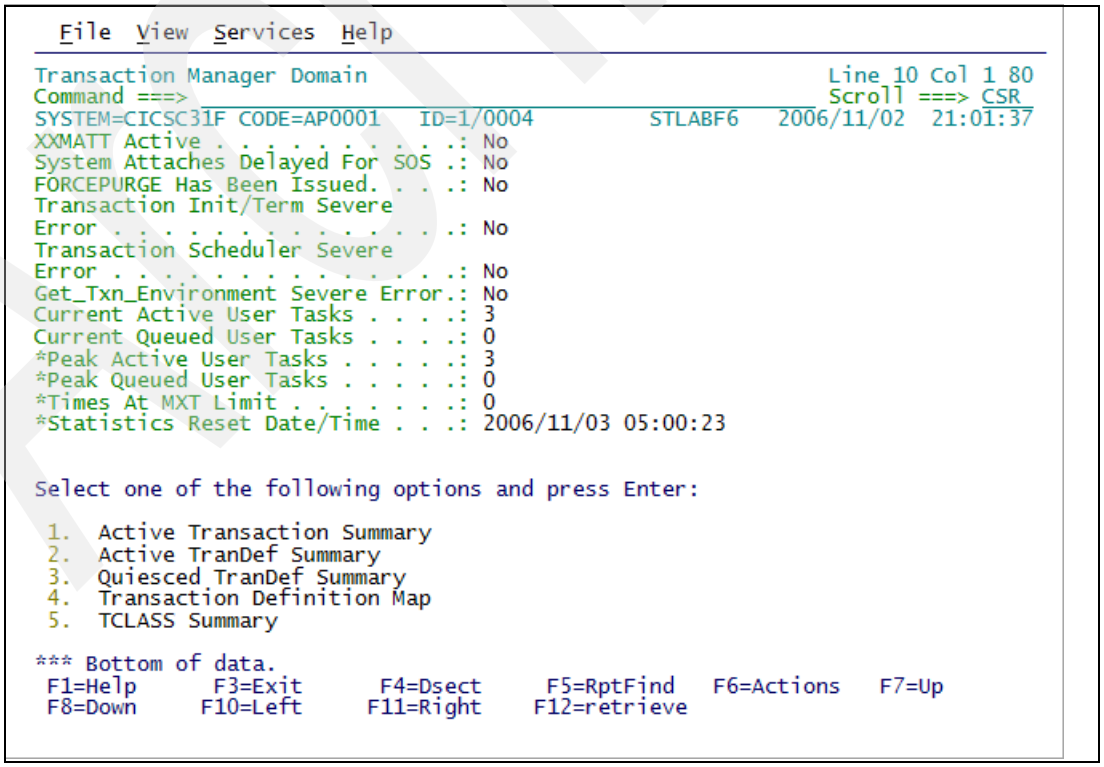


Figure 19-18 Transaction manager (XM) display (2 of 2)

- From here we selected option **4** - Transaction Definition Map (see Figure 19-19) and then selected the **CFA** point-and-shoot field as shown in Figure 19-20.

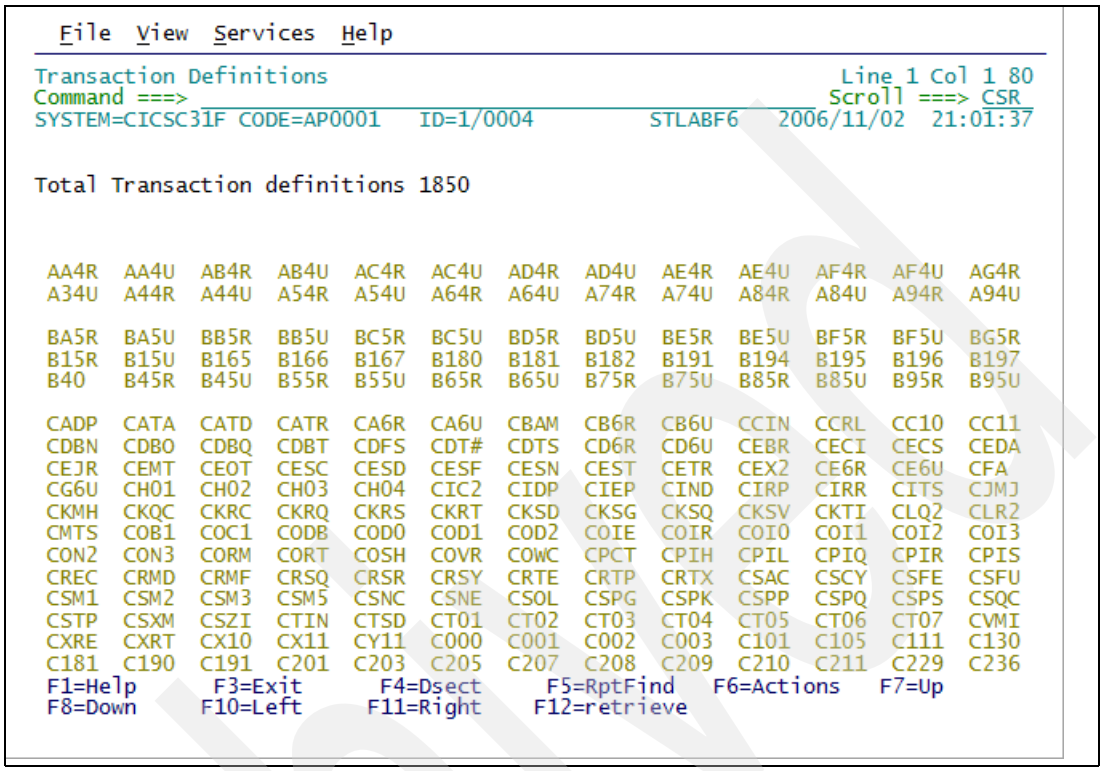


Figure 19-19 Transaction Definition map

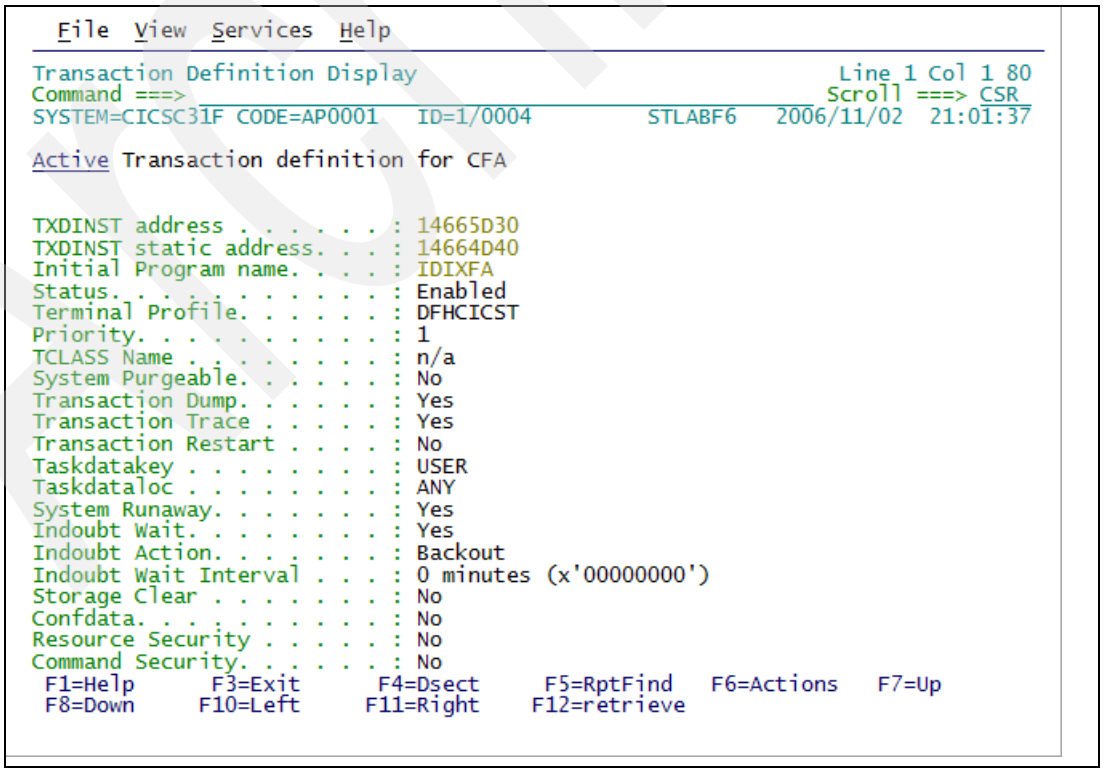


Figure 19-20 CFA Transaction definition

- From here, we select program **IDIXFA** to get the PPTE details for this program as shown in Figure 19-21 and Figure 19-22.

```
  File  View  Services  Help
Program Manager PPTE Entry                                    Line 1 Col 1 80
Command ===>                                                  Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004         STLABF6   2006/11/02  21:01:37


PPTE address 1466C7C0

Program Name. . . . . . . . : IDIXFA
Module Type . . . . . . . . : Program
Defined Language. . . . . . : Assembler
Deduced Language. . . . . . : Assembler
Install Type. . . . . . . . : Built From Group List
CEDF Status . . . . . . . . : NOT Allowed
Availability. . . . . . . . : Enabled
Data Location . . . . . . . : Any
Execution Key . . . . . . . : CICS
Execution Set . . . . . . . : Full API
Reload Status . . . . . . . : Do Not Reload a new copy each use
Dynamic Status. . . . . . . : DPL NOT Permitted
Load Status . . . . . . . . : Loaded
Hold Status . . . . . . . . : Task Lifetime
PWGE Wait Elements. . . . . : None
Use Count . . . . . . . . . : 1
Lock Owner. . . . . . . . . : n/a
Remote Program Id . . . . . : n/a
Remote Sysid. . . . . . . . : n/a
Remote Transid. . . . . . . : n/a
Concurrency (Adjusted). . . : Quasireentrant
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 19-21 PPTE details for program IDIXFA (1 of 2)

```
  File  View  Services  Help
Program Manager PPTE Entry                                   Line 12 Col 1 80
Command ===>                                                 Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004         STLABF6   2006/11/02  21:01:37
Data Location . . . . . . . : Any
Execution Key . . . . . . . : CICS
Execution Set . . . . . . . : Full API
Reload Status . . . . . . . : Do Not Reload a new copy each use
Dynamic Status. . . . . . . : DPL NOT Permitted
Load Status . . . . . . . . : Loaded
Hold Status . . . . . . . . : Task Lifetime
PWGE Wait Elements. . . . . : None
Use Count . . . . . . . . . : 1
Lock Owner. . . . . . . . . : n/a
Remote Program Id . . . . . : n/a
Remote Sysid. . . . . . . . : n/a
Remote Transid. . . . . . . : n/a
Concurrency (Adjusted). . . : Quasireentrant
Concurrency (Defined) . . . : Quasireentrant
API (Adjusted)  . . . . . . : Cicsapi
API (Defined) . . . . . . . : Cicsapi
Run Under JVM . . . . . . . : NO
PLB Address . . . . . . . . : 142E06D4

CPE address: 1466DC68 Most recent APE: 142E2BE0

Loaded from DFHRPL dataset: ADTOOLS.FA710.SIDIAUTH

*** Bottom of data.
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 19-22 PPTE details for program IDIXFA (2 of 2)

- We then entered command **!DS** to look at the dispatcher domain. See Figure 19-23. Next we selected option **1** to view the Task Summary as shown in Figure 19-24.

```
  File  View  Services  Help
Dispatcher Domain                                          Line 1 Col 1 80
Command ===>                                              Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004        STLABF6   2006/11/02  21:01:37


Dispatcher Domain Anchor: 12F29000

Select one of the following options and press Enter:

  1.   Task Summary
  2.   Open TCB Pool Summary
  3.   Open TCB Task Summary

*** Bottom of data.









 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 19-23 Dispatcher domain

```
  File  View  Services  Help
Dispatcher Domain - Summary                                Line 1 Col 1 80
Command ===>                                              Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004        STLABF6   2006/11/02  21:01:37



                                  Resource Resource            Time Of  Timeout  DTA
DS_TOKEN KE_TASK  T S F P TT Type      Name            W Suspend  Due      (DS
00000001 12D8BC00 S S N N -  ENF       NOTIFY          M 14:50:47    -      12F
00060003 12DA8C00 S R                                                     12F
000A0003 12FAFC00 S S N N -  SODOMAIN  SO_NOWORK       M 04:42:49    -      12F
000E0003 142CE480 S S N N -            DLCNTRL         M 14:51:03    -      12F
00120005 1476C880 S S N N -  ZC        DFHZNAC1        S 05:01:23    -      12F
00820003 12F91C00 S S N N -  ICMIDNTE  DFHAPTIM        S 08:00:23    -      12F
00880003 12F29C00 S S N N IN LGHARTBT  LG_MGRST        S 05:01:56 05:02:06 12F
008E0003 14258480 S S N N -  ICEXPIRY  DFHAPTIX        S 05:01:36    -      12F
01080003 14297080 S S N N -  TCP_NORM  DFHZDSP         W 05:01:58    -      12F
018E0005 14297880 S S N N -  CSNC      MROQUEUE        M 14:51:00    -      12F
01900003 12D8B000 S S N N -  TIEXPIRY  DS_NUDGE        S 05:01:36    -      12F
01920005 14258880 S S N N IN SHSYSTEM                  S 04:51:10 05:51:10 12F
02000007 142CE080 S S P N -  USERWAIT  CDB2TIME        M 05:01:36    -      230
02040001 1302C080 S S N N IN SMSYSTEM                  S 04:59:02 05:04:02 230
02060001 1302C480 S S N N -  smsyre    smva_ecb        M 14:50:47    -      230
0F0000E1 14240480 N S N N -  EKCWAIT   SINGLE          W 22:54:18    -      230
0F040001 14240880 N S P N -                            M 05:01:08    -      230
0F0C04BD 14240080 N R                                                     230

 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 19-24 Dispatcher Task Summary

One of the features of Fault Analyzer CICS system dump support is to display times in local time value. For example, the 05:01:08 shown in Figure 19-24 on page 520 is extracted from a dispatcher control block and is a GMT time value. By selecting this field, Fault Analyzer displays the local time value as shown in Figure 19-25.

Note: In this instance, the adjusted time is not an exact multiple of hours as might be expected, which can be very significant when looking at task waits, and so on.

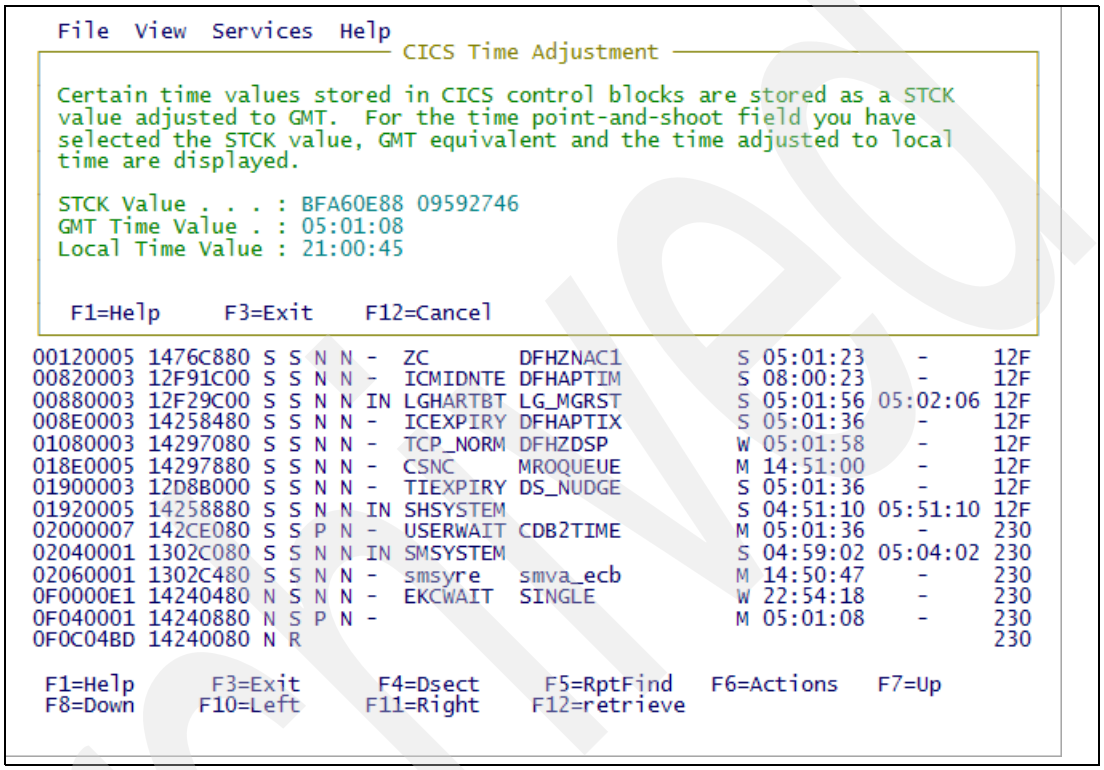


Figure 19-25 Converted Time value

- Finally we looked at the AP summary by entering **AP** on the command line. See Figure 19-26.

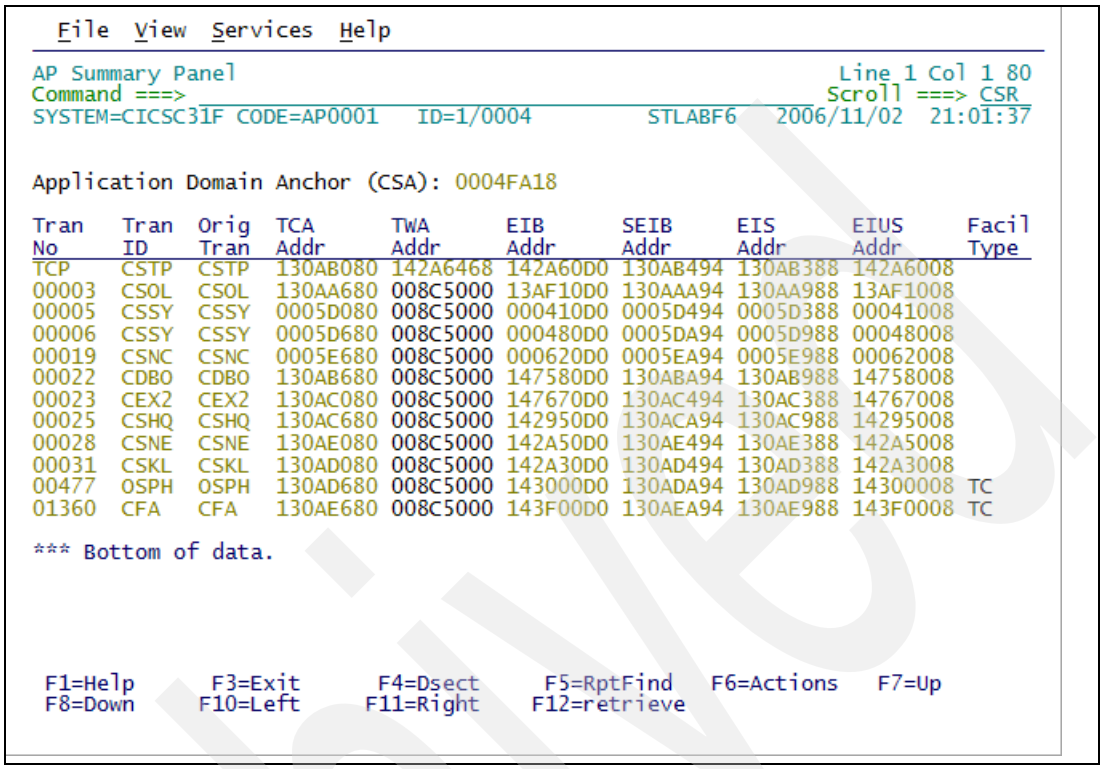


Figure 19-26 AP Summary panel

From here we selected **Tran No 01360** to display the “Task Details” panel for task 1360 as shown in Figure 19-27. One of the options from this panel is to display the CICS trace entries for the task. Selecting this option displays a pop-up window, as shown in Figure 19-28, where full trace selection/filtering can be performed. In our case we simply displayed the Full trace records for task 1360 as shown in Figure 19-29.

```
  File  View  Services  Help
CICS Task Detail                                              Line 1 Col 1 80
Command ===>                                                  Scroll ===> CSR
SYSTEM=CICSC31F CODE=AP0001   ID=1/0004        STLABF6   2006/11/02  21:01:37

                        Tran            Initial  Facility
KE_NUM Status           Number Tran ID  Program  Id        In Error Userid
011D   ***Running**     01360  CFA      IDIXFA   0020      *YES*    SIMCOCK

Attach Date/Time. .: 2006/11/03 05:01:55
Current program . .: IDIXFA

AP Info                 Kernel Info             Storage Info
UTCA      : 130AE680    KE_TASK : 14240080      SMX : 230160F0
STCA      : 130AE780    KE_KTCB : 12DB4000
EIB       : 143F00D0
SEIB      : 130AEA94
EIS       : 130AE988
EIUS      : 143F0008
TWA       : 008C5000

XM Info                 DS Info                 Facility Info
TXN       : 1300A030    DS_TASK : 230B4980      Type    : TC
TXD       : 14665D30                            Name    : 0020
                                                Address : 14496970

PG Link Levels

AP Link Levels
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 19-27 Task details for task 1360

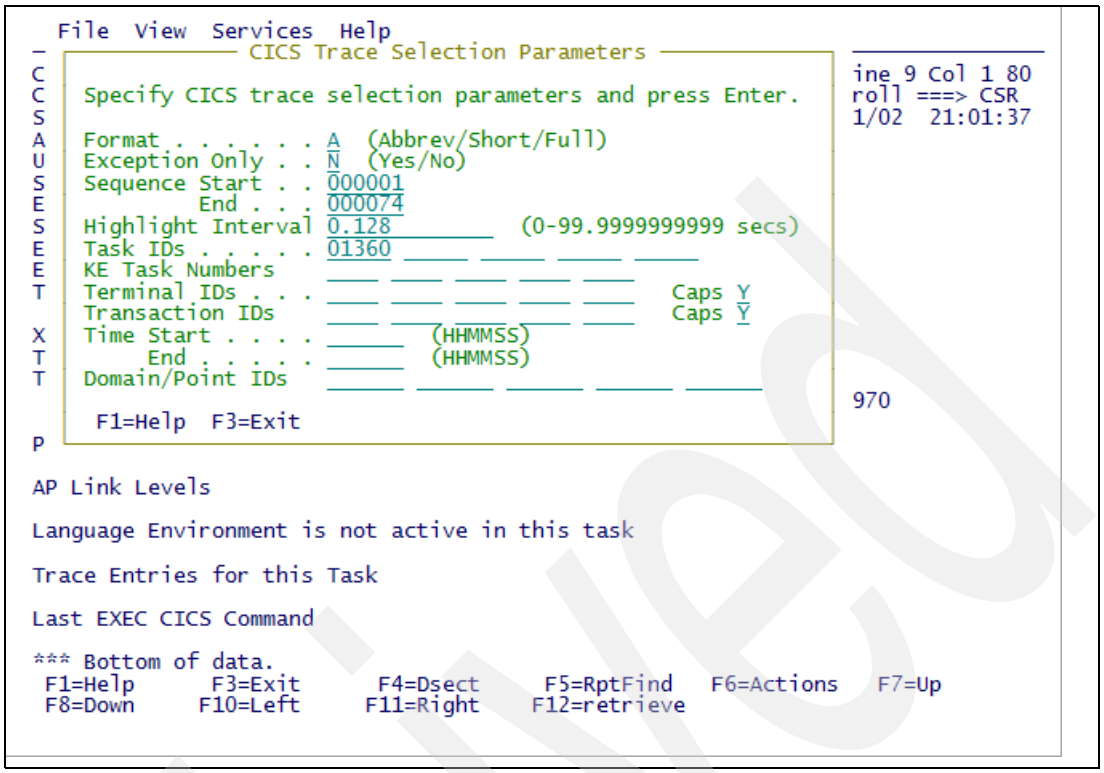


Figure 19-28 Trace selection popup

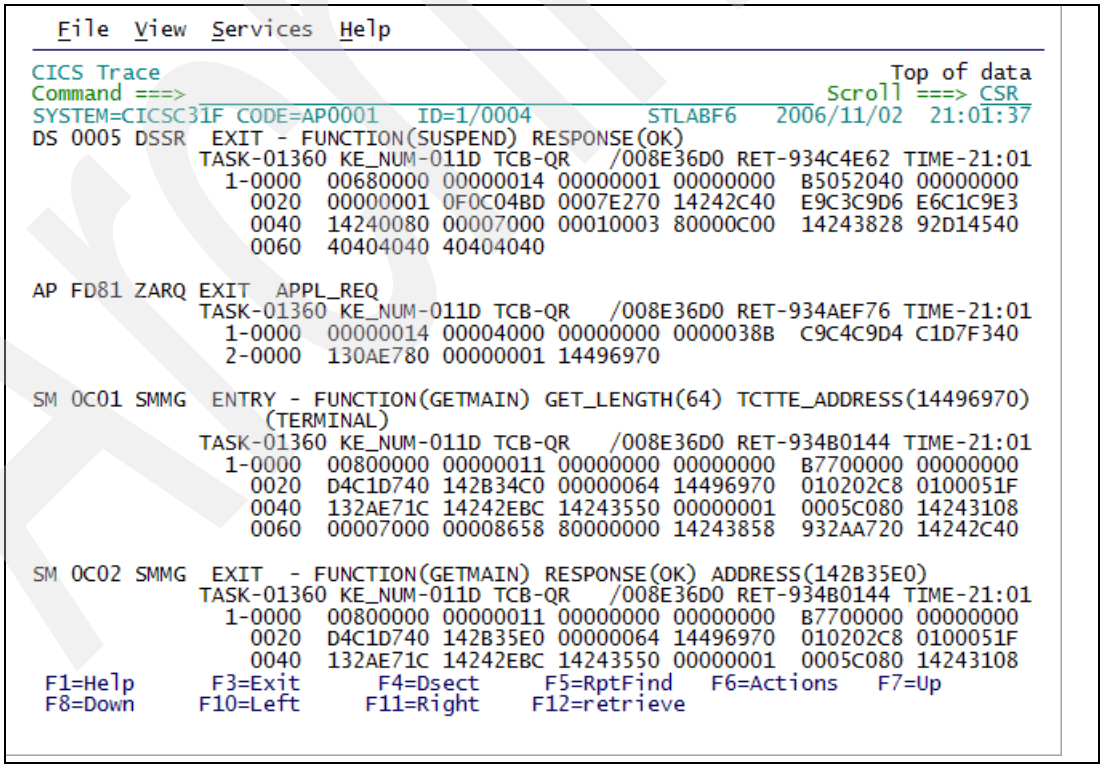


Figure 19-29 Full trace entries for task 1360

20

Fault Analyzer interactive analysis under CICS

In this chapter we discuss using Fault Analyzer interactive analysis under CICS.

20.1 Introduction

As well as the ISPF interface for performing interactive reanalysis, Fault Analyzer also provides a similar interface under CICS. This interface, invoked in the same way as any other CICS transaction, is identical in appearance and functionality as the ISPF interface with only a few exceptions and restrictions.

This interface is primarily intended for users who might not have TSO logon capability on an MVS image, but have to review and analyze history file information on that MVS image.

20.2 Restrictions

The following restrictions and variations apply when using the CICS interactive interface compared with the ISPF interface:

1. Submitting a batch reanalysis of the fault entry is not supported.
2. Functions that invoke ISPF EDIT are not supported:
   - Editing the options data set prior to interactive reanalysis
   - Altering allocated data sets prior to interactive reanalysis
   - EDIT of a User Formatting EXEC from a list of available EXECs
   - EDIT of a DSECT from the DSECT list display
3. Since this component is not running under TSO, no prefixing of data set names is performed. That is, where a data set name can be entered, for example, for a Fault History File or View, the data set name has to be fully qualified with or without quotes.
4. If an action-bar pull-down menu is selected, but no option will be selected, then PF6 should be used to deselect the menu — no other function key is available while a pull-down menu is displayed.
5. ISPF profile changes made while in the Interactive Reanalysis Report, for example, changing the location of the command line, might not be immediately reflected upon return to the Fault Entry List display. However, the profile changes will be detected on the next invocation of the main CICS transaction.

20.3 Resource definitions

A sample CICS DFHCSDUP job is provided as member IDIWCIDI in the IDI.SIDISAM1 data set, that can be used to make the required CICS resource definitions. There is one transaction definition and one associated program definition required. Optionally, there is also a transaction profile definition, which specifies the SCRNSIZE(ALTERNATE) option so that different screen sizes can be used. See Example 20-1.

Example 20-1 CICS resource definitions

```
//IDICICS EXEC PGM=DFHCSDUP,REGION=1024K,
//            PARM='CSD(READWRITE),PAGESIZE(60),NOCOMPAT'
//STEPLIB  DD DISP=SHR,DSN=<xxx>.SDFHLOAD
//DFHCSD   DD DISP=SHR,DSN=<xxx>.DFHCSD
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DEF PROFILE(IDIPROF) GROUP(FACICS) SCRNSIZE(ALTERNATE)
  DEF PROGRAM(IDIXIPV) GROUP(FACICS) EXECKEY(CICS)
      LANGUAGE(ASSEMBLER) CEDF(NO) DATALOCATION(ANY)
  DEF TRANSACTION(IDI) GROUP(FACICS)
      PROGRAM(IDIXIPV) TASKDATALOC(ANY)
      PROFILE(IDIPROF)
  ADD GROUP(FACICS) L(<list-name>)
/*
```

20.4 JCL changes

The following modifications must be made to your CICS JCL as shown in Example 20-2. It is assumed that IDI is the data set name high-level qualifier that was used during Fault Analyzer installation:

1. Add data set IDI.SIDIAUTH to the DFHRPL concatenation of the CICS JCL.
2. Allocate a new profile data set to be assigned to the IPVPROF and IPVTABL DD names below — for example, IDI.IDIPPROF. This data set should be defined as a PDS(E) with LRECL=80 and RECFM=FB.
3. Add the following additional DD names to the CICS JCL.

Example 20-2 CICS JCL Changes

```
//IPVPLIB DD DISP=SHR,DSN=IDI.SIDIIPVP
//       DD DISP=SHR,DSN=IDI.SIDIPLIB
//       DD DISP=SHR,DSN=IDI.SIDIMLIB
//       DD DISP=SHR,DSN=IDI.SIDISLIB
//IPVPROF DD DISP=SHR,DSN=IDI.IDIPPROF
//IPVTABL DD DISP=SHR,DSN=IDI.IDIPPROF
//IPVTLIB DD DISP=SHR,DSN=IDI.SIDITLIB
```

20.5 Using The CICS Interactive Interface

Assuming that the transaction definition shown in Example 20-1 on page 527 was used, then starting the FA CICS Interface is simply a case of logging on to CICS and starting transaction **IDI**. For this book, we used logmode **D4A32XX3** so that we had a comparable screen size as ISPF, as shown in Example 20-3.

Example 20-3 Using LOGMODE for alternate screen size

```
LOGON APPLID(cics_applid) LOGMODE(D4A32XX3)
```

Having started Fault Analyzer, the screen images shown in Figure 20-1 and Figure 20-2 were captured to illustrate that there is no difference between Interactive reanalysis under ISPF and CICS.

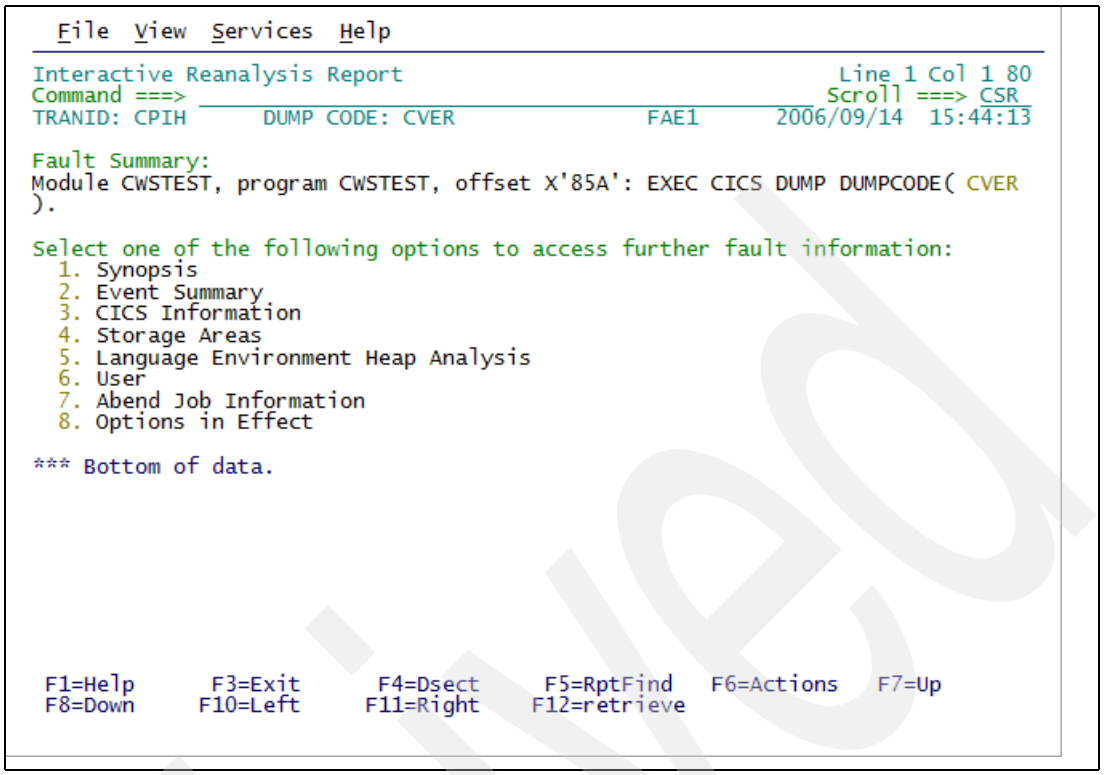


Figure 20-1 Screen image from ISPF interactive reanalysis

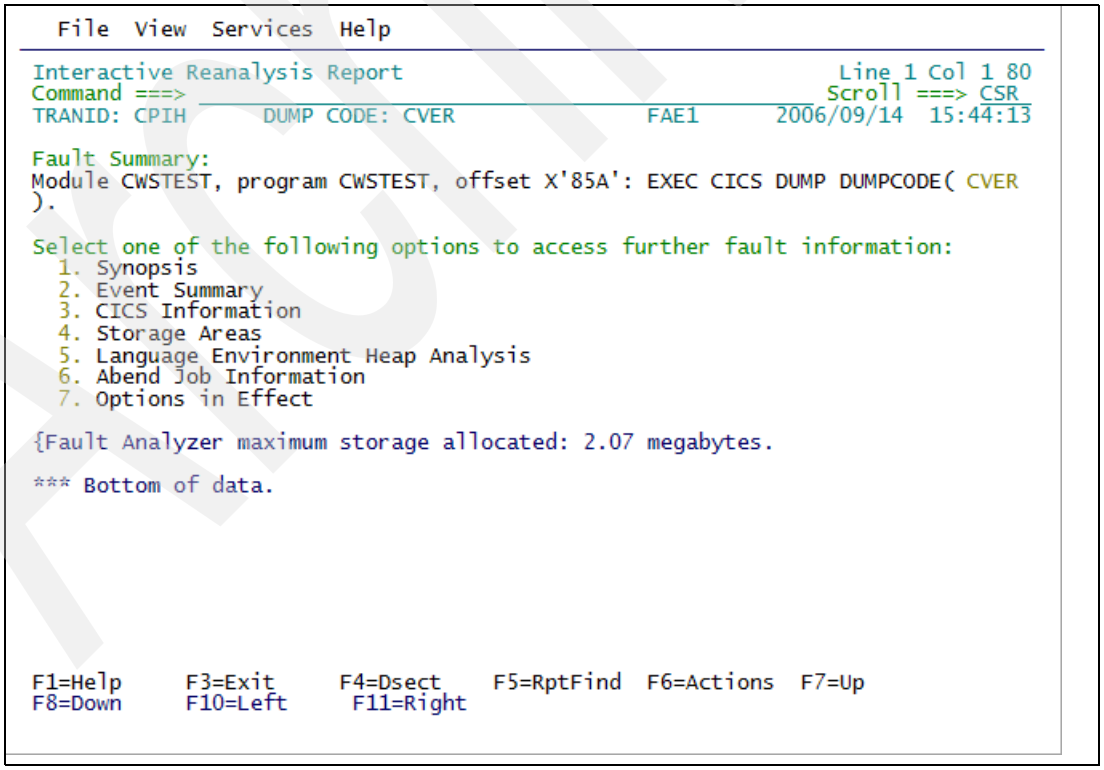


Figure 20-2 Screen image from CICS interactive reanalysis

21

Fault Analyzer and subsystems

In this chapter we describe how Fault Analyzer interacts with the various subsystems. We also explain how to set up and customize FA with CICS, DB2, and IMS.

Also included are details on the Recovery Fault Recording feature of Fault Analyzer.

21.1 Fault Analyzer and CICS

We recommend that you install the resources identified in Table 21-1 using the steps described in *IBM Fault Analyzer for z/OS and its User's Guide and Reference*, under "Customizing the CICS environment".

Table 21-1 Contents of group IDI

Names	Type	Group
IDIPLT	PROGRAM	IDI
IDIXCCEE	PROGRAM	IDI
IDIXCX52	PROGRAM	IDI
IDIXCX53	PROGRAM	IDI
IDIXFA	PROGRAM	IDI
CFA	TRANSACTION	IDI

Fault Analyzer can be invoked three different ways under CICS:

- XPCABND

  Global user exit using program IDIXCX52 or IDIXCX53. This is the main exit provided to invoke Fault Analyzer for CICS transaction fault analysis.

- XDUREQ

  Global user exit using program IDIXCX52 or IDIXCX53. This exit can be used to invoke Fault Analyzer for CICS dumps generated from an EXEC CICS DUMP command. The analysis performed by Fault Analyzer at this exit point is the same as for the XPCABND exit point.

- LE Exit

  LE abnormal termination exit using program IDIXCCEE. This exit is only effective with Language Environment based application programs when the CEECXTAN exit has been set.

  CICS AKCS abends can be analyzed using this exit if the failing program is LE enabled and an entry exists in the CICS dump table for AKCS, specifying that a transaction dump is to be taken.

The first two of these exits are CICS global user exit points, and Fault Analyzer is enabled and disabled at these points using CICS calls. This means that, by default, these exit points are not enabled in a CICS region. They are enabled either by adding an entry to the CICS PLT or by using the supplied CFA transaction once CICS has initialized.

The LE abnormal termination exit, however, requires a modification to LE to invoke Fault Analyzer, and hence its effect is system wide. Fault Analyzer provides a mechanism for controlling the use of this exit at a CICS region level, but in order for this mechanism to work, the LE exit must first be enabled system wide. Once enabled at a system-wide level, then the initial setting in a CICS region is enabled.

IDI.SIDIAUTH must be added to the DFHRPL concatenation of the CICS JCL for any of the foregoing exits to be successfully enabled.

To use Fault Analyzer with CICS, perform the following steps:

1. Configure Language Environment for CICS to invoke Fault Analyzer.
2. Define the required programs to your CICS system.
3. Add the required programs to your startup PLT.
4. Define a transaction for Fault Analyzer.

21.1.1 Configuring Language Environment for CICS to invoke Fault Analyzer

Fault Analyzer provides a Language Environment abnormal termination exit for CICS, IDIXCCEE, as an additional method of invoking Fault Analyzer to the CICS XPCABND global exit. This exit is specific to LE U1xxx or U4xxx-type abend in reference to CICS LE abnormal termination CEECXTAN CSECT exit IDIXCCEE. Defining required program to CICS

The following programs and BMS map must be defined to your CICS system, unless CICS program auto install is active:

- IDIPLT
- IDIPLTD
- IDIPLTS
- IDIXCX52
- IDIXCX53
- IDIXFA
- IDIXMAP (BMS map)

These programs are all assembler programs, and should be defined in a group that is included in a group list used during CICS startup.

A sample job is provided as member IDISCICS in data set IDI.SIDISAM1 to make these definitions using the DFHCSDUP utility.

Keep the following considerations in mind:

- In order for Fault Analyzer to be invoked under CICS, it is necessary to add IDI.SIDIAUTH to the DFHRPL concatenation.
- CICS tracing must be active for Fault Analyzer to display CICS trace information.
- If CICS is used without LE in the LINKLIST, it is necessary to install the IDILEDS USERMOD.
- Fault Analyzer is not invoked for CICS transaction abends if the NODUMP keyword is used on the EXEC CICS ABEND statement that is causing the abend, as shown in Example 21-1.

Example 21-1 EXEC CICS ABEND statement

```
EXEC CICS ABEND ABCODE(<abcd>) NODUMP END_EXEC
```

Note: ABCODE must be used in order for Fault Analyzer to be invoked.

21.1.2 Controlling CICS transaction abend analysis

Once the CFA transaction has been installed it can be used to install or uninstall the following Fault Analyzer invocation exits:

- XPCABND CICS global user exit
- XDUREQ CICS global user exit
- LE abnormal termination exit

In addition, the CFA transaction can be used to install or uninstall the Fault Analyzer SDUMP screening feature.

Prior to installing either the XPCABND or XDUREQ exits, the CFA transaction issues a CICS NEWCOPY command for program IDIXCX52 and IDIXCX53 if both exits are in the <Uninstalled> state. Hence, to load a new copy of IDIXCX52 or IDIXCX53 (for example, after applying maintenance), use the CFA transaction to uninstall the XPCABND and XDUREQ exits, and then re-install either or both exits as required.

There are two ways to interact with the CFA transaction: either from a CICS terminal or from the MVS console. For this book, we only used CFA from a terminal as it provided more information.

21.1.3 Using a CFA from a CICS terminal

To use the CFA transaction from a CICS terminal, simply enter CFA. You are subsequently presented with a display similar to Figure 21-1.

```
                    Fault Analyzer Control Transaction

Options: I=Install U=Uninstall
                   Current Status/Error Message
_ XPCABND          Installed
_ XDUREQ           Installed
_ LE Exit          Installed
_ SDUMP Screening  Installed

         Current     HWM  Setting
Active      0005    0005     0005
Waiting     0012    0012     0020

Option: F=Force purge task
  _ Task(0000045) Elapsed Seconds(004)
  _ Task(0000047) Elapsed Seconds(004)
  _ Task(0000049) Elapsed Seconds(003)
  _ Task(0000051) Elapsed Seconds(003)
  _ Task(0000055) Elapsed Seconds(001)


IDITRACE status   OFF

PF1=Help PF3=Exit PF5=Clear FND Area PF9=IVP PF11=IDITRACE ENTER=Update
```

Figure 21-1 CFA transaction display

Initially, the display shows the current status of the CICS Fault Analyzer exit points, plus details of any active and waiting Fault Analyzer tasks. By entering an **I** or **U** (for **i**nstall or **u**ninstall) next to a specific exit point, its status can be changed accordingly. If there are active analysis tasks (as shown in Figure 21-1), then a CICS TASK FORCEPURGE can be issued for that task by entering an **F** in the input field next to the active task details.

This function is only possible if CICS transaction isolation is INACTIVE, or if ACTIVE, that the IDIXFA program is defined to have an EXECKEY of CICS. For help information about a specific CICS exit, press PF1 on the main panel with the cursor on an exit selection field.

As can also be seen in Figure 21-1 the Active and Waiting tasks have three values, *Current*, *HWM* and *Setting*. The *Setting* values correspond to the Fault Analyzer option:

```
DeferredReport(CICSFATasks(max_slots<,max_waits>))
```

This option allows you to control the maximum number of parallel execution slots and the maximum number of tasks allowed to wait for an available execution slot. The default values are 1,20 (that is, 1 execution slot and up to 20 waiting tasks). For the purposes of this book, we set the maximum execution slot value to 5 and then started 20 instances of an abending transaction. As can be seen in Figure 21-1 on page 532, at the moment of taking the screen capture, there were 5 tasks being analyzed in parallel and 12 tasks waiting to be analyzed.

21.1.4 Ensuring transaction abend analysis is not suppressed by DUMP(NO)

If the active transaction definition for an abending transaction has the DUMP(NO) option specified, then CICS does not call the XPCABND global user exit, and hence Fault Analyzer is not invoked. To check the DUMP setting for a transaction, follow one of these methods:

1. Use the CEDA transaction to view the transaction definition in question and check the DUMP(YES/NO) setting. Care should be taken when using this method, as there might be multiple definitions of the same transaction, and hence the order in which the definitions are installed by CICS is important.
2. Check the active transaction definition in a dump.
3. Use the CICS-supplied transaction, CECI, to check the DUMP setting for the active transaction. This can be done by issuing the following command:

   CECI INQUIRE TRANSACTION(*nnnn*)

   Where *nnnn* is the transaction ID in question.

   Having issued this command, the displayed DUMPING value has the following meaning:

   – A value of 00186 means DUMP(YES).
   – A value of 00187 means DUMP(NO).

21.1.5 Preventing LE from causing the CICS trace to wrap

When a CICS transaction abends and Language Environment is active in the abending enclave, Language Environment by default writes diagnostic information to a transient data queue named CESE. This occurs if the IBM-supplied Language Environment default run-time option TERMTHDACT(TRACE) is in effect. Because these diagnostics are recorded before Fault Analyzer receives control to process the abend, the CICS trace table is liable to wrap around, and application trace data might therefore be lost. Depending on your level of MVS, we recommend that the TERMTHDACT option is set to one of the following:

- TERMTDHACT(TRACE,CICSDDS,...)

  This causes Language Environment to write its diagnostics to the CICS transaction dump data set.

  Note: This option might only be available if the required maintenance has been applied to your CICS system.

- TERMTDHACT(QUIET)

  This suppresses most of the Language Environment diagnostics.

21.1.6 Specifying data sets through the IDIOPTS DDname

To avoid the necessity to recycle CICS in case compiler listing or side file data sets change, specify these via the data sets option in a user options file pointed to by the IDIOPTS DDname.

21.1.7 Installing the MVS post-dump exit IDIXTSEL

This optional Fault Analyzer post-dump exit, IDIXTSEL, is installed in the IEAVTSEL installation exit list. The exit, which is only invoked for SVC dumps, is installed by the USERMOD, IDIWTSEL. It is normally only required to register CICS system abend dumps.

To install this USERMOD, edit and submit the sample job IDIWTSEL. This includes IDIXTSEL in the IEAVTSEL installation exit list. If you have other exits defined in this list, add the IDIXTSEL exit last.

To activate this change, re-IPL or cancel the DUMPSRV address space so that it restarts with the new exit.

For dump registration via this exit to occur, it is necessary to also start the Fault Analyzer subsystem.

Starting the subsystem

To start the subsystem, a simple job, as shown below, can be submitted:

```
//IDISS JOB ...
//IDISSTST EXEC PGM=IDISAMAN,TIME=NOLIMIT

//* (An optional DD statements might follow, as described below.)
```

Alternatively, the subsystem can be established using a started task. The IDIS subsystem dynamically allocates data sets to SYSOUT=*, so it must be run under the job entry subsystem (JES).

Note: Ensure that the TIME=NOLIMIT parameter is specified as shown in the example above to prevent subsystem abend S522.

There is no requirement for any REGION size specification. The default 32-MB region is adequate. If you have more than one version of DB2 installed, or if the DB2 load module library is not in LINKLIST, then you must add DD statements for all DB2 subsystems that are not accessible via LINKLIST, and for which you want Fault Analyzer to perform analysis, as follows:

//DB2*subsystem-id* DD DISP=SHR,DSN=*data-set-name*

Where *subsystem-id* is the DB2 subsystem ID (usually four characters) and *data-set-name* is the associated load module library.

For a data sharing group, the group attach name is used as the subsystem ID.

If, for example, the DB2 subsystem with an ID of DSN1 requires the load library DSN1.LOADLIB, which is not in LINKLIST, then add the following JCL DD statement to the Fault Analyzer subsystem job:

```
//DB2DSN1 DD DISP=SHR,DSN=DSN1.LOADLIB
```

The subsystem requires access to the DB2 PLAN:DSNACLI and READ access to the following SYSIBM catalog tables:

- SYSIBM.SYSDBRM
- SYSIBM.SYSPACKAGE
- SYSIBM.SYSPACKSTMT
- SYSIBM.SYSPLAN
- SYSIBM.SYSSTMT

The subsystem name used by Fault Analyzer is IDIS.

This name does not have to be defined in the IEFSSNxx PARMLIB member, as it is dynamically defined by the IDISAMAN program.

Stopping the subsystem

The subsystem can be cancelled and restarted at any time.

A MODIFY command can also be used to stop the subsystem.

F *name*,STOP

Note: *name* is the appropriate identifier for the MODIFY command, depending on the way in which the subsystem was started. If the FA subsystem is not active or if an incorrect identifier was used on the MODIFY command, MVS issues the message:

```
IEE341I XYZ NOT ACTIVE.
```

If the FA subsystem is already active when another attempt to start it is performed, then the following message is issued to the operator console:

```
IDISAMAN The Fault Analyzer Subsystem is already active in jobname job-id
```

Where *jobname* is the job or started task name of the currently executing FA subsystem and *job-id* is the JES job or started task ID.

21.2 Storage requirements

The real-time execution following an abend requires extra storage in the abending region while the analysis is carried out on the in-storage data.

The following are the requirements for the *minimum* available region size, assuming that neither Language Environment nor Fault Analyzer is available from LPA:

- A minimum of 299 kilobytes below-the-line storage regardless of execution environment
- A minimum of 16 megabytes above-the-line storage for CICS transactions
- A minimum of 14 megabytes above-the-line storage for programs other than CICS transactions

Depending on the type of fault being analyzed and the environment on which this occurs, additional storage might be required.

If the initial amount of required storage is not available, then message IDI0055I is issued and no analysis is performed.

The storage requirements under CICS are for MVS GETMAIN-managed storage, not CICS DSA-managed storage. So, to increase below-the-line MVS GETMAIN-managed storage, you must decrease CICS below-the-line DSA-managed storage (and similarly for above-the-line storage).

Information about the actual storage used by Fault Analyzer is available at the end of the real-time analysis report. However, the amount of storage provided in the report accounts for the explicit allocations performed by Fault Analyzer only and does not include, for example, Language Environment heap and stack storage or storage used for load modules.

In post-abend situations, where the minidump or SYSMDUMP is being processed, only a marginal increase in storage requirements occur over that of the real-time execution, as the result of allocating space for referenced dump pages. The increase is typically less than 500 kilobytes.

For interactive reanalysis, the storage is required in the TSO region.

The minimum available region size above-the-line can be reduced with the size of required modules that are either available from LPA, and therefore does not have to be loaded, or already loaded, if, for example, the abending program uses LE.

Having LE in LPA saves almost 5 megabytes, and Fault Analyzer in LPA almost 7 megabytes, reducing the storage requirement for a typical non-CICS program to less than 3 megabytes.

If the necessary below-the-line size is not available, then message IDI0086E is issued and processing terminates.

If the necessary above-the-line size is not available, then message IDI0055E is issued and processing terminates. Additionally, message IDI0087I might be issued to provide information about storage that could be made available if the command included in the message text is issued to add modules to LPA. The module names likely to be included in the message are the Fault Analyzer modules IDIDA, IDIBOOKR, and IDILANGX. To place these modules in LPA, and save approximately 7 megabytes above-the-line storage, either issue the following MVS operator command:

SETPROG LPA,ADD,MOD=(IDIDA,IDIBOOKR,IDILANGX),DSN=LNKLST

Or add:

IDI.SIDILPA1 to the LPALSTxx PARMLIB member

Note: If Fault Analyzer modules are loaded into LPA, then it is important that the FA modules be placed again in the LPA, which is performed after applying Fault Analyzer maintenance. Failure to perform this step following the installation of maintenance prevents the update of Fault Analyzer LPA modules. Because all Fault Analyzer modules are not in LPA, this can cause a mismatch between the old and the new code, which might lead to undefined behavior.

The MVS IEFUSI exit can be used as a general way to provide additional region size if JCL change is not practical for all jobs. A sample IEFUSI exit is provided as member IDISUSI in the IDI.SIDISAM1 data set. The sample exit increases the region size of all jobs by 16 megabytes.

21.3 Fault Analyzer and DB2

For details on customizing Fault Analyzer for DB2, see “Customizing the DB2 Environment” in the *Fault Analyzers Users Guide*.

21.4 Fault Analyzer and IMS

For details on customizing Fault Analyzer for IMS see “Customizing the IMS Environment” in the *Fault Analyzers Users Guide*.

21.5 Recovery fault recording

The recovery fault recording feature of Fault Analyzer is provided to reduce the number of instances where an abnormal termination problem during real-time analysis prevents a normal fault entry from being created. This might, for example, be in the following situations:

- Insufficient storage (Message IDI0005S issued)
- Fault Analyzer abended (Message IDI0047S issued)
- Fault Analyzer timed out (Message IDI0092S issued)
- Invalid negative storage length request (Message IDI0105S issued)

When a terminating condition is subject to recovery fault recording processing, then a skeleton fault entry is created and an associated IEATDUMP transaction dump written.

The IEATDUMP creates an additional data set, into which MVS writes a dump of the address space. This takes significantly more DASD space than a minidump, but in these situations, Fault Analyzer has failed to gather the minidump. Subsequently, the IEATDUMP data set is used in place of the minidump for reanalysis of the skeleton fault entry.

Note: To enable recovery fault recording processing, the IDIS subsystem must be started and the default history file must be a PDSE.

The history file in which the fault entry is created is either the current history file for the abending job, as determined at the time of the abnormal analysis termination, or the default history file for the IDIS subsystem. The current history file determined for the abending job is attempted first if it is a PDSE. Otherwise, the IDIS subsystem attempts to use the default history file.

Message IDI0126I is issued to indicate in which history file the fault entry was created.

The IEATDUMP data set is created from the abending region, while the skeleton fault entry is written by the IDIS subsystem.

Once the recovery fault recording process starts no user exits are driven for the process, except for any Notification user exits specified in the options available to the IDIS subsystem. These are invoked when creating the skeleton recovery fault recording fault entry. To distinguish a recovery fault recording event from other invocations of Notification user exits, the NFY.NFYTYPE field is set to **R**

Depending on where in the real-time analysis process the problem occurred, reanalysis of the recovery fault recording fault entry is capable of producing a reanalysis report, which is effectively identical to the one that would have been produced if the real-time analysis had completed normally. The fact that a recovery fault recording fault entry was created instead of the normal real-time fault entry is almost transparent to the user for many of the recovery situations.

To illustrate this feature for this book, we put a *Do Forever* loop in a Fault Analyzer REXX Notify exit and then caused a deliberate abend. The predicted result was an IDI0092S message and consequently a recovery fault recording entry.

As can be seen in Figure 21-2, the abend analysis did not complete, and an IDI0092S message was issued.

```
  Display  Filter  View  Print  Options  Help
------------------------------------------------------------------------------
SDSF OUTPUT DISPLAY PLISAMPE JOB35263  DSID     2 LINE 2       COLUMNS 02- 81
COMMAND INPUT ===>                                           SCROLL ===> CSR

18.48.56 JOB35263 ---- WEDNESDAY, 15 NOV 2006 ----
18.48.56 JOB35263  IRR010I  USERID SIMCOCK  IS ASSIGNED TO THIS JOB.
18.48.56 JOB35263  ICH70001I SIMCOCK  LAST ACCESS AT 18:46:06 ON WEDNESDAY, NOVE
18.48.56 JOB35263  $HASP373 PLISAMPE STARTED - INIT 2    - CLASS A - SYS FAE1
18.48.56 JOB35263  IEF403I PLISAMPE - STARTED - TIME=18.48.56
18.48.58 JOB35263  +IDI0001I Fault Analyzer V7R1M0 (MVS 2006/11/02) invoked by I
18.49.02 JOB35263  +IDI0002I Module PLISAMP, program PLISAMP, offset X'D8': Aben
18.50.57 JOB35263  +IDI0092S Time  2 minutes for analysis exceeded, the subtask
18.51.06 JOB35263  IEA794I SVC DUMP HAS CAPTURED:  488
   488             DUMPID=288 REQUESTED BY JOB (PLISAMPE)
   488             DUMP TITLE=SLIP DUMP ID=F092
18.51.06 JOB35263  IGD101I SMS ALLOCATED TO DDNAME (SYS00093)  493
   493                     DSN (SIMCOCK.IDIRFR.FAE1.D061115.T105106.PLISAMPE)
   493                     STORCLAS (PRIMARY) MGMTCLAS (PRIMARY) DATACLAS (DEFAU
   493                     VOL SER NOS= E$US09
18.51.10 JOB35263  IGD104I SIMCOCK.IDIRFR.FAE1.D061115.T105106.PLISAMPE RETAINED
18.51.10 JOB35263  IEA822I COMPLETE TRANSACTION DUMP WRITTEN TO SIMCOCK.IDIRFR.F
18.51.10 JOB35263  +IDI0126I Recovery fault recording fault ID F00222 assigned i
18.51.41 JOB35263  -                                          --TIMINGS (MINS.)--
```

Figure 21-2 Joblog from abending batch job

Following the IDI0092S message, the IGD104I message gives the dump data set name, and the IDI0126I message indicates that the Recovery fault recording happened and was written as fault ID F00222.

We then removed the deliberate loop and submitted the job again. This time the analysis completed and Fault ID F00223 was written. Figure 21-3 shows the two Fault entries with the MVS_Dump_DSN showing to illustrate that a dump was written.

```
  File  Options  View  Services  Help
IBM Fault Analyzer - Fault Entry List                    Line 1 Col 1 80
Command ===> ____________________________________________ Scroll ===> CSR

Fault History File or View  : 'SIMCOCK.A7.HISTORY'

   Fault_ID MVS_Dump_DSN                                        Job_Type Jobname  Pro
__   F00223 n/a                                                 Batch    PLISAMPE PLI
__   F00222 SIMCOCK.IDIRFR.FAE1.D061115.T105106.PLISAMPE        Batch    PLISAMPE PLI
__   F00198 n/a                                                 CICS     CICS6FA1 IDI
__   F00197 n/a                                                 CICS     CICS6FA1 IDI
__   F00196 n/a                                                 CICS     CICS6FA1 IDI
__   F00193 n/a                                                 CICS     CICS6FA1 ABC
__   F00192 n/a                                                 CICS     CICS6FA1 DFH
__   F00191 n/a                                                 CICS     CICS6FA1 n/a
__   F00190 n/a                                                 CICS     CICS6FA1 n/a
__   F00189 n/a                                                 CICS     CICS6FA1 IDI
__   F00188 n/a                                                 CICS     CICS6FA1 IDI
__   F00187 n/a                                                 CICS     CICS6FA1 n/a
__   F00186 n/a                                                 CICS     CICS6FA1 n/a
__   F00174 n/a                                                 CICS     CICS6FA1 MAS
__   F00176 n/a                                                 CICS     CICS6FA1 MAS
__   F00185 n/a                                                 CICS     CICS6FA1 MAS
```

Figure 21-3 MVS_DUMP_DSN

To illustrate that the interactive reanalysis for either Fault Entry produces the same result, Figure 21-4 and Figure 21-5 are the main and event summary screens for Fault ID F00222, while Figure 21-6 and Figure 21-7 are the corresponding screens for Fault ID F00223.

```
  File  View  Services  Help
Interactive Reanalysis Report                                  Line 1 Col 1 80
Command ===>                                                  Scroll ===> CSR
JOBNAME: PLISAMPE  SYSTEM ABEND: 0C9          FAE1      2006/11/15  18:48:57

Fault Summary:
Module PLISAMP, program PLISAMP, compiler listing file
SIMCOCK.TEST.PLILIST(PLISAMP) statement # 8 : Abend S0C9 (Fixed-Point-Divide
Exception).

Select one of the following options to access further fault information:
  1. Synopsis
  2. Event Summary
  3. Open Files
  4. Storage Areas
  5. Messages
  6. Language Environment Heap Analysis
  7. Abend Job Information
  8. Fault Analyzer Options

*** Bottom of data.
```

Figure 21-4 Main interactive analysis screen of Recovery Fault Recording Entry F00222

```
  File  View  Services  Help
Event Summary                                                  Line 1 Col 1 80
Command ===>                                                  Scroll ===> CSR
JOBNAME: PLISAMPE  SYSTEM ABEND: 0C9          FAE1      2006/11/15  18:48:57

Event          Fail  Module   Program  EP
#  Type        Point Name     Name     Name     Event Location (*)     Loaded
 1 Call              CEEBLIIA n/a      n/a      M+4F4                  CEE.SC
 2 Call              CEEPLPKA n/a      n/a      M+A9AE8                LPA
 3 Call              IBMREV10 n/a      CEEEV010 E+310                  CEE.SC
 4 Call              IBMRLIB1 n/a      IBMRPMIA E+51E                  CEE.SC
 5 Abend S0C9  ***** PLISAMP  PLISAMP  PLISAMP  S#8 P+D8               SIMCOC

(*) One or more of the following abbreviations might appear in the "Event
    Location" column:

    F#n  Source file number (refer to detailed event information for file
         identification)
    L#n  Source file line number
    S#n  Listing file statement number (refer to detailed event information
         for file identification)
    M+x  Offset from start of load module
```

Figure 21-5 Event Summary of Recovery Fault Recording Entry F00222

```
  File  View  Services  Help
Interactive Reanalysis Report                                  Line 1 Col 1 80
Command ===>                                                  Scroll ===> CSR
JOBNAME: PLISAMPE  SYSTEM ABEND: 0C9            FAE1      2006/11/15  20:00:23

Fault Summary:
Module PLISAMP, program PLISAMP, compiler listing file
SIMCOCK.TEST.PLILIST(PLISAMP) statement # 8 : Abend S0C9 (Fixed-Point-Divide
Exception).

Select one of the following options to access further fault information:
  1. Synopsis
  2. Event Summary
  3. Open Files
  4. Storage Areas
  5. Messages
  6. Language Environment Heap Analysis
  7. Abend Job Information
  8. Fault Analyzer Options

*** Bottom of data.
```

Figure 21-6 Main interactive analysis screen of completed Fault Entry F00223

```
  File  View  Services  Help
Event Summary                                                  Line 1 Col 1 80
Command ===>                                                  Scroll ===> CSR
JOBNAME: PLISAMPE  SYSTEM ABEND: 0C9            FAE1      2006/11/15  20:00:23

Event           Fail  Module   Program  EP
#  Type         Point Name     Name     Name     Event Location (*)     Loaded
 1 Call               CEEBLIIA n/a      n/a      M+4F4                  CEE.SC
 2 Call               CEEPLPKA n/a      CEEBBEXT E+1B6                  LPA
 3 Call               IBMREV10 n/a      CEEEV010 E+310                  CEE.SC
 4 Call               IBMRLIB1 n/a      IBMRPMIA E+51E                  CEE.SC
 5 Abend S0C9  ***** PLISAMP  PLISAMP  PLISAMP  S#8 P+D8                SIMCOC


(*) One or more of the following abbreviations might appear in the "Event
    Location" column:

    F#n  Source file number (refer to detailed event information for file
         identification)
    L#n  Source file line number
    S#n  Listing file statement number (refer to detailed event information
         for file identification)
    M+x  Offset from start of load module
```

Figure 21-7 Event Summary of completed Fault Entry F00223

Part 6

File Export for z/OS

In this part of the book, we explain how to use File Export for z/OS.

22

Using File Export for z/OS

In this chapter we provide detailed information on customizing File Export to assist you in creating extracts. We also discuss specific scenarios on how to use the product to help a new user understand the concepts and flow of File Export.

We discuss the following topics:

- Customizing Option Blocks
- Creating a DB2 extract utilizing only DB2 RI as defined in the DB2 catalog
- Creating a DB2 RI Extract with Application Relationships
- Creating an IMS Extract Using Logical Relationships
- Creating an IMS Extract Using Application Relationships

22.1 Customizing Option Blocks

To simplify the use of File Export, Option Blocks have been established to contain site-specific default values for parameters required to complete an extract or load request. Option Blocks are load modules that are assembled and linked into the product load library. When an extract or load is executed, Option Blocks are used to provide default values to necessary parameters used by the ABXUTIL program.

When an Option Block is in use, site-specific default values are used in processing extracts and loads, relieving the user from having to provide them. However, processing options can be overridden by the user on the ISPF panels.

Customizing the product Option Blocks should take place after installation has been completed. However, anyone with UPDATE authority on the product data sets can create or change an Options Block to customize the product.

There are three types of Option Blocks used within File Export. The Global Option Block, DB2 Option Block, and IMS Option Block. The contents and the specific instructions on editing, assembling and linking the three separate Option Block modules can be found in the IBM File Export for z/OS Reference Manual in the chapter titled Customization Tasks.

22.1.1 Global Option Block

The Global Option Block contains parameters necessary for processing all extract and load requests. Parameters such as SYSOUT CLASS, WORK UNIT, WORK VOLUME and job card information can be specified within the Global Option Block. The Global Option Block is assembled using ABX.SABXSAMP member ABXOGLB0.

Example 22-1 shows the Global Options that have been defined for the scenarios explained in this chapter.

Example 22-1 The following parameters were used in the Global Options Block

```
****************************** Top of Data ******************************
ABXOGBL0 TITLE 'File Export for z/OS : Global Processing Options'
*********************************************************************
-  -  -  -  -  -  -  -  -  -  -  -  -  -  -  - 18 Line(s) not Displayed
*
         PRINT GEN
         ABXMGBL0 AUDIT=NO,                                           +
               FILL=BLANKS,                                           +
               OTDDN=ABXOUT,                                          +
               SESSLIB=%SYSUID.ABX.SESSLIB,                           +
               JCLLIB=%SYSUID.ABX.JCLLIB,                             +
               SMFNO=255
*
         END   ABXOGBL0
***************************** Bottom of Data ****************************
```

AUDIT=NO Indicates that SMF audit records are not to be written to the system SMF datasets.

FILL=BLANKS The character to be used as the pad or fill character when format changes to an output record require the inclusion of filler or slack bytes, either to provide correct alignment or to complete a request to lengthen the record. This parameter is valid for all processing types. The default value of BLANKS cause (x'40') to be used as the fill character.

OTDDN=ABXOUT Indicates the ddname to be used to define the output dataset when using processing tpes VSAM, SEQUENTIAL or IMS. The default value is ABXOUT. Any one to eight character string can be used. The string must follow MVS ddname naming rules and must not conflict with other specified ddnames. This value cannot be overridden in the user interface.

SESSLIB=%SYSUID.ABX.SESSLIB

Specifies the name of the default dataset to be used to hold all selected options that pertain to Extract and Load Requests. The symbolic %SYSUID results in the user's TSO ID being used within the dataset name. This dataset name can be overridden using the Settings panels in the user interface.

JCLLIB=%SYSUID.ABX.JCLLIB

Specifies the name of the default dataset to be used to hold generated Extract and Load Request JCL. The symbolic %SYSUID results in the user's TSO ID being used within the dataset name. This dataset name can be overridden using the Settings panels in the user interface.

SMFNO=255 There is no default value for this parameter. It must be included when the AUDIT parameter has a value of Yes or FORCE. THis parameter indicates the SMF record number that should be used by the Extract Engine when writing records to the system FSMF datasets. The value should be determined after consulting with the site system programmer to avoid number conflicts with other program products.

For a complete list of those parameters that can be specified within the Global Option Block, refer to the IBM File Export for z/OS Reference Manual in the appropriate Appendix.

22.1.2 DB2 Option Block

The DB2 Option Block contains parameters specific to processing DB2 data, as shown in Example 22-2. Parameters include the DB2 plan name(s) for the File Export user interface and the ABXUTIL program. Parameters also include the DB2 DSN load library name and DSNUPROC utility parameters specific for each SSID on which File Export is to be used. The DB2 Option Block is assembled using ABX.SABXSAMP member ABXODB20.

Example 22-2 The following parameters were used in the DB2 Options Block

```
****************************** Top of Data ******************************
ABXODB2O TITLE 'FILE EXPORT FOR Z/OS : DB2 PROCESSING OPTIONS'
*********************************************************************
-  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  - 25 Line(s) not Displayed
************************************************************
**  Indicate/initialize the start of entry processing      **
************************************************************
         ABXMDB2O TYPE=START
```

```
*
         ABXMDB20 TYPE=ENTRY,                                          X
               SSID=R71A,                                              X
               LOADLIB=DSN.V710.SDSNLOAD,                              X
               UIPLAN=ABXWRK12,                                        X
               BATCHPLAN=ABXWRK12,                                     X
               COMMENT='R71A using WRK0120 Libraries'
*
************************************************************
**  Indicate that all TYPE=ENTRY instances have been coded **
************************************************************
         ABXMDB20 TYPE=FINAL
         END   ABXODB20
***************************** Bottom of Data ***************************
```

SSID=R71A The DB2 subsystem ID that are used for the Option Entry.

LOADLIB=DSN.V710.SDSNLOAD

Specifies the dataset name of the DB2 subsystem library. The LOADLIB parameter is used when File Export is looking for a DB2 program.

UIPLAN=ABXWRK12

Specifies the DB2 plan name for the File Export user interface to use. The default plan name that is shipped within bind job ABXBIND is ABXPLAN. The default plan name that is shipped within bind job ABXBIND3 is ABXDBUT. The plan names are only suggestions and can be changed.

BATCHPLAN=ABXWRK12

Specifies the DB2 plan name for the Extract Engine to use when processing a batch job. The default plan name that is shipped within bind job ABXBIND is ABXPLAN. The default plan name that is shipped within bind job ABXBIND3 is ABXDBUT. The plan names are only suggestions and can be changed.

For a complete list of those parameters that can be specified within the DB2 Option Block, please refer to the IBM File Export for z/OS Reference Manual in the appropriate Appendix.

22.1.3 IMS Option Block

The IMS Option Block contains parameters specific to processing IMS data, as shown in Example 22-3. Parameters include the IMS SSID, the corresponding MDALIB, DBDLIB, RESLIB and IMS region processing options. The IMS Option Block is assembled using ABX.SABXSAMP member ABXOIMS0.

Example 22-3 The following parameters were used in the IMS Options Block

```
***************************** Top of Data ******************************
ABXOIMS0 TITLE 'File Export for z/OS : IMS Processing Options'
*********************************************************************
*
* Option block for:  RSTEST.ABX.ABXOIMS0.STEPLIB
*
*********************************************************************
*
```

```
************************************************************
**  Indicate/initialize the start of entry processing     **
************************************************************
         ABXMIMSO TYPE=START
*
         ABXMIMSO TYPE=ENTRY,IMSID=I8QA,                              X
               COMMENT='ENTRY FOR IMS I8QA',                          X
               AGN=IVP,                                               X
               DBRC=YES,                                              X
               IRLM=NO,                                               X
               IRLMNAME=IRLM,                                         X
               DLTDUPS=YES,                                           X
               DYNPSB=YES,                                            X
               DPSBPFX=ABXPSB,                                        X
               DPSBSFX=09,                                            X
               NBA=010,                                               X
               OBA=010,                                               X
               PROCOPT=G,                                             X
               DPSBMAC=IMS.V810.SDFSMAC,                              X
               DBDLIB=RSTEST.ABX.IMS.I8QA.DBDLIB,                     X
               DOPTLIB=RSTEST.ABX.IMS.I8QA.DOPTLIB,                   X
               RESLIB1=IMS.U8A1.SDFSRESL,                             X
               RESLIB2=RSTEST.ABX.IMS.I8QA.MDALIB,                    X
               DFSVSAMP=RSTEST.ABX.IMS.I8QA.PROCLIB(DFSVSAMP)
*
*
************************************************************
**  Indicate that all TYPE=ENTRY instances have been coded **
************************************************************
         ABXMIMSO TYPE=FINAL
         END   ABXOIMSO
***************************** Bottom of Data ***************************
```

IMSID=I8QA Specifies the IMS subsystem name the entry relates to.

COMMENT='ENTRY FOR IMS I8QA'

A comment that can be used to describe the subsystem and is displayed when the IMS Option Block contents are displayed by the File Export user interface.

AGN=IVP Specifies the Application Group Name used for interregion communication security.

DBRC=YES Specifies to use DBRC when processing in DLI or DBB mode.

IRLM=NO Specifies to use IRLM when processing in DLI or DBB mode.

IRLMNAME=IRLM Specifies the name of the IRLM subsystem that should be used if IRLM is active for this subsystem.

DLTDUPS=YES Specifies whether the Extract Engine should automatically delete any duplicate segments that are found in an IMS database when writing to that database. This parameter is used when processing in DLI, DBB or BMP mode.

DYNPSB=YES Specifies if the Dynamic PSB feature should be used for this subsystem when processing in DLI, DBB or BMP mode.

DPSBPFX=ABXPSB Specifies the prefix to be used for the names of the PSBs that are generated by the Dynamic PSB feature. The 6-character prefix along with a 2-character suffix (supplied by parameter DPSBSFX below) comprise the PSB's name, and is used when processing in DLI, DBB or BMP mode.

DPSBSFX=09 Specified the suffix to be used for the names of the PSBs that are generated by the Dynamic PSB feature when processing in DLI, DBB or BMP mode.

NBA=010 The number of IMS buffers that should be used when processing Fast Path databases when processing in BMP mode.

OBA=010 The number of IMS buffers that should be used as overflow when processing Fast Path databases if the NBA value is insufficient.

PROCOPT=G Specifies the IMS processing options that should be used to read databases when using a dynamically-built PSB in DLI, DBB and BMP region processing modes.

DPSBMAC=IMS.V810.SDFSMAC

Specifies the partitioned dataset that contains the IMS macros used to generate PSBs when using the Dynamic PSB feature using DLI, DBB or BMP modes.

Defining the Required Repository Objects

The name of the partitioned dataset that contains the DBDLIB members associated with the specified IMS subsystem when processing in DLI,DBB or BMP regions using the Dynamic PSB feature. This is the first dataset name to be searched when File Export is looking for an IMS DBD schema definition.

DOPTLIB=RSTEST.ABX.IMS.I8QA.DOPTLIB

Specifies the dataset name of the partitioned dataset that contains the ACBLIB members that are created by File Export when the Dynamic PSB feature is used while processing in BMP mode.

RESLIB1=IMS.U8A1.SDFSRESL

The name of the IMS SVC dataset that contains the IMS nucleus and required service modules when processing in DLI, DBB or BMP modes. When used in a concatenation, this dataset appears first.

RESLIB2=RSTEST.ABX.IMS.I8QA.MDALIB

The name of the IMS dataset that contains any IMS dynamic allocation members, or DFSMDA members, when processing in DLI, DBB, or BMP mode. When used in a concatenation, this dataset appears second.

DFSVSAMP=RSTEST.ABX.IMS.I8QA.PROCLIB(DFSVSAMP)

The name of the dataset that contains the buffer and subpool information for databases that are processed using DLI and DBB modes. The length of the parameter permits both a dataset name and a member name, including the parenthesis.

For a complete list of those parameters that can be specified within the IMS Option Block, please refer to the IBM File Export for z/OS Reference Manual in the appropriate Appendix.

22.2 Creating a DB2 extract utilizing only DB2 RI as defined in the DB2 catalog

Use the following steps to see how easy it is to create a related-data extract using DB2 RI only. The following extract utilizes the DB2 Options Block described in the beginning of this chapter. Therefore, setting DB2 Options are not discussed.

22.2.1 Defining the required repository objects

Task 1: Create a Related Subset in a Repository. The Related Subset contains a set of objects that are related to each other and have related data extracted from each.

Step 1: Select Option **1** Metadata Management from the Primary Option menu and press Enter.

Step 2: From the Metadata Management panel, select option **4** Manage Related Subsets as shown in Figure 22-1 and press Enter. Remember that a Related Subset is simply a group of related objects that you want to extract related data from.

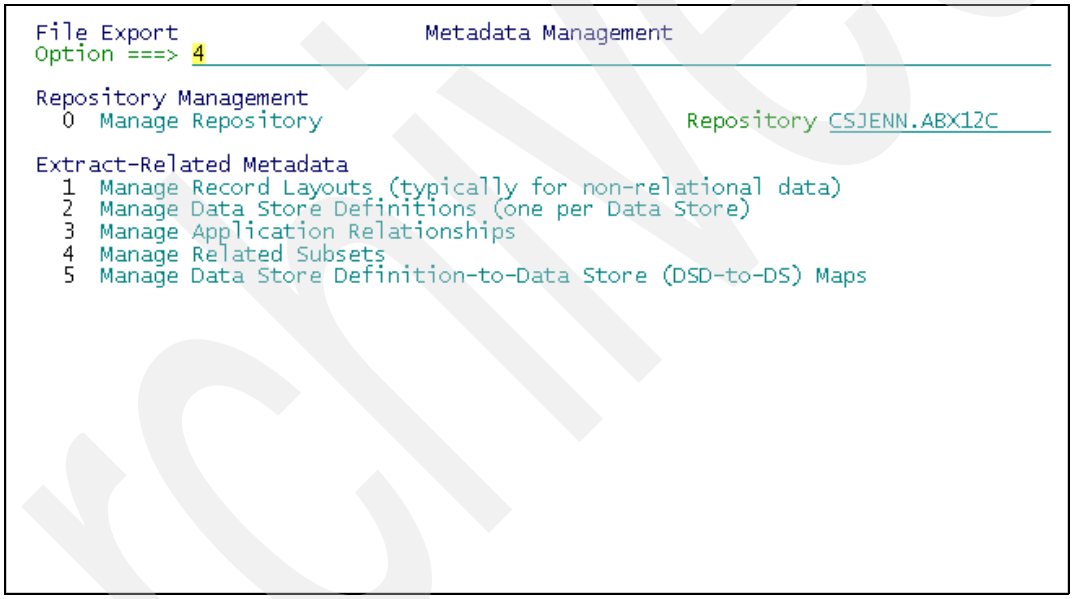


Figure 22-1 Step 2: Select Manage Related Subset to create a new one

Step 3: Provide a name for the new Related Subset, and type **C** on the Command Line to indicate you want to create the Related Subset. Press Enter to continue as shown in Figure 22-2.

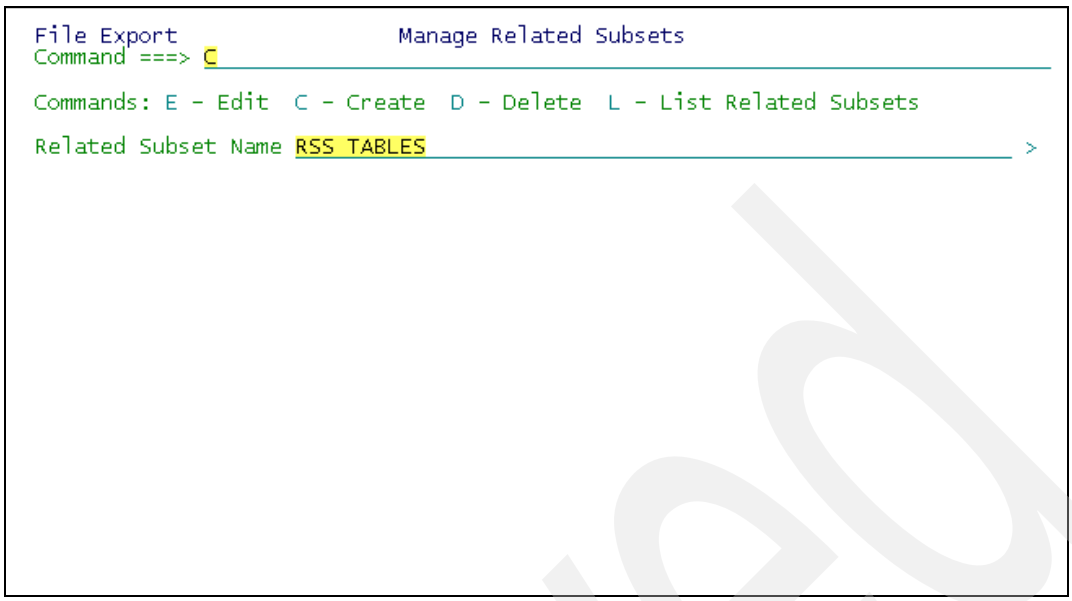


Figure 22-2 Step 3: Provide a name to call the new Related Subset

Step 4: Specify the table to begin the extract with.

This is the driving table from which all related data are extracted and is referenced to as the start DSD. Also specify if you want to include both Parents and Children of the starting object, and optionally how many levels of relationships to traverse as shown in Figure 22-3. Press Enter.

Note: If the Maximum Relationship Levels field is left blank, all related objects are retrieved.

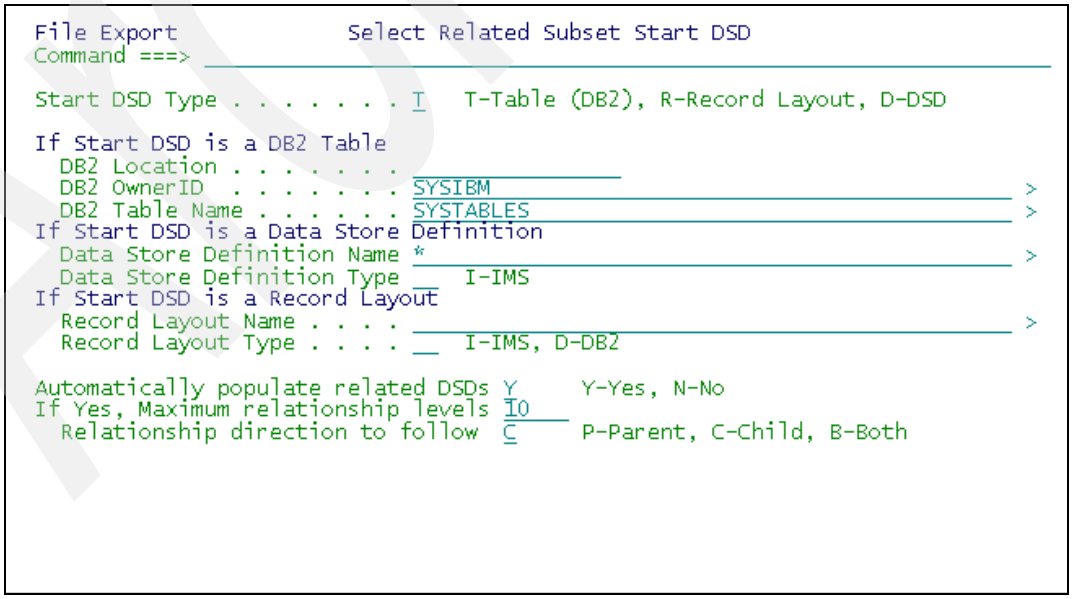


Figure 22-3 Step 4: Specify the object to start the extract from

When Enter is pressed from the Select Related Subset Start DSD panel, the resultant Related Subset panel is displayed, as shown in Figure 22-4.

```
File Export          Create Related Subset Definition
Command ===> _____________________________________________ Scroll ===> CSR

Commands:      A - Add Data Store Definitions  R - Relationship Usage
Line Commands: G - Get Related Definitions  D - Delete  C - Selection Criteria
               K - Key File  S - Scramble

Related Subset Name . : RSS TABLES
DSD-to-DS Map Name  . . _____________________________________________ >
                                                          Row 1 of 31
  ---- Data Store Definition ----                                Every
        Table / RL / DSD          Type Status  Start Ref Skip    Limit   Nth
_ SYSIBM.SYSTABLES                DB2          Y     N   _______ _______ _____
_ SYSIBM.SYSTABAUTH               DB2          N     N   _______ _______ _____
_ SYSIBM.SYSRELS                  DB2          N     N   _______ _______ _____
_ SYSIBM.SYSCOLUMNS               DB2          N     N   _______ _______ _____
_ SYSIBM.SYSTRIGGERS              DB2          N     N   _______ _______ _____
_ SYSIBM.SYSCONSTDEP              DB2          N     N   _______ _______ _____
_ SYSIBM.SYSTABCONST              DB2          N     N   _______ _______ _____
_ SYSIBM.SYSSYNONYMS              DB2          N     N   _______ _______ _____
_ SYSIBM.SYSINDEXES               DB2          N     N   _______ _______ _____
```

Figure 22-4 The objects related to SYSIBM.SYSTABES are displayed in the Related Subset

Although there are many aspects of the Related Subset that can be specified (for example, adding selection criteria, or specifying which columns to scramble), we keep this extract simple and stop at this point.

Once a Related Subset has been created, the End command (typically PF3) can be used to save the Related Subset and back out to the Metadata Management panel.

Task 2: Create a Data Store Definition-to-Data Store Map (DSD-to-DS Map). Remember that a DSD-to-DS Map, or DDM, is a map for File Export to know where to find where the actual data resides for each logical entity (DSD) in the Related Subset. In short, it maps the logical data definition to the physical data.

Step 1: From panel Metadata Management, select option **5** Manage Data Store Definition-to-Data Store (DSD-to-DS) Maps, and press Enter as shown in Figure 22-5.

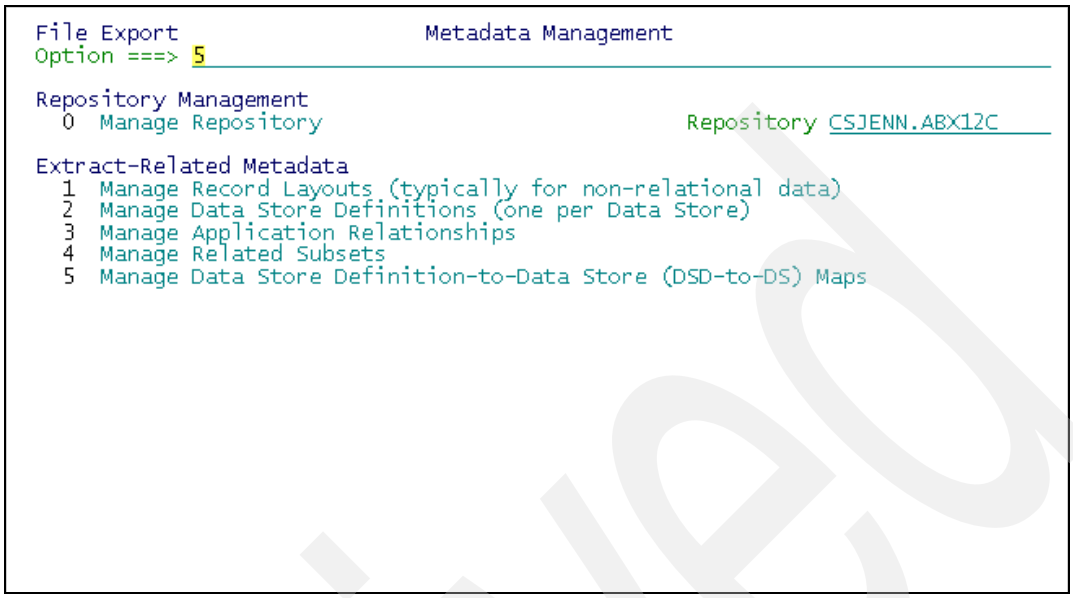


Figure 22-5 Create a DDM for the data contained in the Related Subset

Step 2: On the Manage DSD-to-DS Map panel, provide the name for the new DDM. In addition, tell File Export which Related Subset to base the map on, as shown in Figure 22-6. Press Enter.

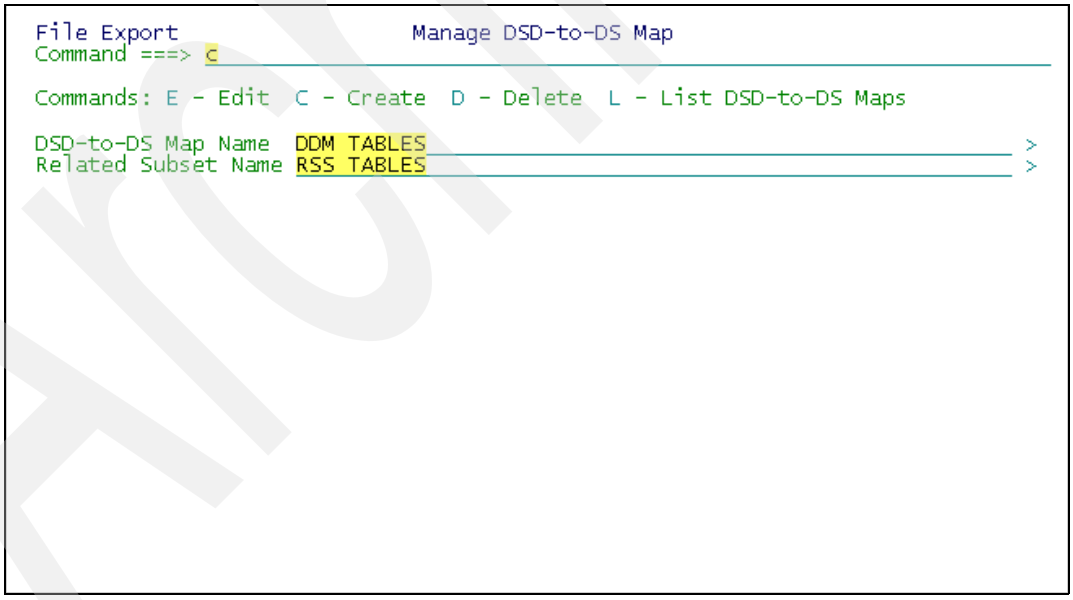


Figure 22-6 Step 2: Provide a name for the DDM and the Related Subset of objects to include

In Figure 22-7, the tables in the left-hand column are the logical names that describe the structure of the data to be extracted, i.e., for DB2, table DDL is the data definition. The tables in the right-hand column are the physical tables where the actual data resides.

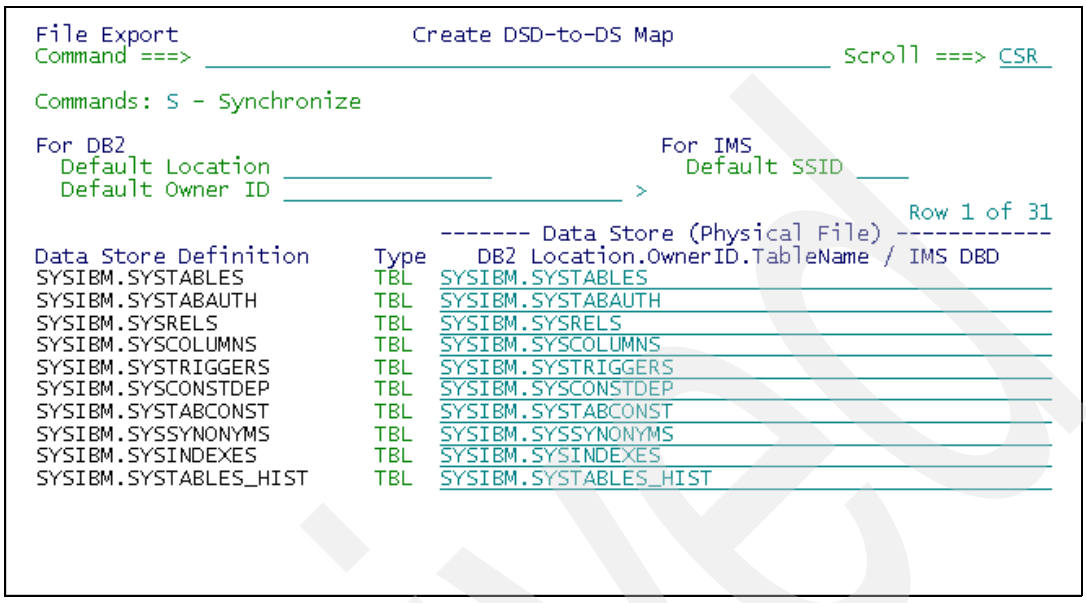


Figure 22-7 The structural definition of the data mapped to the physical location of the data

Related Subsets can be used in Data Store Definitions-to-Data Store Maps if the structural definition of a data source is the same as the objects defined in the Related Subset. That is, they describe a specific catalog, but the definitions could be used to extract from any catalog.

For example, if user Jane Doe had a copy of the catalog tables qualified by her user ID, and the Related Subset, RSS Tables, already exists containing the catalog tables qualified by SYSIBM, then Jane Doe could use the existing Related Subset. This is because Jane's tables are defined the same as the DB2 tables listed in Related Subset, RSS Tables. See Figure 22-8 and Figure 22-9 for further explanation.

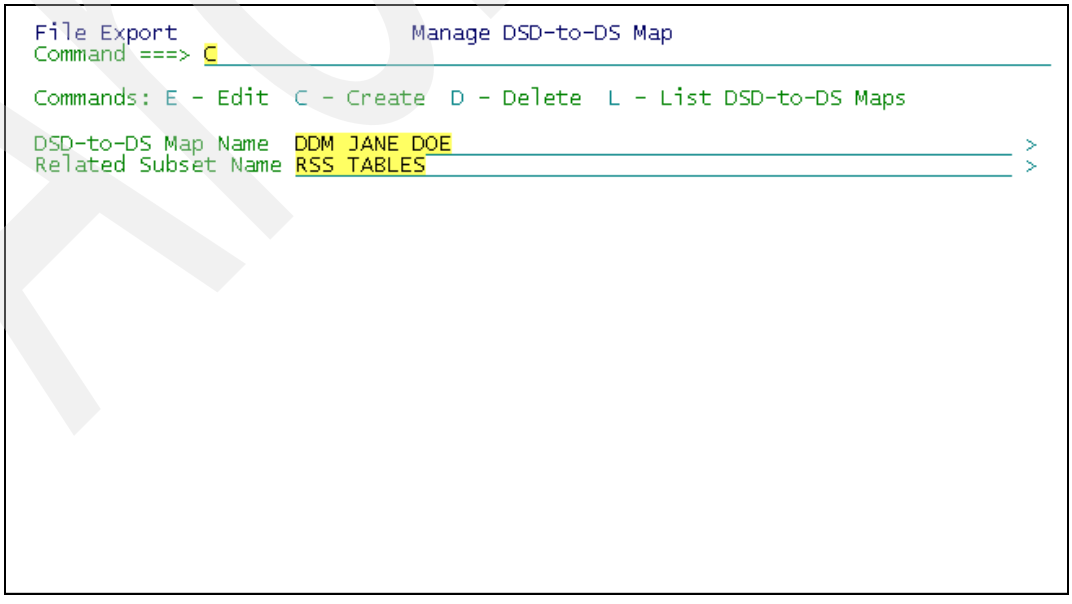


Figure 22-8 Related Subsets of data can be used to create any DDM

Figure 22-9 illustrates how a user can use a Related Subset of objects in defining a DDM. The Table names on the left-hand side are just a structural definition of the tables so that File Export knows how the source object is defined. If Jane Doe's tables are defined the same way, then all Jane has to do is change the Default Table Owner to her ID to tell File Export where to find the actual data for Jane's extract request.

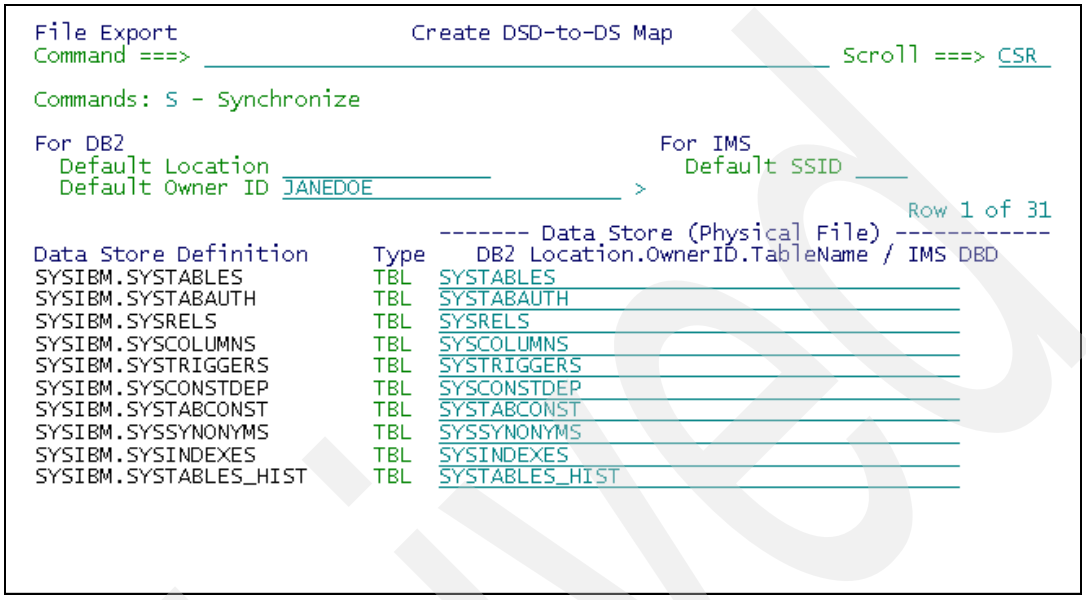


Figure 22-9 The table names on the left-hand side are just a structural definition of the tables, and the right-hand side is the physical location of the data

22.2.2 Creating an extract request

Create an Extract Request using the RSS and DDM we have just created.

Step 1: From the File Export Primary Option Menu, Select option **3** Multiple Data Store Extract and Load as shown in Figure 22-10. Press Enter.

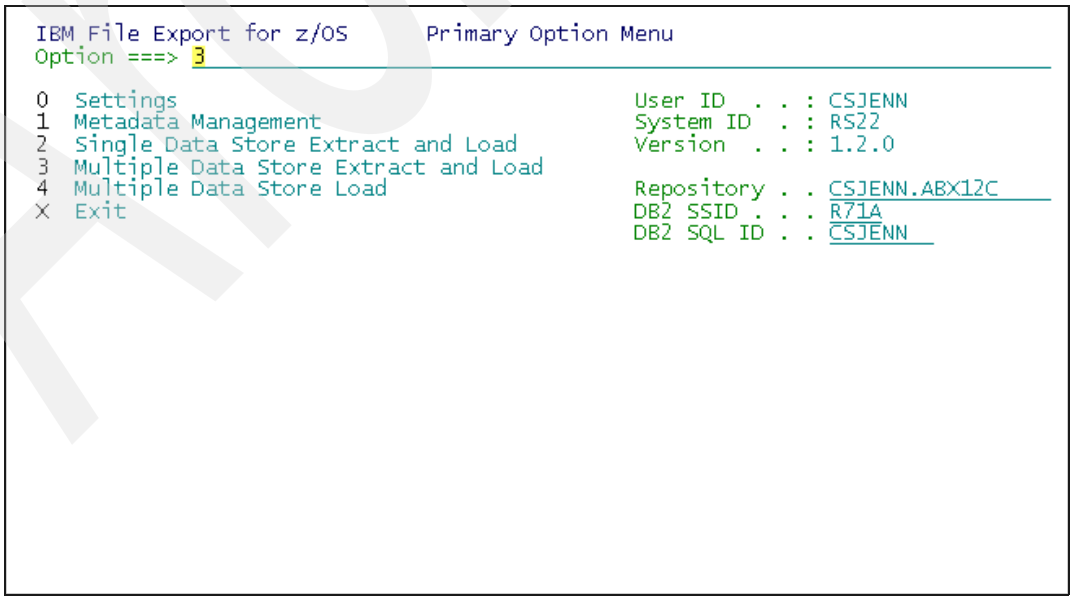


Figure 22-10 Step 1: Select option 3 Multiple Data Store Extract and Load

Step 2: On the Manage Multi-Data Store Extract Request panel, provide a name for the Extract Request Name and an optional description. Specify the Related Subset and DSD-to-DS Map name that contains the objects and map you want to create the Extract Request for. The Extract File Name is the dataset name to which a low level qualifier is appended in the Extract Request JCL as shown in Figure 22-11.

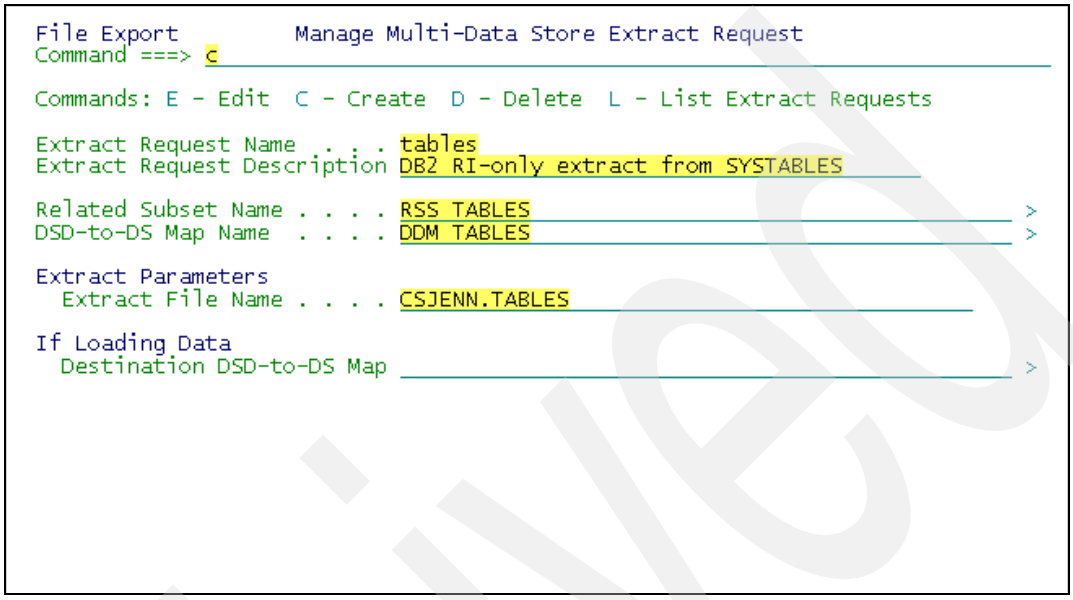


Figure 22-11 The Related Subset and DDM can be shared among users

Step 3: On the panel Multi-Data Store Extract shown in Figure 22-12, provide a JCL data set name to generate the Extract JCL into. File Export creates the JCL dataset if it does not exist. You can overwrite this value. Additionally, provide a member name to generate the JCL into if one is not provided. Select No for the option Generate Load Step, as that is covered in a later example. Press Enter to generate the extract JCL.

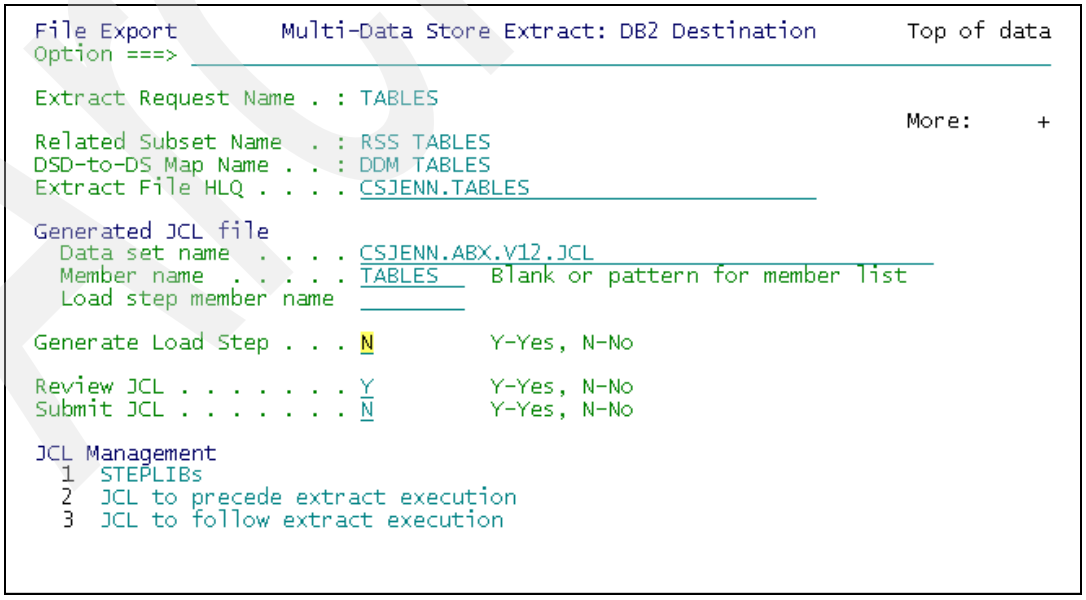


Figure 22-12 Step 2: Provide a JCL dataset and member name to generate the JCL into

The generated JCL is displayed in Example 22-4.

Example 22-4 The resulting JCL for the extract beginning with SYSIBM.SYSTABLES

```
****** ***************************** Top of Data ******************************
000001 //CSJENNE JOB CSJENN,CLASS=A,NOTIFY=&SYSUID,
000002 //       MSGCLASS=X,MSGLEVEL=(1,1)
000003 //*
000004 //*
000005 //STEP1    EXEC PGM=ABXUTIL
000006 //STEPLIB  DD   DISP=SHR,DSN=RSQA.ABX120.IBMTAPE.SABXLOAD
000007 //         DD   DISP=SHR,DSN=RSRTE.XML.HXML180.SIXMLOD1
000008 //         DD   DISP=SHR,DSN=CSWILK.ABX.GLOBAL.INSTLBLK.V2
000009 //         DD   DISP=SHR,DSN=DSN.V710.SDSNLOAD
000010 //*
000011 //*
000012 //ABXCTL   DD   *
000013 OPTIONS(DB2, PLAN(JNABX120));
000014 SOURCEDB2(R71A);
000015 TARGETDB2(R71A);
000016 DB2RESULTSET(RESULTSET1) SOURCE(SYSIBM.SYSTABLES)
000017 TARGET(SYSIBM.SYSTABLES) DD:TABL0001;
000018 SELECT * FROM SYSIBM.SYSTABLES;
000019 DB2RESULTSET(RESULTSET2) SOURCE(SYSIBM.SYSTABAUTH)
000020 TARGET(SYSIBM.SYSTABAUTH) DD:TABL0002 CHILD OF RESULTSET1 WHERE
000021 RESULTSET2.TCREATOR, TTNAME = RESULTSET1.CREATOR, NAME;
000022 SELECT * FROM SYSIBM.SYSTABAUTH;
000023 DB2RESULTSET(RESULTSET3) SOURCE(SYSIBM.SYSRELS)
000024 TARGET(SYSIBM.SYSRELS) DD:TABL0003 CHILD OF RESULTSET1 WHERE
000025 RESULTSET3.CREATOR, TBNAME = RESULTSET1.CREATOR, NAME OR
000026 CHILD OF RESULTSET1 WHERE RESULTSET3.REFTBCREATOR, REFTBNAME =
000027 RESULTSET1.CREATOR, NAME;
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -   116 Line(s) not Displayed
000144 //*
000145 //ABXLOADC DD   DSN=CSJENN.TABLES.LOADCNTL,
000146 //            DISP=(NEW,CATLG,DELETE),
000147 //            UNIT=SYSDA,
000148 //            SPACE=(TRK,(1,1))
000149 //ABXPRINT DD  SYSOUT=X
000150 //SYSABEND DD  SYSOUT=X
000151 //TABL0001 DD  DSN=CSJENN.TABLES.TABL0001,
000152 //            DISP=(NEW,CATLG,DELETE),
000153 //            UNIT=SYSDA,
000154 //            SPACE=(CYL,(10,10),RLSE)
000155 //TABL0002 DD  DSN=CSJENN.TABLES.TABL0002,
000156 //            DISP=(NEW,CATLG,DELETE),
000157 //            UNIT=SYSDA,
000158 //            SPACE=(CYL,(10,10),RLSE)
000159 //TABL0003 DD  DSN=CSJENN.TABLES.TABL0003,
000160 //            DISP=(NEW,CATLG,DELETE),
000161 //            UNIT=SYSDA,
000162 //            SPACE=(CYL,(10,10),RLSE)
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -   112 Line(s) not Displayed
000275 /*
****** **************************** Bottom of Data ****************************
```

For the sake of page space, only the first three tables and their corresponding intermediate datasets are displayed.

In the example above, you see table SYSTABLES listed first. It is the driving object, or the object from which the extract begins. Its keys are held in RESULTSET1, which is used when the extract from the second table, SYSTABAUTH, is processed to find related data.

The DD ABXLOADC is where the load control information, also known as SYSPUNCH, is held. The three TABL000n DDs contain the extracted DB2 data before it is loaded into the target tables using the DB2 LOAD utility.

22.3 Creating a DB2 extract with application relationships

In this scenario, the DB2 lab that is shipped with File Export, along with the training data, is used to create a more elaborate DB2 extract. You can locate the DB2 lab and follow along by viewing ABX.SABXSAMP(ABXLABS).

This extract is utilizes the DB2 Option Block that is displayed in the chapter titled Introduction to File Export. Therefore, setting DB2 options is skipped.

22.3.1 Defining the required repository objects

In this scenario, there are three tables; ORG, STAFF and IT_ASSETS. Both ORG and STAFF are related through referential integrity defined in the DB2 catalog. Table IT_ASSETS is not related to any table. An application relationship is created between STAFF and IT_ASSETS to demonstrate how all three tables can be included in an extract.

Task 1: Create Application Relationships. Application relationships that are used in this lab must first be defined to File Export before related data can be extracted from two objects.

Step 1: Select option **3** Manage Application Relationships from the Metadata Management panel as shown in Figure 22-13. Press Enter.

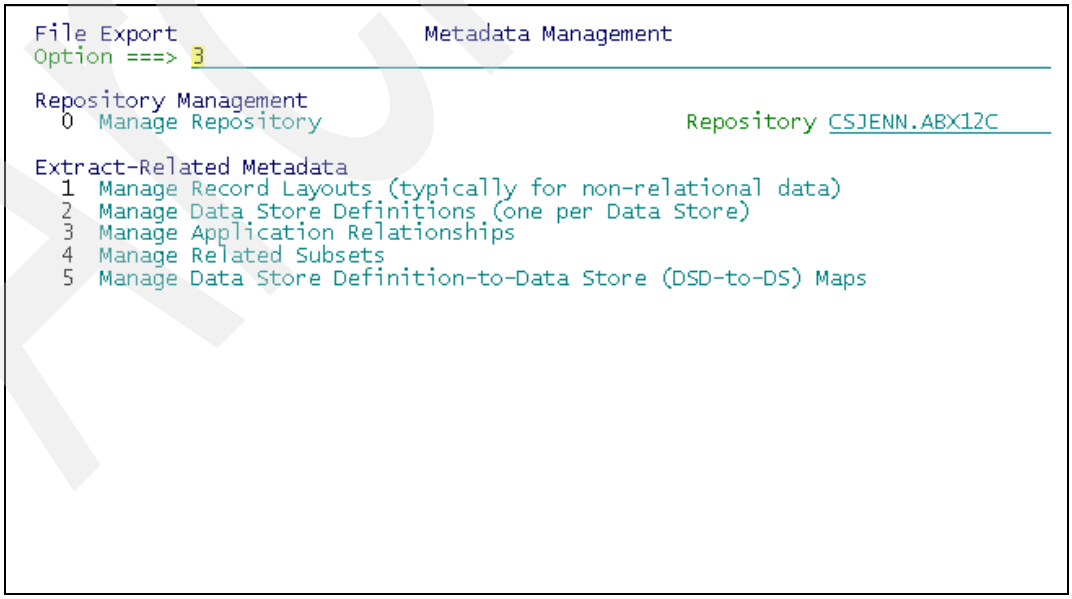


Figure 22-13 Use option 3 Manage Application Relationships to create a relationship between two unrelated objects

Step 2: Provide the new Application Relationship with a name and specify the parent object. In this scenario, the table STAFF is the parent to table IT_ASSETS as shown in Figure 22-14.

```
File Export            Manage Application Relationships
Command ===> _____________________________________________________________
                                                                More:     +
Commands: E - Edit  C - Create  D - Delete  L - List Application Relationships

Relationship Name STAFF2ITASSETS                                          >

Parent Data Store Definition  (mutually exclusive, specify only one)
  If using DB2 Table
    DB2 Location . . . . . . . . . _______________
    DB2 OwnerID  . . . . . . . . . CSJENN                                 >
    DB2 Table Name . . . . . . . . STAFF                                  >
  If using Record Layout
    Record Layout Name . . . . . . _____________________________________  >
    Record Layout Type . . . . . . _  I-IMS, D-DB2
  If using Data Store Definition
    Data Store Definition Name . . _____________________________________  >
    Data Store Definition Type . : I  I-IMS
    Record Layout Name . . . . . . _____________________________________  >
    If using IMS Data Store Definition
      IMS Segment  . . . . . . . . ________

Child Data Store Definition   (mutually exclusive, specify only one)
  If using DB2 Table
```

Figure 22-14 Provide the name of the parent object first

Step 3: Scroll down the panel to see the same fields for the child object as shown in Figure 22-15. In this scenario, the table IT_ASSETS is the child to table STAFF. Type a **C** on the Command Line to create the new Application Relationship.

```
File Export            Manage Application Relationships
Command ===> C____________________________________________________________
                                                                More:     -
    Data Store Definition Name . . _____________________________________  >
    Data Store Definition Type . : I  I-IMS
    Record Layout Name . . . . . . _____________________________________  >
    If using IMS Data Store Definition
      IMS Segment  . . . . . . . . ________

Child Data Store Definition   (mutually exclusive, specify only one)
  If using DB2 Table
    DB2 Location . . . . . . . . . _______________
    DB2 OwnerID  . . . . . . . . . CSJENN                                 >
    DB2 Table Name . . . . . . . . IT_ASSETS                              >
  If using Record Layout
    Record Layout Name . . . . . . _____________________________________  >
    Record Layout Type . . . . . . _  I-IMS, D-DB2
  If using Data Store Definition
    Data Store Definition Name . . _____________________________________  >
    Data Store Definition Type . : I  I-IMS
    Record Layout Name . . . . . . _____________________________________  >
    If using IMS Data Store Definition
      IMS Segment  . . . . . . . . ________
```

Figure 22-15 The child object is specified second

Step 4: Once the objects to participate in the application relationship have been specified, the fields on which to base the relationship are specified.

The option Switch Mode allows you to toggle between a view all fields in both objects with those comprising the relationship numbered in column format, as shown in Figure 22-16, or a view that only shows fields participating in the relationship.

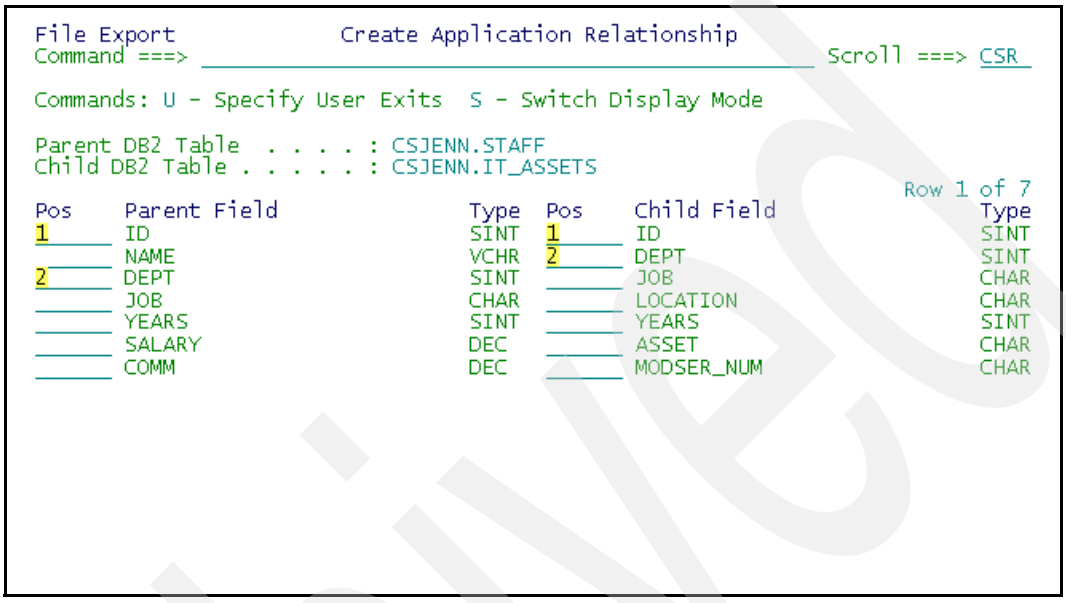


Figure 22-16 Switch Mode displays both objects' fields in columnar format

Task 2: Create a Related Subset of objects to extract related data from.

A Related Subset is a set of objects that are related through DB2 RI or application relationships. A test environment can be created using a Related Subset of data encompassing related objects from a production environment.

Step 1: Select option **4** Manage Related Subsets from the Metadata Management panel as shown in Figure 22-17.

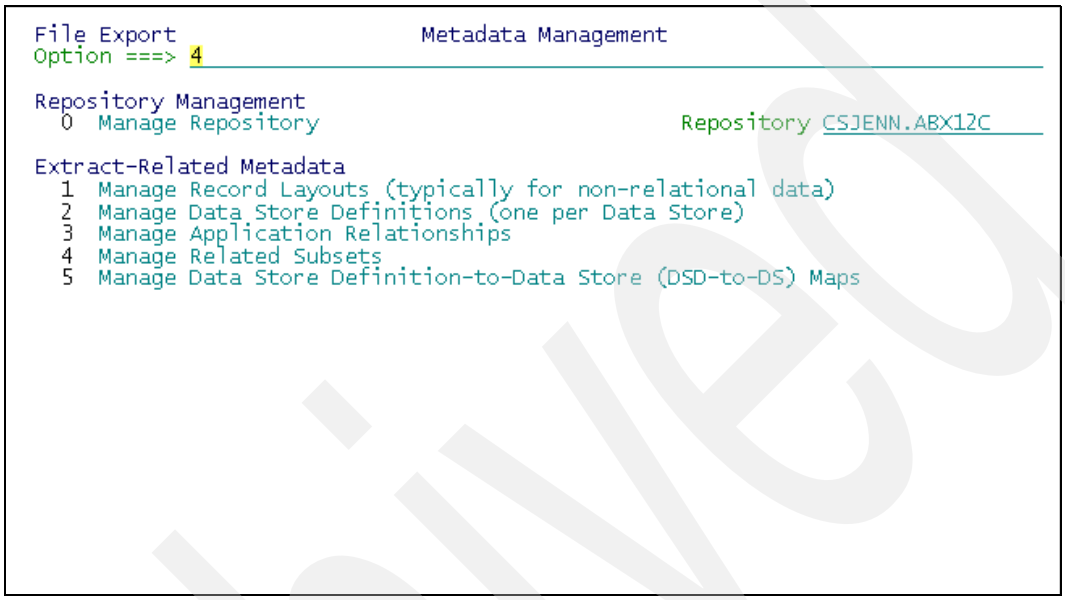


Figure 22-17 Related Subsets are objects that are related that are to be included in an extract request

Step 2: Provide the Related Subset a name and type **C** on the Command Line to create the Related Subset as shown in Figure 22-18.

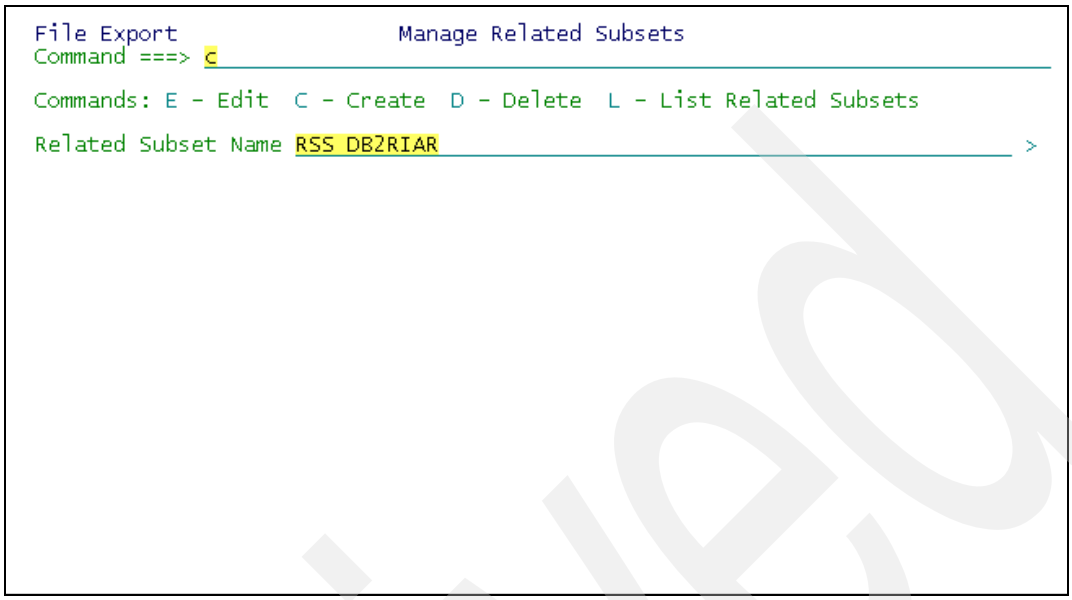


Figure 22-18 Creating a related subset of objects

Step 3: Define the driving object of the Related Subset. This object is the starting point in the extract to find related objects.

Parent relationships, child relationships or both can be searched for. In addition, the maximum levels of relationships can be specified. If no value is placed in the field Maximum relationship levels, all related objects are populated into the Related Subset as shown in Figure 22-19.

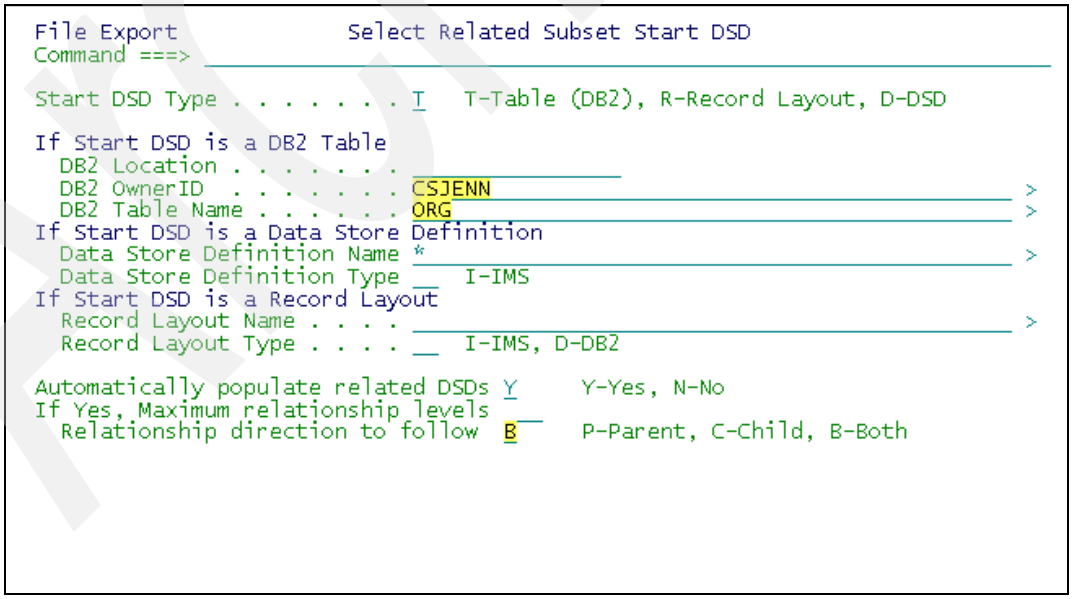


Figure 22-19 Define the driving object to begin the relationship traversal

Step 2: The table ORG was the driving object from which the related objects were populated into the Related Subset. Table STAFF had a direct relationship through DB2 RI as a child to table ORG, so STAFF was populated into the Related Subset. Table IT_ASSETS was not directly related to table ORG, but because it had an application relationship defined to table STAFF, it too was populated into the Related Subset.

Selection criteria can now be added to the objects in the Related Subset.

Step 4: Tab down next to table ORG and type a **C** to add selection criteria as shown in Figure 22-20.

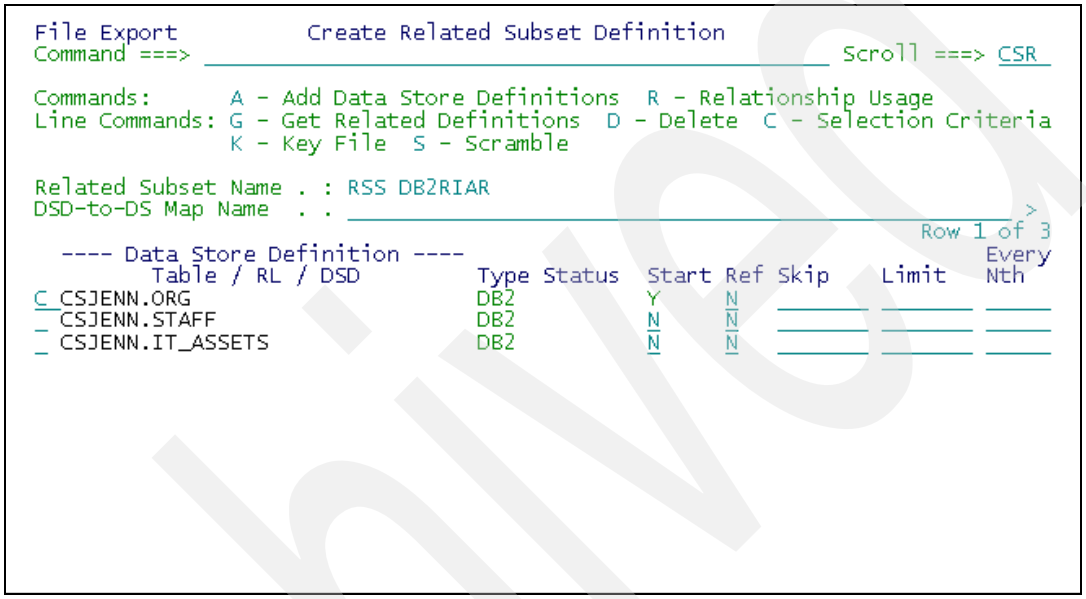


Figure 22-20 Selection criteria can be added to each object from this panel

Step 5: On the Specify Simple Selection Criteria, operators and column values can be specified as shown in Figure 22-21 and Figure 22-23.

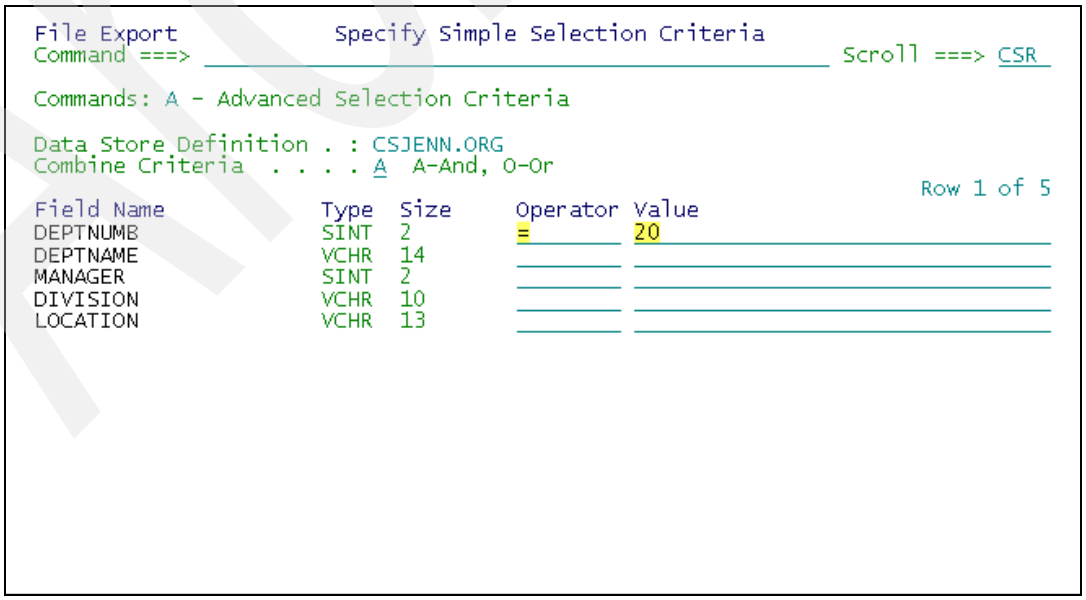


Figure 22-21 Selection criteria can be specified this way

Selection criteria can optionally be specified on each object to help narrow down the search criteria on related data as shown in Figure 22-22 and Figure 22-23.

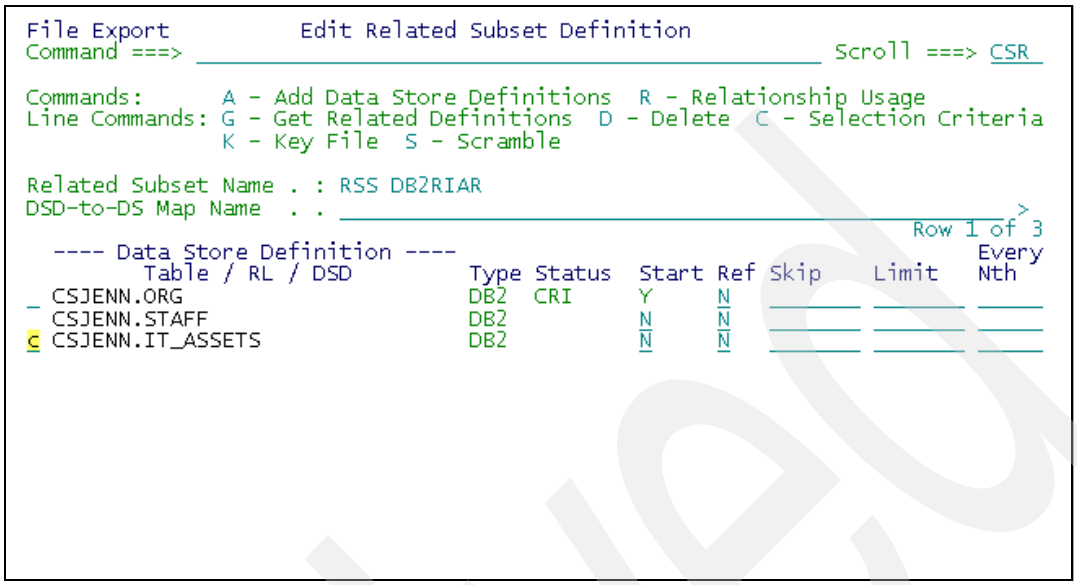


Figure 22-22 Selection criteria can be placed on individual objects within the Related Subset

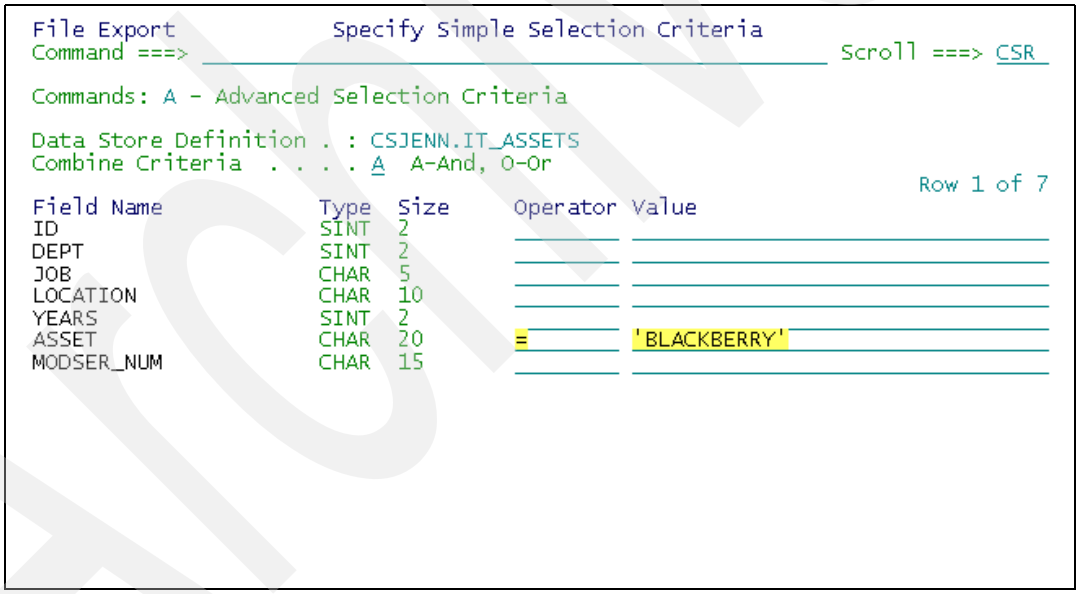


Figure 22-23 Narrow down your search results

Task 3: Create a DSD-to-DS Map for the Source data.

Remember that a DSD-to-DS Map is used to map “logical” Data Store Definitions to their “physical” counterparts, or underlying files. This map tells File Export what the structural definition of the source looks like and where to find the data.

Step 1: First create the DSD-to-DS Map that is used for the source data. That is, from the Metadata Management panel, select option **5** Manage Data Store Definition-to-Data Store (DSD-to-DS) Map as shown in Figure 22-24. Press Enter.

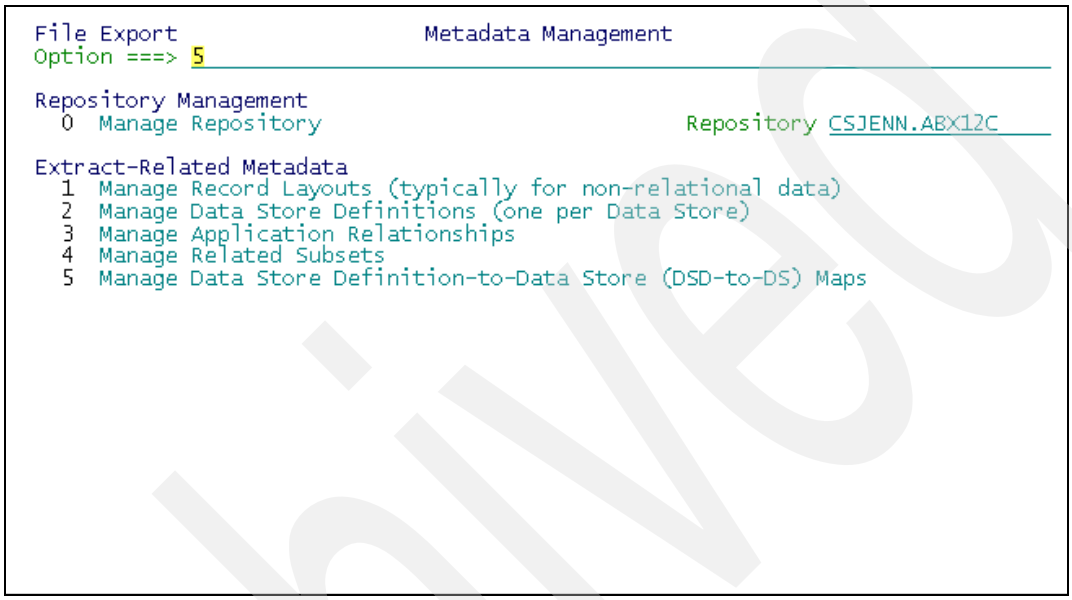


Figure 22-24 Step 1: Select Option 5 to create a DSD-to-DS Map

Step 2: Once on the Manage DSD-to-DS Map panel, provide the Map a name and specify the Map to use the Related Subset that was just created as shown in Figure 22-25. Press Enter.

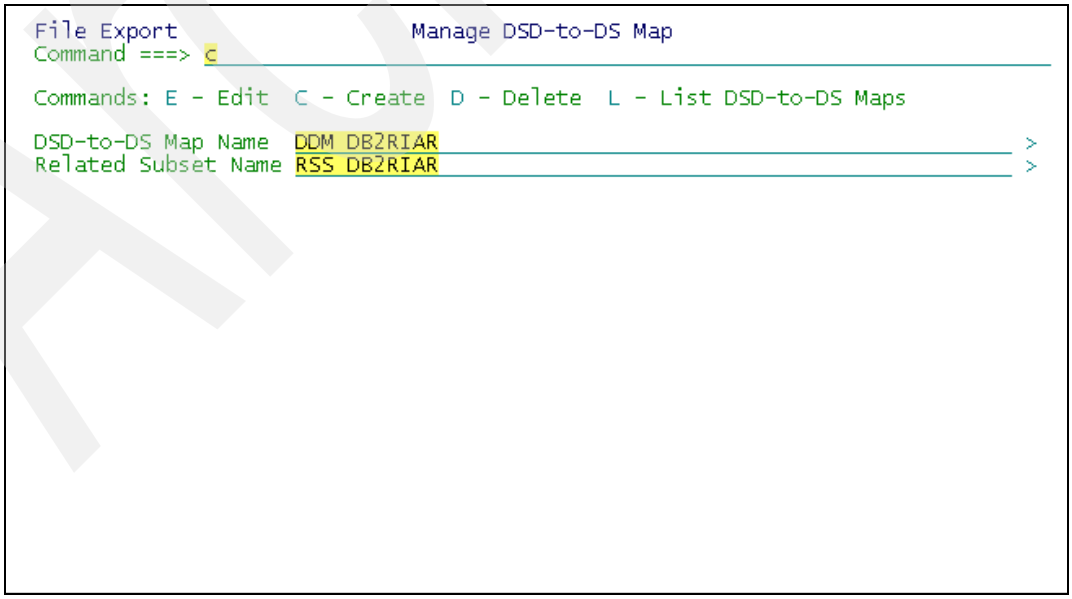


Figure 22-25 Step 2: Create a Map to point to the Related Subset of objects that was just created

Step 3: When the DSD-to-DS Map has been created, the Related Subset of objects is populated into the Map telling File Export what the structural definition of the source objects are and the physical location of the data. See Figure 22-26.

```
File Export                    Create DSD-to-DS Map
Command ===> _____________________________________________ Scroll ===> CSR

Commands: S - Synchronize

For DB2                                        For IMS
  Default Location _______________               Default SSID ____
  Default Owner ID _____________________ >
                                                              Row 1 of 3
                                 ------- Data Store (Physical File) ------------
Data Store Definition      Type    DB2 Location.OwnerID.TableName / IMS DBD
CSJENN.ORG                 TBL  CSJENN.ORG
CSJENN.STAFF               TBL  CSJENN.STAFF
CSJENN.IT_ASSETS           TBL  CSJENN.IT_ASSETS
```

Figure 22-26 Specify the source objects from which data is extracted

In this example, File Export looks at the DDL in the DB2 catalog to see how the source objects are defined. Because the data also reside in the DB2 tables, the table qualifier and table name are specified. In this scenario, the “logical” object and the “physical” location of the data are the same.

Task 4: Create a DSD-to-DS Map for where the data is loaded, that is, the target (or destination) objects.

The target DDM tells File Export the structural definition of the target objects and where to load the physical data. Just as the Source DDM specified the logical object with the physical data to extract, the Target DDM defines the logical object with the physical location to load the data.

Step 1: Create a Data Store Definition-to-Data Store Map for the target object. While still on the Manage Data Store Definition-to-Data Store (DSD-to-DS) Map panel, provide another name for the Destination Map, and use the same Result Subset name. Type a **C** on the command line to create the DDM and press Enter. See Figure 22-27 for an illustration.

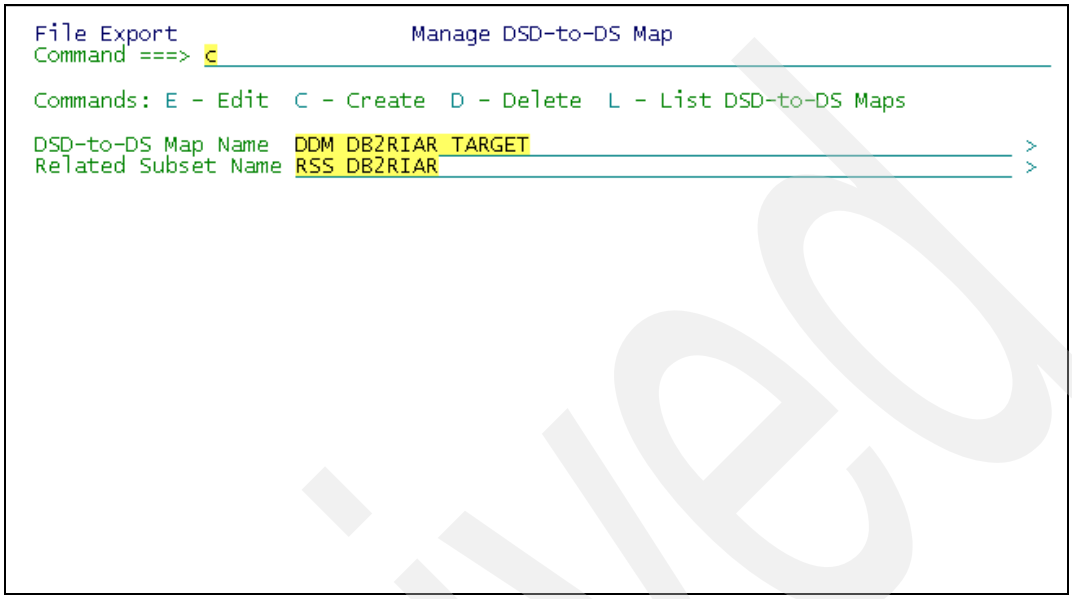


Figure 22-27 Create a DSD-to-DS Map for the Target data

Step 2: Upon pressing Enter, panel Create DSD-to-DS Map is displayed showing the table qualifier and names containing the data in the right-hand column. If the target table qualifier is different than the source table qualifier, delete the existing qualifier and simply put the new qualifier in the field Default Owner ID. File Export then prepends the Default Owner ID to the table name as shown in Figure 22-28.

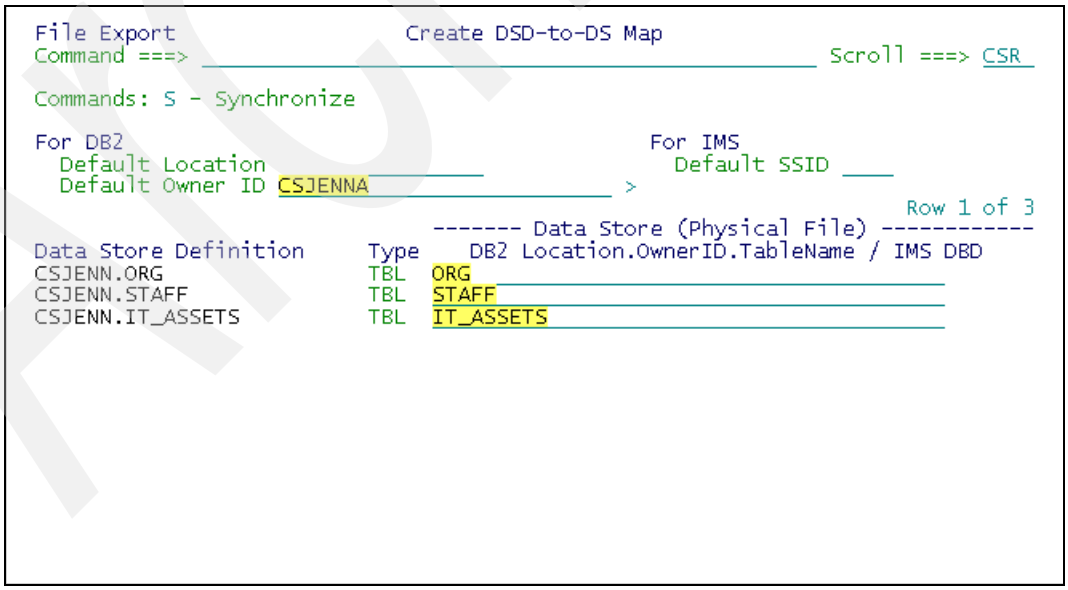


Figure 22-28 Use the Default Owner ID field to change the target table qualifier

22.3.2 Create an extract and load request

Now you are ready to create an Extract Request using the Related Subset we just created.

Step 1: From the File Export Primary Option menu, select option **3** Multiple Data Store Extract and Load. Press Enter. Figure 22-29. Both an extract job and a load job are created.

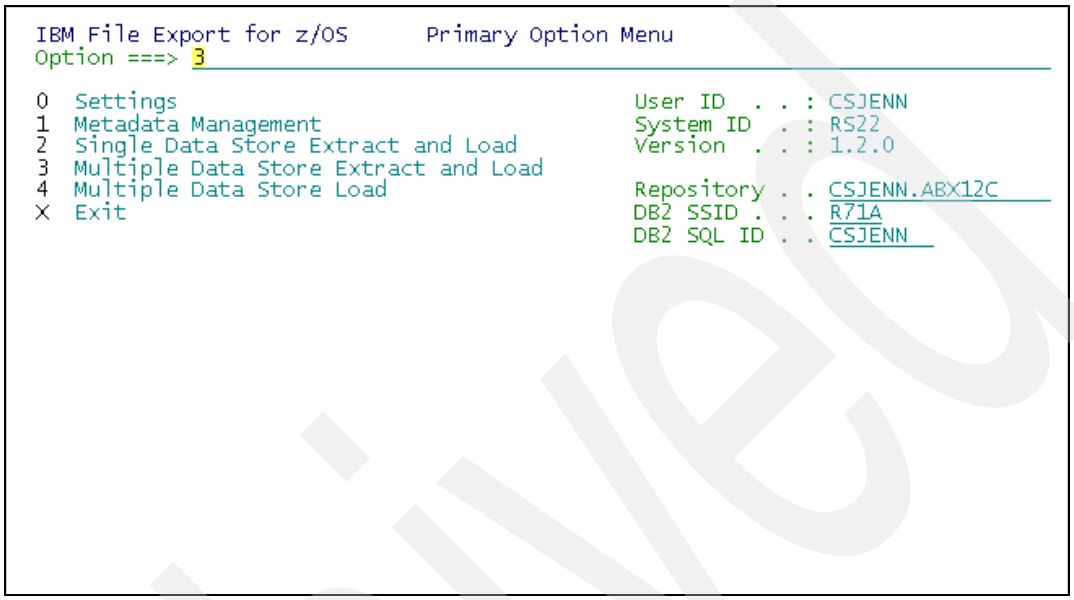


Figure 22-29 Multiple Data Store Extract and Load

Step 2: Provide a name for the Extract Request and an optional description. Specify the Related Subset of objects to extract data from. Provide the name of the DSD-to-DS Map for the source tables, and the DSD-to-DS Map for the target tables. The value in the field Extract File Name is used to define the load control dataset and the intermediate datasets where the extracted data is held until it is loaded into the target tables. When all fields are filled in, type **C** on the command line to create the Extract Request. See Figure 22-30 for an illustration.

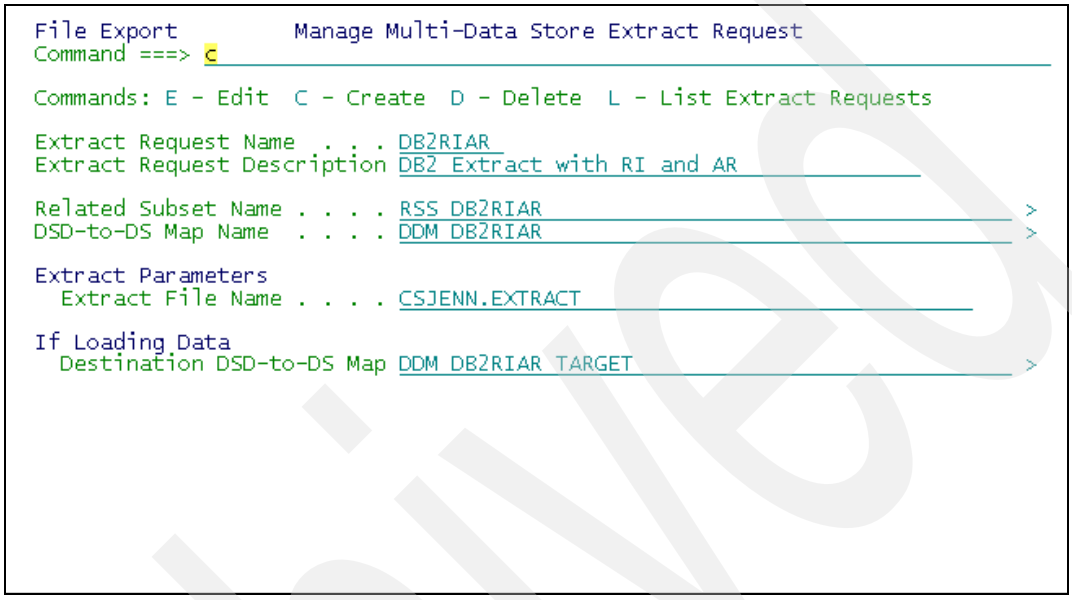


Figure 22-30 Specify the RSS and DDM to create an Extract Request

Step 3: In Figure 22-31, the fields under Generated JCL File tell File Export where to store the generated Extract and load Request JCL. If the JCL dataset in the field Data Set Name does not exist, then File Export creates it as a FB 80 PDS. An existing dataset name can also be provided.

The value in the field Member Name is the PDS member name of the Extract Request JCL. The field Load Step Member Name is the PDS member name of the Load Request, if a separate job for the Load is requested.

Set Generate Load Step to **Y** so that a Load Request is generated, and Review JCL to **Y** to receive an ISPF edit session to display the generated Extract Request JCL.

Note: If the field LoadStep Member Name is left blank, but the option Generate Load Step is set to **Y**, the Load step is included in the same JCL as the Extract step.

Scroll down the panel to see more options and select option **4** to specify the DB2 Settings for the target DB2 subsystem, and press Enter.

```
File Export        Multi-Data Store Extract: DB2 Destination
Option ===> 4

Extract Request Name . : DB2RIAR
                                                              More:     +
Related Subset Name  . : RSS DB2RIAR
DSD-to-DS Map Name . . : DDM DB2RIAR
Extract File HLQ . . . . CSJENN.EXTRACT

Generated JCL file
  Data set name  . . . . CSJENN.ABX.V12.JCL
  Member name  . . . . . DB2RIAR   Blank or pattern for member list
  Load step member name  DB2RIARL

Generate Load Step . . . Y         Y-Yes, N-No

Review JCL . . . . . . . Y         Y-Yes, N-No
Submit JCL . . . . . . . N         Y-Yes, N-No

JCL Management
  1  STEPLIBs
  2  JCL to precede extract execution
  3  JCL to follow extract execution
```

Figure 22-31 Parameters to generate Extract and Load Request JCL

Step 4: On the DB2 Settings panel, specify command line S and press Enter to see a list of target DB2 subsystems. Select a subsystem by typing S next to it and press Enter. The DB2 Settings panel is returned with the selected DB2 subsystem parameters populated into fields DB2 Subsystem and DB2 Loadlib.

To have File Export check the referential integrity of the data, set option Enforce RI to **Y** as shown in Figure 22-32. Otherwise, the Target Database and Tablespace must be specified so that the Check Data utility can be generated into the Load Request job.

```
File Export        Multi-Data Store Extract: DB2 Settings
Command ===>

Destination DB2
  DB2 Subsystem  . . . R71A
  DB2 Loadlib  . . . . DSN.V710.SDSNLOAD

DB2 Load Parameters
  RESUME . . . . . . . Y    Y-Yes, N-No
  REPLACE  . . . . . . _    Y-Yes, N-No
  REUSE  . . . . . . . _    Y-Yes, N-No
  Enforce RI . . . . . Y    Y-Yes, N-No (No implies running CHECK DATA utility)

If CHECK DATA utility needed
  Target Database  . . . . . ________
  Target Tablespace  . . . . ________

DSNUPROC parameters:
  Region Size  . . . . . . . _____
  Unique Identifier  . . . . ________________
  Restart Command Option . . . ________________   blank, PREVIEW, RESTART,
                                                 RESTART(CURRENT), or
                                                 RESTART(PHASE)
```

Figure 22-32 Specify the Target DB2 Subsystem options

Step 5: When the DB2 Settings panel has been filled in, press PF3 to process the options and return to the DB2 Destination panel. On the DB2 Destination panel, press Enter to generate Extract and Load Request JCL.

The first ISPF edit session returns the Extract Request JCL. It can be submitted immediately, put into a job scheduler to run during a batch window, or stored for later submission as shown in Figure 22-33. Press PF3 to exit the Extract Request JCL and view the Load Request JCL.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CSJENN.ABX.V12.JCL(DB2RIAR) - 01.00            Columns 00001 00072
Command ===>                                               Scroll ===> CSR
****** ***************************** Top of Data ********************************
000001 //DB2RIAR JOB CSJENN,CLASS=A,NOTIFY=&SYSUID,
000002 //       MSGCLASS=X,MSGLEVEL=(1,1)
000003 //*
000004 //*
000005 //STEP1    EXEC PGM=ABXUTIL
000006 //STEPLIB  DD   DISP=SHR,DSN=RSQA.ABX120.IBMTAPE.SABXLOAD
000007 //         DD   DISP=SHR,DSN=RSRTE.XML.HXML180.SIXMLOD1
000008 //         DD   DISP=SHR,DSN=CSWILK.ABX.GLOBAL.INSTLBLK.V2
000009 //         DD   DISP=SHR,DSN=DSN.V710.SDSNLOAD
000010 //*
000011 //*
000012 //ABXCTL   DD   *
000013 OPTIONS(DB2, PLAN(JNABX120));
000014 SOURCEDB2(R71A);
000015 TARGETDB2(R71A);
000016 DB2RESULTSET(RESULTSET1) SOURCE(CSJENN.ORG)
000017 TARGET(CSJENN.ORG) DD:TABL0001 LOAD(RESUME);
000018 SELECT * FROM CSJENN.ORG WHERE (DEPTNUMB = 20);
000019 DB2RESULTSET(RESULTSET2) SOURCE(CSJENN.STAFF)
```

Figure 22-33 The generated Extract Request JCL

The Load Request JCL, as seen in Figure 22-34, utilizes the ABXCNTL dataset and TABL000n datasets that were created by running the Extract Request job. The ABXCNTL dataset that contains the Load control syntax is populated into the Load Request JCL as the SYSIN dataset. The TABL000n datasets that contain the extracted data are utilized as input to the LOAD utility.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CSJENN.ABX.V12.JCL(DB2RIARL) - 01.00           Columns 00001 00072
Command ===>                                               Scroll ===> CSR
****** ***************************** Top of Data ********************************
000001 //DB2RIARL JOB CSJENN,CLASS=A,NOTIFY=&SYSUID,
000002 //       MSGCLASS=X,MSGLEVEL=(1,1)
000003 //*
000004 //*
000005 //*
000006 //STEPLOAD EXEC DSNUPROC,COND=(4,LT),
000007 //        LIB=DSN.V710.SDSNLOAD,SIZE=0M,
000008 //        SYSTEM=R71A,UID='',UTPROC=''
000009 //DSNTRACE DD  SYSOUT=*
000010 //SYSPRINT DD  SYSOUT=*
000011 //UTPRINT  DD  SYSOUT=*
000012 //SORTLIB  DD  DSN=SYS1.SORTLIB,DISP=SHR
000013 //SYSIN    DD  DSN=CSJENN.EXTRACT.LOADCNTL,
000014 //             DISP=(OLD,KEEP)
000015 //SYSUT1   DD  UNIT=SYSDA,SPACE=(4000,(20,20),,,ROUND)
000016 //SORTOUT  DD  UNIT=SYSDA,SPACE=(4000,(20,20),,,ROUND)
000017 //SORTWK01 DD  UNIT=SYSDA,SPACE=(4000,(20,20),,,ROUND)
000018 //SORTWK02 DD  UNIT=SYSDA,SPACE=(4000,(20,20),,,ROUND)
000019 //SORTWK03 DD  UNIT=SYSDA,SPACE=(4000,(20,20),,,ROUND)
```

Figure 22-34 The load job utilizes datasets created in the Extract Request JCL

The Load Request job can be run multiple times to populate a test environment repeatedly as required.

22.4 Creating an IMS extract using logical relationships

IMS DBDs have logical relationships defined in the segments. Logical relationships can be utilized by File Export as Application Relationships using the Import function to extract related data without the user having to redefine the logical relationships to File Export.

This extract utilizes the IMS Option Block that is displayed in the chapter titled Introduction to File Export. Therefore, setting IMS options are not described.

22.4.1 Defining the required repository objects

In this scenario, the existing IMS logical relationships are used to create a related data extract. The field definitions for each IMS DBD Segment are defined into a Record Layout, and those Segments that are defined to the same DBD are grouped into a Data Store Definition.

Task 1: Create Data Store Definitions for the IMS DBDs that contain logical relationships.

Step 1: Select option **2** from the Metadata Management panel to create Data Store Definitions for those IMS DBDs that contain related data as shown in Figure 22-35. Press Enter.

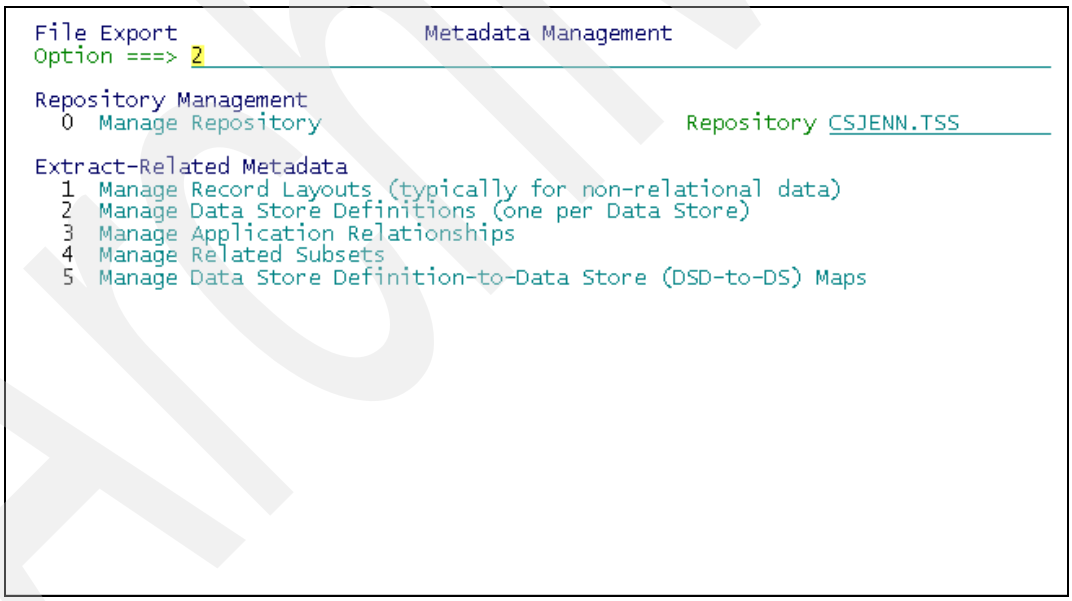


Figure 22-35 Create Data Store Definitions for those IMS DBDs that are extracted from

Step 2: Option Import allows File Export to find logical relationships from the IMS DBD schema definitions, and import the relationships into the Repository for use in creating Related Subsets and Extract Requests.

Specify Command option I to import the logical relationships from the IMS DBDSRC as shown in Figure 22-36. Press Enter.

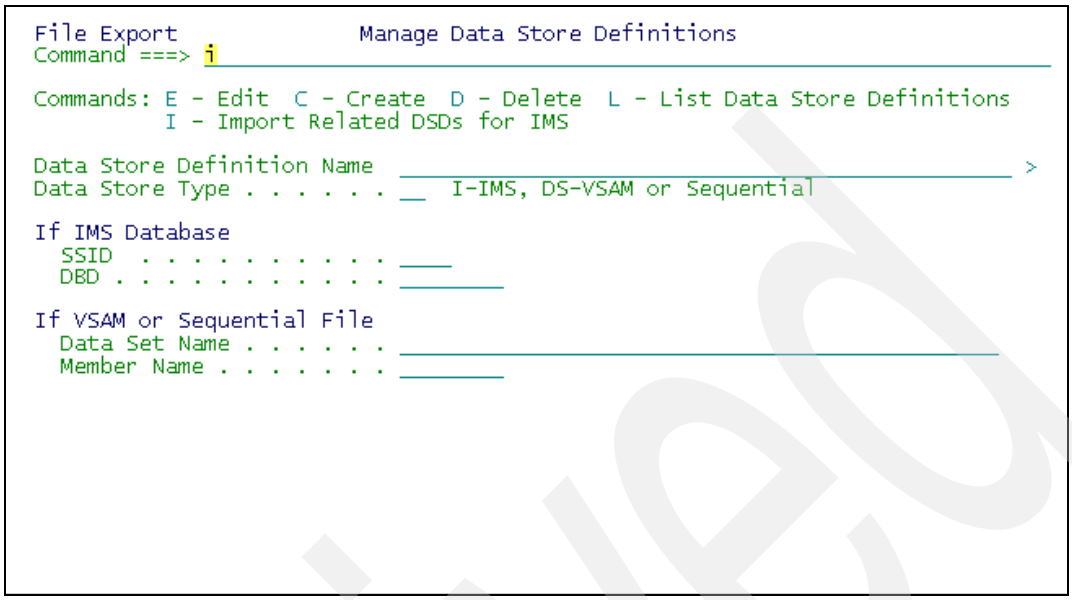


Figure 22-36 Specify I to import IMS logical relationships

Step 3: Specify the IMS SSID on which the DBDs reside. The field SSID can be wild carded to any degree using an asterisk as shown in Figure 22-37. Press Enter.

Figure 22-37 The IMS SSID and DBD can be wild carded to any degree using an asterisk

Step 4: A pop-up window displays the available IMS SSIDs available to select. The IMS subsystems and their corresponding region processing options are derived from the IMS Option Block assembled and linked into the ABX.SABXLOAD dataset. To add subsystems and / or change processing options, see the section titled “Customizing Option Blocks” on page 544 which explains assembling an IMS Option Block.

Select an IMS SSID that contains the IMS DBDs you want to import relationships from as shown in Figure 22-38, and press Enter.

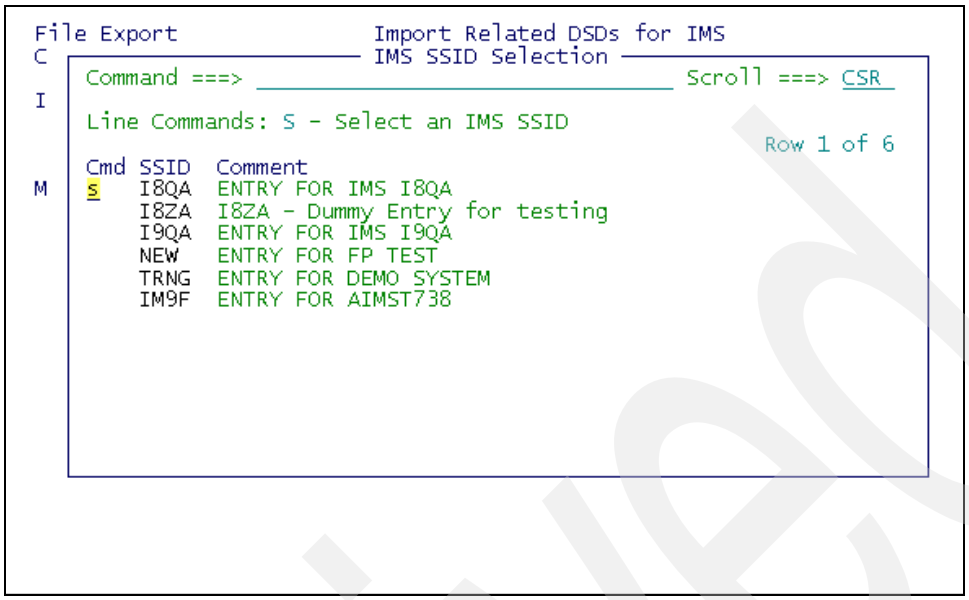


Figure 22-38 A pop-up window displays those IMS SSIDs that have been defined in the IMS Option Block

Step 5: The DBD Selection panel displays all available IMS DBDs within the selected SSID. The Find command is similar to the ISPF command in that the keyword **FIND** or **F** can be used along with a value to locate an object.

Issue the **FIND** command to look for a specific IMS DBD from which to begin importing logical relationships as shown in Figure 22-39.

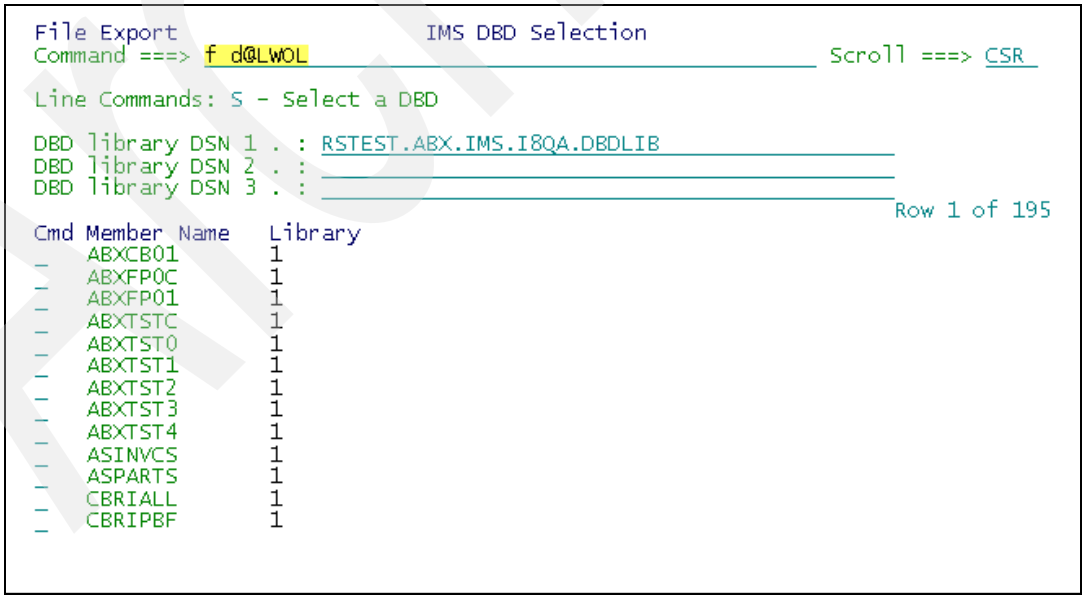


Figure 22-39 Use the FIND command to look for a DBD name

If additional DBD libraries are required, they can be added to the fields DBD Library DSN *n* to broaden the scope for importing IMS DBDs that reside on the same IMS SSID.

To begin populating the Repository with the related IMS DBDs, select the DBD from which to begin the relationship chain as shown in Figure 22-40.

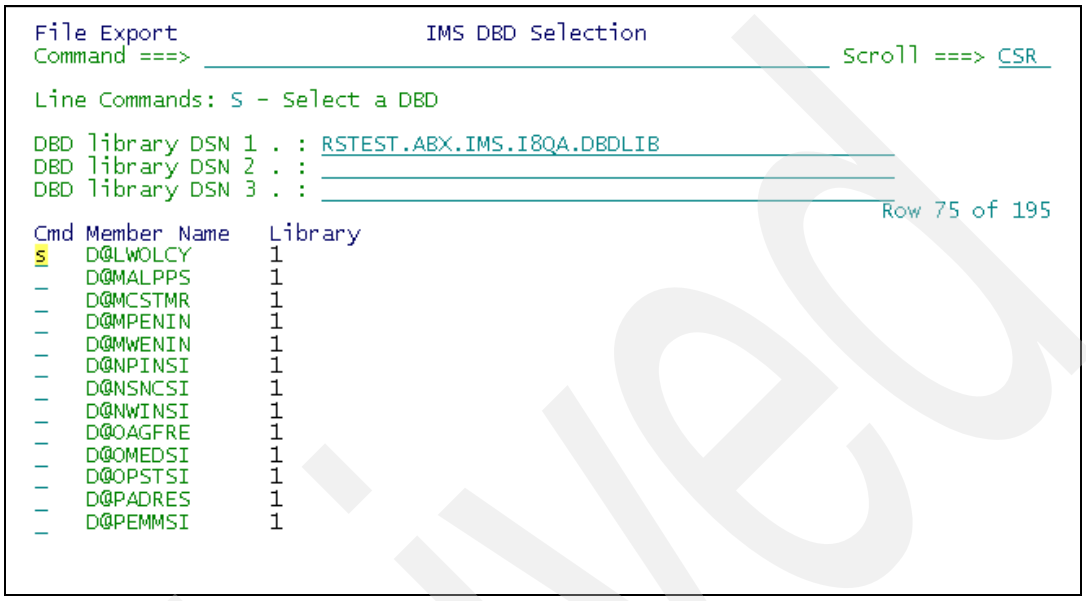


Figure 22-40 Select the IMS DBD when it is located

Step 6: Additionally, specify the maximum level of relationships to traverse when importing the logical relationships. For the field, Maximum relationship levels, leaving it blank results in all logically related IMS DBDs to each be populated into the Repository as individual Data Store Definitions as shown in Figure 22-41.

Press Enter when all field values have been specified.

Figure 22-41 Specify the number of levels of logical relationships to search

When importing IMS logical relationships into the Repository is complete, a message is displayed. To view all Data Store Definitions that have been created, enter Command option L to List all Data Store Definitions that have been created for each IMS DBD as shown in Figure 22-42. Press Enter.

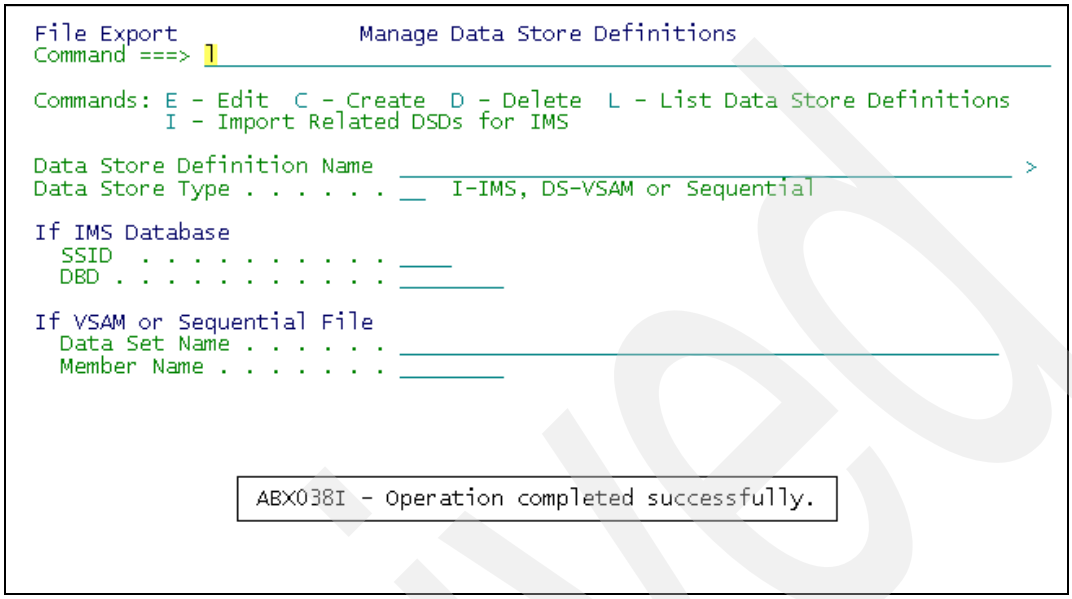


Figure 22-42 List the Data Store Definitions available to choose from

The Data Store Definition Selection panel is displayed listing all IMS DBDs that were imported into the Repository.

To view a Data Store Definition and the underlying Record Layouts for all Segments for each IMS DBD, select a Data Store Definition as shown in Figure 22-43. Press Enter.

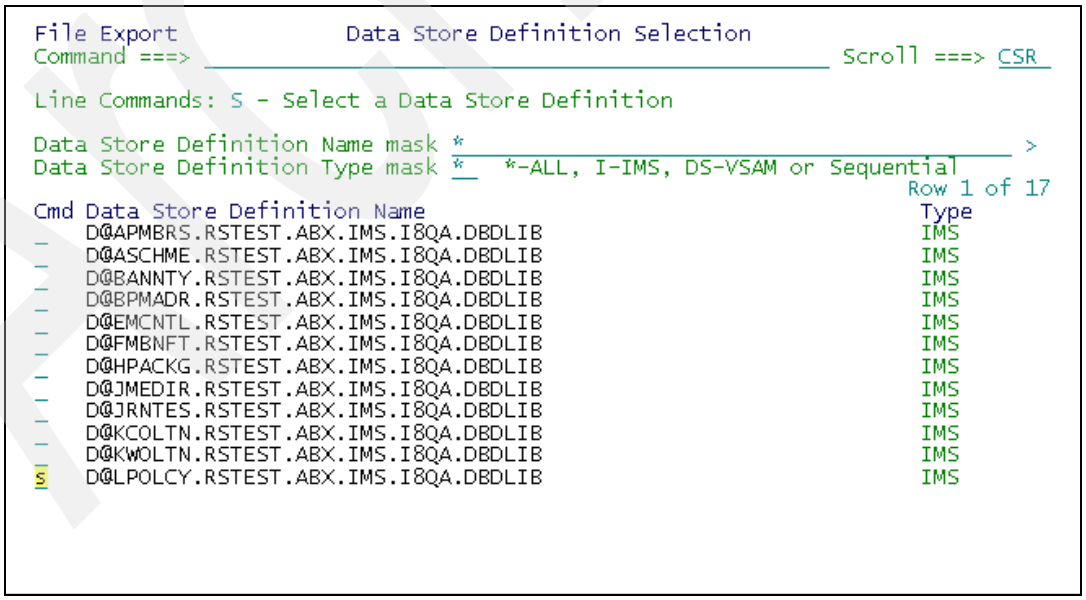


Figure 22-43 To view the underlying Record Layout, select a DSD from the list

The selected DSD name is then populated in the field Data Store Definition Name on the Manage Data Store Definition panel as shown in Figure 22-44. You can then use the **E** Command to edit the current Data Store Definition and view the Record Layouts for all segments within the IMS database for which the DSD was created.

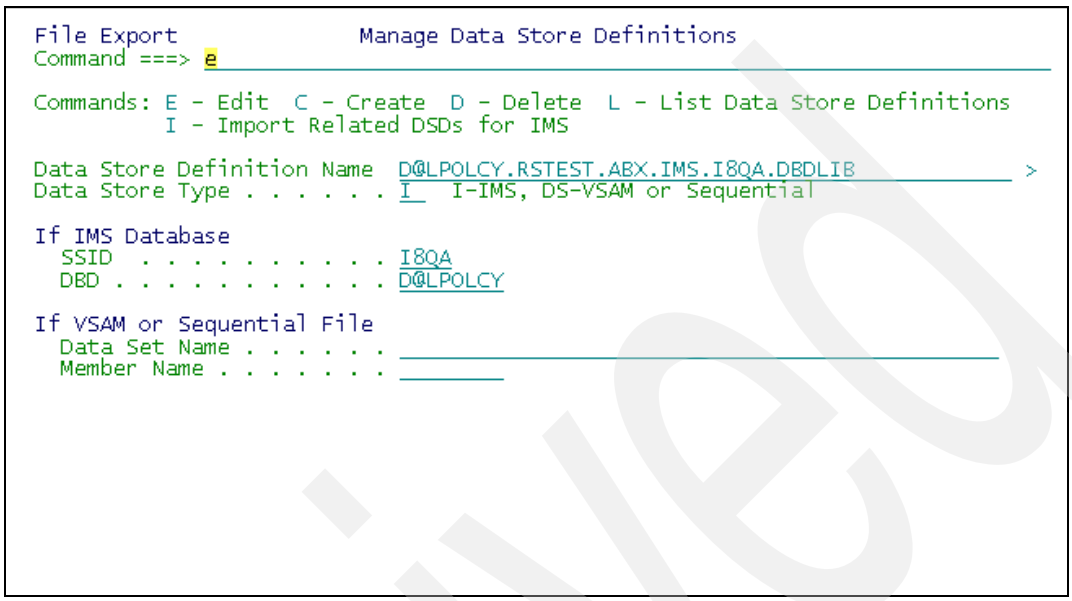


Figure 22-44 Command E allows the Data Store Definition to be edited or viewed

For the selected IMS DSD (which is correlated to an IMS Database), there are three defined segments, each with their own Record Layout, which was also created as part of the Import process. To view a Record Layout for a segment, type V next to a Record Layout and press Enter as shown in Figure 22-45.

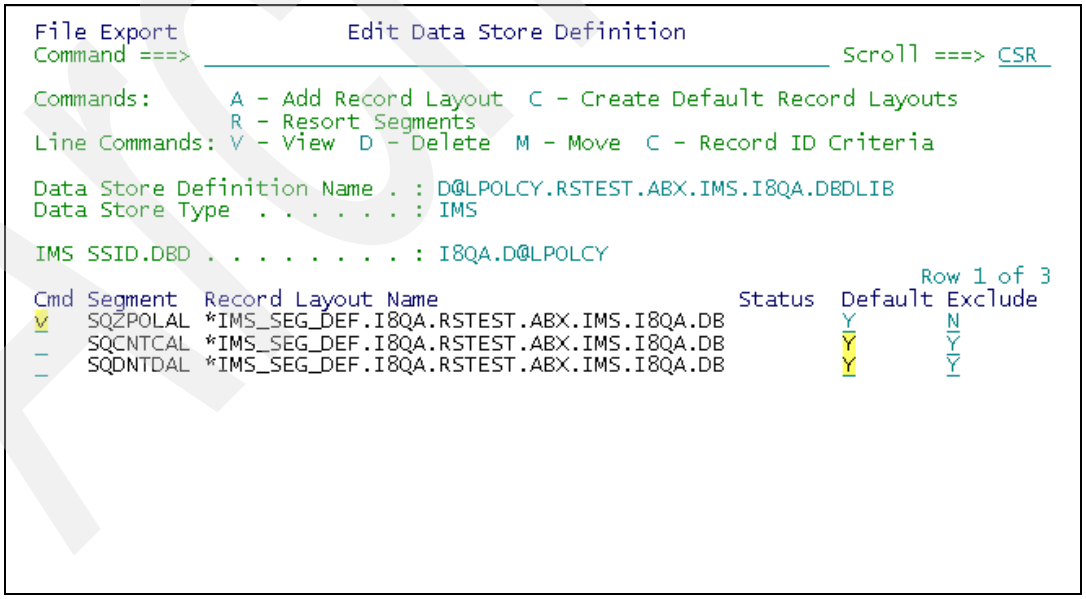


Figure 22-45 IMS DBD Segment definitions can be viewed

Figure 22-46 shows the field definitions that the Record Layout was defined with within IMS. Fields can also be viewed. Because this is the default Record Layout, that is, the field definitions as they were defined to IMS, it cannot be edited. However, another Record Layout can be defined on an IMS segment to redefine the fields.

Use the End command (typically PF3) to return to the Metadata Management panel.

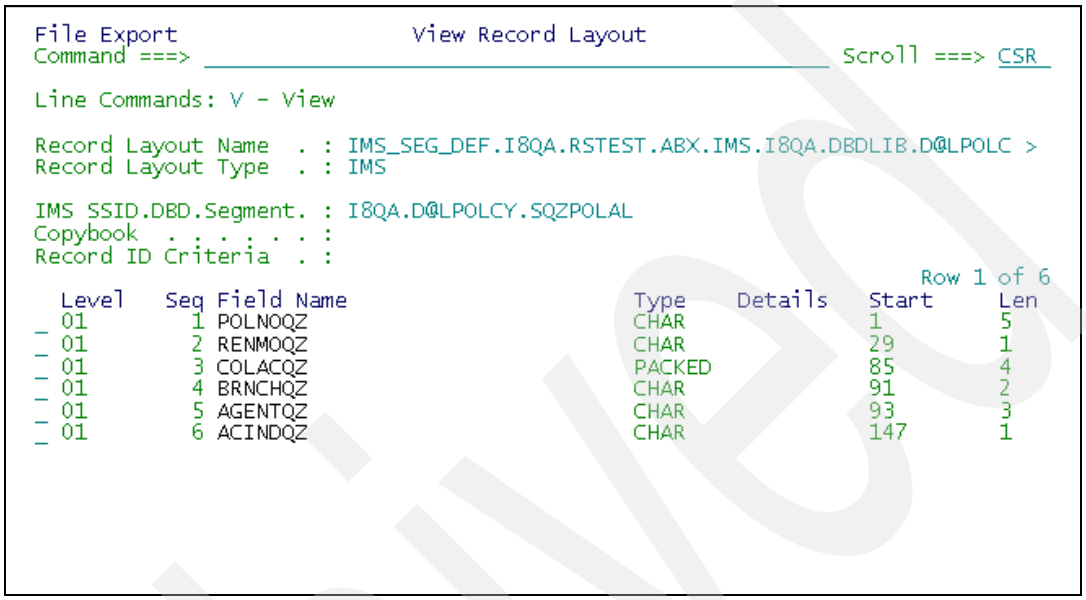


Figure 22-46 The view of a Segment as it is defined in IMS

Task 2: Create a Related Subset of objects that are logically related. Remember that a Related Subset is a group of objects whose related data is extracted as a single request. Begin building the Related Subset from an IMS DBD that will be the start object of an extract.

Step 1: Select option **4** from the Metadata Management panel to Manage Related Subsets as shown in Figure 22-47. Press Enter.

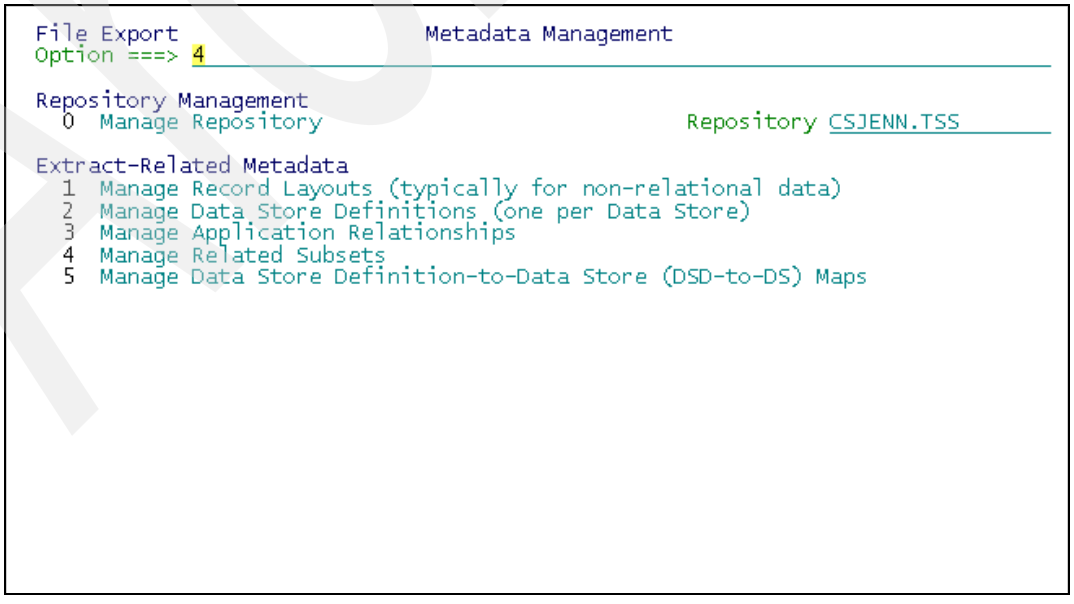


Figure 22-47 Create a Related Subset to contain IMS objects to extract from

Step 2: Provide a name for the Related Subset, and type **C** on the Command line to create it as shown in Figure 22-48. Press Enter.

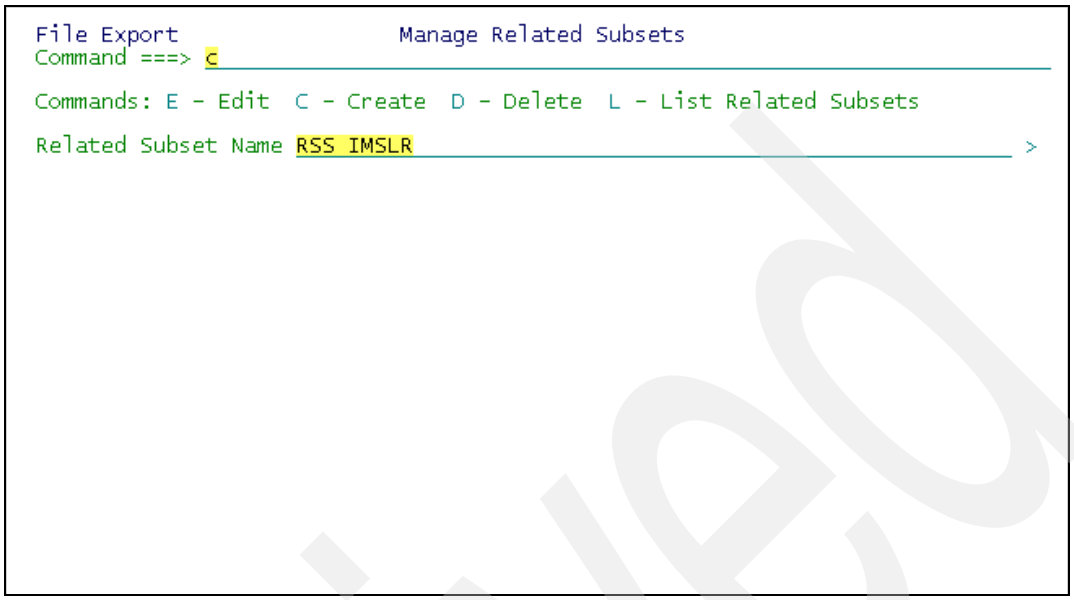


Figure 22-48 Specify a name for the Related Subset

Step 3: Specify option D in the field Start DSD Type to indicate that the Relates Subset is to be populated by Data Store Definitions. Wildcard the Data Store Definition subfield Data Store Definition Name with an asterisk to select the IMS DBD from a list. Specify the Data Store Definition Type as IMS by specifying **I** in the subfield as shown in Figure 22-49. Press Enter.

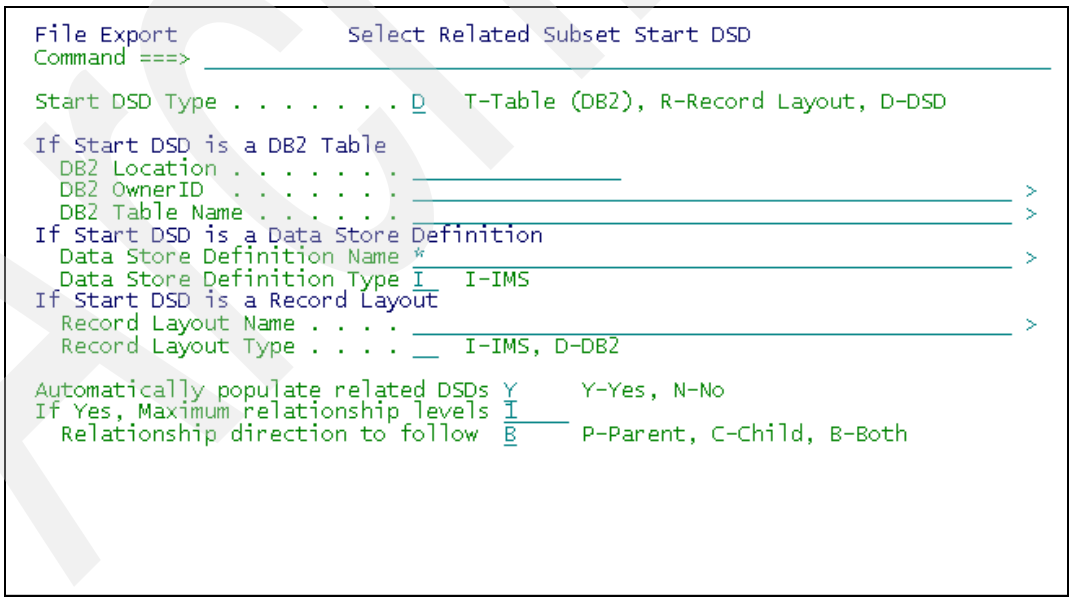


Figure 22-49 Start type D creates a Related Subset from Data Store Definitions

Step 4: A list of IMS-type Data Store Definitions is displayed allowing you to choose the starting IMS DSD from which to begin the extract. A value in the field Data Store Definition Name Mask narrows down the Data Store Definitions displayed in the list. Select the IMS DSD to use as the starting object in a Related Subset as shown in Figure 22-50. Press Enter.

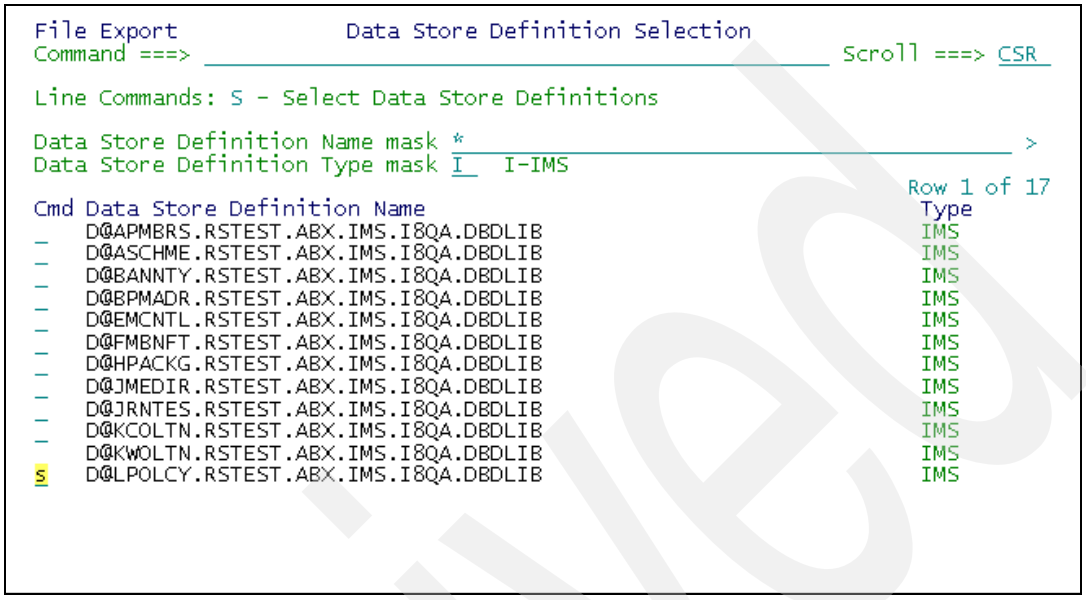


Figure 22-50 Select the Data Store Definition of the IMS DBD Segment to begin the extract from

Step 5: Those IMS DSDs related to the start object (via Application Relationships created by the Import process) are populated into the Related Subset as shown in Figure 22-51. This list of objects comprises the set of objects that related data is extracted from.

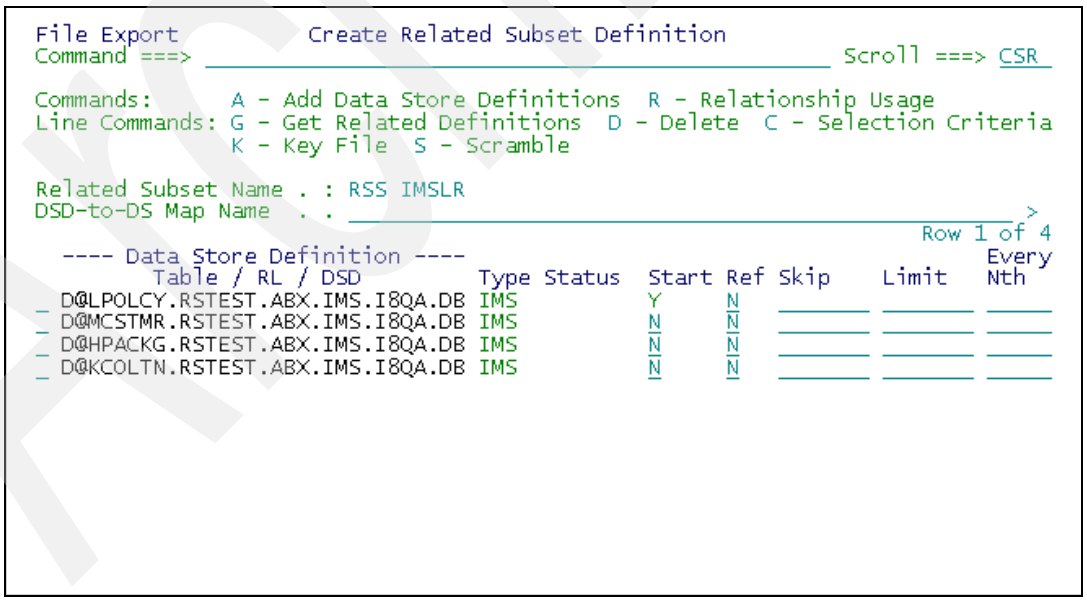


Figure 22-51 All related objects are populated into the Related Subset

Note: Additional *unrelated* objects can be added to the list by specifying A on the Command line. Additional *related* objects can be added to the list by specifying Line Command G next to an object in the list.

Task 3: Create a Source Data Store Definition-to-Data Store Map or DDM. Remember that a DDM maps logical structures to their physical file locations. It describes the structural definition of the object and tells File Export where to physically find the object's data.

Step 1: From the Metadata Management panel, select option **5** to create a Data Store Definition-to-Data Store Map as shown in Figure 22-52. Press Enter.

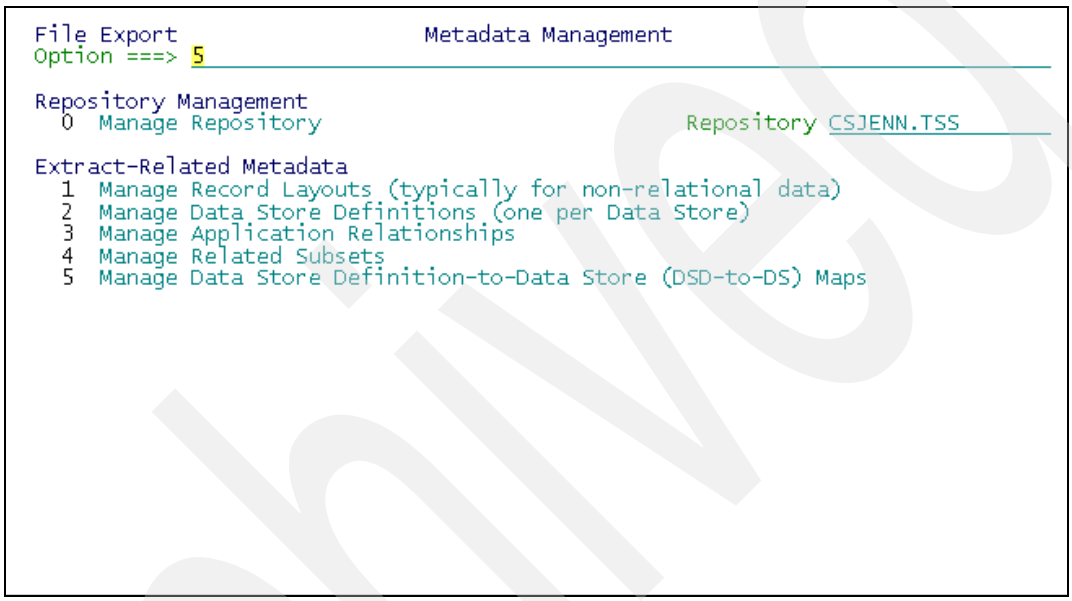


Figure 22-52 Source and destination data maps must be created

Step 2: Specify a name for the DDM and specify which Related Subset is to be used to include the source objects. Type **C** on the Command line to create the Data Store Definition-to-Data Store Map as shown in Figure 22-53. Press Enter.

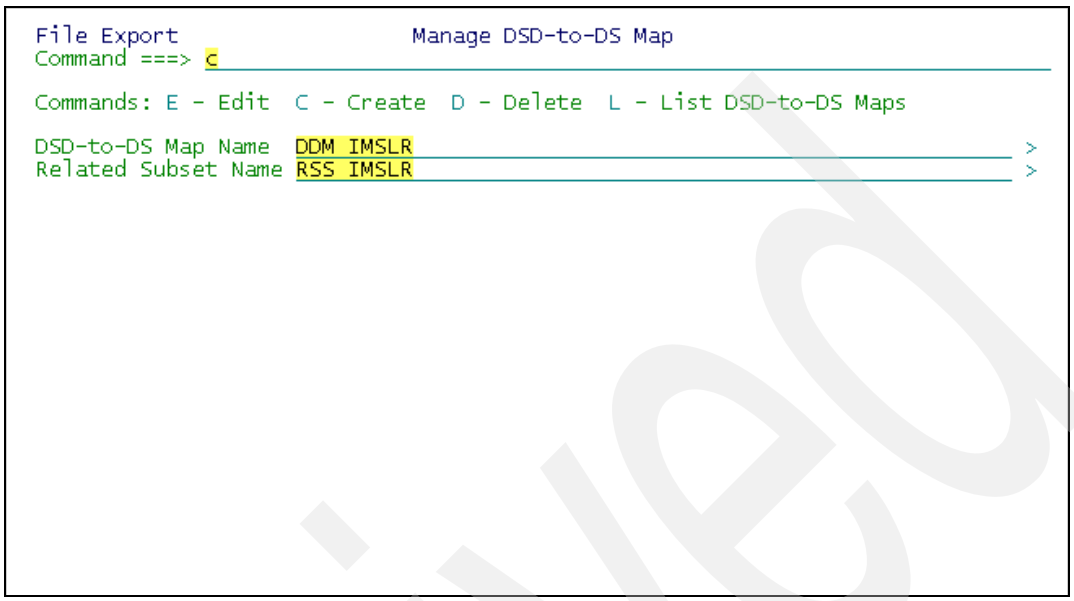


Figure 22-53 Provide a name for the DDM, and use the newly created Related Subset

In Figure 22-54, the Create DSD-to-DS Map panel shows the Data Store Definition name on the left hand side of the panel and the DBD name on the right hand side of the panel. The Data Store Definitions are the logical structural definition of the data. The IMS DBD names are the physical location of the data.

This is how File Export knows how the data is defined, and where to find the actual data. DDMs can be used with more than one Related Subset, and a Related Subset can be used with more than one DDM depending on the related data that must be extracted.

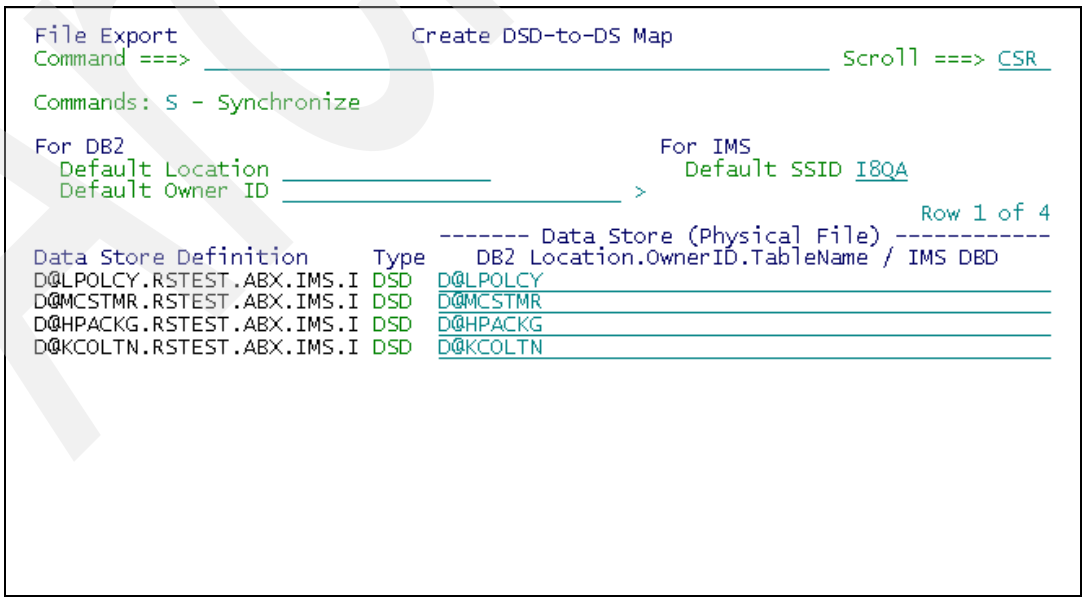


Figure 22-54 The structural definition is on the left, the IMS DBD where the data is located is on the right

Task 4: Create a Target (or destination) Data Store Definition-to Data Store Map to tell File Export where to load the extracted data. The same Related Subset is used to define the DDM. This is typical when the structure of the source and destination data is the same.

Step 1: On the Manage DSD-to-DS Map panel, provide a name for the target DDM, and specify the same Related Subset of objects that was used to create the source DDM. Type **C** on the Command line, as shown in Figure 22-55, and press Enter.

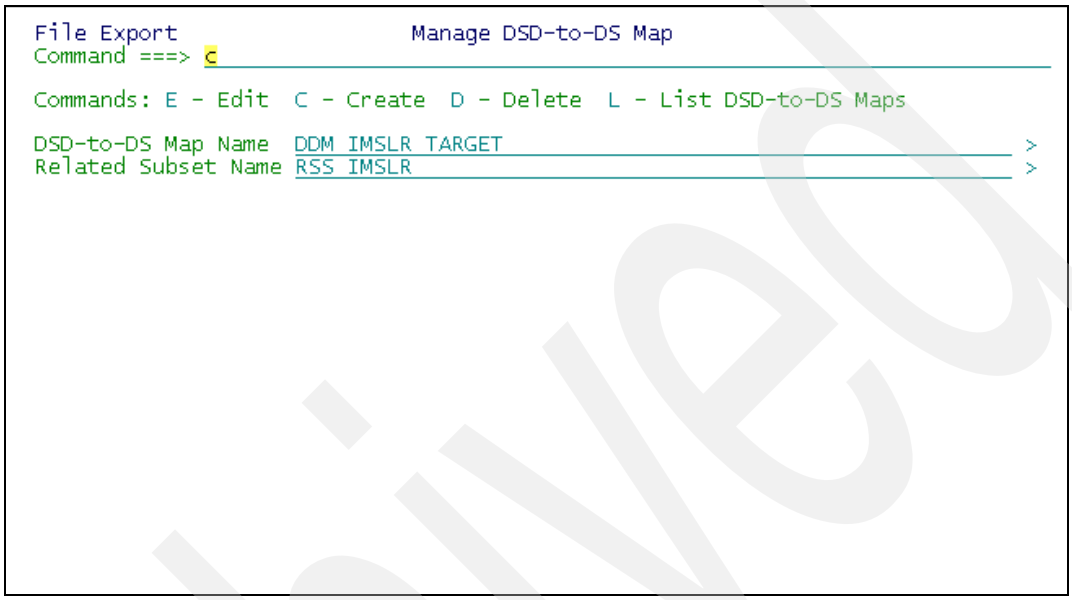


Figure 22-55 Create a DDM for the destination objects using the same Related Subset

Step 2: To change the target IMS subsystem ID, specify an asterisk in the Default SSID field for IMS as shown in Figure 22-56. Press Enter.

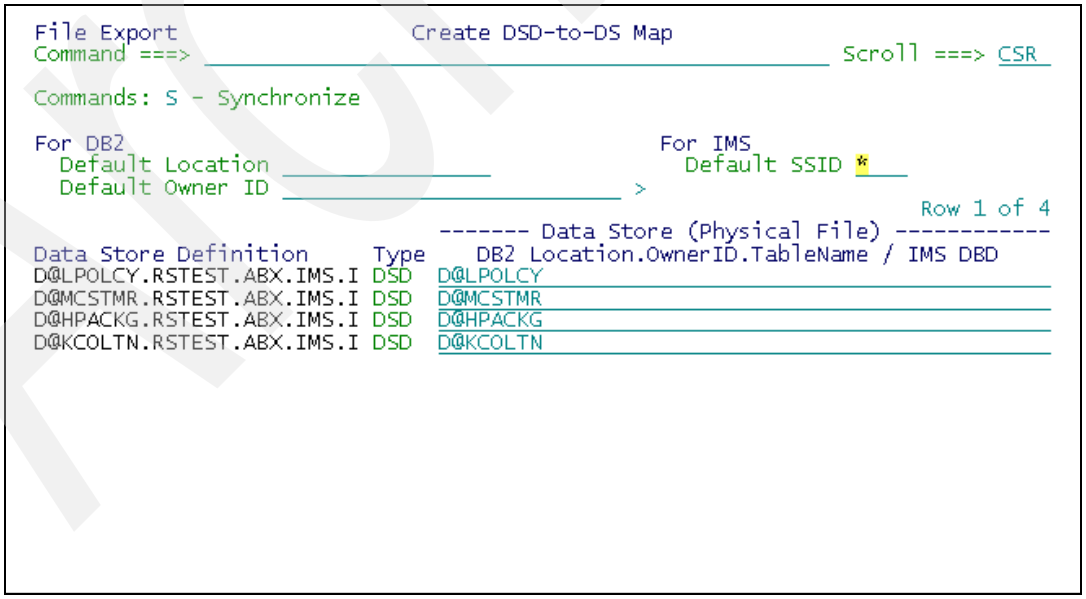


Figure 22-56 Another IMS SSID can be specified as the target location

A pop-up window displays the available IMS SSIDs available to select as the target IMS SSID as shown in Figure 22-57. The IMS subsystems and their corresponding region processing options are derived from the IMS Option Block assembled and linked into the ABX.SABXLOAD dataset. To add subsystems and / or change processing options, see the section titled “Customizing Option Blocks” on page 544 which explains assembling an IMS Option Block. Select an IMS SSID as the destination. Press Enter.

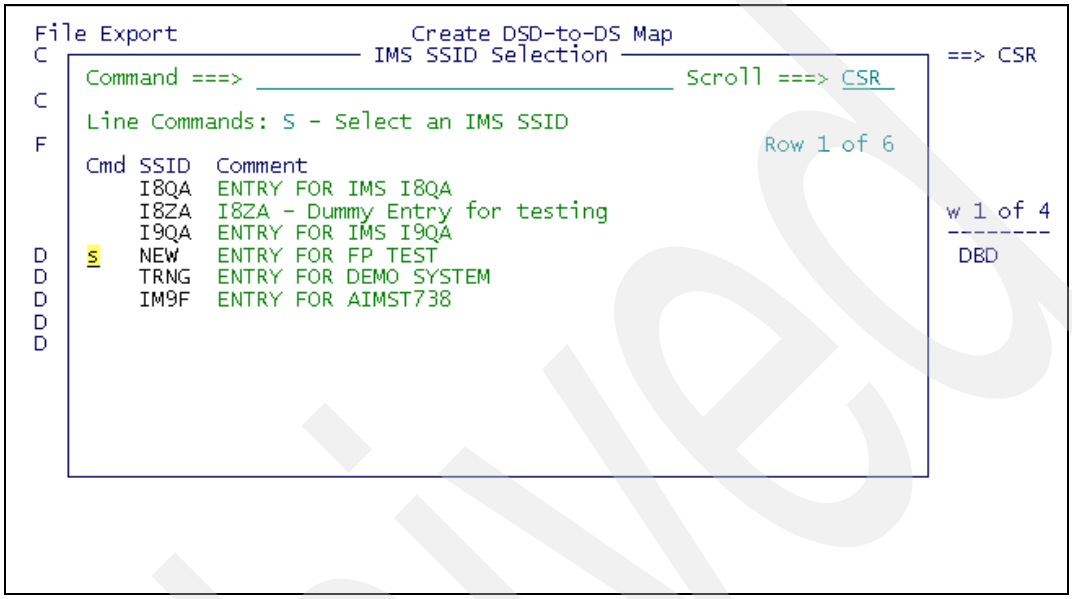


Figure 22-57 Select the target IMS SSID destination to load the extracted data

The target, or destination, DSD-to-DS Map displays the target IMS SSID as shown in Figure 22-58. If the IMS DBD names on the Target IMS SSID are the same as the Source IMS DBDs, then no edits to the IMS DBDs have to be made. However, if the IMS DBD names on the Target IMS SSID are different than the names of the Source IMS DBDs, edit them now.

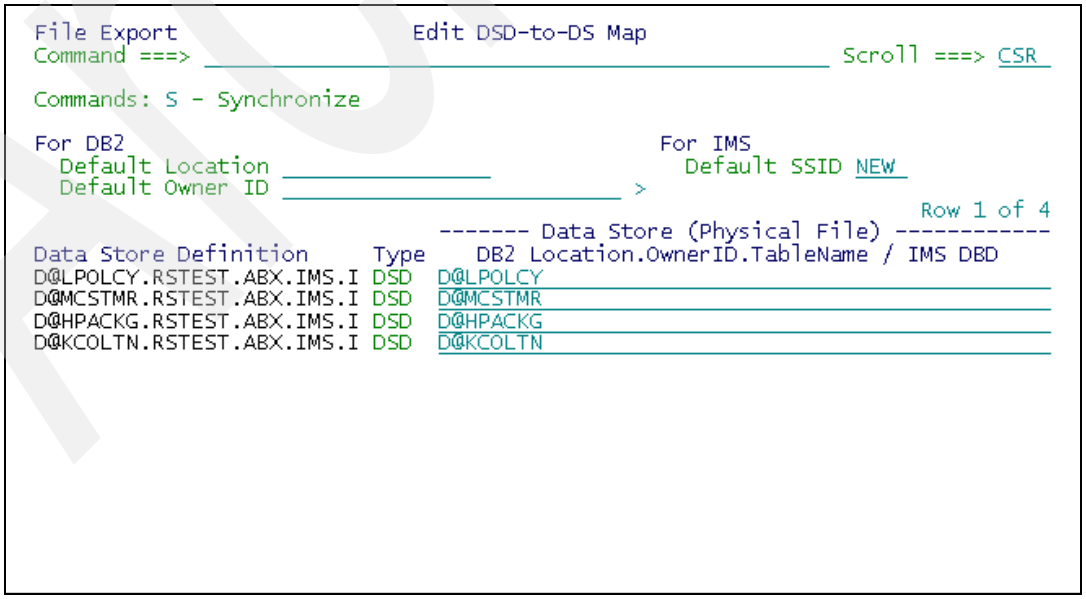


Figure 22-58 The IMS DBDs on the right hand side can be edited if they differ on the target system

22.4.2 Creating an extract and load request

Create an IMS related data Extract Request using the objects we just defined.

Step 1: From the File Export Primary Option menu, select option **3** to create a Multiple Data Store Extract and Load, as shown in Figure 22-59. Press Enter.

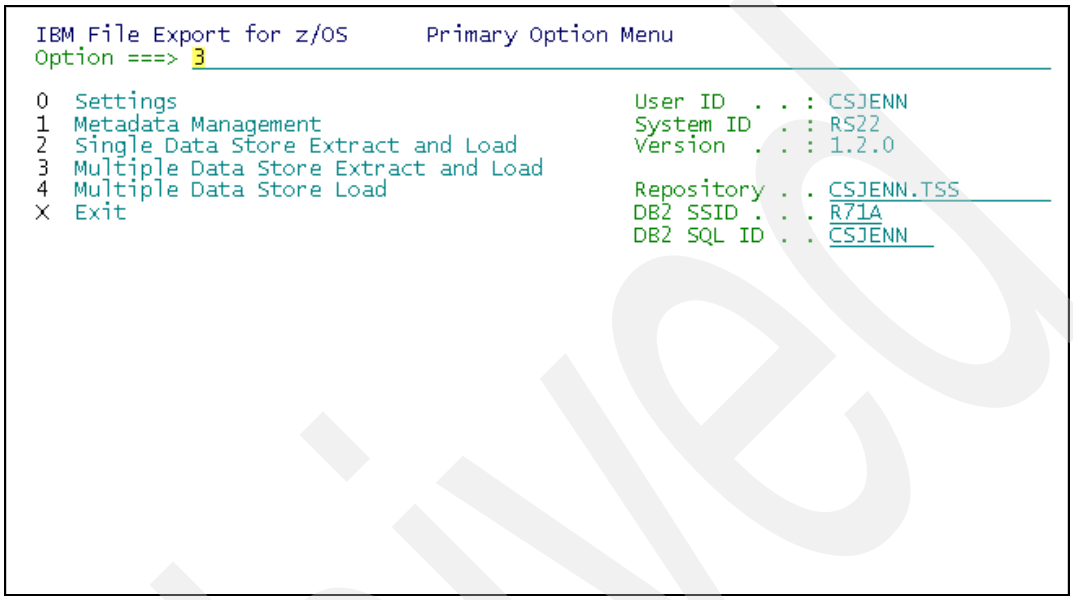


Figure 22-59 Create an Extract and Load Request for the related IMS data

Step 2: Provide a name for the Extract Request and an optional description. Specify the Related Subset of objects to extract data from. Provide the name of the DSD-to-DS Map for the source Segments, and the DSD-to-DS Map for the target segments (if a load is to be done). The value in the field Extract File Name is used to define the load control dataset and the intermediate datasets where the extracted data is held until it is loaded into the target segments.

When all fields are filled in, type `C` on the command line to create the Extract Request as shown in Figure 22-60. Press Enter.

```
File Export          Manage Multi-Data Store Extract Request
Command ===> c

Commands: E - Edit  C - Create  D - Delete  L - List Extract Requests

Extract Request Name  . . . IMSLR
Extract Request Description IMS LOGICAL RELATIONSHIP EXTRACT

Related Subset Name . . . . RSS IMSLR                                    >
DSD-to-DS Map Name  . . . . DDM IMSLR                                    >

Extract Parameters
  Extract File Name . . . . CSJENN.IMSLR

If Loading Data
  Destination DSD-to-DS Map DDM IMSLR TARGET                             >
```

Figure 22-60 Creating a multi-data store IMS extract

Step 3: In Figure 22-61, the fields under Generated JCL file tell File Export where to store the generated Extract and Load Request JCL. If the JCL dataset in the field Data Set Name does not exist, then File Export creates it as a FB 80 PDS. An existing dataset name can also be provided.

The value in field Member Name is the PDS member name of the Extract Request JCL. The field Load Step Member Name is the PDS member name of the Load Request, if a separate job for the Load is requested.

Set Generate Load Step to **Y** so that a Load Request is generated, and Review JCL to **Y** to receive an ISPF edit session to display the generated Extract and Load Request JCL.

```
File Export        Multi-Data Store Extract: IMS Destination        Top of data
Option ===>

Extract Request Name . : IMSLR
                                                                    More:     +
Related Subset Name  . : RSS IMSLR
DSD-to-DS Map Name   . : DDM IMSLR

Generated JCL file
  Data set name  . . . . CSJENN.ABX.PTF118.JCL
  Member name  . . . . . IMSLR     Blank or pattern for member list
  Load step member name  IMSLRL

Generate Load Step . . . Y         Y-Yes, N-No

Review JCL . . . . . . . Y         Y-Yes, N-No
Submit JCL . . . . . . . N         Y-Yes, N-No

JCL Management
  1  STEPLIBs
  2  JCL to precede extract execution
  3  JCL to follow extract execution

IMS-specific Processing Specifications
```

Figure 22-61 IMS-Specific processing options

Step 4: Scroll down the panel to see more options, then select option **5** to specify IMS Target SSID processing options as shown in Figure 22-62. Press Enter.

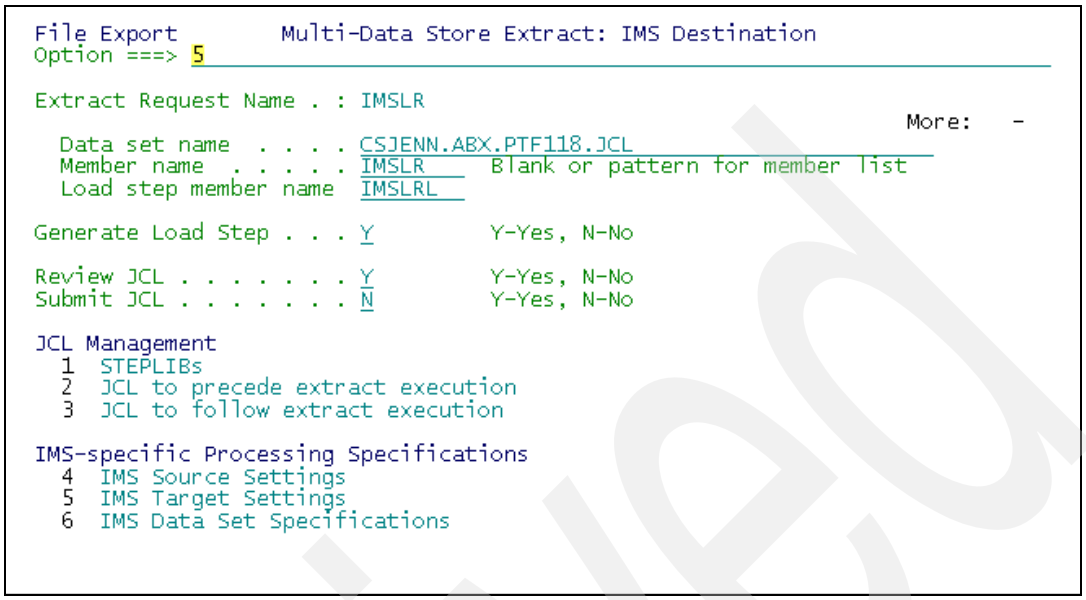


Figure 22-62 IMS Target object settings

Step 5: Type S on the Command line to see a list of available IMS SSID to select as the target. Press Enter. A pop-up window displays the IMS SSIDs available to be selected as the target IMS SSID. Select an IMS subsystem, as shown in Figure 22-63, and press Enter.

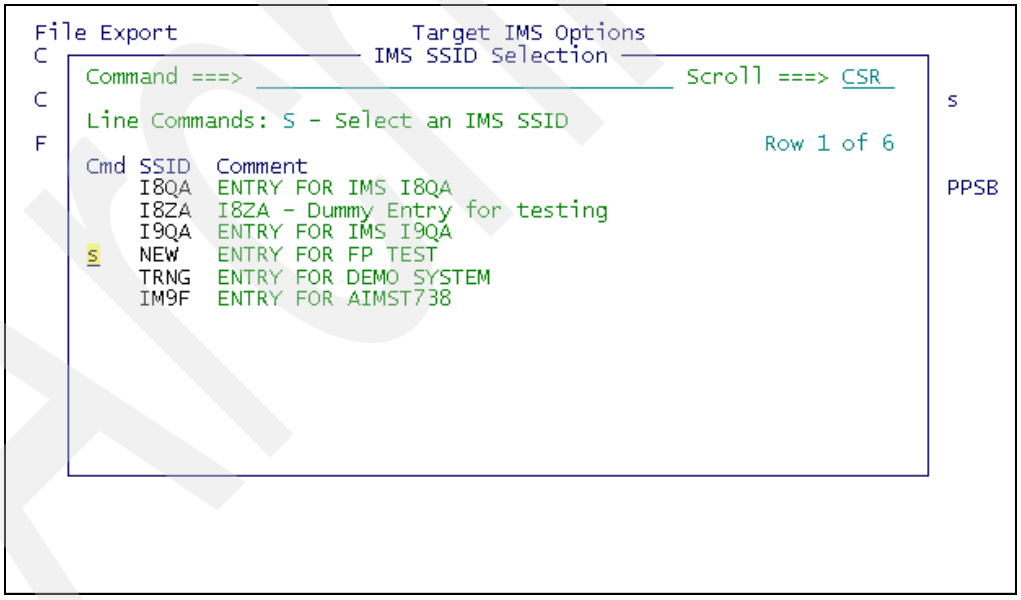


Figure 22-63 Select a target IMS SSID

Any necessary edits to the IMS region processing options can be made on this panel as shown in Figure 22-64, and only affects this extract. Use the End command (typically PF3) when any edits are completed to return to the Multi-Data Store Extract Request panel.

```
File Export                  Target IMS Options
Command ===>

Commands: S - Select Target IMS SSID  R - Set Region Processing Options

File Export IMS processing Options:
  SSID Name . . . . . . . . : NEW        Get request PROCOPT . . . . G
  Use Dynamic PSBs  . . . . . Y          Dynamic PSB name prefix . . FPPSB
  Dynamic PSB name suffix . . 9          DLT Duplicate Segs  . . . . Y
  Alternate DFSRRC00 name . . ________

  PSB Macro DSN . . . . IMS.V810.ADFSMAC
  DBD library DSN . . . ABX.FP.JULY2905.DBDLIB
  DOPT ACBLIB DSN . . . RSTEST.ABX.IMS.I8QA.DOPTLIB
  Reslib1 DSN . . . . . ABX.FP.JULY2905.RESLIB
  Reslib2 DSN . . . . . ABX.FP.TARGET.MDALIB
  Reslib3 DSN . . . . . RSTEST.ABX.IMS.I9QA.EXITLIB
  DFSVSAMP DSN/MBR  . . RSTEST.ABX.IMS.I8QA.PROCLIB(DFSVSAMP)
```

Figure 22-64 Target IMS SSID processing options can be edited if necessary

Step 6: On the Multi-Data Store Extract Request panel, select option **6** to provide dataset parameters for the File Export unload-format dataset. The File Export unload-format dataset is the interim staging dataset where the IMS data is held before loading as shown in Figure 22-65. Press Enter.

```
File Export       Multi-Data Store Extract: IMS Destination
Option ===> 6

Extract Request Name . : IMSLR
                                                              More:   -
  Data set name  . . . . CSJENN.ABX.PTF118.JCL
  Member name  . . . . . IMSLR     Blank or pattern for member list
  Load step member name  IMSLRL

Generate Load Step . . . Y         Y-Yes, N-No

Review JCL . . . . . . . Y         Y-Yes, N-No
Submit JCL . . . . . . . N         Y-Yes, N-No

JCL Management
  1  STEPLIBs
  2  JCL to precede extract execution
  3  JCL to follow extract execution

IMS-specific Processing Specifications
  4  IMS Source Settings
  5  IMS Target Settings
  6  IMS Data Set Specifications
```

Figure 22-65 Specify IMS processing dataset parameters

Specify **E** next to the dataset name for which you want to edit the dataset allocation parameters as shown in Figure 22-66. Press Enter.

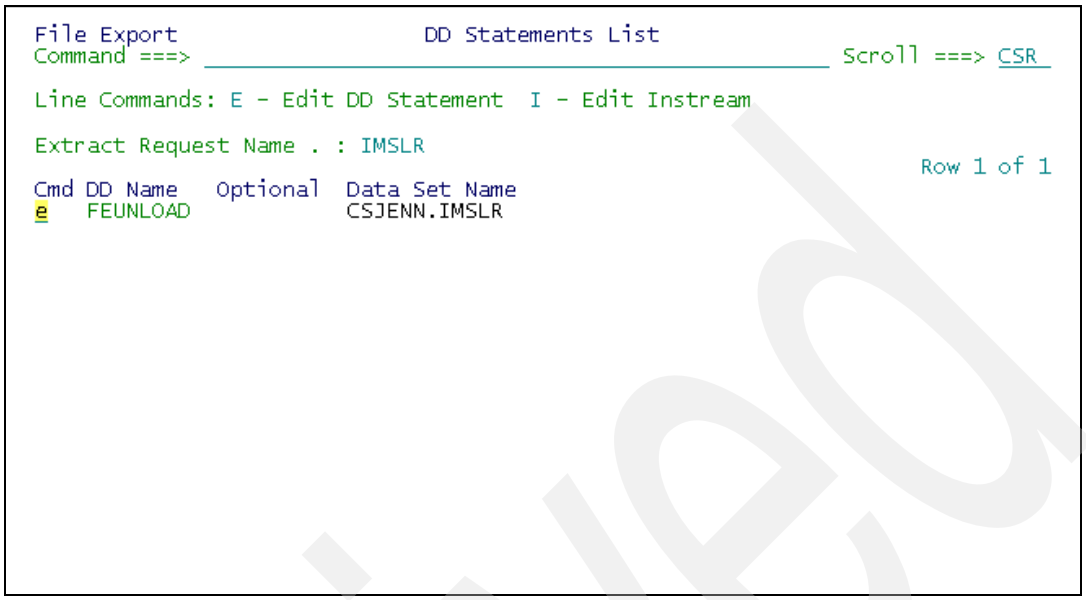


Figure 22-66 Edit the FEUNLOAD dataset to add allocation parameters

Option **W** invokes the Dataset Allocation Wizard to assist in specifying dataset parameters as shown in Figure 22-67. Press Enter.

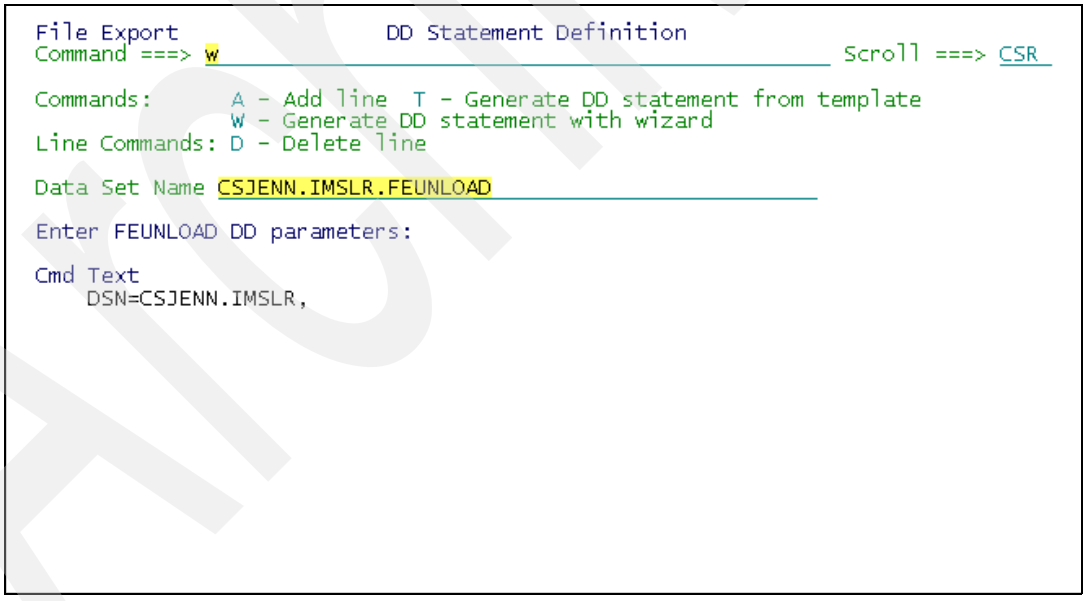


Figure 22-67 Use the Wizard to specify the FEUNLOAD dataset parameters

Provide the sizing parameters for the FEUNLOAD dataset as shown in Figure 22-68. Press PF3.

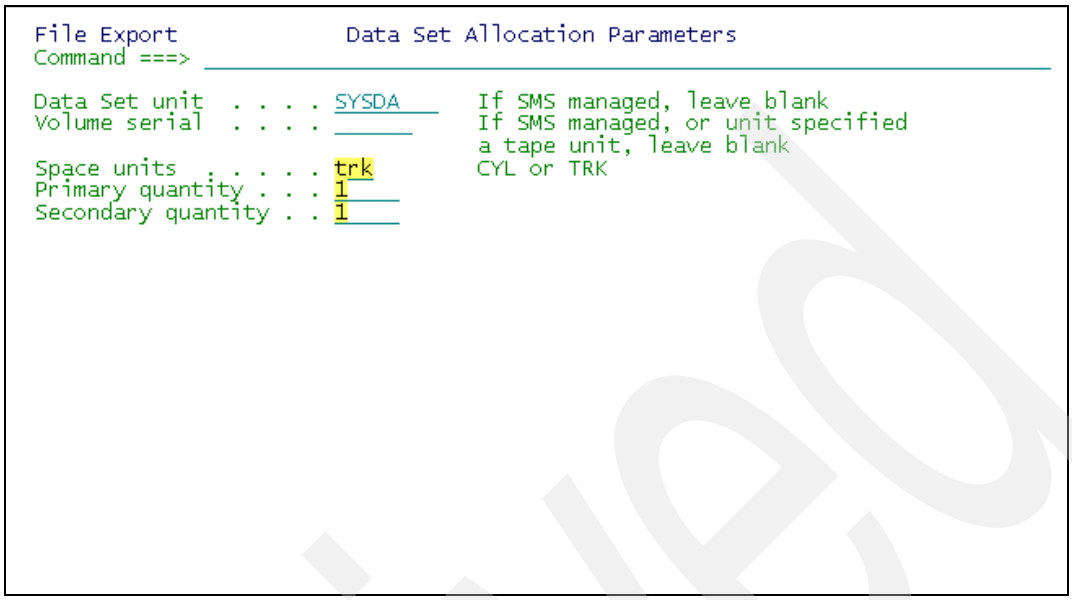


Figure 22-68 Specify the sizing parameters that are required to create the FEUNLOAD dataset.

The DD statement definition now reflects the sizing attributes that are generated into the Extract Request JCL as shown in Figure 22-69. Use the End command (typically PF3) to return to the Multi-Data Store Extract Request panel.

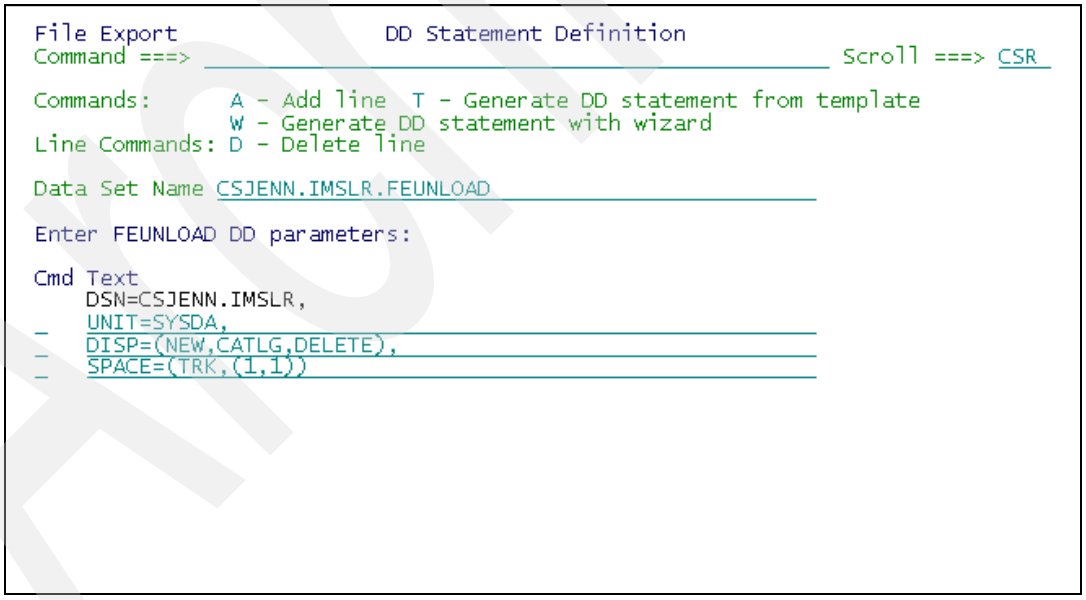


Figure 22-69 The completed DD definition

Step 7: Press Enter to generate Extract and Load Request JCL as show in Figure 22-70.

```
File Export        Multi-Data Store Extract: IMS Destination
Option ===>

Extract Request Name . : IMSLR
                                                                More:     +
Related Subset Name  . : RSS IMSLR
DSD-to-DS Map Name   . : DDM IMSLR

Generated JCL file
  Data set name  . . . . CSJENN.ABX.PTF118.JCL
  Member name  . . . . . IMSLR     Blank or pattern for member list
  Load step member name  IMSLRL

Generate Load Step . . . Y         Y-Yes, N-No

Review JCL . . . . . . . Y         Y-Yes, N-No
Submit JCL . . . . . . . N         Y-Yes, N-No

JCL Management
  1  STEPLIBs
  2  JCL to precede extract execution
  3  JCL to follow extract execution

IMS-specific Processing Specifications
```

Figure 22-70 Press Enter to generate JCL

The first JCL job to be generated is the Extract Request JCL as shown in Figure 22-71. In the generated Extract Request JCL, there are RESULTSET keywords that refer to the extracted data for a given segment. There are also CHILD OF RESULTSET*n* syntax that indicate the object referred to in the RESULTSET statement are children of a previous RESULTSET whose extraction is dependent upon data from a parent object.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CSJENN.ABX.PTF118.JCL(IMSLR) - 01.00            Columns 00001 00072
Command ===>                                                Scroll ===> CSR
****** ******************************** Top of Data ********************************
000001 //IMSLR   JOB CSJENN,CLASS=A,NOTIFY=&SYSUID,
000002 //       MSGCLASS=X,MSGLEVEL=(1,1)
000003 //*
000004 //*
000005 //STEP1    EXEC PGM=ABXUTIL
000006 //STEPLIB  DD   DISP=SHR,DSN=RSQA.ABX120.IBMTAPE.SABXLOAD
000007 //         DD   DISP=SHR,DSN=RSRTE.XML.HXML180.SIXMLOD1
000008 //         DD   DISP=SHR,DSN=CSWILK.ABX.GLOBAL.INSTLBLK.V2
000009 //*
000010 //*
000011 //ABXOUT   DD   DSN=CSJENN.IMSLR.FEUNLOAD,
000012 //             UNIT=SYSDA,
000013 //             DISP=(NEW,CATLG,DELETE),
000014 //             SPACE=(TRK,(1,1))
000015 //ABXPRINT DD   SYSOUT=*
000016 //ABXCTL   DD   *
000017 OPTIONS((DLI, FEUNLOAD), DECIMAL, PERCENT, NOQUIET);
000018 IMSID(I8QA);
000019 SOURCEDBD(D@LPOLCY);
```

Figure 22-71 The generated IMS extract job

When the End command is pressed to exit from the Extract Request JCL, the Load Request JCL is generated and displayed as shown in Figure 22-72. The FEUNLOAD dataset from the Extract Request JCL is utilized in the Load Request JCL as input. The target IMS SSID is displayed in the Control Syntax indicating to File Export where the target DBDs are located.

The INCLUDE statements contain the Source DBD names as well as the corresponding Target DBD names.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CSJENN.ABX.PTF118.JCL(IMSLRL) - 01.00            Columns 00001 00072
Command ===>                                                 Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 //IMSLRL  JOB CSJENN,CLASS=A,NOTIFY=&SYSUID,
000002 //       MSGCLASS=X,MSGLEVEL=(1,1)
000003 //*
000004 //*
000005 //*
000006 //STEPLOAD EXEC PGM=ABXFELD0
000007 //STEPLIB  DD   DISP=SHR,DSN=RSQA.ABX120.IBMTAPE.SABXLOAD
000008 //         DD   DISP=SHR,DSN=RSRTE.XML.HXML180.SIXMLOD1
000009 //         DD   DISP=SHR,DSN=CSWILK.ABX.GLOBAL.INSTLBLK.V2
000010 //*
000011 //ABXIN    DD   DISP=SHR,DSN=CSJENN.IMSLR.FEUNLOAD
000012 //ABXPRINT DD   SYSOUT=*
000013 //ABXCTL   DD   *
000014 OPTIONS(DLI);
000015 IMSID(NEW);
000016 INCLDBD((D@LPOLCY,D@LPOLCY));
000017 INCLDBD((D@MCSTMR,D@MCSTMR));
000018 INCLDBD((D@HPACKG,D@HPACKG));
000019 INCLDBD((D@KCOLTN,D@KCOLTN));
```

Figure 22-72 The matching generated IMS load job

22.5 Creating an IMS extract using application relationships

Application Relationships can be defined between IMS objects. Application Relationships can also be used in conjunction with logical relationships imported from IMS Database schema definitions. That is to say, Application Relationships can be created between two unrelated IMS DBDs and used with other Application Relationships based on logical relationships defined in IMS Database schema definitions by using the Import function.

This extract utilizes the IMS Option Block described in the beginning of this chapter. Therefore, setting IMS Options is not discussed.

22.5.1 Defining the required repository objects

Task 1: Create Application Relationships to be used in an Extract and Load Request.

Step 1: From the File Export Primary Option Menu, select option **1** Metadata Management to create an Application Relationship. Press Enter.

Step 2: Select option **3** Manage Application Relationships from the Metadata Management panel as shown in Figure 22-73. Press Enter.

```
File Export                    Metadata Management
Option ===> 3

Repository Management
  0  Manage Repository                          Repository CSJENN.TSS

Extract-Related Metadata
  1  Manage Record Layouts (typically for non-relational data)
  2  Manage Data Store Definitions (one per Data Store)
  3  Manage Application Relationships
  4  Manage Related Subsets
  5  Manage Data Store Definition-to-Data Store (DSD-to-DS) Maps
```

Figure 22-73 Select option 3 to create an Application Relationship

Step 3: Provide a name for the Application Relationship. Now specify the parent object. Tab down to the subfield Data Store Definition Name type an asterisk. In the subfield Data Store definition Type, select option **I** to retrieve IMS Data Store Definitions as shown in Figure 22-74. Press Enter.

```
File Export           Manage Application Relationships
Command ===>
                                                              More:     +
Commands: E - Edit  C - Create  D - Delete  L - List Application Relationships

Relationship Name RSS IMSAR                                             >

Parent Data Store Definition  (mutually exclusive, specify only one)
  If using DB2 Table
    DB2 Location . . . . . .
    DB2 OwnerID  . . . . . .                                            >
    DB2 Table Name . . . . .                                            >
  If using Record Layout
    Record Layout Name . . . . . .                                      >
    Record Layout Type . . . . . _  I-IMS, D-DB2
  If using Data Store Definition
    Data Store Definition Name . . *                                    >
    Data Store Definition Type . : I  I-IMS
    Record Layout Name . . . . . .                                      >
    If using IMS Data Store Definition
      IMS Segment  . . . . . .

Child Data Store Definition   (mutually exclusive, specify only one)
  If using DB2 Table
```

Figure 22-74 An asterisk provides a list of available Data Store Definitions

Step 4: The Data Store Definition Selection panel displays all IMS Data Store Definitions, or DSDs. The DSDs that are displayed are those that were created in the previous section.

To create a Data Store Definition for an IMS DBD that is not in the list, see the section titled “Defining the required repository objects” on page 571 for detailed instructions on how to create a new DSD.

The **FIND** command can be used to find a specify IMS DSD. Select the IMS DSD to use as the parent in the Application Relationship as shown in Figure 22-75. Press Enter.

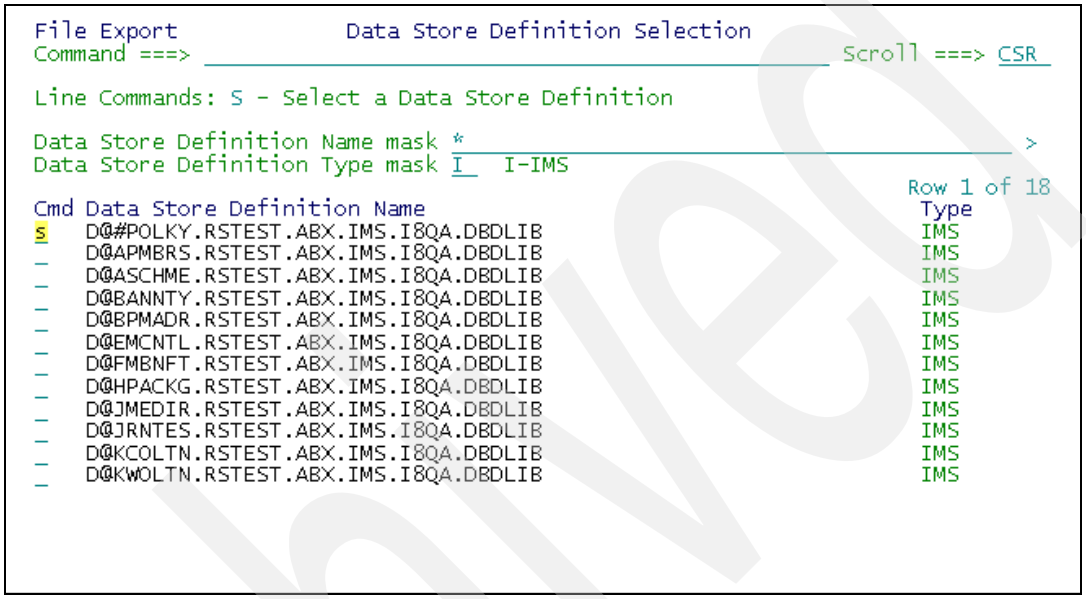


Figure 22-75 Select the IMS DBD to use as the parent in the Application Relationship

Step 5: Each IMS DBD Segment has a default Record Layout, that is, the definition of the Segment as it is defined in IMS. Select the Default Record Layout as shown in Figure 22-76. Press Enter.

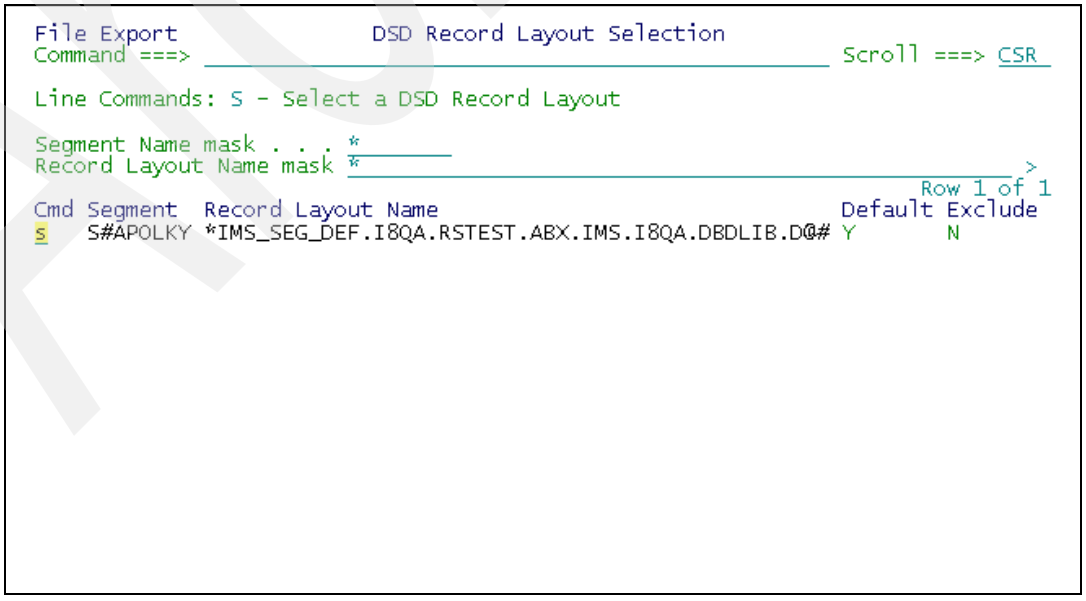


Figure 22-76 Select the Record Layout for the segment to use as the parent

The parent IMS Data Store Definition contains the corresponding Record Layout that is used for the extract. Now specify a child IMS DBD to use in the Application Relationship.

Step 6: On the Manage Application Relationship panel, scroll down the panel by pressing PF8. The Child Data Store Definition fields are displayed. Tab down to subfield Data Store Definition Name and enter an asterisk to search for the child DSD from a list as shown in Figure 22-77. Press Enter.

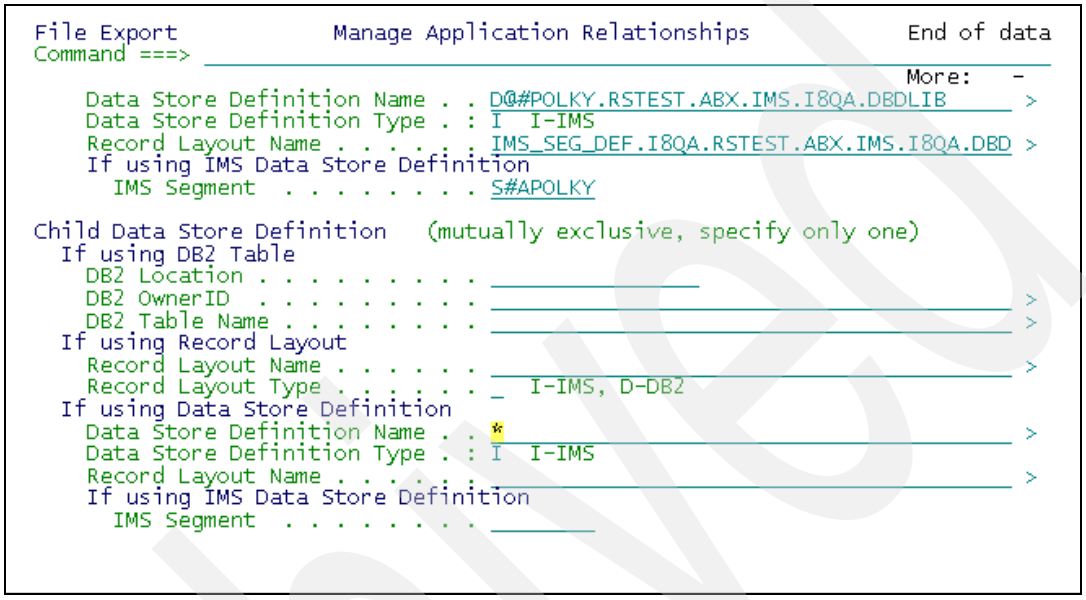


Figure 22-77 Now specify the child object to use in the Application Relationship

Step 7: The same Data Store Definition Selection panel is displayed. Select the child IMS DSD to use as the child object in the Application Relationship as shown in Figure 22-78. Press Enter.

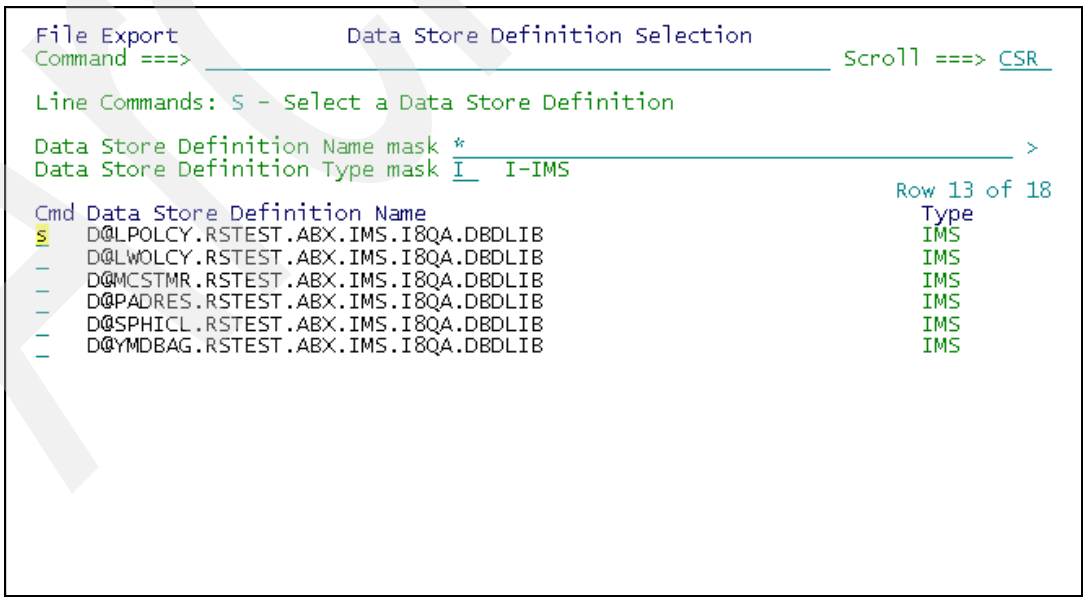
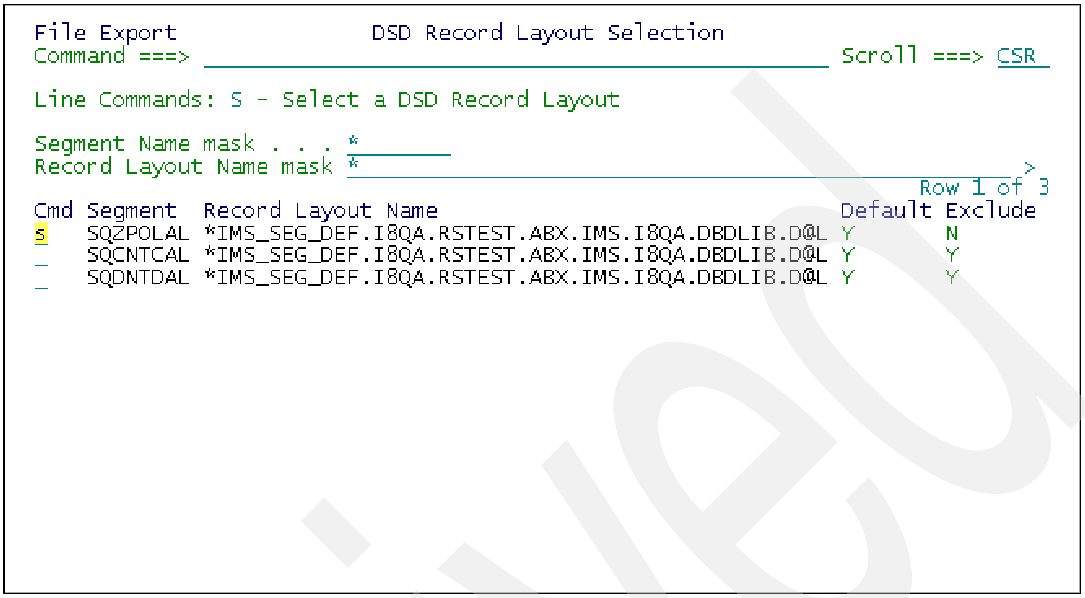


Figure 22-78 Select the Child IMS DBD to use in the Application Relationship

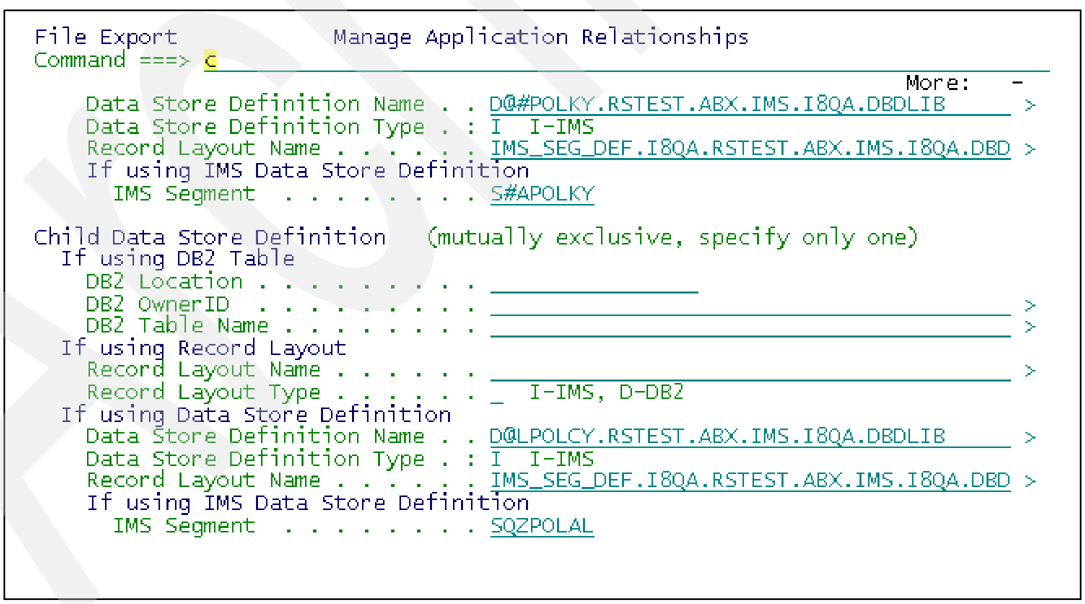
Step 8: Each IMS DBD Segment has a default Record Layout, that is, the definition of the Segment as it is defined in IMS. Select the Default Record Layout as shown in Figure 22-79. Press Enter.



File Export DSD Record Layout Selection
Command ===> Scroll ===> CSR
Line Commands: S - Select a DSD Record Layout
Segment Name mask . . . *
Record Layout Name mask *
Row 1 of 3
Cmd Segment Record Layout Name Default Exclude
S SQZPOLAL *IMS_SEG_DEF.I8QA.RSTEST.ABX.IMS.I8QA.DBDLIB.D@L Y N
SQCNTCAL *IMS_SEG_DEF.I8QA.RSTEST.ABX.IMS.I8QA.DBDLIB.D@L Y Y
SQDNTDAL *IMS_SEG_DEF.I8QA.RSTEST.ABX.IMS.I8QA.DBDLIB.D@L Y Y

Figure 22-79 Select the default Record Layout for the child segment

Step 9: With both the Parent Data Store Definition and Child Data Store Definition filled in, type a **C** on the Command line to create the Application Relationship as shown in Figure 22-80. Press Enter.



File Export Manage Application Relationships
Command ===> C
More: -
Data Store Definition Name . . D@#POLKY.RSTEST.ABX.IMS.I8QA.DBDLIB
Data Store Definition Type . : I I-IMS
Record Layout Name . . . . . . IMS_SEG_DEF.I8QA.RSTEST.ABX.IMS.I8QA.DBD
If using IMS Data Store Definition
IMS Segment . . . . S#APOLKY
Child Data Store Definition (mutually exclusive, specify only one)
If using DB2 Table
DB2 Location . . . .
DB2 OwnerID . . . .
DB2 Table Name . . . .
If using Record Layout
Record Layout Name . . . . . .
Record Layout Type . . . . . . _ I-IMS, D-DB2
If using Data Store Definition
Data Store Definition Name . . D@LPOLCY.RSTEST.ABX.IMS.I8QA.DBDLIB
Data Store Definition Type . : I I-IMS
Record Layout Name . . . . . . IMS_SEG_DEF.I8QA.RSTEST.ABX.IMS.I8QA.DBD
If using IMS Data Store Definition
IMS Segment . . . . SQZPOLAL

Figure 22-80 Type a C on the Command line to create the Application Relationship

Upon entering the Create Application Relationship panel, you can specify the **S** command to Switch the view. Press Enter. The fields from both IMS DSDs are displayed side-by-side in columnar format.

Step 10: Numerically select the field names from each DSD or Record Layout to create the Application Relationship as shown in Figure 22-81. Use the End command (typically PF3) when the fields have been selected.

```
File Export              Create Application Relationship
Command ===> ____________________________________________ Scroll ===> CSR

Commands: U - Specify User Exits  S - Switch Display Mode

Parent DSD  . . . . . . . : D@#POLKY.RSTEST.ABX.IMS.I8QA.DBDLIB
Child DSD . . . . . . . . : D@LPOLCY.RSTEST.ABX.IMS.I8QA.DBDLIB
                                                          Row 1 of 7
Pos    Parent Field              Type  Pos    Child Field        Type
       *                         REC          *                  REC
1      PPACK#A                   CHAR  1      POLNOQZ            CHAR
       PCHAR#A                   CHAR         RENMOQZ            CHAR
                                              COLACQZ            PACK
                                              BRNCHQZ            CHAR
                                              AGENTQZ            CHAR
                                              ACINDQZ            CHAR
```

Figure 22-81 Switch Display Mode shows the fields in column format

Task 2: Create a Related Subset of objects that are both logically related and related through the Application Relationship that was just created. Remember that a Related Subset is a group of objects whose related data is extracted as a single request. Begin building the Related Subset from an IMS DSD that is the beginning object of an extract.

Step 1: Select option **4** from the Metadata Management panel to Manage Related Subsets as shown in Figure 22-82. Press Enter.

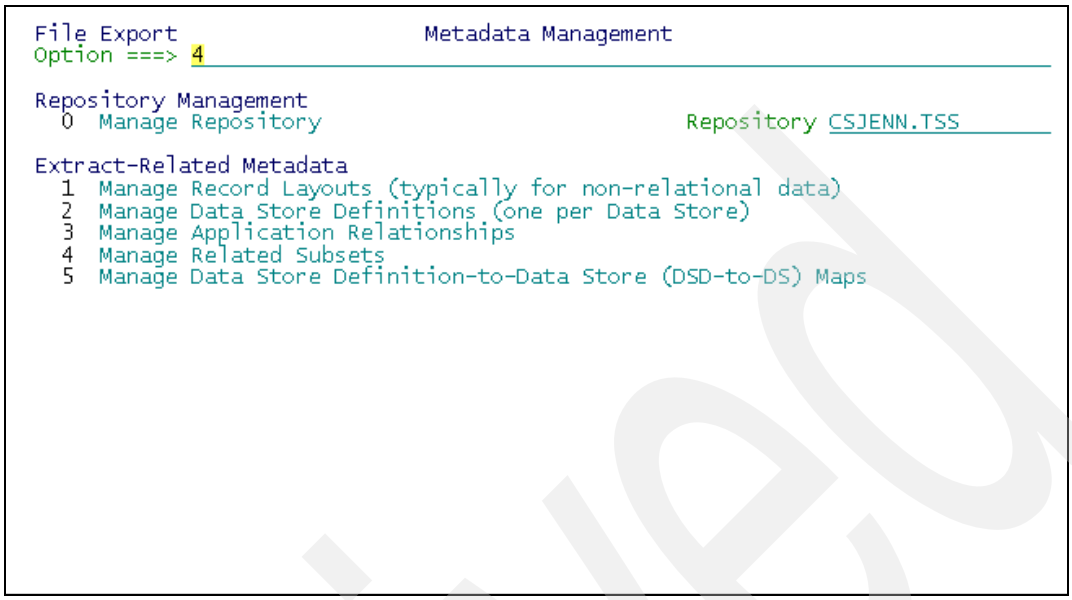


Figure 22-82 Create a Related Subset to contain IMS objects to extract from

Step 2: Provide a name for the Related Subset and type option C on the Command line to create it as shown in Figure 22-83. Press Enter.

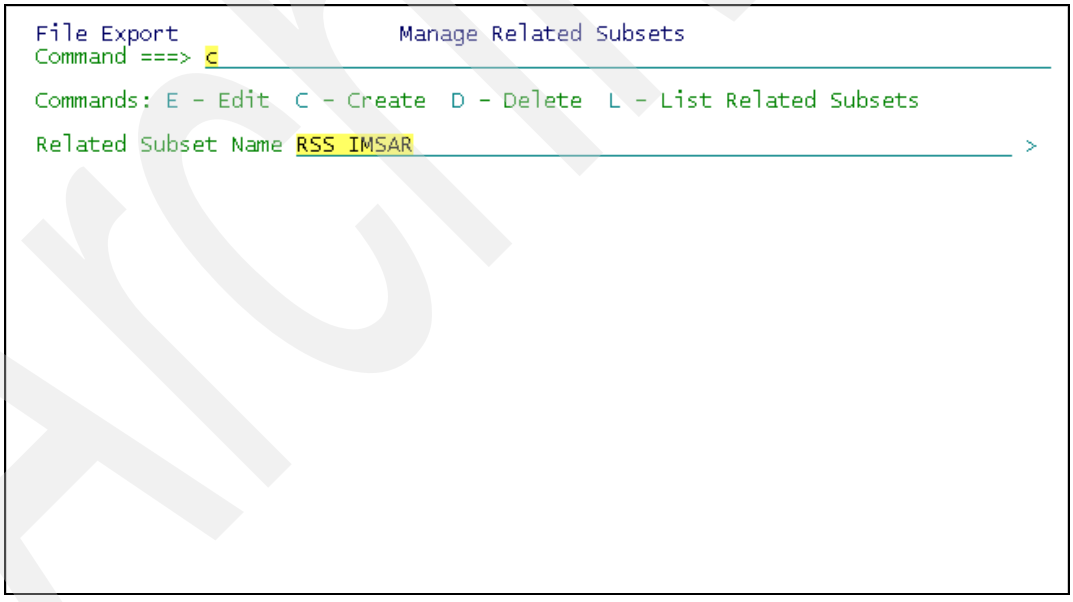


Figure 22-83 Specify a name for the Related Subset

Step 3: Specify option D in the field Start DSD Type to indicate that the Relates Subset is to be populated by Data Store Definitions. Wildcard the Data Store Definition subfield Data Store Definition Name with an asterisk to select the IMS DBD from a list. Specify the Data Store Definition Type as IMS by specifying **I** in the subfield as shown in Figure 22-84. Press Enter.

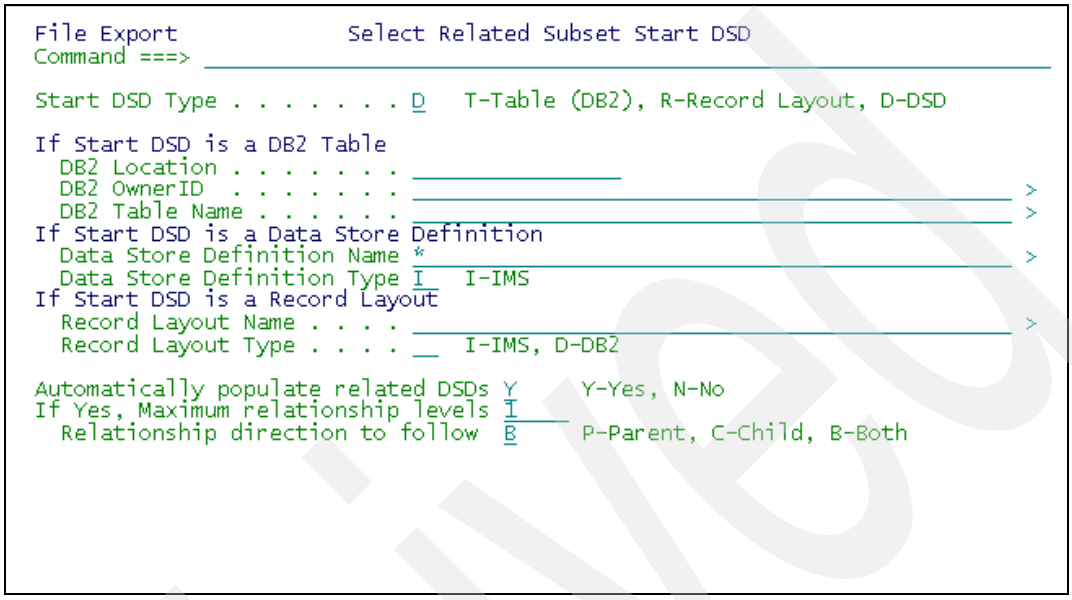


Figure 22-84 Start Type D creates a Related Subset from Data Store Definitions.

Step 4: A list of IMS-type Data Store Definitions is displayed allowing you to choose the starting IMS DSD (which correlates to an actual IMS DBD) from which to begin the extract. A value in the field Data Store Definition Name Mask can be utilized to narrow down the Data Store Definitions displayed in the list. Select the IMS DBD to use as the starting object in a Related subset as shown in Figure 22-85. Press Enter.

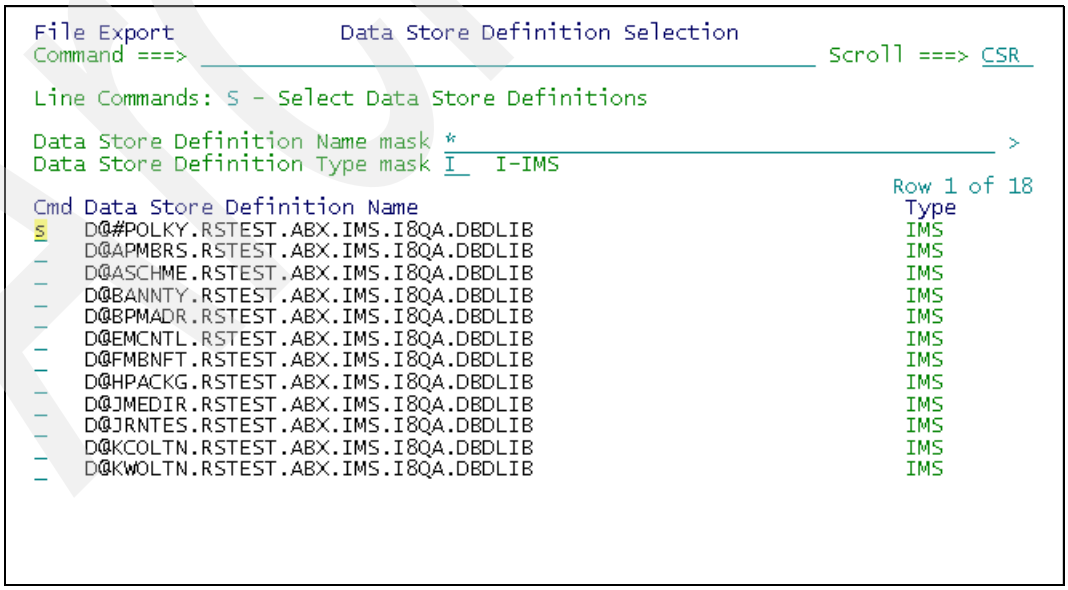


Figure 22-85 Select the Data Store Definition of the IMS DBD Segment to begin the extract from

Step 5: Those IMS DSDs related to the starting object by the Application Relationship are populated into the Related Subset. This list of objects comprises the set of objects that related data is extracted from as shown in Figure 22-86.

Note: Additional *unrelated* objects can be added to the list by specifying A on the Command line. Additional *related* objects can be added to the list by specifying Line Command G next to an object in the list.

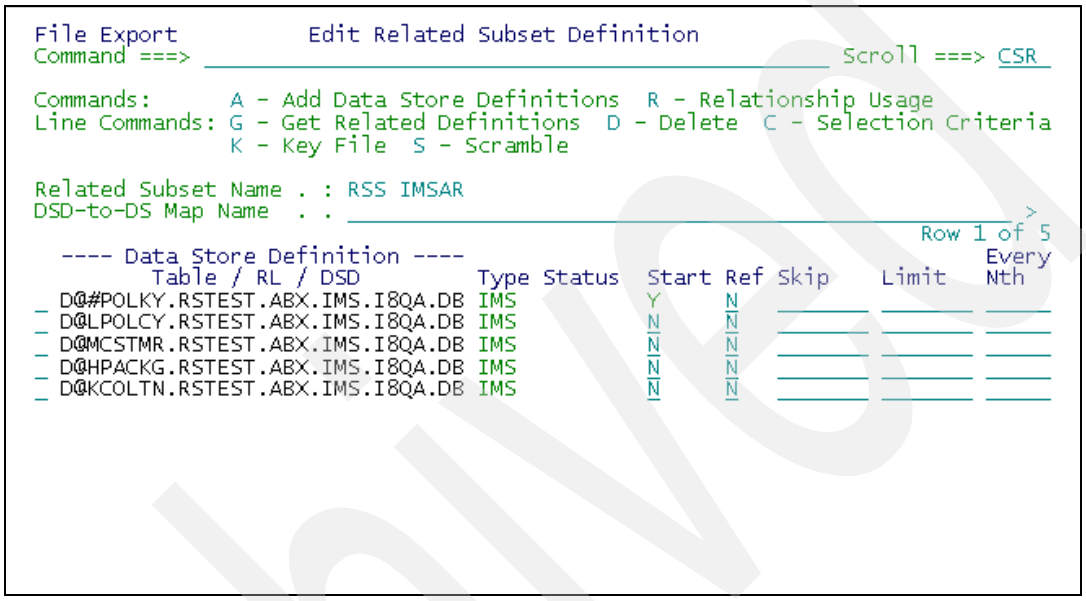


Figure 22-86 All related objects are populated into the Related Subset

Task 3: Create a Data Store Definition-to-Data Store Map or DDM for the Source IMS DBDs. Remember that a DDM maps logical structures to their physical file locations. It describes the structural definition of the object and tells File Export where to physically find the object's data.

Step 1: From the Metadata Management panel, select option **5** to create a Data Store Definition-to-Data Store Map that specifies the DBDs for each DSD in the Related Subset as shown in Figure 22-87. Press Enter.

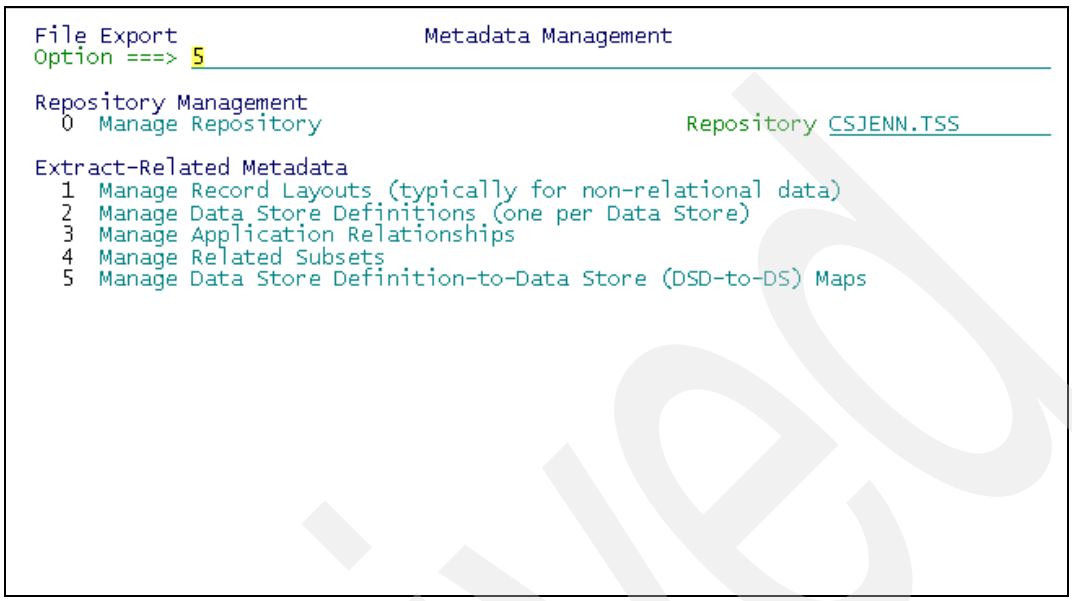


Figure 22-87 Source and Destination data maps must be created

Step 2: Specify a name for the DDM and specify which Related Subset to use. Type **C** on the Command line to create the Data Store Definition-to-Data Store Map as seen in Figure 22-88. Press Enter.

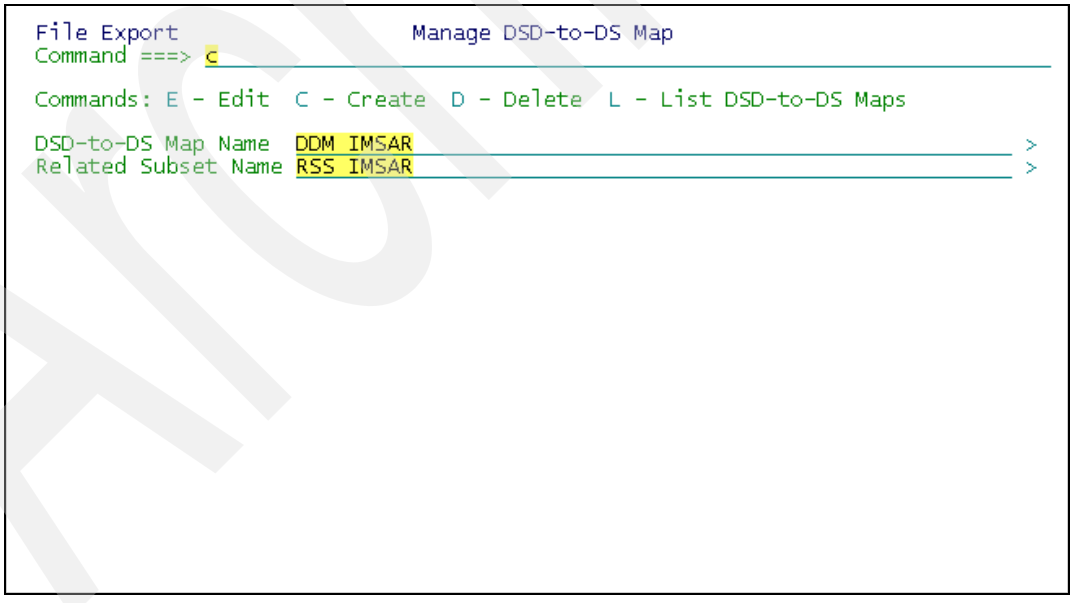


Figure 22-88 Provide a name for the Source DDM and use the newly created Related Subset

In Figure 22-89, the Create DSD-to-DS Map panel shows the Data Store Definition names on the left hand side of the panel and the corresponding DBD names on the right hand side of the panel. The Data Store Definitions are the logical structural definition of the data. The IMS DBD names are the physical location of the data.

This is how File Export knows how the data is defined, and where to find the actual data. DDMs can be used with more than one Related Subset, and a Related Subset can be used with more than one DDM depending on the related data that must be extracted.

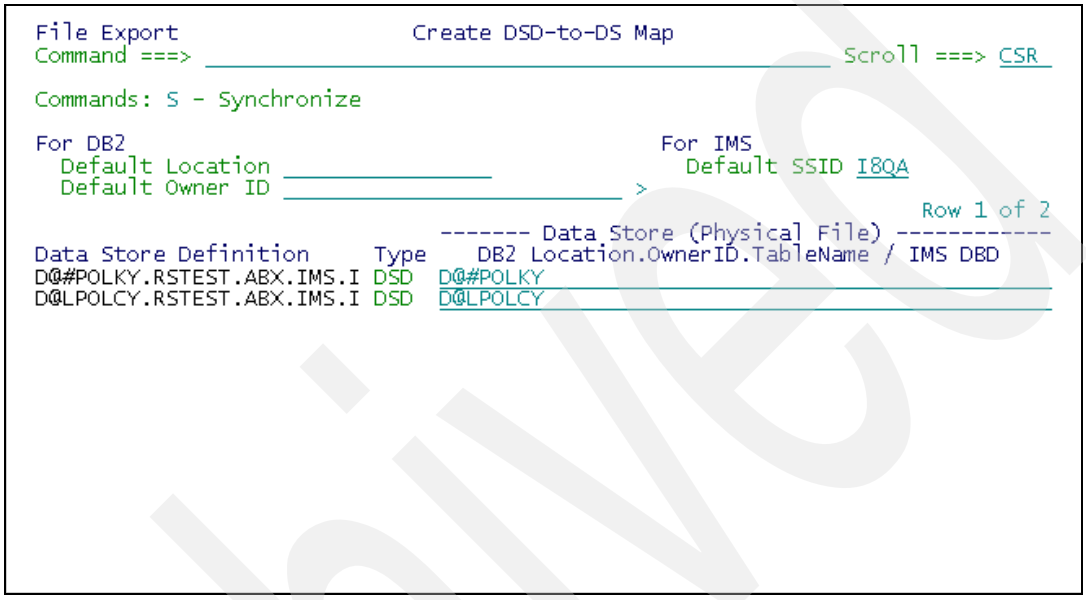


Figure 22-89 The structural definition is on the left, the physical location of the data is on the right

Task 4: Create a Data Store Definition-to Data Store Map for the Target IMS DBDs to tell File Export where to load the extracted data. The same Related Subset is used to define the Target DDM because the structure of the source and destination are the same.

Step 1: On the Manage DSD-to-DS Map panel, provide a name for the target DDM, and specify the same Related Subset of objects that was used to create the source DDM. Type **C** on the Command line, as shown in Figure 22-90, and press Enter.

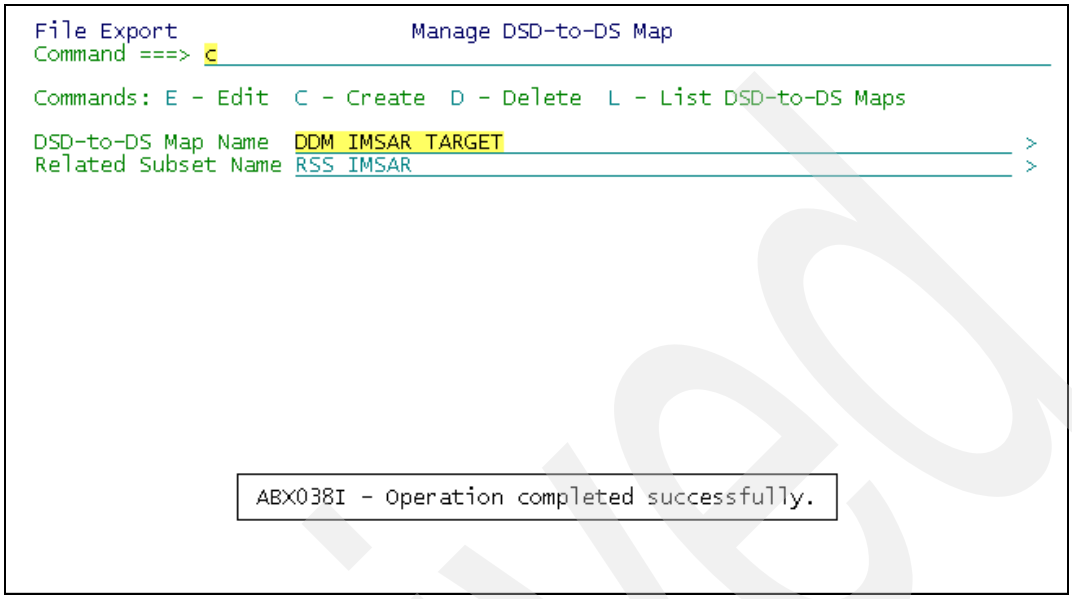


Figure 22-90 Create a DDM for the destination objects using the same Related Subset of data

Step 2: To change the Target IMS subsystem ID where the data is to be loaded, specify an asterisk in the Default SSID field for IMS as shown in Figure 22-91. Press Enter.

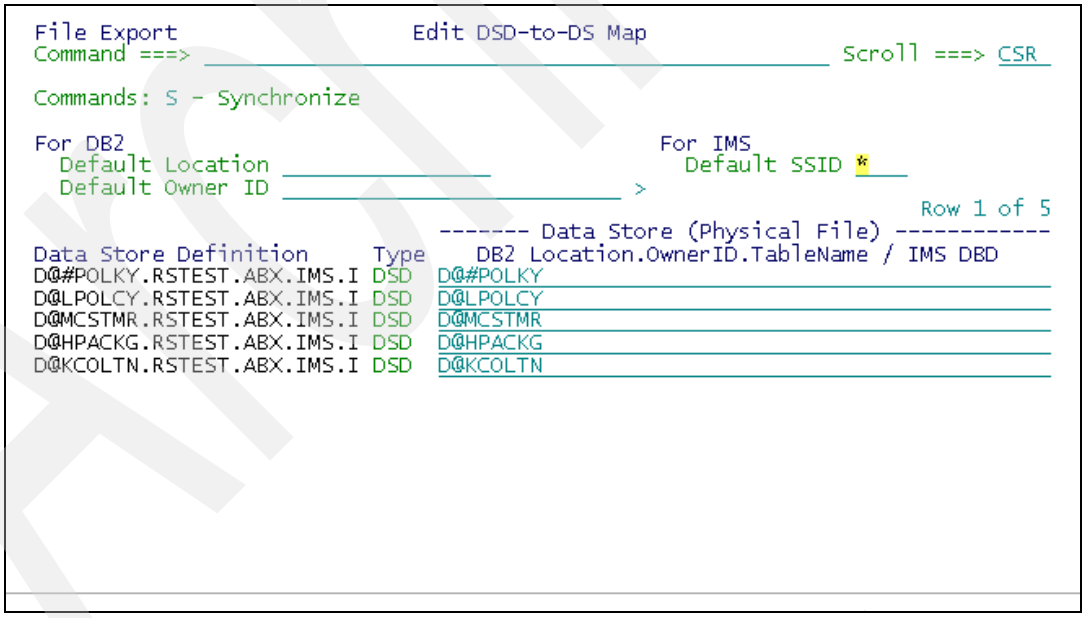


Figure 22-91 Another SSID can be specified as the target location

A pop-up window displays the available IMS SSIDs from which one can be selected as the target IMS SSID. The IMS subsystems and their corresponding region processing options are derived from the IMS Option Block assembled and linked into the ABX.SABXLOAD dataset. To add subsystems and / or change processing options, see the section titled “Customizing Option Blocks” on page 544 which explains assembling an IMS Option Block.

Select an IMS SSID as the destination as shown in Figure 22-92. Press Enter.

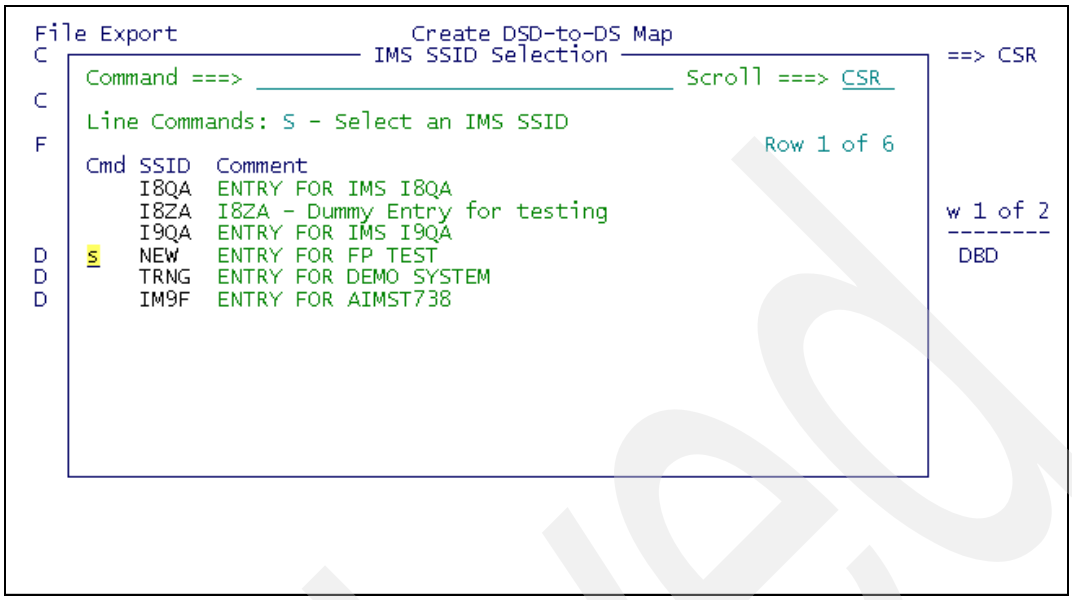


Figure 22-92 Select the Target IMS SSID destination to load the extracted data

The destination DSD-to-DS Map displays the target IMS SSID. If the IMS DBD names on the Target IMS SSID are the same as the Source IMS DBDs, then no edits to the IMS DBDs have to be made. However, if the IMS DBD names on the Target IMS SSID are different than the names of the Source IMS DBDs, edit them now as shown in Figure 22-93.

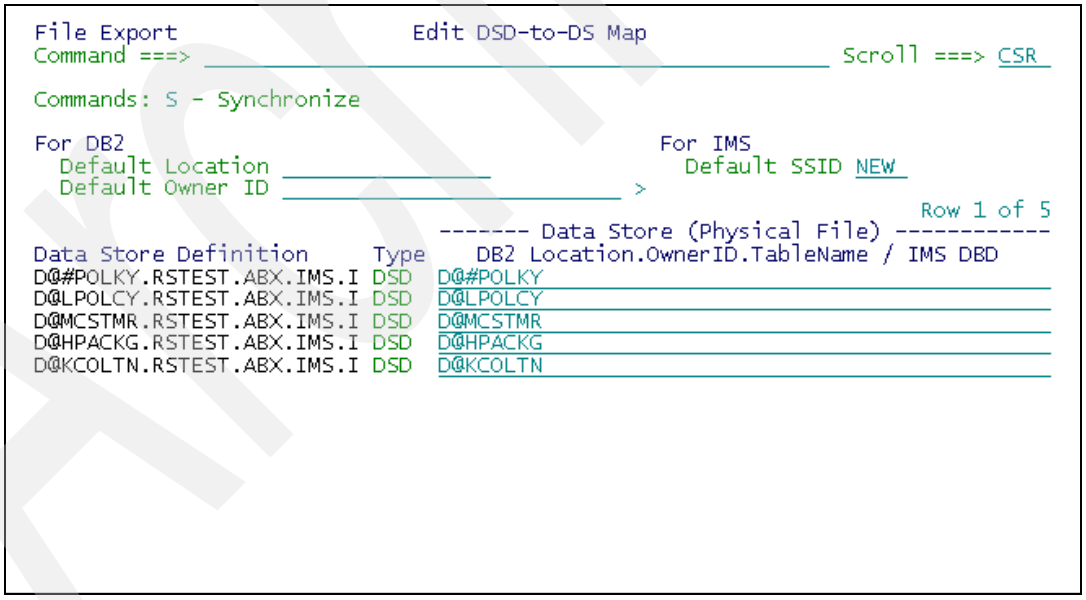


Figure 22-93 The IMS DBDs on the right hand side can be edited if they differ on the target system

22.5.2 Creating an extract and load request

Create an IMS related data extract using the Application Relationships that were created by importing the logical IMS relationships defined in the IMS Database schema.

Step 1: From the File Export Primary Option menu, select option **3** to create a Multiple Data Store Extract and Load as shown in Figure 22-94. Press Enter.

```
IBM File Export for z/OS      Primary Option Menu
Option ===> 3

0  Settings                                  User ID  . . : CSJENN
1  Metadata Management                       System ID  . : RS22
2  Single Data Store Extract and Load        Version  . . : 1.2.0
3  Multiple Data Store Extract and Load
4  Multiple Data Store Load                  Repository . . CSJENN.TSS
X  Exit                                      DB2 SSID . . . R71A
                                             DB2 SQL ID . . CSJENN
```

Figure 22-94 Create an Extract and Load Request for the related IMS data

Step 2: Provide a name for the Extract Request and an optional description. Specify the Related Subset of objects to extract data from. Provide the name of the DSD-to-DS Map for the source Segments, and the DSD-to-DS Map for the target segments if both an Extract Request and a Load Request is chosen. The value in the field Extract File Name is used to define the load control dataset and the intermediate datasets where the extracted data is held until it is loaded into the target segments.

When all fields are filled in, type `C` on the command line to create the Extract Request as shown in Figure 22-95. Press Enter.

```
File Export         Manage Multi-Data Store Extract Request
Command ===> c

Commands: E - Edit  C - Create  D - Delete  L - List Extract Requests

Extract Request Name  . . . IMSAR
Extract Request Description IMS LOGIAL RELATIONSHIPS W/APPL RELS

Related Subset Name . . . . RSS IMSAR                                  >
DSD-to-DS Map Name  . . . . DDM IMSAR                                  >

Extract Parameters
  Extract File Name . . . . CSJENN.IMSAR

If Loading Data
  Destination DSD-to-DS Map DDM IMSAR TARGET                           >
```

Figure 22-95 Create a multi-data store IMS extract

Step 3: In Figure 22-96, the fields under Generated JCL File tell File Export where to store the generated Extract and Load Request JCL. If the JCL dataset in the field Data Set Name does not exist, then File Export creates it as a FB 80 PDS. An existing dataset name can also be provided.

The value in field Member Name is the PDS member name of the Extract Request JCL. The field Load Step Member Name is the PDS member name of the Load Request, if a separate job for the Load is requested.

Set Generate Load Step to **Y** so that a Load Request is generated, and Review JCL to **Y** to receive an ISPF edit session to display the generated Extract and Load Request JCL.

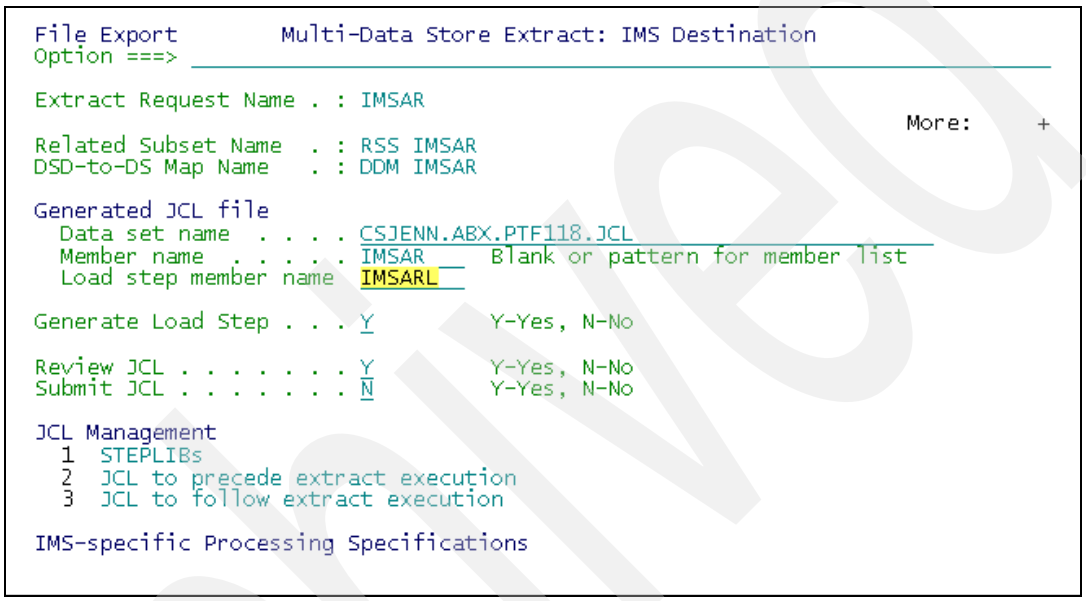


Figure 22-96 IMS-specific processing options

Step 4: Scroll down the panel to see additional options. Select option **5** to specify IMS Target SSID processing options as shown in Figure 22-97. Press Enter.

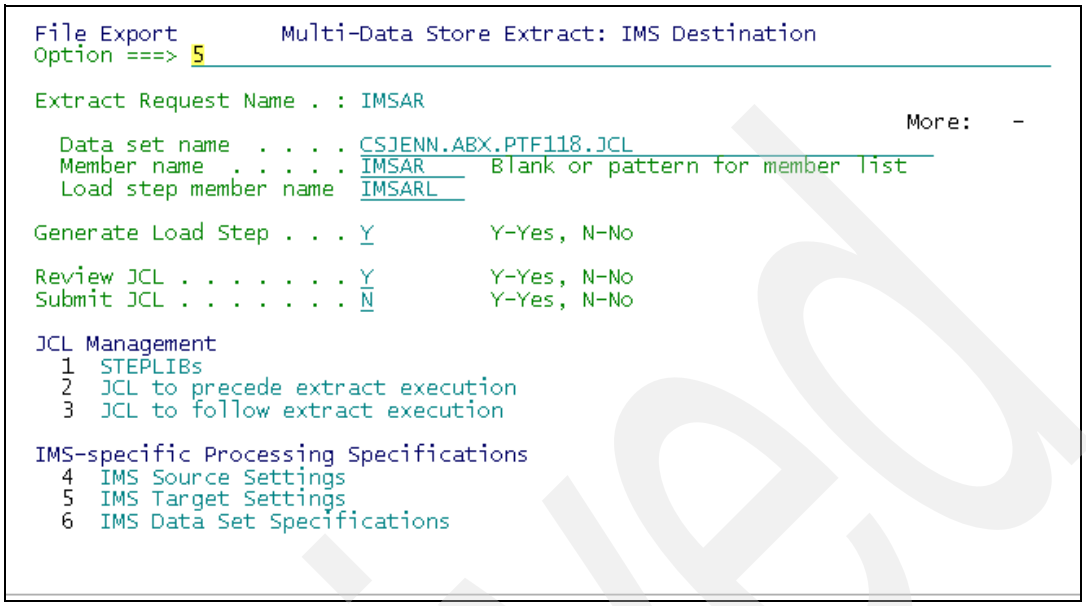


Figure 22-97 IMS target object settings

Step 5: Type S on the Command line to see a list of available IMS SSID to select as the target as shown in Figure 22-98. Press Enter.

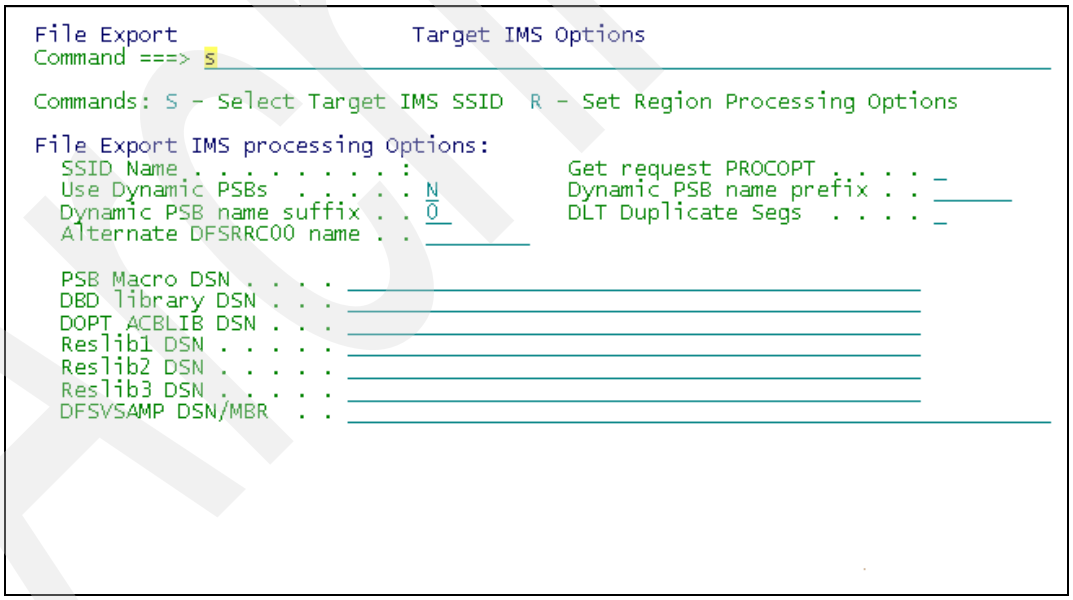


Figure 22-98 Type S to get a list of IMS SSIDs to load the extracted data to

A pop-up window displays the IMS SSIDs available for selection as the target IMS SSID. Select an IMS subsystem, as shown in Figure 22-99, and press Enter.

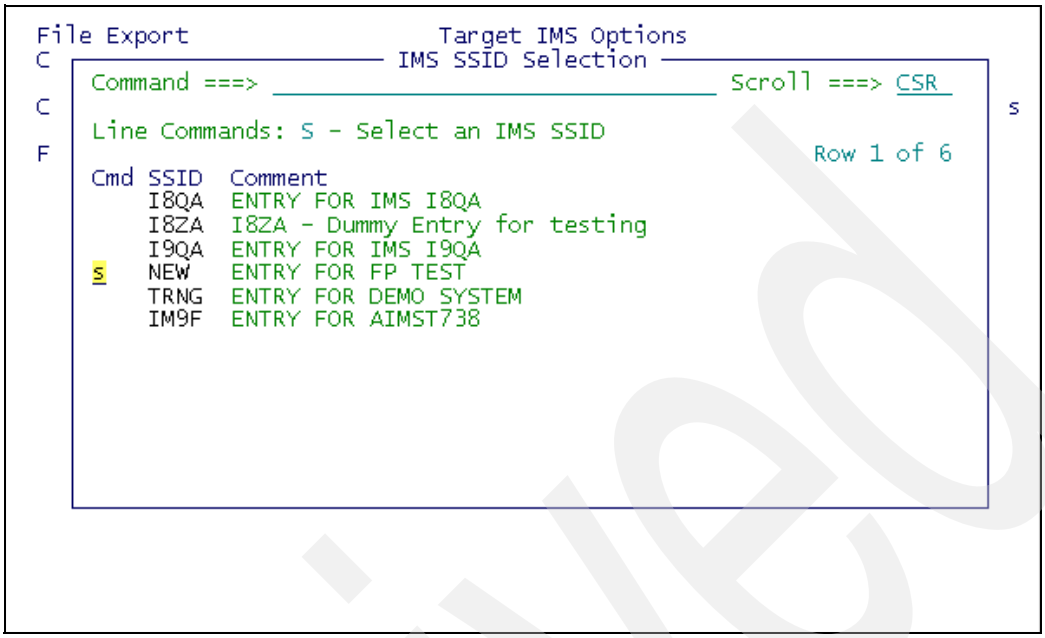


Figure 22-99 Select a target IMS SSID

Corresponding IMS region processing options are displayed for the selected IMS SSID. Any necessary edits to the IMS region processing options can be made and only affects this extract. Use the End command (typically PF3) when any edits are completed to return to the Multi-Data Store Extract Request panel.

Step 6: On the Multi-Data Store Extract Request panel, select option **6** to provide dataset parameters for the File Export unload-format dataset. The File Export unload-format dataset is the interim staging dataset where the IMS data is held before loading as shown in Figure 22-100. Press Enter.

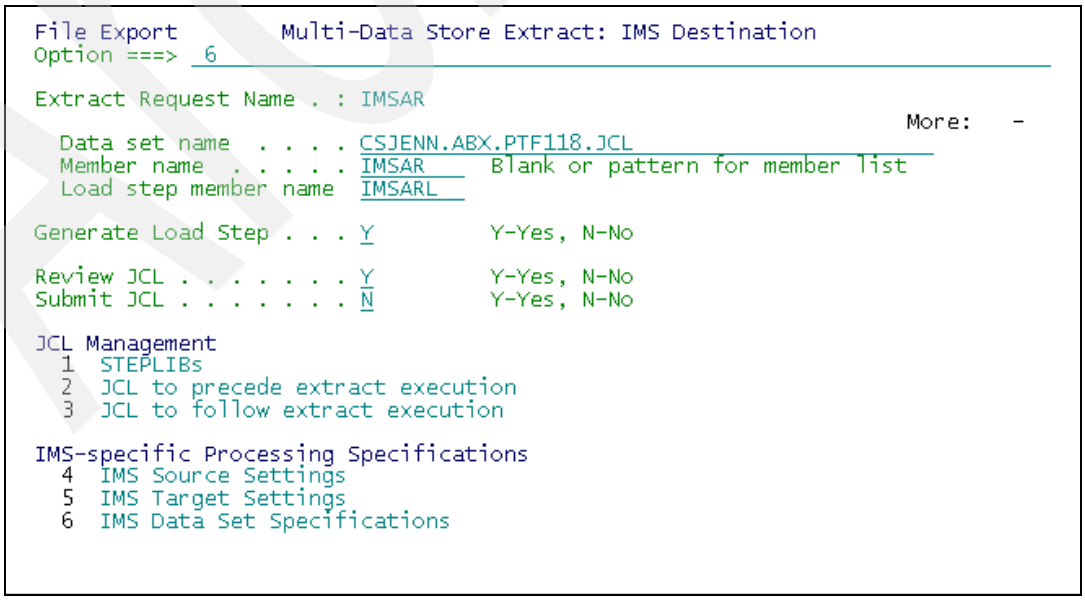


Figure 22-100 Specify IMS dataset parameters

Specify E next to the dataset name to edit the dataset allocation parameters as shown in Figure 22-101. Press Enter.

```
File Export                    DD Statements List
Command ===> ____________________________________________ Scroll ===> CSR

Line Commands: E - Edit DD Statement  I - Edit Instream

Extract Request Name . : IMSAR
                                                          Row 1 of 1
Cmd DD Name   Optional  Data Set Name
e   FEUNLOAD           CSJENN.IMSAR
```

Figure 22-101 Edit the FEUNLOAD dataset to add allocation parameters

Option **W** invokes the Dataset Allocation Wizard to assist in specifying dataset parameters, as shown in Figure 22-102. Press Enter.

```
File Export              DD Statement Definition
Command ===> w_____________________________________ Scroll ===> CSR

Commands:       A - Add line  T - Generate DD statement from template
                W - Generate DD statement with wizard
Line Commands: D - Delete line

Data Set Name CSJENN.IMSAR.FEUNLOAD_____________________

Enter FEUNLOAD DD parameters:

Cmd Text
    DSN=CSJENN.IMSAR
```

Figure 22-102 Use the Wizard to specify the FEUNLOAD dataset parameters

Provide the sizing parameters for the FEUNLOAD dataset as shown Figure 22-103. Use the End command (typically PF3).

```
File Export             Data Set Allocation Parameters
Command ===> ____________________________________________________________

Data Set unit  . . . . SYSDA___   If SMS managed, leave blank
Volume serial  . . . . ______     If SMS managed, or unit specified
                                  a tape unit, leave blank
Space units    . . . . trk        CYL or TRK
Primary quantity . . . 1___
Secondary quantity . . 1___
```

Figure 22-103 Specify the sizing parameters that are required to create the FEUNLOAD dataset

The DD statement definition now reflects the sizing attributes that is generated into the Extract Request JCL. Use the End command (typically PF3) to return to the Multi-Data Store Extract Request panel. Press Enter to generate Extract and Load Request JCL.

The first JCL job to be generated is the Extract Request JCL. In the generated Extract Request JCL, there are RESULTSET keywords that refer to the extracted data for a given segment. There are also CHILD OF RESULTSET*n* syntax that indicate the object referred to in the RESULTSET statement are children of a previous RESULTSET whose extraction is dependent upon data from a parent object as shown in Figure 22-104.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CSJENN.ABX.PTF118.JCL(IMSAR) - 01.00            Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
****** ******************************** Top of Data ********************************
000001 //IMSAR   JOB CSJENN,CLASS=A,NOTIFY=&SYSUID,
000002 //       MSGCLASS=X,MSGLEVEL=(1,1)
000003 //*
000004 //*
000005 //STEP1    EXEC PGM=ABXUTIL
000006 //STEPLIB  DD   DISP=SHR,DSN=RSQA.ABX120.IBMTAPE.SABXLOAD
000007 //         DD   DISP=SHR,DSN=RSRTE.XML.HXML180.SIXMLOD1
000008 //         DD   DISP=SHR,DSN=CSWILK.ABX.GLOBAL.INSTLBLK.V2
000009 //*
000010 //*
000011 //ABXOUT   DD   DSN=CSJENN.IMSAR.FEUNLOAD,
000012 //             UNIT=SYSDA,
000013 //             DISP=(NEW,CATLG,DELETE),
000014 //             SPACE=(TRK,(1,1))
000015 //ABXPRINT DD   SYSOUT=*
000016 //ABXCTL   DD   *
000017 OPTIONS((DLI, FEUNLOAD), DECIMAL, PERCENT, NOQUIET);
000018 IMSID(I8QA);
000019 SOURCEDBD(D@#POLKY);
```

Figure 22-104 The generated IMS Extract Request JCL

When the End command is pressed to exit from the Extract Request JCL, the Load Request JCL is generated and displayed. The FEUNLOAD dataset from the Extract Request JCL is utilized in the Load Request JCL as input. The target IMS SSID is displayed in the Control Syntax indicating to File Export where the target DBDs are located. The INCLUDE statements contain the Source DBD names as well as the corresponding Target DBD names as shown in Figure 22-105.

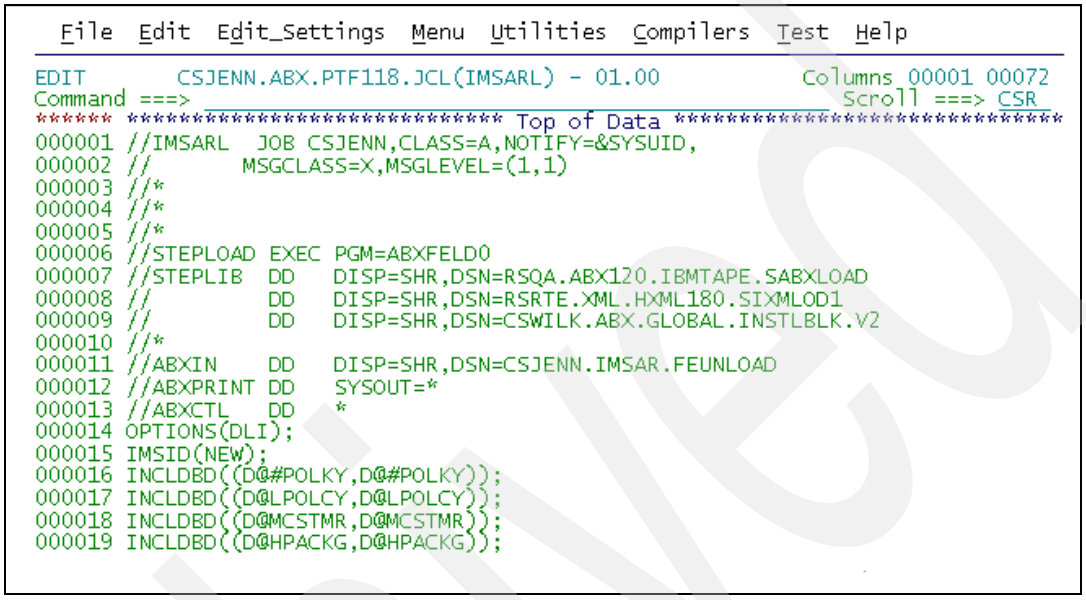


Figure 22-105 The matching generated IMS Load Request JCL

23

File Export: Beyond the basics

In this chapter we describe how to use additional features of File Export such as Key Value File for providing alternate selection criteria, scrambling sensitive data, as well as other hints and tips for using File Export. The information in this chapter is more in-depth and assumes the reader is familiar with the basic tasks in File Export, such as creating a Repository, and creating an Extract and Load Request.

We discuss the following topics:

- Using key files to extract data
- Scrambling fields to hide sensitive data
- Record ID criteria for selecting appropriate record layout
- Binding File Export DB2 plans
- Exporting from and importing to a repository

23.1 Using key files to extract data

Key Value Files are datasets that contain values from which are used to select records and thus drive the extract. That is, only those values from the key value file that appear in the starting objects are selected for extract. In this manner, the key value file is the driving selection criteria that determines which records are extracted from the related objects.

To specify that a key value file is to be used in an Extract Request, follow these steps.

Step1: On the Metadata Management panel, select option **4** to create a Related Subset as shown in Figure 23-1.

Note: An existing Related Subset can also be used. To use an existing Related Subset, begin by editing an existing Related Subset as described in Step 5.

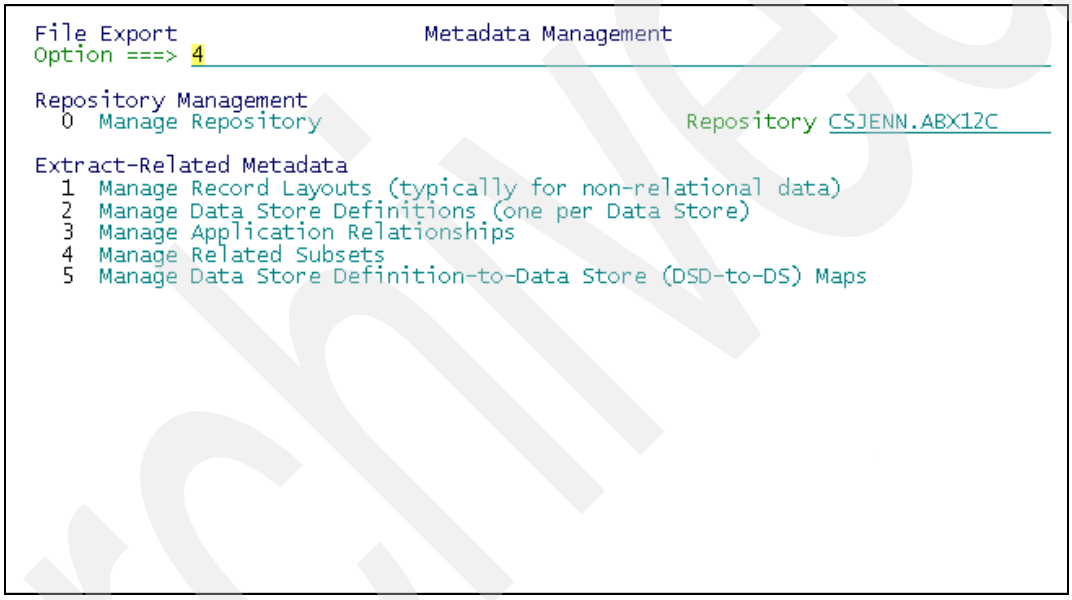


Figure 23-1 Edit an existing Related Subset or create a new one

Step 2: Provide a name for the new Related Subset, and type **C** on the Command Line as shown in Figure 23-2. Press Enter to continue.

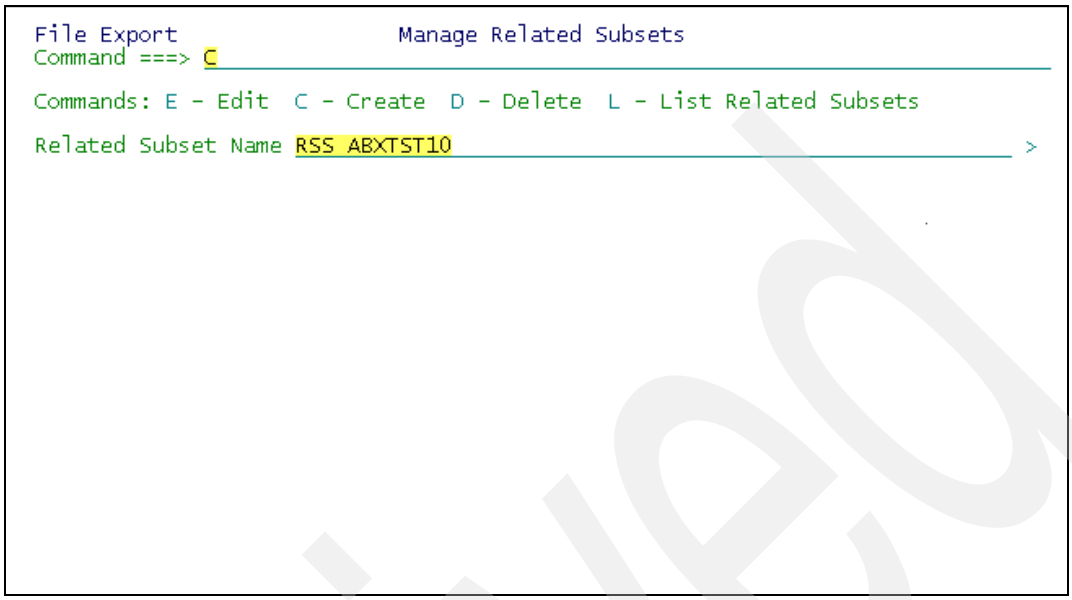


Figure 23-2 Provide a new Related Subset with a name

Step 3: Specify the objects' start type. In this example, it is a Data Store Definition.

Specify the object to begin the extract from. This is the driving object from which all related data is extracted. The Data Store Definition can be partially wild carded to any degree by using an asterisk. Also specify if you want to traverse both Parents and Children of the starting object, and optionally how many levels of relationships to traverse as shown in Figure 23-3. Press Enter.

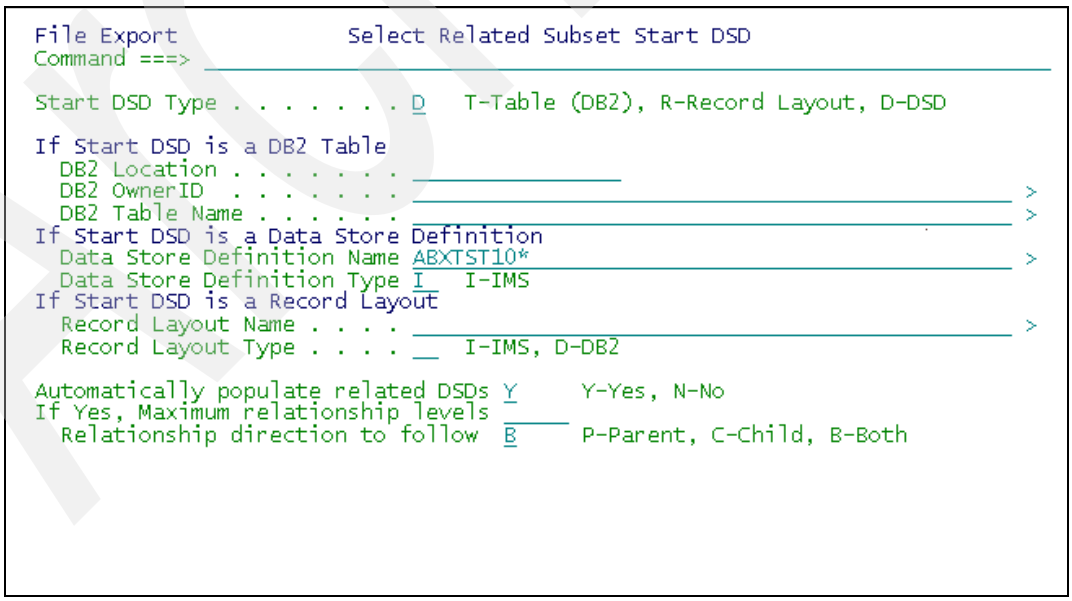


Figure 23-3 Use a partially wild carded DSD name to get a list

Step 4: A selection list is returned displaying all Data Store Definitions that match the wildcard pattern. Select the DSD to use as the start object, as shown in Figure 23-4, and press Enter.

When Enter is pressed from the Select Related Subset Start DSD panel, the resultant Related Subset panel is displayed containing all objects related to the starting object.

```
File Export            Data Store Definition Selection
Command ===> _____________________________________________ Scroll ===> CSR

Line Commands: S - Select Data Store Definitions

Data Store Definition Name mask ABXTST10*_____________________________________ >
Data Store Definition Type mask I_  I-IMS
                                                                  Row 1 of 1
Cmd Data Store Definition Name                                    Type
s   ABXTST10.RSDEMO.ABX.TRAINING.DBDLIB                         IMS
```

Figure 23-4 Select the Data Store Definition to use as the driving object

Step 5: Tab down next to the first object in the list and type a K to be able to specify a Key File to use as input to the Extract Request as shown in Figure 23-5. Press Enter.

```
File Export          Create Related Subset Definition
Command ===> _____________________________________________ Scroll ===> CSR

Commands:      A - Add Data Store Definitions  R - Relationship Usage
Line Commands: G - Get Related Definitions  D - Delete  C - Selection Criteria
               K - Key File  S - Scramble

Related Subset Name . : RSS ABXTST10
DSD-to-DS Map Name  . . _____________________________________________________ >
                                                                  Row 1 of 3
  ---- Data Store Definition ----                                      Every
       Table / RL / DSD          Type Status  Start Ref Skip    Limit   Nth
k ABXTST10.RSDEMO.ABX.TRAINING.DB IMS          Y     N
_ ABXTST06.RSDEMO.ABX.TRAINING.DB IMS          N     N
_ ABXTST05.RSDEMO.ABX.TRAINING.DB IMS          N     N
```

Figure 23-5 Typing K next to an object allows you to provide a Key File value

Step 6: A list of DBD Segments is displayed allowing you to specify which segment to use a Key File on. Since there is only one Segment defined to the DBD in this example, only one segment is displayed.

Type **K** next to the segment to use the Key File against, as shown in Figure 23-6, and press Enter.

Figure 23-6 Select the Segment of the DBD you want to use a Key File on

Step 6: On the Key File Specification panel, provide the information required to use the key value file as input to the Extract Request. Information such as the name of the dataset that contains the key values, the position, length and data type of the key values in the dataset and the Segment field name to use the key values on are required fields. Press the End command (usually PF3) when all fields have been completed as shown in Figure 23-7.

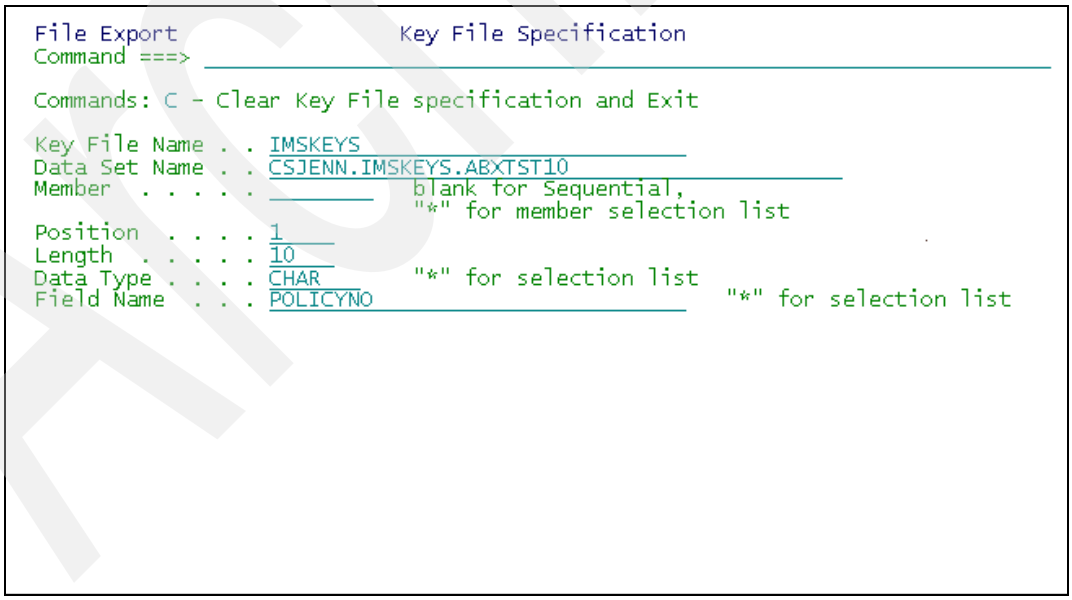


Figure 23-7 Provide the Key File values

The field Status on the Edit Related Subset Definition panel displays the value KF indicating that a key value file was specified on the object and is used in the Extract Request as shown in Figure 23-8.

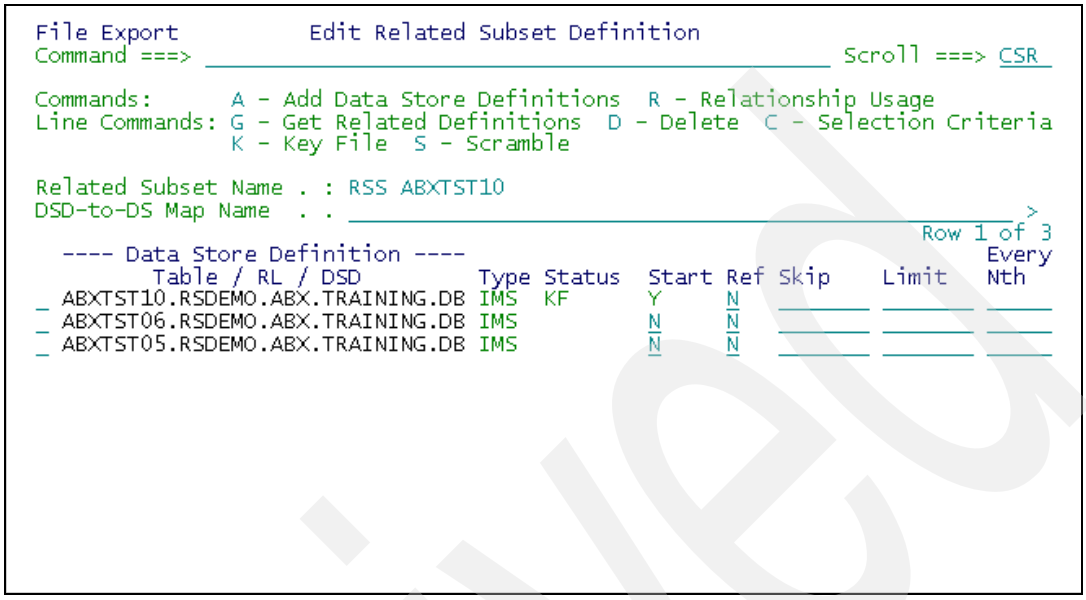


Figure 23-8 The status KF indicates at-a-glance that a Key File is used

When Extract Request JCL is generated using the Related Subset of objects that contains the Key File as input, the generated JCL looks like the JCL in Example 23-1.

Example 23-1 Generated JCL using a Key File

```
****** ***************************** Top of Data ******************************
000001 //IMSKEYF JOB CSJENN,CLASS=A,NOTIFY=&SYSUID,
000002 //      MSGCLASS=X,MSGLEVEL=(1,1)
000003 //*
000004 //*
000005 //STEP1    EXEC PGM=ABXUTIL
000006 //STEPLIB  DD   DISP=SHR,DSN=RSQA.ABX120.IBMTAPE.SABXLOAD
000007 //         DD   DISP=SHR,DSN=RSRTE.XML.HXML180.SIXMLOD1
000008 //         DD   DISP=SHR,DSN=CSWILK.ABX.GLOBAL.INSTLBLK.V2
000009 //*
000010 //ABX00001 DD   DISP=SHR,DSN=CSJENN.IMSKEYS.ABXTST10
000011 //*
000012 //ABXOUT   DD   DISP=SHR,DSN=CSJENN.IMSKEYF.EXTRACT
000013 //ABXPRINT DD   SYSOUT=*
000014 //ABXCTL   DD   *
000015 OPTIONS((DLI, FEUNLOAD), DECIMAL, PERCENT, NOQUIET);
000016 IMSID(TRNG);
000017 RELATIONSHIP DD:ABX00001
000018  IMSKEYS = (1,10,CHAR);
000019 SOURCEDBD(ABXTST10);
000020 SEGMENT(POLICY);
000021 RESULTSET (RESULTSET1);
000022 SELECT * FROM POLICY WHERE POLICYNO IN (IMSKEYS);
000023 SOURCEDBD(ABXTST06);
000024 SEGMENT(POLICY);
000025 RESULTSET (RESULTSET2) CHILD OF RESULTSET1 WHERE RESULTSET2.POLICYNO =
```

```
000026 RESULTSET1.POLICYNO;
000027 SELECT * FROM POLICY;
000028 SEGMENT(BENEFIC);
000029 RESULTSET (RESULTSET3);
000030 SELECT * FROM BENEFIC;
000031 SOURCEDBD(ABXTST05);
000032 SEGMENT(BENEFIC);
000033 RESULTSET (RESULTSET4) CHILD OF RESULTSET3 WHERE RESULTSET4.BENID =
000034 RESULTSET3.BENID;
000035 SELECT * FROM BENEFIC;
000036 /*
****** ************************** Bottom of Data **************************
```

In the generated JCL above, the Control Syntax in this example begins with the keyword OPTIONS and ends with the last SELECT statement. The Control Syntax provides information to the Extract Engine about the name of the SOURCE to find the data to extract, the relationships between the IMS DBD Segments, the SELECT criteria to find specific data, and the Key File values to use to begin searching for related data.

The Control Syntax keywords are defined below for clarity.

RELATIONSHIP DD:ABX0000n

The DD name of the Key File dataset that contains the Key values to use as input for searching for related data in the specified objects.

IMSKEYS = (1,10,CHAR)

The name given to the Relationship that, when referenced, points the Extract Engine to the specified DD to find related data. The positional numbers tell the Extract Engine where to look in the named DD to find the Key values. The specified data type CHAR tells the Extract Engine what type data is contained in the specified position and length.

SOURCE(<IMS DBD>)

The IMS DBD from which related data is to be extracted.

SEGMENT(<IMS Segment>)

The IMS DBD Segment name that to be included in the extract.

RESULTSET (RESULTSETn)

Indicates that the extracted data from the named SOURCE is held in virtual memory identified by the specified resultset name.

SELECT * FROM WHERE <Field> IN (<Relationship Name>)

The statement that tells the Extract Engine what field's value is to be found in the Key File in order for the record to be extracted.

CHILD OF RESULTSETb WHERE RESULTSETa.<column> = RESULTSETb.<column>

Syntax that indicates an object is a child of the object referenced in the resultset name. The WHERE clause behaves much like SQL syntax in that the column from the child extract must match the column from the parent extract.

23.2 Scrambling fields to hide sensitive data

File Export contains a scrambling algorithm to alter values in a test environment from their original values in a production environment. The scrambling algorithm is deterministic and repeatable, that is, a given value always scrambles to the same value in an extract. In this way, key values always scramble to the same value, and related data is still properly related.

Scrambling values in an object can be crucial in copying production data for use in a test environment which by its very nature is less secure. The actual value of the data is probably not pertinent to testing an application, while the actual data type would be.

For example, a field containing social security numbers can be used in testing an application. The actual numbers themselves are not pertinent to testing an application, but the data type can be important. Using File Export's scrambling feature allows you to alter the value of the social security numbers to other numbers (not actual social security numbers) enabling related data for a given social security number to still be extracted.

To specify that a field or fields are to be scrambled in an Extract Request, follow these steps.

Step1: On the Metadata Management panel, select option **4** to Edit a Related Subset.

Note: A new Related Subset of objects can also be created. To create a new Related Subset, begin with Step 1 in the section titled "Using key files to extract data" on page 612.

Step 2: Provide the name for the Related Subset to edit and type an E on the Command Line. Press Enter to continue.

Step 3: On the Edit Related Subset Definition panel, tab down next to an object in the list, and type an **S** to specify the fields to scramble in an Extract Request as shown in Figure 23-9. Press Enter.

```
File Export          Edit Related Subset Definition
Command ===> _____________________________________________ Scroll ===> CSR

Commands:      A - Add Data Store Definitions  R - Relationship Usage
Line Commands: G - Get Related Definitions  D - Delete  C - Selection Criteria
               K - Key File  S - Scramble

Related Subset Name . : RSS ABXTST10
DSD-to-DS Map Name  . . _____________________________________________>
                                                          Row 1 of 3
  ---- Data Store Definition ----                                Every
        Table / RL / DSD          Type Status  Start Ref Skip    Limit   Nth
s ABXTST10.RSDEMO.ABX.TRAINING.DB IMS  KF      Y     N   _______ _______ _____
_ ABXTST06.RSDEMO.ABX.TRAINING.DB IMS          N     N   _______ _______ _____
_ ABXTST05.RSDEMO.ABX.TRAINING.DB IMS          N     N   _______ _______ _____
```

Figure 23-9 Select a DSD on which to scramble fields

Step 4: A list of DBD segments is displayed allowing you to specify the segment whose fields are to be scrambled. Since there is only one segment defined to the DBD in this example, only one segment is displayed.

Type an **S** next to the segment whose fields are to be scrambled, as shown in Figure 23-10, and press Enter.

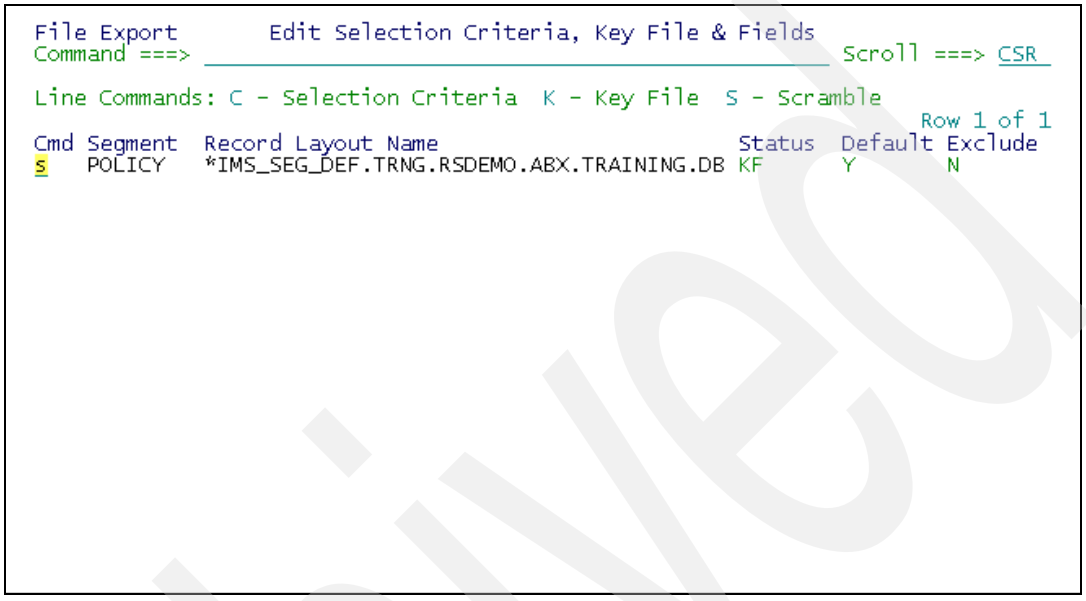


Figure 23-10 Select the Segment whose fields are to be scrambled

Step 5: On the Edit Fields panel, tab down to the field that should be scrambled, and type a Y in the Scramble column as shown in Figure 23-11. Press the End command to accept the changes and return back to the previous panel.

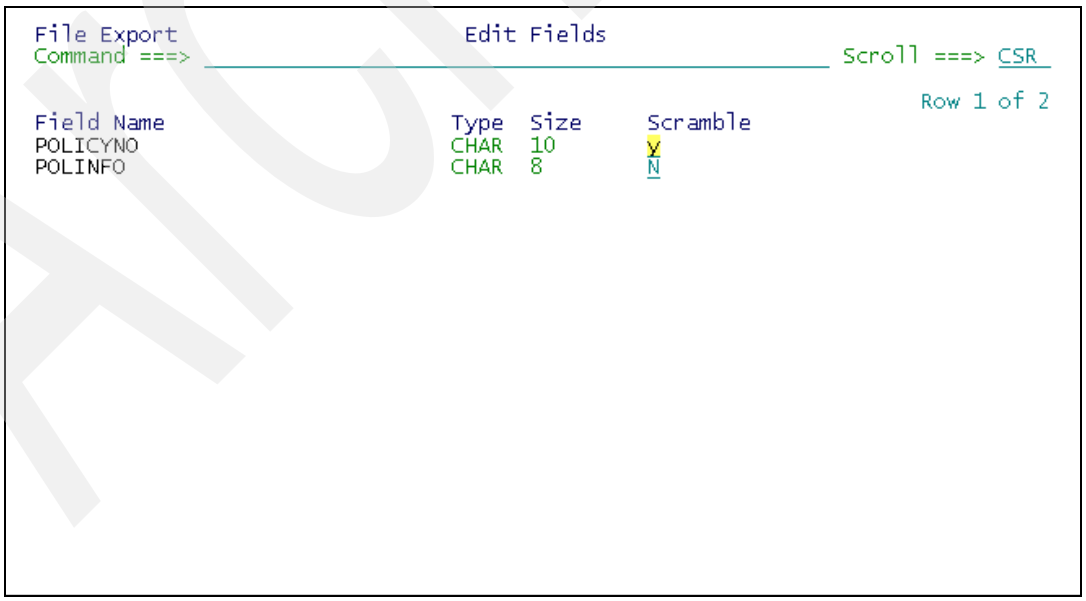


Figure 23-11 Type Y' under the column Scramble

The field Status on the Edit Related Subset Definition panel displays the value SCR indicating that scrambling was specified on one or all of the objects' fields as shown in Figure 23-12.

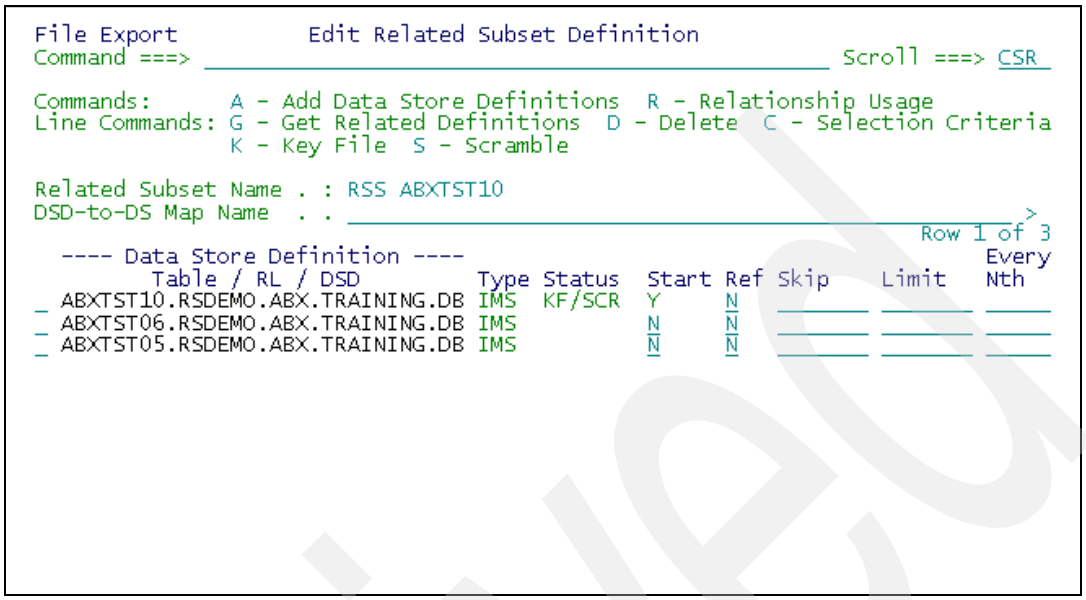


Figure 23-12 The status SCR indicates at-a-glance that scrambling was chosen

When Extract Request JCL is generated using the Related Subset of objects that contains scrambled fields, the generated JCL looks like the JCL in Example 23-2.

Example 23-2 Example JCL showing the SCRAMBLE syntax

```
****** ***************************** Top of Data ******************************
000001 //IMSSCRA JOB CSJENN,CLASS=A,NOTIFY=&SYSUID,
000002 //       MSGCLASS=X,MSGLEVEL=(1,1)
000003 //*
000004 //*
000005 //STEP1    EXEC PGM=ABXUTIL
000006 //STEPLIB  DD   DISP=SHR,DSN=RSQA.ABX120.IBMTAPE.SABXLOAD
000007 //         DD   DISP=SHR,DSN=RSRTE.XML.HXML180.SIXMLOD1
000008 //         DD   DISP=SHR,DSN=CSWILK.ABX.GLOBAL.INSTLBLK.V2
000009 //*
000010 //ABX00001 DD   DISP=SHR,DSN=CSJENN.IMSKEYS.ABXTST10
000011 //*
000012 //ABXOUT   DD   DISP=SHR,DSN=CSJENN.IMSSCRA
000013 //ABXPRINT DD   SYSOUT=*
000014 //ABXCTL   DD   *
000015 OPTIONS((DLI, FEUNLOAD), DECIMAL, PERCENT, NOQUIET);
000016 IMSID(TRNG);
000017 RELATIONSHIP DD:ABX00001
000018  IMSKEYS = (1,10,CHAR);
000019 SOURCEDBD(ABXTST10);
000020 SEGMENT(POLICY);
000021 RESULTSET (RESULTSET1);
000022 SELECT * SCRAMBLING(POLICYNO) FROM POLICY WHERE POLICYNO IN (IMSKEYS);
000023 SOURCEDBD(ABXTST06);
000024 SEGMENT(POLICY);
000025 RESULTSET (RESULTSET2) CHILD OF RESULTSET1 WHERE RESULTSET2.POLICYNO =
000026 RESULTSET1.POLICYNO;
```

```
000027 SELECT * FROM POLICY;
000028 SOURCEDBD(ABXTST05);
000029 SEGMENT(BENEFIC);
000030 RESULTSET (RESULTSET4) CHILD OF RESULTSET3 WHERE RESULTSET4.BENID =
000031 RESULTSET3.BENID;
000032 SELECT * FROM BENEFIC;
000033 /*
****** ***************************** Bottom of Data ****************************
```

In the generated JCL above, the Control Syntax in this example begins with the keyword OPTIONS and ends with the last SELECT statement. The Control Syntax provides information to the Extract Engine about the name of the SOURCE to find the data to extract, the relationships between the IMS DBD Segments, the SELECT criteria to find specific data, the Key File values to use to begin searching for related data, and the fields whose content should be scrambled.

The Control Syntax keywords are defined below for clarity.

RELATIONSHIP DD:ABX0000n

The DD name of the Key File dataset that contains the Key values to use as input for searching for related data in the specified objects.

IMSKEYS = (1,10,CHAR)

The name given to the Relationship that, when referenced, points the Extract Engine to the specified DD to find related data. The positional numbers tell the Extract Engine where to look in the named DD to find the Key values. The specified data type CHAR tells the Extract Engine what type data is contained in the specified position and length.

SOURCE(<IMS DBD>)

The IMS DBD from which related data is to be extracted.

SEGMENT(<IMS Segment>)

The IMS DBD Segment name to be included in the extract.

RESULTSET (RESULTSETn)

Indicates that the extracted data from the named SOURCE is held in virtual memory identified by the specified resultset name.

SELECT * SCRAMBLING(<Field>) FROM WHERE <Field> IN (<Relationship Name>)

The statement that tells the Extract Engine what field's value is to be found in the Key File in order for the record to be extracted.

The statement also tells the Extract Engine what fields are to be scrambled upon extract.

CHILD OF RESULTSETb WHERE RESULTSETa.<column> = RESULTSETb.<column>

Syntax that indicates an object is a child of the object referenced in the resultset name. The WHERE clause behaves much like SQL syntax in that the column from the child extract must match the column from the parent extract.

23.3 Record ID criteria for selecting appropriate record layout

Record ID criteria allow you to select a Record Layout when a specified condition is met. It might be the case that in sequential and VSAM files there are records that contain different kinds of data, and therefore there are multiple Record Layouts defined. For example, there might be records that contain insurance policy information as well as policy holder information contained in one data source. In such a case, there is usually a Record ID that specifies what kind of data that record contains.

For example, in position 1 for a length of two, there might be a Record ID of '01' to indicate that the record contains information on an insurance policy. Record ID '02' might indicate that the record contains information about the policy holder. And perhaps Record ID '03' indicates records that contain information on insurance claims.

In such a scenario, Record IDs can be used in selecting the appropriate Record Layout so the records to be extracted can be interpreted properly. That is, only the records whose ID matches the specified value are extracted and used in a related data extract.

In the example that follows, a sequential file containing multiple types of records has a field for each entry which identities the kind of data each record contains. In this example, a COBOL copybook contains field definitions that describe the layout of each kind of record. To ease the process of creating record layouts for each kind of record, the copybook can be parsed to create Record Layouts. However, Record Layouts can also be created manually by a user familiar enough with the data to define each field in the sequential file.

Task 1: To specify that a Record Layout is to have criteria used in evaluating types of records in an Extract Request, follow these steps.

Step1: On the Metadata Management panel, select option **1** Manage Record Layouts to create a new Record Layout.

Step 2: On the Manage Record Layouts panel, provide the name for the Record Layout and specify **DS** as the Underlying Data Store Type as shown in Figure 23-13.

```
File Export                    Manage Record Layouts
Command ===> c
                                                                    More:     +
Commands: E - Edit  C - Create  V - View  D - Delete  L - List Record Layouts
          S - Specify Copybook SYSLIB Data Sets

Record Layout Name  . . . . RL_SEQ03                                         >
Underlying Data Store Type  DS  I-IMS, DS-VSAM or Sequential, D-DB2
Save Data Store Association Y   Y-Yes, N-No

If using a Copybook
  Compiler to Use . . . . . C   C-COBOL, P-PL/I
  Compiler Options  . . . .                                                  >
  Data Set Name . . . . . . ABX.TRAINING.COPYB
  Member Name . . . . . . . ABXSEQ03  Blank or pattern for member list

If IMS Database
  SSID  . . . . . . . . . .
  DBD . . . . . . . . . . .
  Segment . . . . . . . . .
If VSAM or Sequential Data Set
  Data Set Name . . . . . . ABX.TRAINING.ABXSEQ03.INPUT
  Member Name . . . . . . .           Blank or pattern for member list
```

Figure 23-13   Create a Record Layout by parsing a Copybook

In the section If using a Copybook, specify **C** as the language the copybook is written in, and provide the dataset name where the copybook can be found. You can optionally specify copybook compiler options.

Tab down to the section titled If VSAM or Sequential Data Set and provide the name of the file that contains the actual data that the field definitions in the copybook define. Press Enter.

A temporary copy of the copybook is displayed in an ISPF edit session for you to make any final edits to before it is parsed. To begin parsing the copybook, press the End command (usually PF3) to accept the copybook and exit the ISPF Edit session as shown in Figure 23-14.

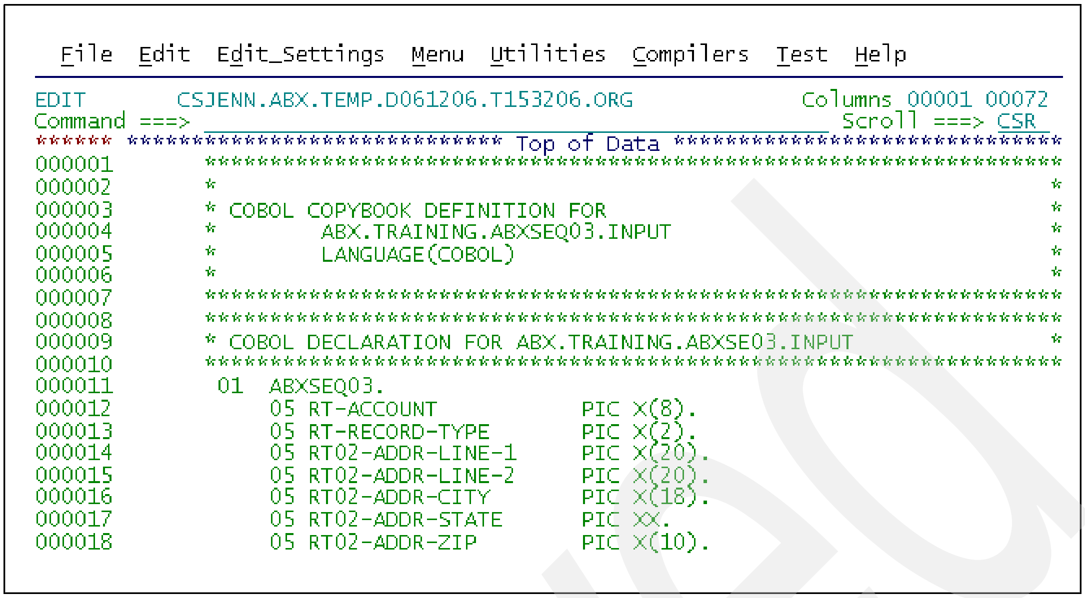

File Edit Edit_Settings Menu Utilities Compilers Test Help
EDIT CSJENN.ABX.TEMP.D061206.T153206.ORG Columns 00001 00072
Command ===> Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 **
000002 * *
000003 * COBOL COPYBOOK DEFINITION FOR *
000004 * ABX.TRAINING.ABXSEQ03.INPUT *
000005 * LANGUAGE(COBOL) *
000006 * *
000007 **
000008 **
000009 * COBOL DECLARATION FOR ABX.TRAINING.ABXSE03.INPUT *
000010 **
000011 01 ABXSEQ03.
000012 05 RT-ACCOUNT PIC X(8).
000013 05 RT-RECORD-TYPE PIC X(2).
000014 05 RT02-ADDR-LINE-1 PIC X(20).
000015 05 RT02-ADDR-LINE-2 PIC X(20).
000016 05 RT02-ADDR-CITY PIC X(18).
000017 05 RT02-ADDR-STATE PIC XX.
000018 05 RT02-ADDR-ZIP PIC X(10).

Figure 23-14 A temporary copy of the copybook allows you to make last-minute edits.

When the parse has completed, the Create Record Layout panel is displayed listing all fields that were parsed from the copybook. Any additional fields that have to be created or any edits to existing fields can be done from this panel.

Step 3: In the field Record ID Criteria, specify the field and the value to qualify the field so that the Extract Engine only selects those records whose Record ID criteria is met as shown in Figure 23-15. Press the End command to save the Record Layout and exit this panel.

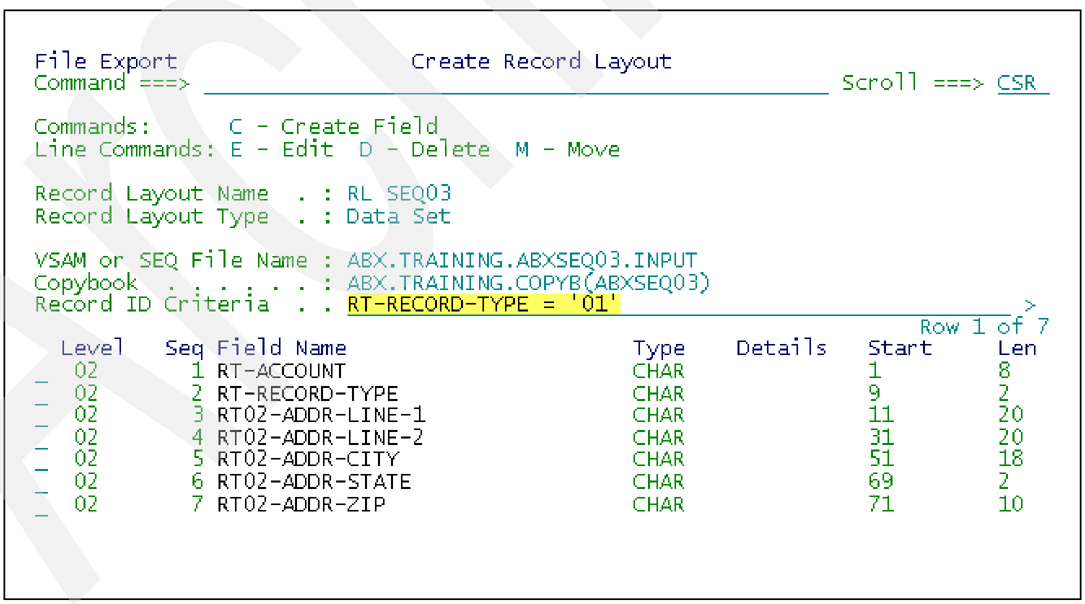

File Export Create Record Layout
Command ===> Scroll ===> CSR
Commands: C - Create Field
Line Commands: E - Edit D - Delete M - Move
Record Layout Name . : RL SEQ03
Record Layout Type . : Data Set
VSAM or SEQ File Name : ABX.TRAINING.ABXSEQ03.INPUT
Copybook . . . . . . : ABX.TRAINING.COPYB(ABXSEQ03)
Record ID Criteria . . RT-RECORD-TYPE = '01'
Row 1 of 7
Level Seq Field Name Type Details Start Len
02 1 RT-ACCOUNT CHAR 1 8
02 2 RT-RECORD-TYPE CHAR 9 2
02 3 RT02-ADDR-LINE-1 CHAR 11 20
02 4 RT02-ADDR-LINE-2 CHAR 31 20
02 5 RT02-ADDR-CITY CHAR 51 18
02 6 RT02-ADDR-STATE CHAR 69 2
02 7 RT02-ADDR-ZIP CHAR 71 10

Figure 23-15 Specify the Record ID Criteria for the new Record Layout

To create multiple record layouts, repeat steps 1 through 3. In our example, four Record Layouts have been created to manipulate the extracted data in a specific manner in an Extract Request

Task 2: Once all Record Layouts required for your extract have been created, create a Data Store Definition to group the Record Layouts you want to process as a single request.

Step 1: select option **2** from the Metadata Management panel to create a Data Store Definition as shown in Figure 23-16. Press Enter.

Step 2: On the Manage Data Store Definition panel, provide a Data Store name in the field Data Store Definition Name, and specify **DS** as the Data Store Type.

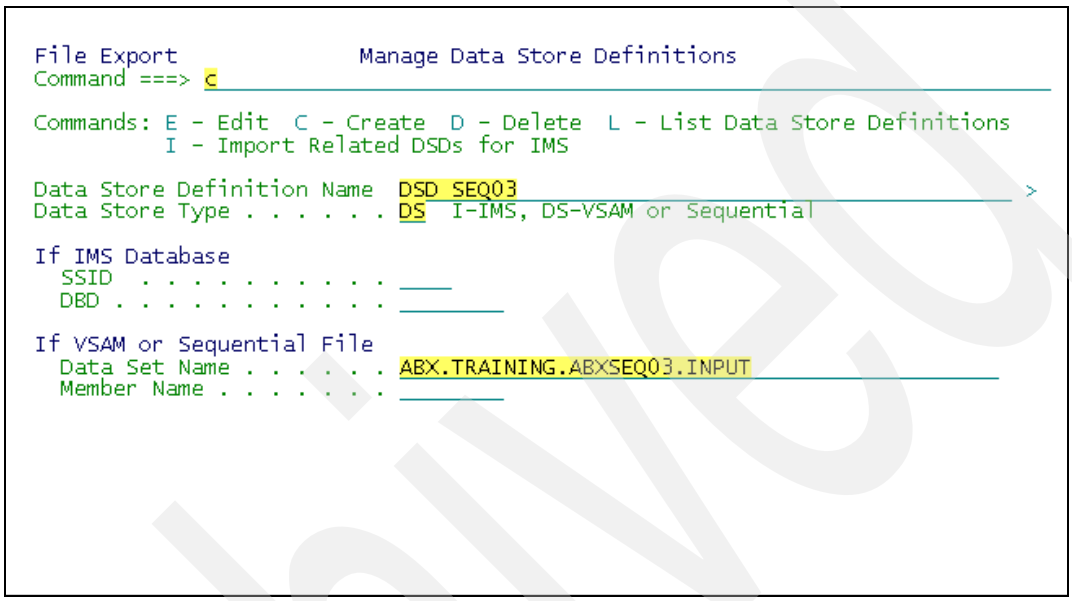


Figure 23-16 Manage Data Store Definitions

Tab down to the section titled If VSAM or Sequential File and provide the Data Set name where the actual data resides. Once the fields are filled in, type option **C** on the command line to Create a Data Store. Press Enter.

Step 4: The panel Create Data Store Definition is displayed with the DSD name and physical data file used to create the Data Store Definition. To populate the DSD with the Record Layouts we created in Task 1, type A on the command line, as shown in Figure 23-17, and press Enter.

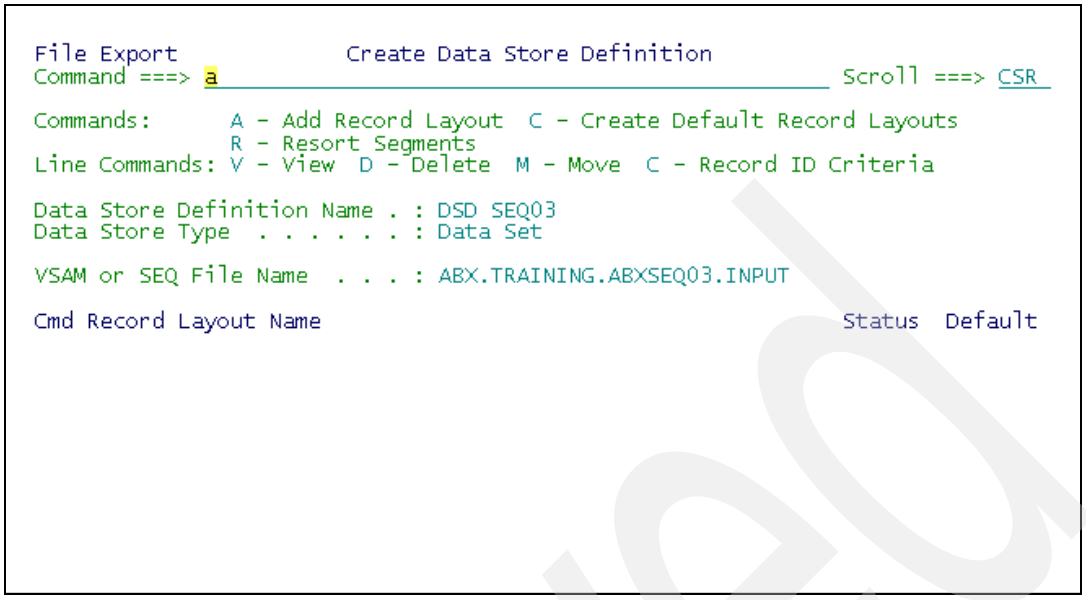


Figure 23-17 Add Record Layouts to the Data Store Definition

A pop-up window titled Record Layout Selection allows you to partially qualify a DSD name, and specify the data store type to narrow the results. Provide a Record Layout name qualified to any degree, and/or specify DS to only return Record Layouts that define VSAM or sequential files as shown in Figure 23-18. Press Enter.

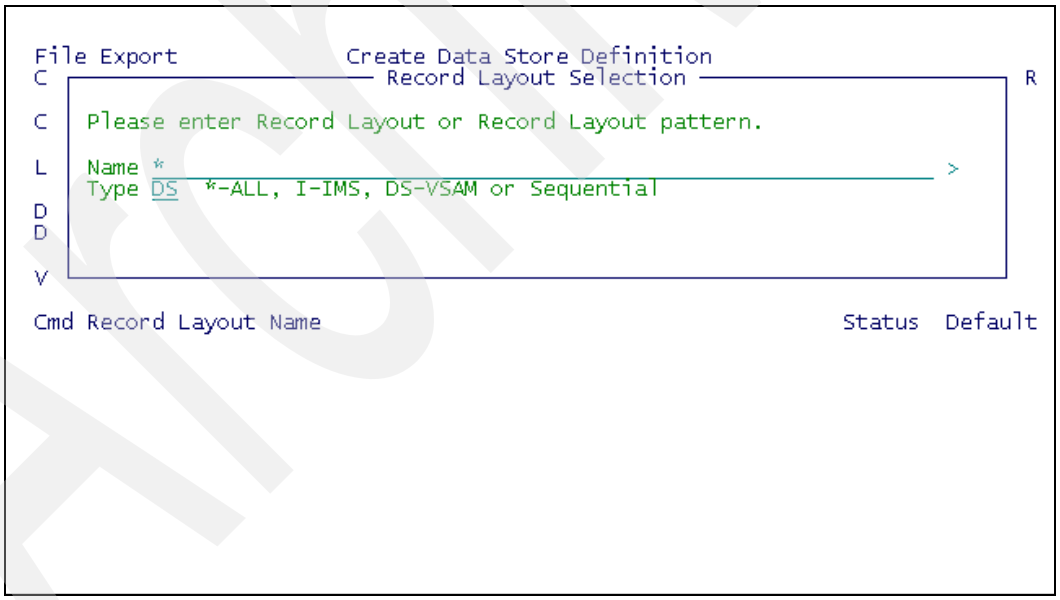


Figure 23-18 A wildcard can be used to find a pattern

The same pop-up window displays the Record Layouts that match the wild card name and/or the Record Layout Type initially specified. Select as many items on the Record Layout Selection pop-up, as shown in Figure 23-19, and press Enter.

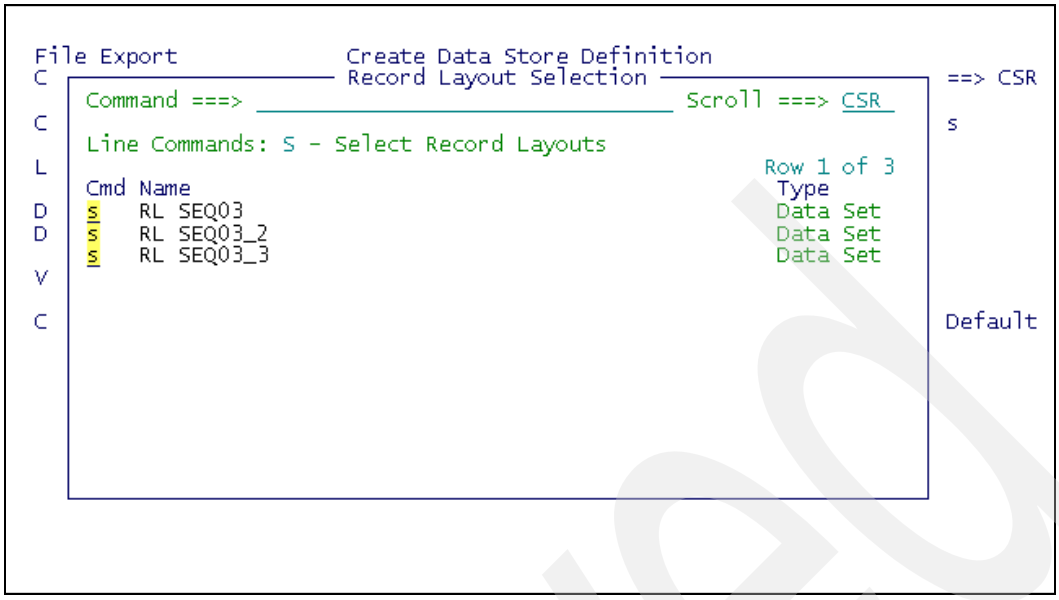


Figure 23-19 Select as many Record Layouts as required

The Create Data Store Definition panel is populated with the Record Layouts selected from the previous panel as shown in Figure 23-20. Press the End command to save the newly created Data Store Definition, and exit the panel.

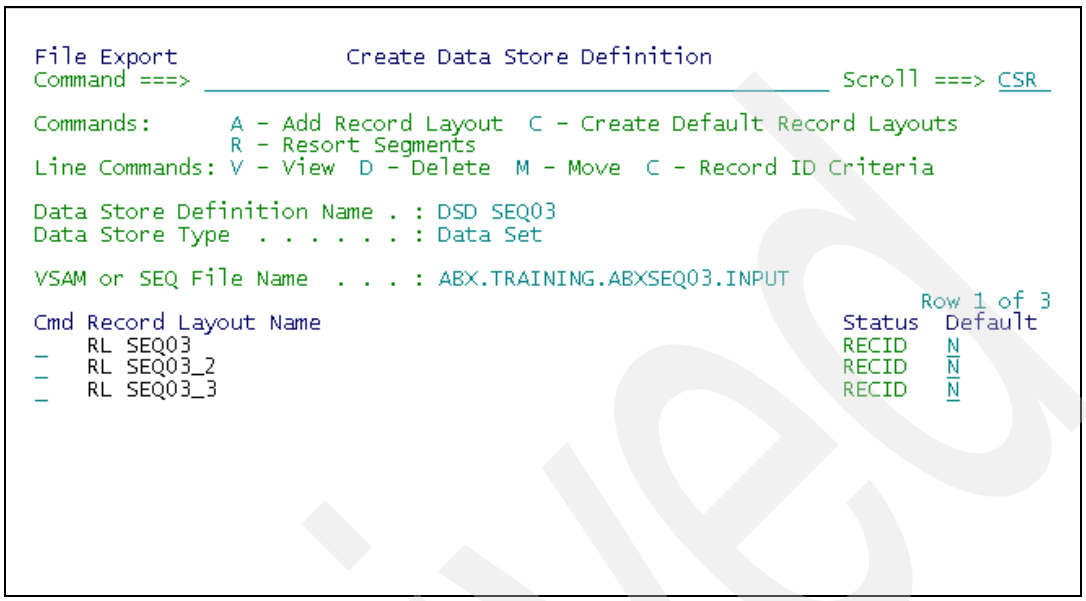


Figure 23-20 A completed Data Store Definition grouping Record Layouts

When Extract Request JCL is generated using the Data Store Definition that contains Record Layouts using Record ID Criteria, the generated JCL looks like the JCL in Example 23-3.

Example 23-3 Example JCL showing Record ID Criteria syntax

```
****** ***************************** Top of Data ******************************
000001 //JOBCARD JOB CSJENN,CLASS=A,NOTIFY=&SYSUID,
000002 //       MSGCLASS=X,MSGLEVEL=(1,1)
000003 //*
000004 //*
000005 //STEP1    EXEC PGM=ABXUTIL
000006 //STEPLIB  DD   DISP=SHR,DSN=RSQA.ABX120.IBMTAPE.SABXLOAD
000007 //         DD   DISP=SHR,DSN=RSRTE.XML.HXML180.SIXMLOD1
000008 //         DD   DISP=SHR,DSN=CSWILK.ABX.GLOBAL.INSTLBLK.V2
000009 //*
000010 //ABXIN    DD   DISP=SHR,DSN=ABX.TRAINING.ABXSEQ03.INPUT
000011 //ABXOUT   DD   DISP=SHR,DSN=CSJENN.ABXSEQ.OUTPUT
000012 //ABXPRINT DD   SYSOUT=*
000013 //ABXCTL   DD   *
000014 OPTIONS((SEQ, SEQ), DECIMAL, PERCENT, NOQUIET);
000015 AUDIT(NO);
000016 WUNIT(SYSDA);
000017 TYPE (RL SEQ03) WHEN RT-RECORD-TYPE = '01';
000018 FIELD(RT-ACCOUNT) = (1,8,CHAR);
000019 FIELD(RT-RECORD-TYPE) = (9,2,CHAR);
000020 FIELD(RT02-ADDR-LINE-1) = (11,20,CHAR);
000021 FIELD(RT02-ADDR-LINE-2) = (31,20,CHAR);
000022 FIELD(RT02-ADDR-CITY) = (51,18,CHAR);
000023 FIELD(RT02-ADDR-STATE) = (69,2,CHAR);
000024 FIELD(RT02-ADDR-ZIP) = (71,10,CHAR);
000025 SELECT * FROM RECORD;
```

```
000026 TYPE (RL SEQ03_2) WHEN RT-RECORD-TYPE = '99';
000027 FIELD(RT-ACCOUNT) = (1,8,CHAR);
000028 FIELD(RT-RECORD-TYPE) = (9,2,CHAR);
000029 FIELD(RT02-ADDR-LINE-1) = (11,20,CHAR);
000030 FIELD(RT02-ADDR-LINE-2) = (31,20,CHAR);
000031 FIELD(RT02-ADDR-CITY) = (51,18,CHAR);
000032 FIELD(RT02-ADDR-STATE) = (69,2,CHAR);
000033 FIELD(RT02-ADDR-ZIP) = (71,10,CHAR);
000034  EXCLUDE;
000035 TYPE (RL SEQ03_3) WHEN RT-RECORD-TYPE = '00';
000036 FIELD(RT-ACCOUNT) = (1,8,CHAR);
000037 FIELD(RT-RECORD-TYPE) = (9,2,CHAR);
000038 FIELD(RT02-ADDR-LINE-1) = (11,20,CHAR);
000039 FIELD(RT02-ADDR-LINE-2) = (31,20,CHAR);
000040 FIELD(RT02-ADDR-CITY) = (51,18,CHAR);
000041 FIELD(RT02-ADDR-STATE) = (69,2,CHAR);
000042 FIELD(RT02-ADDR-ZIP) = (71,10,CHAR);
000043  EXCLUDE;
000044 TYPE(DEFAULT);
000045  EXCLUDE;
000046 /*
****** ***************************** Bottom of Data ******************************
```

In the generated JCL above, the Control Syntax in this example begins with the keyword OPTIONS and ends with the keyword EXCLUDE. The Control Syntax provides information to the Extract Engine about the name of the SOURCE to find the data to extract, the defined Record Layouts and the records to extract when the Record ID criteria is matched.

The Control Syntax keywords are defined below for clarity.

TYPE (RL SEQ03) The name of the Record Layout to which the following Record ID Criteria syntax and corresponding field definitions applies.

WHEN <Field Name> = '<value>'

The criteria that is to be met before a record is selected for extract, or excluded from an extract.

SELECT * FROM RECORD

Select statement similar to SQL that tells File Export what fields to select for an extract. Specific field names can be identified by using Selection Criteria.

FIELD(<field name>) = (<position>,<length>,<data type>)

The data fields as they are defined in the named Record Layout.

EXCLUDE Indicates to the Extract Engine that the preceding Record Layouts' records are to be excluded from the extract when the Record ID Criteria is met.

23.4 Binding File Export DB2 plans

File Export uses a set of DBRMs to access the DB2 catalog when creating a DB2 Related Subset or processing a DB2 Extract Request. Therefore, plans must be bound for File Export to access the DB2 catalog.

The ABX.SABXSAMP library contains two bind jobs. The first job, ABXBIND, contains one plan bind step and encompasses all DBRMs into one plan as shown in Example 23-4.

Example 23-4 Bind job ABXBIND for a single plan

```
****** ***************************** Top of Data ****************************
000100 //ABXBIND  JOB 'BIND ABXUTIL',CLASS=A,MSGCLASS=X,
000200 //            NOTIFY=&SYSUID
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  - 43 Line(s) not Displayed
004800 //ABXUTIL  EXEC  PGM=IKJEFT01,DYNAMNBR=20
004900 //STEPLIB  DD  DISP=SHR,DSN=DB2.LOADLIB
005000 //SYSTSIN  DD  *
005100    DSN SYSTEM(DB2SUBSYS)
005200    BIND PLAN (ABXPLAN)  -
005300       MEMBER (ABXSQLX   -
005400               ABXDBUTX  -
005500               ABXDB2IX) -
005600       ACTION (REPLACE)  -
005700       RETAIN            -
005800       EXPLAIN    (NO)   -
005900       ISOLATION  (CS)   -
006000       SQLRULES   (DB2)  -
006100       OWNER(PLANOWNER)
006110    RUN PROGRAM(DSNTIAD) -
006120       PLAN(DSNTIAXX)    -
```

The second job shown in Example 23-5, ABXBIND3, contains two plan bind steps with the DBRMs split into the two separate plans based on function. Do not edit the placement of the DBRMs, as they are bound to the plans specific to their functions.

In ABXBIND3, there are two separate plan binds; plan ABXUTIL runs under the Extract Engine, and plan ABXD2UI runs under the User Interface. In this manner, a site can restrict which users can build Extract Requests through the User Interface, and which users can run the Extract Requests in the generated JCL.

Example 23-5 Bind job ABXBIND3 for binding two separate plans

```
****** ***************************** Top of Data ****************************
000100 //ABXBIND3 JOB 'BIND ABXUTIL',CLASS=A,MSGCLASS=X,
000200 //            NOTIFY=&SYSUID
000300 //*
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  - 39 Line(s) not Displayed
004500 //ABXUTIL  EXEC  PGM=IKJEFT01,DYNAMNBR=20
004600 //STEPLIB  DD  DISP=SHR,DSN=DB2.LOADLIB
004700 //SYSTSIN  DD  *
004800    DSN SYSTEM(DB2SUBSYS)
004900    BIND PLAN (ABXUTIL)  -
005000       MEMBER (ABXSQLX)  -
005100       ACTION (REPLACE)  -
005200       RETAIN            -
```

```
005300       EXPLAIN    (NO)   -
005400       ISOLATION  (CS)   -
005500       SQLRULES   (DB2)  -
005600       OWNER(PLANOWNER)
005700
005800    BIND PLAN (ABXD2UI)  -
005900       MEMBER (ABXDBUTX  -
005910               ABXDB2IX) -
006000       ACTION (REPLACE)  -
006100       RETAIN            -
006200       EXPLAIN    (NO)   -
006300       ISOLATION  (CS)   -
006400       SQLRULES   (DB2)  -
006500       OWNER(PLANOWNER)
007510    RUN PROGRAM(DSNTIAD) -
007520       PLAN(DSNTIAXX)    -
```

The name of each plan is arbitrary and can be changed to suit your shop's requirements. The plan names that are shipped with the product BIND jobs (that is, ABXUTIL and ABXD2UI) in the ABX.SABXSAMP library are suggestions only.

23.5 Exporting from and importing to a repository

After creating a Repository, the requirement might arise to resize or rename an existing Repository, or move its contents to a different Repository altogether. In such cases, there is an Export/Import function that allows you to accomplish these tasks.

23.5.1 Exporting a repository

Follow these steps to Export a Repository as a prerequisite step to resizing, renaming or moving the contents of an existing Repository:

Step 1: From the Metadata Management panel, select option **0** to Manage Repositories as shown in Figure 23-21. Press Enter.

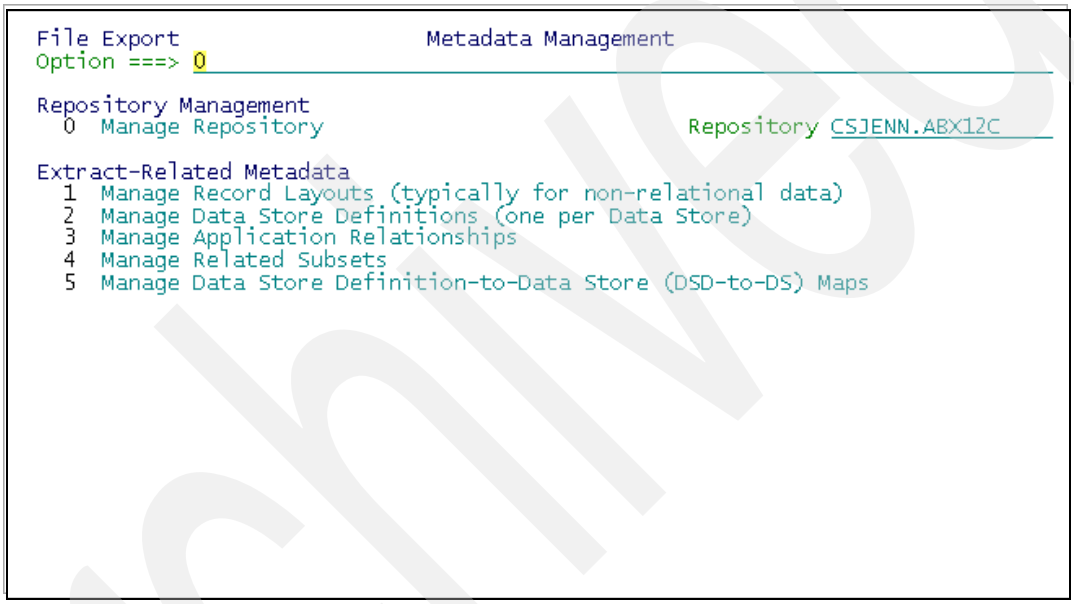


Figure 23-21 Select option 0 to Manage Repositories

Step 2: From the Manage Repository Panel, select option **3** to Export an existing Repository as shown in Figure 23-22. Press Enter.

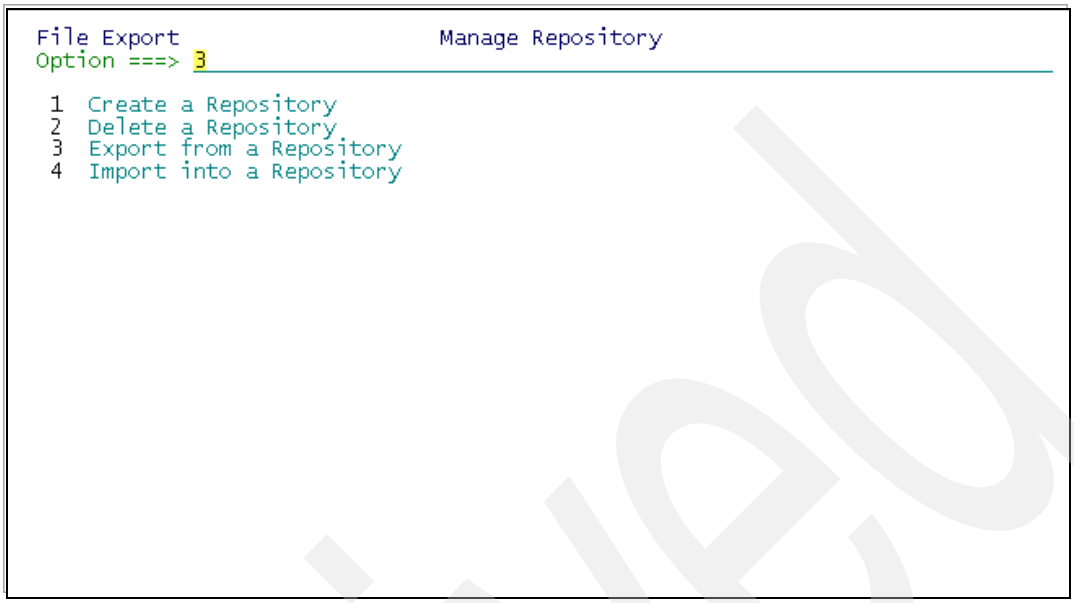


Figure 23-22 Select option 3 to Export an existing Repository's contents

Step 3: On the Export Repository panel, provide the name of the Repository to export to a temporary dataset as shown in Figure 23-23.

Note: The Repository name can be partially qualified by a wildcard, as well as the Description field, to find the Repository name from a list of valid Repositories that match the specified value.

The XML Encoding and the IBM Codepage options can be specified but are optional. If left blank, the default values are used.

The Target XML Dataset and Export Log Dataset can be named anything and do not have to exist. If they do not exist, File Export creates these datasets for you. If they already exist, then the XML datasets' contents are replaced, while the LOG datasets' contents are appended to.

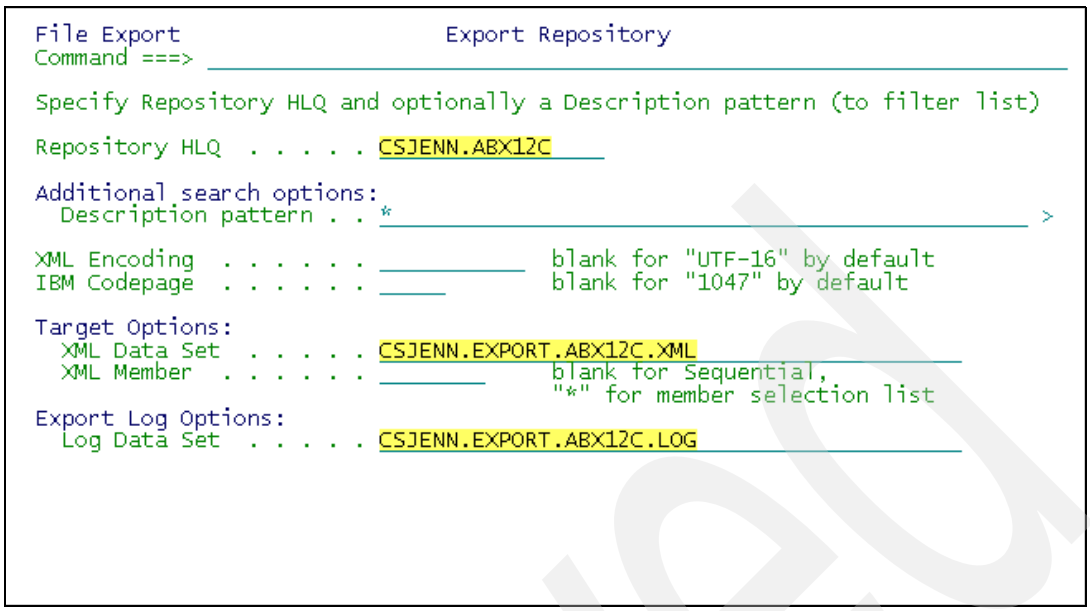


Figure 23-23 Specify the Repository's HLQ from which to export its contents

Press the End command (usually PF3) to process the options on the panel. The message `Operation Completed Successfully` is displayed on the Manage Repository panel when the contents of the Repository have been exported to the XML dataset successfully.

23.5.2 Resizing an existing repository

If you want to resize an existing Repository, there is currently not an automated facility to do this task. You must first Export the contents of the Repository by selecting option 3 as discussed in the beginning of this section in preparation for resizing an existing Repository.

Once the contents of a Repository have been exported to an XML dataset, the existing Repository must be deleted, recreated with larger sizing parameters, and the contents Imported back into the newly created Repository.

Deleting an existing repository

Step 1: After the contents of a Repository have been Exported as described in section "Exporting a repository" on page 632, select option **2** to Delete a Repository from the Manage Repository panel as shown in Figure 23-24. Press Enter.

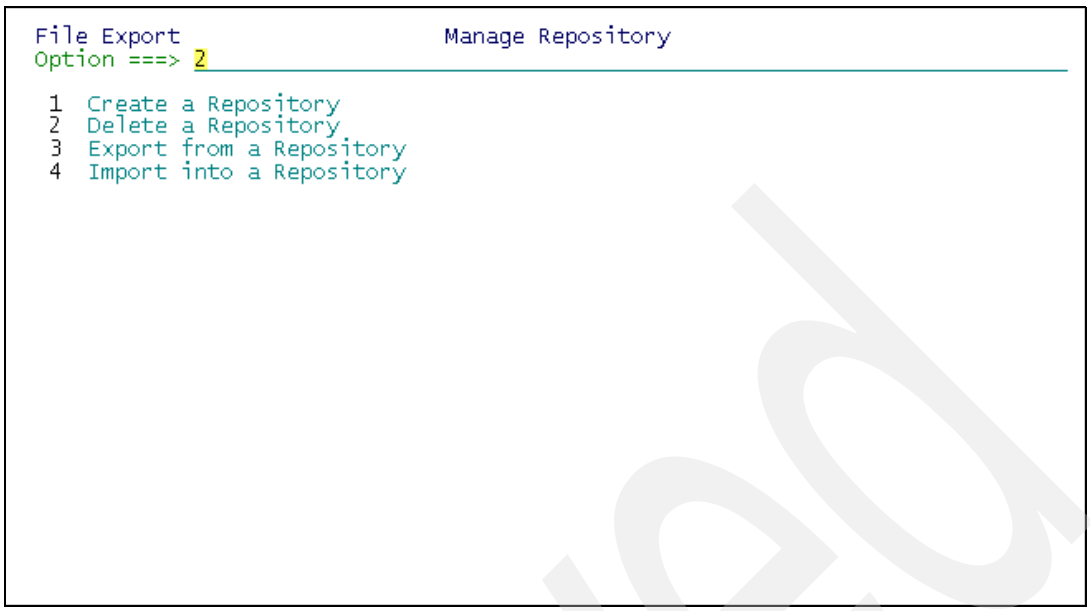


Figure 23-24 Select option 2 to Delete a Repository

Step 2: On the Delete Repository panel, provide the name of the Repository HLQ that you wish to delete and re-create with larger as shown in Figure 23-25. Press Enter. A pop-up window asks you to confirm your Repository HLQ. Press Enter to continue deleting the specified Repository's underlying datasets.

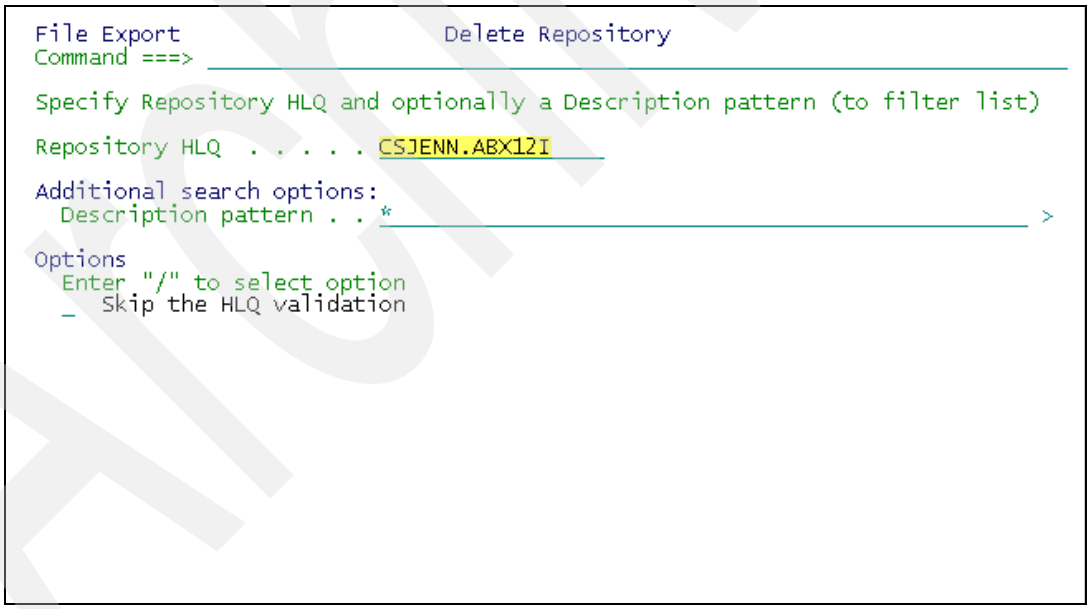


Figure 23-25 Specify the Repository HLQ to be deleted.

Creating a new repository

Step 1: On the Manage Repository panel, select option **1** to Create a Repository as shown in Figure 23-26. Press Enter.

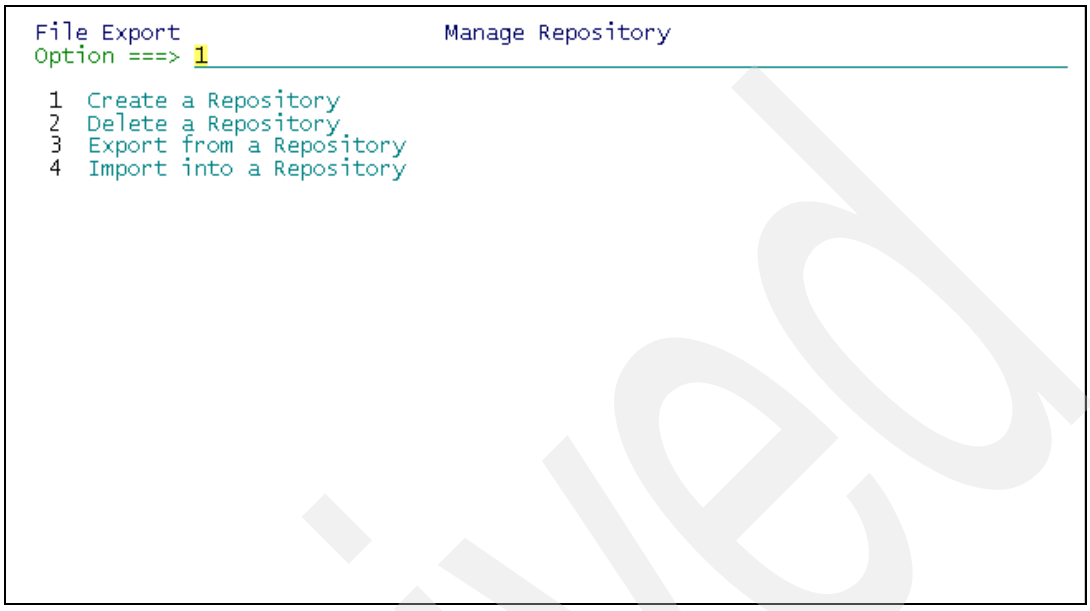


Figure 23-26 Select option 1 to Create a Repository

Step 2: On the Create Repository panel, provide a high level qualifier of up to 17 characters to name the new Repository. Provide an optional description that describes what this Repository is used for.

The fields Number of Record Layouts and Average Number of Fields per Record helps size the Repository's key lengths that are used for the alternate indexes, not the size of the Repository's VSAM datasets.

For SMS-managed datasets, also provide values for Management Class, Storage Class and Data Class. Optionally provide a Volume Serial if one is required.

When all values have been filled in, as shown in Figure 23-27, press PF3 to process the parameters on the panel.

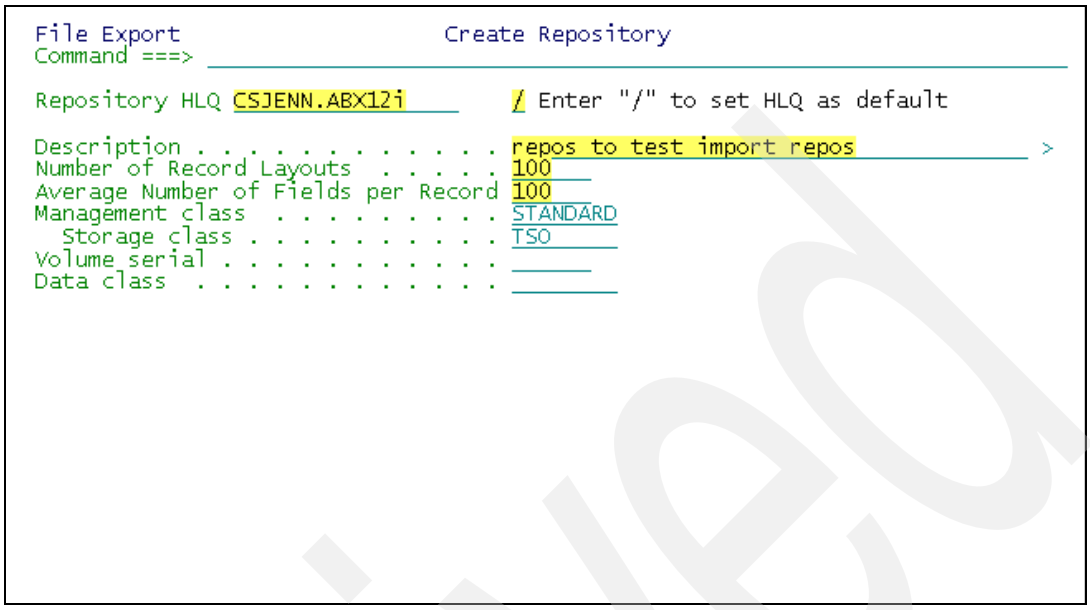


Figure 23-27 Specify the parameters to create a new Repository

When the new Repository has been created, follow the steps in the section titled “Moving the contents of an existing repository” on page 637 to Import the contents of the old Repository into the newly resized Repository.

23.5.3 Renaming an existing repository

If you want to rename an existing Repository at any point after a Repository has been created and filled with content, the Repository’s contents must be first exported, as described in the section “Exporting a repository” on page 632.

Next, create a new Repository with the new required name as described in the section titled “Creating a new repository” on page 636.

Finally, Import the contents of the old Repository into the new Repository as described in the section titled “Moving the contents of an existing repository” below.

23.5.4 Moving the contents of an existing repository

If you want to simply move the contents of one Repository to another Repository, then after completing the instructions under the section titled “Exporting a repository” on page 632, follow these instructions to Import the XML contents into another Repository.

Step 1: On the Manage Repository panel, select option **4** to Import the XML contents from a Repository into a new Repository as shown in Figure 23-28. Press Enter.

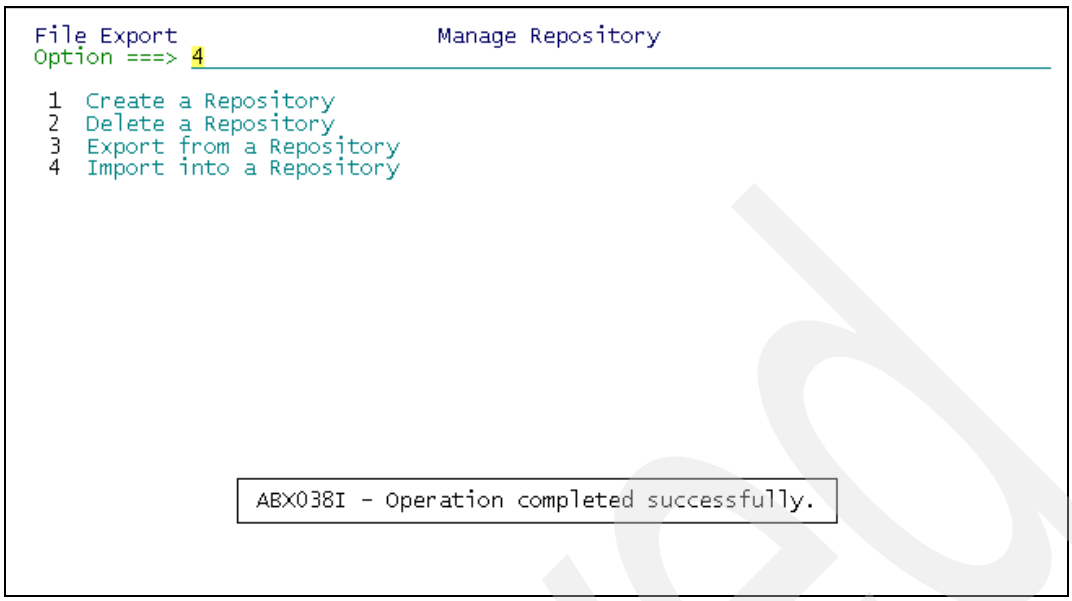


Figure 23-28 Option 4 Import Into A Repository' moves existing content

Step 2: On the Import Repository panel in the field Repository HLQ, enter the name of the Repository that File Export is to import the contents to as shown in Figure 23-29.

The Repository specified in the field Repository HLQ does not have to exist. If it does not exist, File Export creates it and sizes it appropriately for the contents to be imported. If the Repository name specified in the field Repository HLQ already exists, File Export simply imports the contents into the specified Repository.

The XML dataset should be the name of the XML dataset created during the Export Repository process. The contents of the XML dataset from the Export Repository process are loaded into the new Repository that is listed in the field Repository HLQ.

The LOG dataset can be the name of the LOG dataset create during the Export Repository process. If the LOG dataset already exists, File Export simply appends log information from the Import process to the LOG dataset. If the LOG dataset does not exist, File Export creates it.

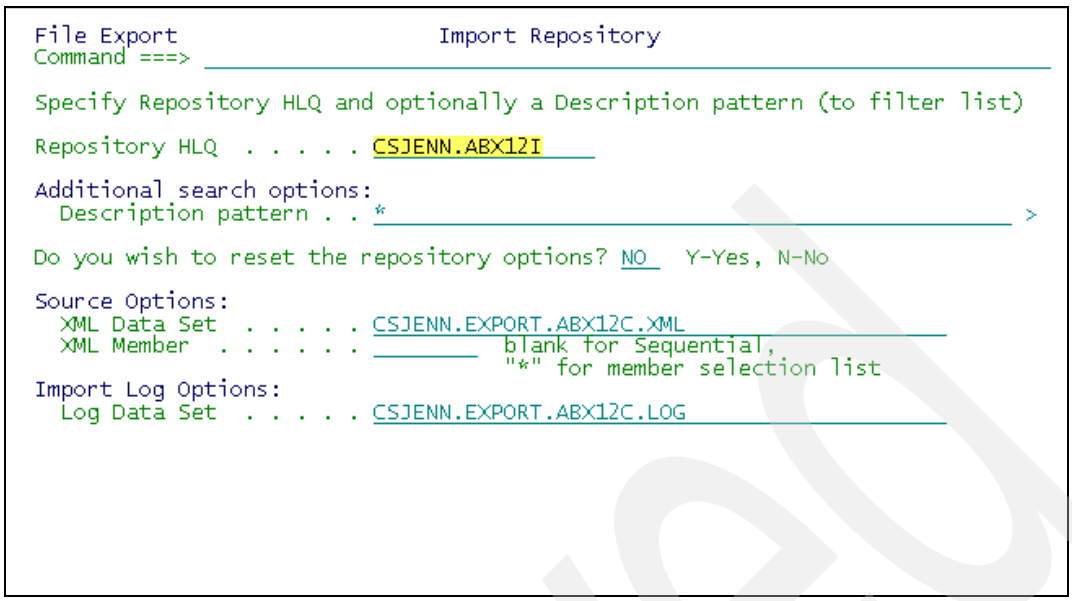


Figure 23-29 Specify the Repository to Import the content into

When all information has been specified on the Import Repository panel, press PF3 to begin the Import process.

The message `Operation Completed Successfully` is displayed when the Import process is done.

Part 7

File Manager for z/OS

In this part of the book, we discuss File Manager for z/OS.

24

Using File Manager

You can use File Manager to display, edit, update, create, copy, compare, print, and erase data. For many of these tasks, you can use templates to enhance File Manager's processing options. In addition to using templates, you can enhance File Manager processing by writing your own custom procedures using DFSORT control statements, REXX programming language, or both.

The File Manager keyword PROC=* is used to indicate that an in-stream routine is being supplied. If using a DFSORT procedure, the first line must be *FASTPROC. If using both DFSORT and REXX procedures, the DFSORT must come first and the first line of the REXX procedure must be *REXXPROC. No identifying tag is required when only REXX is being used.

File Manager utilizes COBOL or PL/I copybooks stored in partitioned data sets and external library management systems to define data set records more efficiently, or dynamically define your own record structure in a template. A COBOL Compiler has been added inside FM to support Customers that have not a license (COBOL) on a Production LPAR

24.1 Performance guidelines

You can easily enhance File Manager with your own procedures, built using either DFSORT or REXX. If there is a choice, always use DFSORT. Performance of many print and copy actions can be enhanced by supplying a procedure containing statements from a subset of the DFSORT statements supported by File Manager, as an alternative to using an equivalent REXX procedure. REXX procedures might be the only option for the more complex select and change types of operations.

DFSORT is used internally wherever possible by File Manager, for copying and printing data sets and for editing large QSAM data files via an auxiliary data set. It is used when all of the following conditions have been met:

- IBM DFSORT R14 is installed (it does not have to be licensed) at the correct maintenance level, PTF UQ90054, APAR PQ68263 for z/OS, or later.
- The input data set is not partitioned.
- The record format of the input data set is not undefined.

File Manager supports only the following DFSORT statements:

- INCLUDE/OMIT
- INREC/OUTREC
- OUTFIL

You cannot use DFSORT for PDSE. If DFSORT is unavailable or unsuitable for a copy or print task, you should consider whether the required function can be achieved using template processing before you resort to a REXX procedure.

When using a REXX procedure, CPU usage and run time can be reduced by observing the following:

- Compiled REXX runs considerably faster than interpreted REXX. By supplying a DD card for the REXX compiler library (if it is installed), instruct File Manager to compile your REXX before running it.
- Always use RETURN in REXX — never use EXIT, because EXIT causes the entire REXX environment to shut down, and it then has to be rebuilt to process the next record.

Be prepared to make two passes through the data, rather than one, if this makes processing more efficient.

24.2 Conventions used

The examples of batch jobs and reports in this chapter adhere to the following conventions:

- In all of the File Manager batch examples presented in this and other chapters, we include STEPLIB references to the File Manager load library and to the COBOL compiler load library. This is done for accuracy and completeness.
- An explicit reference to the File Manager load library is required only if File Manager is not installed in LINKLIST. An explicit reference to the COBOL compiler load library is required only if the COBOL compiler is not installed in LINKLIST, and when copybooks are processed into templates.
- If File Manager and the COBOL compiler are installed in LINKLIST at your site, your system programmer should modify the ISPF skeleton, FMNFTEXC. You can either comment out or remove the STEPLIB statement.

- In all of the File Manager report output examples presented in this and other chapters, we remove the title page, and all pages not pertinent to the example. This is done for brevity.

24.3 Simple copy

System Programmer are usually using File Manager to perform Dataset Copy. This utility is accessed using the Copy utility screen via option **3.3** from the primary option menu, as shown in Figure 24-1.

```
 Process   Options   Help
------------------------------------------------------------------------------
FMNPDSCI ger                    Copy Utility
Command ===> ____________________________________________________________

From Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . 'BAUMBAC.ADLAB.CUSTFILE.KSDS'                  +
   Member  . . . . . . . . ________      (Blank or pattern for member list)
   Volume serial . . . . . ______
   Start key . . . . . . . ____________________________ key or slot
   Skip count  . . . . . . 0_______     number of records to be skipped
   Copy count  . . . . . . ALL_____     number of records to be copied
From Copybook or Template:
   Data set name . . . . . FMN.DATA_____________________________________
   Member  . . . . . . . . AAA3COPY__    (Blank or pattern for member)
Processing Options:
   Copybook/template    Enter "/" to select option
   2  1. Above          /  Batch execution          Advanced member selection
      2. None              Use proc *_____          Skip member name list
      3. Create dynamic    Ignore length mismatch   REXX member selection: P
   _  Edit template        JCL Source format        Directory integrity
   _  Type (1,2,S)         Export mode           /  Report PDS record counts
                                                 _  Binary mode, reclen

 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap      F10=Actions  F12=Cancel
```

Figure 24-1 FMNPDSCI - Initial copy utility Panel

Note: Each FM Panel has been identified by a code (such as FMNPDSCI). It is shown if you type `PANELID` on the Command Line.

This shows a fully qualified or generic data set name, or a path to HFS file or directory. In a generic data set name use the percent sign (%) to represent a single character and an asterisk (*) to represent any number of characters within one qualifier. Two asterisks (**) represent any number of characters within any number of qualifiers.

The TSO prefix (as defined for your user ID) is used as the high-level qualifier for any name that is not enclosed in quotation marks. For performance reasons you should qualify the data set name as much as you can.

If both data set and panel allow for a member specification, the member name can be combined with the data set name in form DSN(Member).

If a path is used, specify the names of the directories starting from the root (absolute path specification). A slash (/) must precede all directory levels and the file name. The path name is case-sensitive.

The choice shown, *Report PDS record counts*, is to copy using batch execution with a report of PDS record counts.

Enter the name of the file you wish to copy to, as shown below. If this file does not exist, you are presented with panels that allow you to allocate it, as shown in Figure 24-2 and Figure 24-3.

```
 Process   Options   Help
-------------------------------------------------------------------------------
FMNPDSCO  BAUMBAC.ADLAB.CUSTFILE.KSDS
Command ===> ___________________________________________________________________

To Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . 'denardi.book2006.custfile'                        +
   Member name (or mask) . ________
   Volume serial . . . . . ______

Processing Options:
   Disposition             Execution "/" options          ISPF Packing
   1  1. Old or Reuse         Replace members             1  1. Asis
      2. Mod                  Binary mode, reclen            2. Pack
                           _  Stats Off                      3. Unpack
                                                             4. None
                                                             5. Skip









 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap      F10=Actions  F12=Cancel
```

Figure 24-2 Specifying the output Dataset

```
 Process   Options   Help
FMNPALLN DENARDI.BOOK2006.CUSTFILE
Command ===>

New Data Set Organization:
   Select option          Instructions
   1 1. KSDS              The above data set does not exist.
     2. ESDS              To define or allocate a new data set select a data
     3. RRDS              set organization and press ENTER or press PF3/EXIT
     4. VRRDS             or PF12/CANCEL to return without allocation.
     5. LDS
     6. Non VSAM
                          For a new data set, enter a data set name
                          below to copy existing allocation attributes.


Existing Data Set:
   Like data set . . . . . 'BAUMBAC.ADLAB.CUSTFILE.KSDS'
   Volume serial . . . . . ______




 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap      F10=Actions  F12=Cancel
```

Figure 24-3 Allocation Panel being displayed to accommodate data set not found

When File Manager determines that the output data set for a function does not exist, it displays a series of panels that enable you to create a new data set as shown in Figure 24-4.

```
 Process   Options   Help
FMNPSCKD ger                      VSAM Define
Command ===>

VSAM Catalog Entry:
   Data set name  . . 'DENARDI.BOOK2006.CUSTFILE'
   Catalog ID . . . . ____________________________________________
                                                          More:     +
Basic Information:
   VSAM data type . . KSDS       Expiration date  . ________
   Data component . . ____________________________________________
   Index component  . ____________________________________________
VSAM Cluster Attributes:
   Key length . . . . 6          Key offset . . . . 0
   CI size  . . . . . 26624      size of the data control intervals
   Buffer space . . . 53760      buffer space to be allocated at open time
   Shr cross region . 1          cross system . . . 3          Reuse . . . . N
   Recovery . . . . . Y          Spanned  . . . . . N          Erase . . . . N
   Writecheck . . . . N
VSAM Data Allocation:
   Allocation unit  . CYL        REC, KB, MB, TRK, or CYL
   Space primary  . . 1          secondary  . . . . 1
   Recsize average  . 2048       maximum  . . . . . 2048
   Freespace  % CI  . 0          % CA . . . . . . . 0
   Volume serial(s) . ______     ______  ______  ______  ______
VSAM Key Definition:
   Low key 1 . . . ______________    High key 1  . . ______________
    (he
 F1=Hel  Press ENTER to define the catalog entry or EXIT to cancel   own
 F9=Swap      F10=Actions  F12=Cancel
```

Figure 24-4 VSAM displayed Panel when data set not found

VSAM entry detail

When you invoke the alter, define, or list service or line command for a VSAM catalog entry, detailed catalog information is displayed for the selected entry, as shown in Figure 24-5. The specific information displayed varies according to the type of VSAM entry selected. For most fields you can view a description of the field by placing the cursor on the field and pressing F1 (Help).

The usage depends on the command:

ALTER — Use scroll commands to display the information available, change the IDCAMS parameters as desired, and press Enter to update the catalog. If the data or index name is not specified, it is generated.

DEFINE — Enter the name of the new entry, use scroll commands to display the information available, change the IDCAMS parameters as desired, and press Enter to define the entry.

LIST — Produce a list.

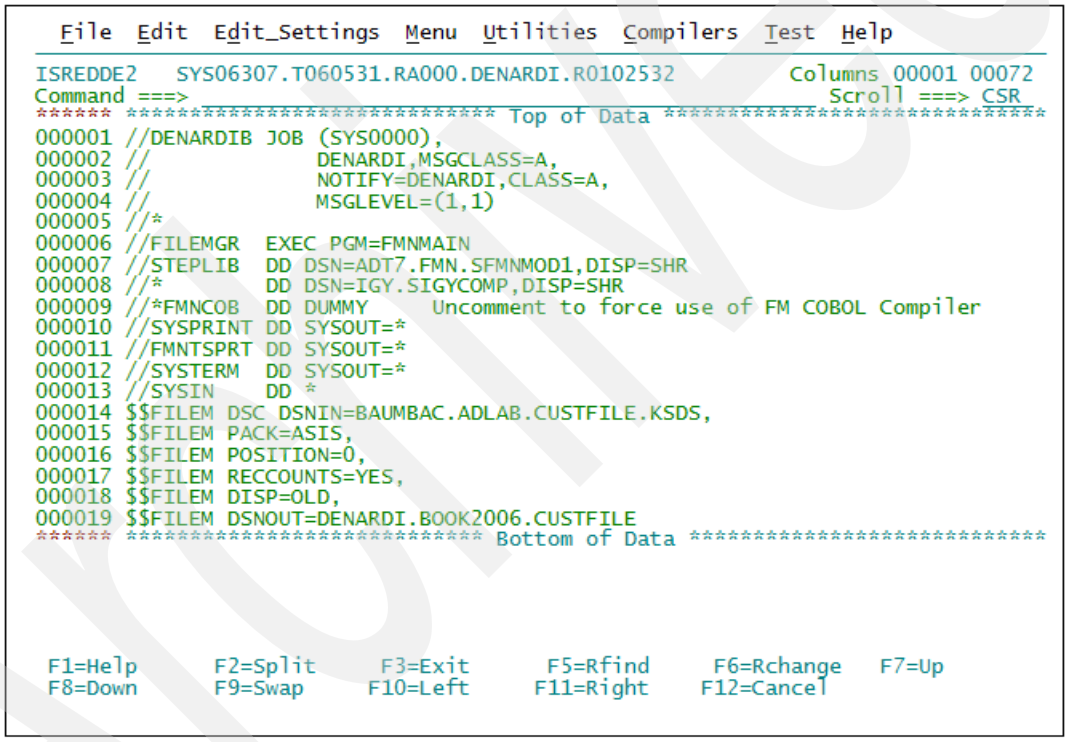


Figure 24-5 Temporary data set where JOB JCL ready for submission, performing in batch mode

Successful copying of the file is reported, and DFSORT has been automatically used internally by File Manager to perform the copy as shown in Figure 24-6.

```
  Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY DENARDIB JOB01236  DSID   102 LINE 1       COLS 02- 81
 COMMAND INPUT ===>                                           SCROLL ===> CSR
IBM File Manager for z/OS


 DEFAULT SET FUNCTION VALUES:

PRINTOUT=SYSPRINT  HEADERPG=YES  ASCII=BOTH     RECLIMIT=(1,*)
PRINTLEN=132       PAGESKIP=NO   PAD=OFF        EOD=/*
PAGESIZE=60        DATAHDR=YES   DBCSPRT=OFF    LANGUAGE=ENGLISH
PRTTRANS=ON        DUMP=UPDOWN   TAPELBL=SL     CYLHD=ABSOLUTE
SMFNO=253          PRTDISP=MOD   USEIOX=DISABLE IOX=
TEMPHLQ=
ABENDCC=NONE

IBM File Manager for z/OS
$$FILEM DSC DSNIN=BAUMBAC.ADLAB.CUSTFILE.KSDS,
$$FILEM PACK=ASIS,
$$FILEM POSITION=0,
$$FILEM RECCOUNTS=YES,
$$FILEM DISP=OLD,
$$FILEM DSNOUT=DENARDI.BOOK2006.CUSTFILE
175 record(s) read
175 record(s) copied: 0 truncated: 0 fields truncated
******************************** BOTTOM OF DATA ********************************


 PF 1=HELP      2=SPLIT     3=END       4=RETURN    5=IFIND     6=BOOK
 PF 7=UP        8=DOWN      9=SWAP     10=LEFT     11=RIGHT    12=RETRIEVE
```

Figure 24-6 Results displayed within the SYSPRINT of executed job

24.4 How to create files using another as a model

When modifications to your application require you to create a new file, or when a testing effort requires a clean copy, you can model it based on an existing file that has common attributes. The following scenario illustrates how to create a file using another as model:

- Access FM/Base and select Catalog Service (option **3.4**) and list the files.
- Select a file that has attributes that resemble those of the file that you want to create, as shown in Figure 24-7, Figure 24-8 and Figure 24-9.

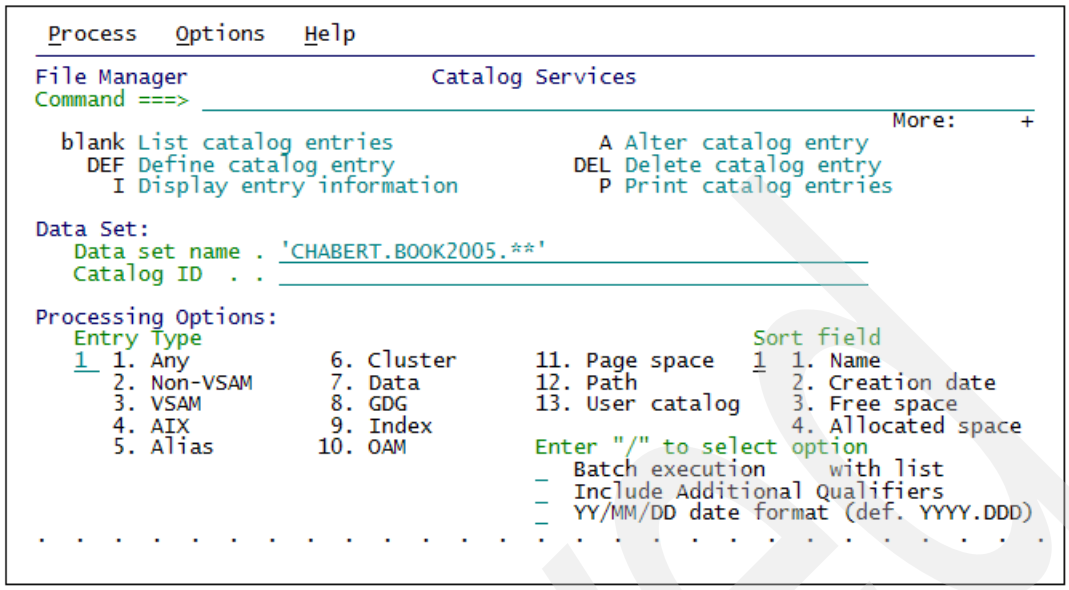


Figure 24-7 Display the files to choose from, part 1 of 3

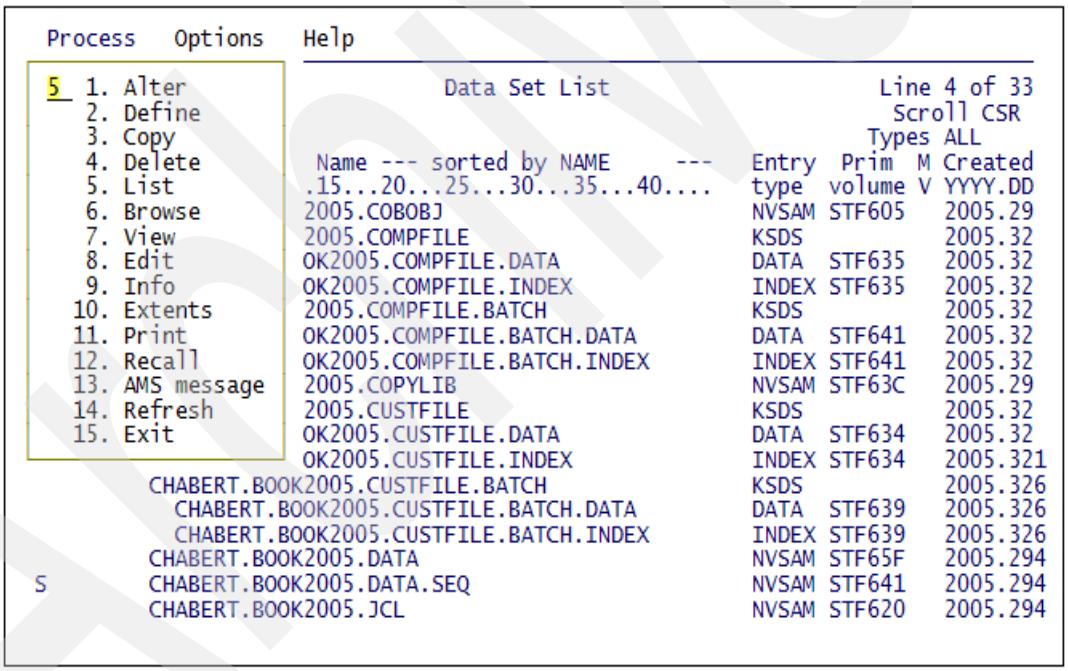


Figure 24-8 Display the files to choose from, part 2 of 3

Note: If you are going to use the pull-down menus, your cursor must be on the same line as the data set name. You can either scroll the list until the file you want to work with is the first one displayed, or position your cursor and press PF6 (PROCESS) to display the process pull-down.

Type LIST in the line commands area or select the Process pull-down and select **List**. The Entry Detail panel, as shown in Figure 24-9, is displayed with information for the current file.

```
 Process   Options   Help
File Manager              Non-VSAM Entry Detail                End of data
Command ===>

Catalog Entry:
   Data set name  . . 'CHABERT.BOOK2005.DATA.SEQ'
   Catalog ID . . . . 'SYS1.ICFCAT.VSTF601'

Additional information available with Extents(F11) and ASsocs(F6) commands.
                                                                More:   -
   Creation date . . . 2005/10/21 (294)
   Expiration date . . ***None***
   Referenced date . . 0000/00/00 (000)
   Change indicator  . NO

Extended Attributes:
   Data attributes    Extended  . N      Compressed  . . . N
                      Striped . . N      Stripe count  . . -

Associations:
   GDG base . . . . .
   Alias  . . . . . .
. . . . . .
```

Figure 24-9 Display the files to choose from, part 3 of 3

When you finish checking the definitions inside your selected file, press PF3 to return to the Data Set List panel, as shown in Figure 24-10, and select Define in the Process pull-down as option. If you type DEFINE in the line command, you obtain the same result.

```
 Process   Options   Help
 2  1. Alter              Data Set List                  Line 19 of 33
    2. Define                                             Scroll CSR
    3. Copy                                              Types ALL
    4. Delete    Name --- sorted by NAME     ---   Entry  Prim  M Created
    5. List      .15...20...25...30...35...40....   type  volume V YYYY.DD
    6. Browse    2005.DATA.SEQ                      NVSAM STF641   2005.29
    7. View      2005.JCL                           NVSAM STF620   2005.29
    8. Edit      2005.LOAD                          NVSAM STF63C   2005.29
    9. Info      2005.PDF                           NVSAM STF607   2005.31
   10. Extents   2005.PLILIST                       NVSAM STF634   2005.29
   11. Print     2005.PLIMOD                        PDSE  STF641   2005.29
   12. Recall    2005.PLIOBJ                        NVSAM STF64F   2005.29
   13. AMS message 2005.SIDEFILE                    NVSAM STF634   2005.29
   14. Refresh   2005.SOURCE                       NVSAM STF632   2005.29
   15. Exit      2005.SYSDEBUG                      PDSE  STF610   2005.31
                 2005.TRANFILE                      NVSAM STF607   2005.306
         CHABERT.BOOK2005.TRANFILE.DB2              NVSAM STF607   2005.321
         CHABERT.BOOK2005.TRANFILE.IMS              NVSAM STF65C   2005.321
         CHABERT.BOOK2005.TRANFILE.NEW              NVSAM STF65F   2005.321
         CHABERT.BOOK2005.TRANFILE.VSAM             NVSAM STF631   2005.321
         ****  End of data  ****
```

Figure 24-10 Using PROCESS pull-down to define a new object

During this stage you can select the type of data set entry. We choose to allocate the dataset using the option **2** as shown in Figure 24-11. Non-VSAM allocate is shown in Figure 24-12.

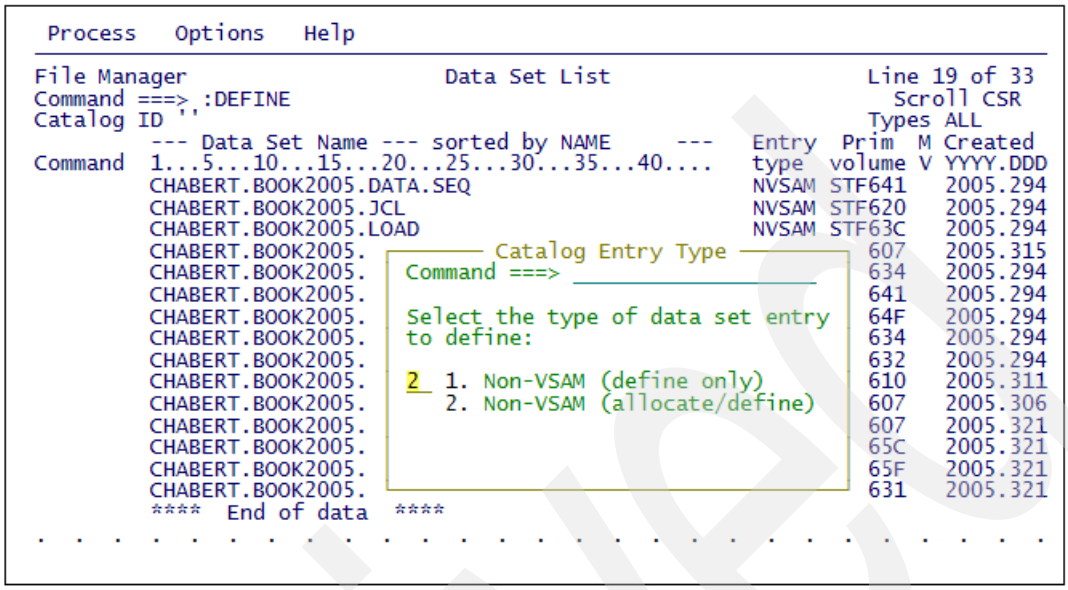


Figure 24-11 Display file to choose 3.3

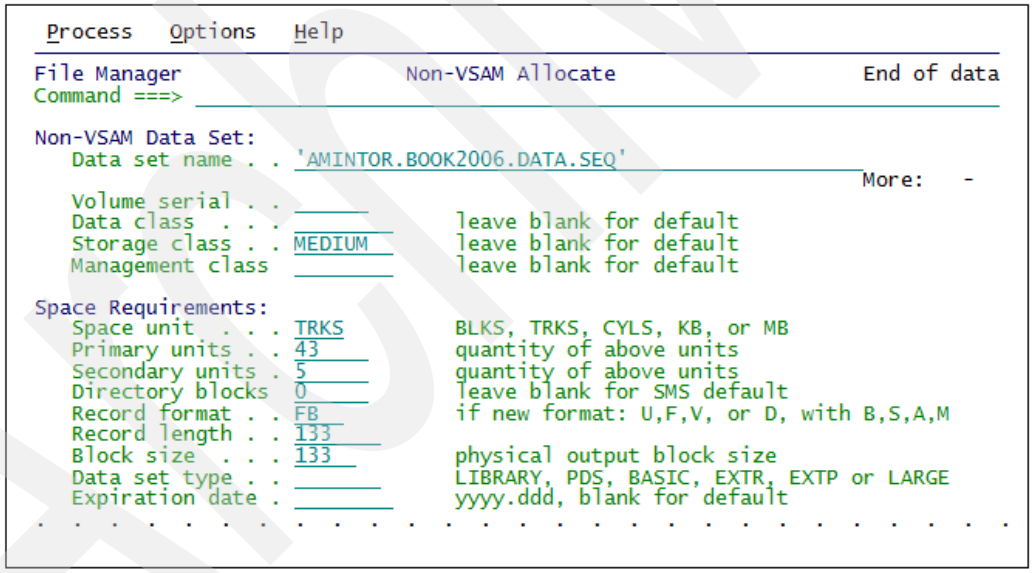


Figure 24-12 Define panel, updating data set to new one

Change the data name, as well as the data and the index names, to the new file's corresponding names, by typing over the existing information.

Note: In the lab, if we did not erase the value in the Catalog ID field, we could not locate the file without explicitly pointing to the catalog and the volume. Have your system programmer validate the rules at your site with your storage management group during a post-installation review.

You can modify any of the other file attributes as required, before pressing ENTER to confirm.

24.5 Define an alternate index for a VSAM (AIX) file

From time to time, developers have to gather information, modify data structures, or even add new fields or keys to resolve requests. Because business requirementss and processes change, customers must then adapt to changing scenarios.

Often, when we modify or create a new index, the processes results and the search activities could be better than before and sometimes the response time improves together to the performances.

This next section describes step by step how to build an alternate index (AIX) for an existing VSAM file. To create a new index we have to perform the following steps:

1. Access FM/Base and select Catalog Service (option **3.4**) and list the files.
2. Type the data set name (and optionally the catalog name) for the existing entry. Select the Entry Type **4** to process a new alternate index (AIX).
3. To run the define function in batch, select the Batch execution option.
4. Enter DEF on the Command line as shown in Figure 24-13. Press Enter.

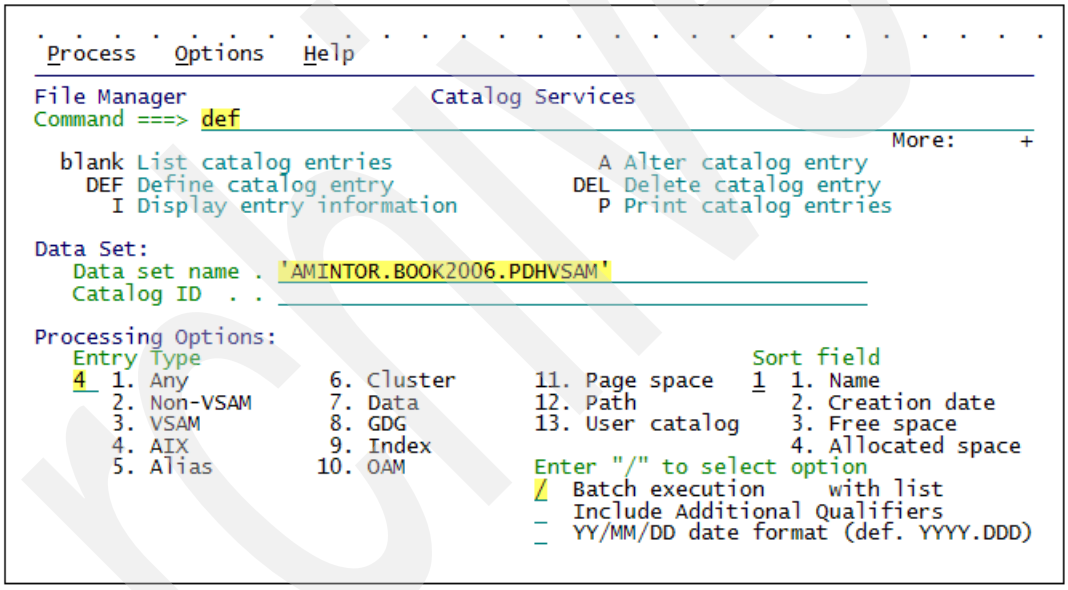


Figure 24-13 Panel to define a new Catalog Entry

When you are defining an alternate index for a VSAM file (AIX), you can also request that the newly defined AIX is built following the define, or you can delete an existing AIX, redefine and rebuild it. The define, build, and redefine functions are available for online and batch processing. In this section you define an alternate index for a VSAM file (AIX). When you choose the Entry Type **4**, File Manager displays a pop-up panel as shown in Figure 24-14.

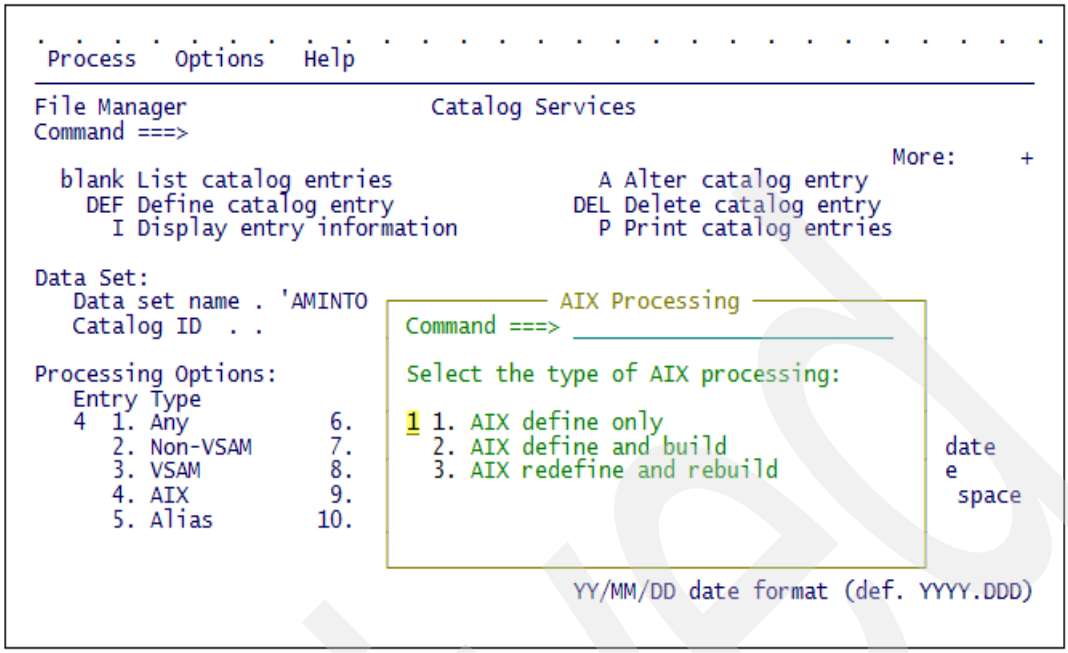


Figure 24-14 Panel to process an Alternate index

- Select the type of AIX processing as follows:
  - Select **1** if you want to define only an AIX.
  - Select **2** to define and build an AIX.
  - Select **3** to delete, recreate, and then rebuild an existing AIX (redefine and rebuild).

In this case we are selecting option **1** to define a new alternate index. Type ENTER, and the define panel for the entry type of your existing data set is displayed, containing the information copied from your model. For example, if your existing data set was a VSAM KSDS file, the VSAM Define panel is displayed as shown in Figure 24-15.

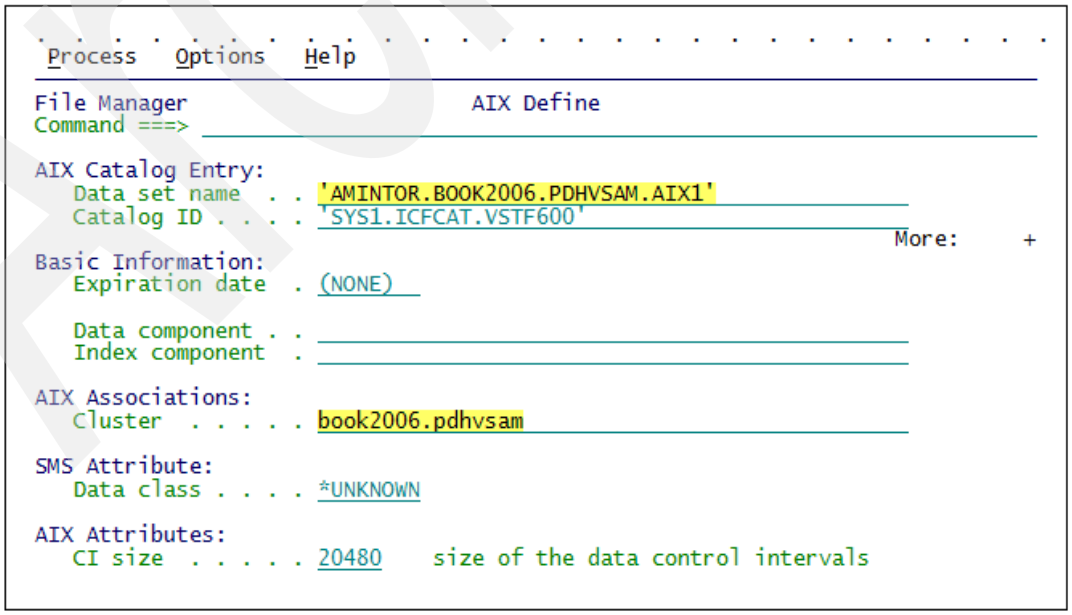


Figure 24-15 Panel to define Cluster name and relate VSAM

- Alter the contents of the Data set name field, according to your new index name, to specify a new, unique data set name.

Regarding the Catalog ID field, we have two choices:

1. If SMS is active on the system, the new value for the catalog ID is copied from the model data set but prefixed with an asterisk (*) to indicate a “comment” value. this value is ignored unless you remove the asterisk.
2. If SMS is not active on the system, the catalog name is copied from the model data set. Do one of the following actions:
    - Blank the field out, to allow the system to assign your data set using the catalog selection order for IDCAMS DEFINE.
    - Remove the asterisk (*) from the existing entry, to assign your data set to the same catalog. If the specified catalog is not the same as the related user catalog for this Entry type, a warning message is issued. You can exit from the message (F3) and change or blank out the catalog name, or you can press Enter to continue using the initial catalog name. If you choose to continue, you have to supply the catalog name in all future searches for the Entry.
    - Specify an alternative Catalog ID.

In this case, we assign the same Catalog ID from the existing VSAM file select before.

To modify the Index and Data Component field, you can:

- For VSAM data sets, alter the component name or names, listed under the Basic Information heading, to unique names suited to your data set.
- Remove or make blank out the component name or names listed under the Basic Information heading. File Manager generates new component names based upon the following AMS rules:
    - IF the last qualifier of the Entry data set name is CLUSTER, it replaces the last qualifier with DATA for the data component and INDEX for the index component.
    - ELSE if the cluster name is less than or equal to 38 characters, then append .DATA to the end of the cluster name for the data component and .INDEX for the index component.
    - ELSE if the cluster name is between 39 and 42 characters inclusive, then append .D to the end of the cluster name for the data component and .I for the index component.
    - ELSE if the name is longer than 42 characters, and the last qualifier is not CLUSTER, use the first (N-1) qualifiers of the cluster, alternate index, or user catalog name up to the first four qualifiers, and append as many 8-character qualifiers as necessary to produce a 5-qualifier name.

After a name is generated, AMS searches the catalog to ensure that the name is unique. If a duplicate name is found, AMS continues generating new names until a unique name is produced.

In this panel, we have removed with blank the name for these field, because these fields (which are not empty) normally assumed the value according to the existing entry. Alter you press PF8 to modify or keep the others remaining parameters (copied from the existing entry) to suit your new entry.

- Press Enter to process your define request and the next step generates the following JCL, as shown in Figure 24-16.

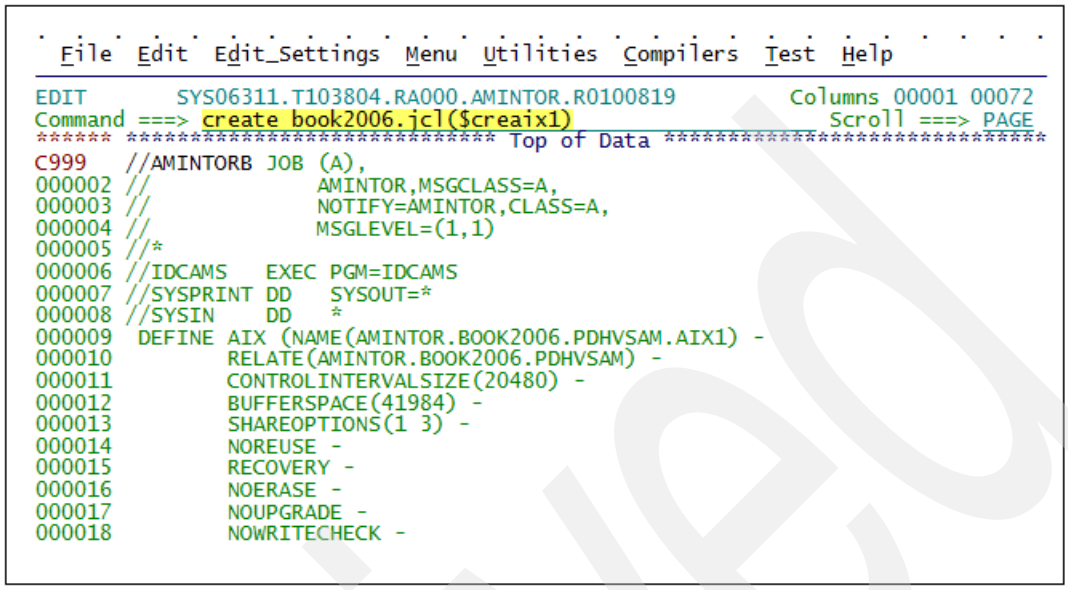


Figure 24-16 Alternate Index definitions

You can save the JCL into a work's library. The following sample, as shown in Example 24-1, shows the Job and SYSOUT.

Example 24-1 Report of DSG low-value record creation

```
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (A),
000002 //            AMINTOR,MSGCLASS=A,
000003 //            NOTIFY=AMINTOR,CLASS=A,
000004 //            MSGLEVEL=(1,1)
000005 //*
000006 //IDCAMS   EXEC PGM=IDCAMS
000007 //SYSPRINT DD   SYSOUT=*
000008 //SYSIN    DD   *
000009  DEFINE AIX (NAME(AMINTOR.BOOK2006.PDHVSAM.AIX1) -
000010         RELATE(AMINTOR.BOOK2006.PDHVSAM) -
000011         CONTROLINTERVALSIZE(20480) -
000012         BUFFERSPACE(41984) -
000013         SHAREOPTIONS(1 3) -
000014         NOREUSE -
000015         RECOVERY -
000016         NOERASE -
000017         NOUPGRADE -
000018         NOWRITECHECK -
000019             ) -
000020         DATA( -
000021           CYLINDERS(1 0) -
000022           RECORDSIZE(4089 4089) -
000023           FREESPACE(0 0) -
000024           VOLUME(STF624    ) -
000025           KEYS(3 1) -
000026           UNIQUEKEY -
000027             ) -
000028         INDEX( -
000029           CONTROLINTERVALSIZE(1024) -
```

```
000030           TRACKS(1 0) -
000031           VOLUME(STF624    ) -
000032             ) -
000033         CATALOG(SYS1.ICFCAT.VSTF600)
****** **************************** Bottom of Data ****************************


SDSF OUTPUT DISPLAY AMINTORB JOB03619  DSID     4 LINE 1       COLUMNS 02- 81
 COMMAND INPUT ===>                                            SCROLL ===> CSR
ICH70001I AMINTOR  LAST ACCESS AT 11:19:38 ON TUESDAY, NOVEMBER 7, 2006
IEF236I ALLOC. FOR AMINTORB IDCAMS
IEF237I JES2 ALLOCATED TO SYSPRINT
IEF237I JES2 ALLOCATED TO SYSIN
IEF142I AMINTORB IDCAMS - STEP WAS EXECUTED - COND CODE 0000
IEF285I   AMINTOR.AMINTORB.JOB03619.D0000102.?         SYSOUT
IEF285I   AMINTOR.AMINTORB.JOB03619.D0000101.?         SYSIN
IEF373I STEP/IDCAMS  /START 2006311.1249
IEF374I STEP/IDCAMS  /STOP  2006311.1249 CPU    0MIN 00.02SEC SRB    0MIN 00.00S
IEF375I  JOB/AMINTORB/START 2006311.1249
IEF376I  JOB/AMINTORB/STOP  2006311.1249 CPU    0MIN 00.02SEC SRB    0MIN 00.00S
IDCAMS  SYSTEM SERVICES                                       TIME: 12:49:05

 DEFINE AIX (NAME(AMINTOR.BOOK2006.PDHVSAM.AIX1) -
        RELATE(AMINTOR.BOOK2006.PDHVSAM) -
        CONTROLINTERVALSIZE(20480) -
        BUFFERSPACE(41984) -
        SHAREOPTIONS(1 3) -
        NOREUSE -
        RECOVERY -
        NOERASE -
        NOUPGRADE -
        NOWRITECHECK -
            ) -
        DATA( -
          CYLINDERS(1 0) -
          RECORDSIZE(4089 4089) -
          FREESPACE(0 0) -
          VOLUME(STF624    ) -
          KEYS(3 1) -
          UNIQUEKEY -
            ) -
        INDEX( -
          CONTROLINTERVALSIZE(1024) -
          TRACKS(1 0) -
          VOLUME(STF624    ) -
            ) -
        CATALOG(SYS1.ICFCAT.VSTF600)
IDC0508I DATA ALLOCATION STATUS FOR VOLUME STF648 IS 0
IDC0509I INDEX ALLOCATION STATUS FOR VOLUME STF648 IS 0
IDC0512I NAME GENERATED-(D) AMINTOR.BOOK2006.PDHVSAM.AIX1.DATA
IDC0512I NAME GENERATED-(I) AMINTOR.BOOK2006.PDHVSAM.AIX1.INDEX
IDC0181I STORAGECLASS USED IS MEDIUM
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
******************************** BOTTOM OF DATA ********************************
```

We have generated a new Alternate Index Unique. In the next panel, as shown in Figure 24-17, when you try to insert a duplicate key A25, you can see that you receive a VSAM error message: `VSAM PUT RC X"08". error Code (duplicate key).`

```
 Process   Options   Help
------------------------------------------------------------------------------
Edit            AMINTOR.BOOK2006.PDHVSAM.AIX1                 Rec 1 of 10
Command ===>                                                   Scroll CSR
VSAM PUT RC X"08", Error Code X"08" (duplicate key)
       Col 1         Insert length 4089
      ----+<=>-1----+----2----+----3----+----4----+----5----+----6----+----7--
000001 0000 A25Bianchi Franco         corso emanuele f.     145
000002 0000 A02Marianna Rebecca       via del mare           44   Roma
000003 0000 A03Cinzia Margherita      piazzale ernerio       44   Roma
000004 0000 A04Carmine Giulio         via del mare           44   Roma
000005 0000 A05Viaggiante Ada         piazza bra            747   Verona
000006 0000 A06Marylen Chabert        via del pino           98   Latina
000007 0000 A08Bontega Mario          via Vicenza            98   Varese
000008 0000 A22Alessi Giorgio         viale trastevere      118   Roccagorga
000009 0000 A23Zaffarani Marco        via Po                 77   Latina
000010 0000 A25D'antrassi Danila      St Ponte lungo         69   Montpellier
000011 ****  End of data  ****
```

Figure 24-17 VSAM error code X"08" (duplicate key)

24.6 Selective copying from a sequential to a VSAM dataset

Many data set copy requirements are selective in that only some of the records from the input data set are required in the output. You can still use DFSORT for this, simply by providing a DFSORT procedure. To indicate the requirement for a procedure, select **Use proc**, as shown in Figure 24-18. The asterisk (*) indicates that the procedure is dynamically created.

```
 Process   Options   Help
------------------------------------------------------------------------------
File Manager                  Copy Utility
Command ===>

From Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . 'AMINTOR.BOOK2006.TRANFILE.SEQ'                +
   Member  . . . . . . . .        (Blank or pattern for member list)
   Volume serial . . . . .
   Start key . . . . . .                                  key or slot
   Skip count  . . . . . .        number of records to be skipped
   Copy count  . . . . . ALL      number of records to be copied
From Copybook or Template:
   Data set name . . . . . AMINTOR
   Member  . . . . . .            (Blank or pattern for member)
Processing Options:
   Copybook/template    Enter "/" to select option
   2  1. Above          /  Batch execution          Advanced member selection
      2. None           /  Use proc *               Skip member name list
      3. Create dynamic    Ignore length mismatch   REXX member selection: P
   _  Edit template        JCL Source format        Directory integrity
   _  Type (1,2,S)         Export mode           /  Report PDS record counts
                                                 _  Binary mode, reclen
```

Figure 24-18 The panel shows how to Select a procedure during a selective Dataset copy file

When you press Enter, FM provides you with the panel shown in Figure 24-19, in which the required selection statements can be entered.

Figure 24-19 An example of statement specification

Proceeding with the copy, Figure 24-20 shows the allocation and creation dataset stage.

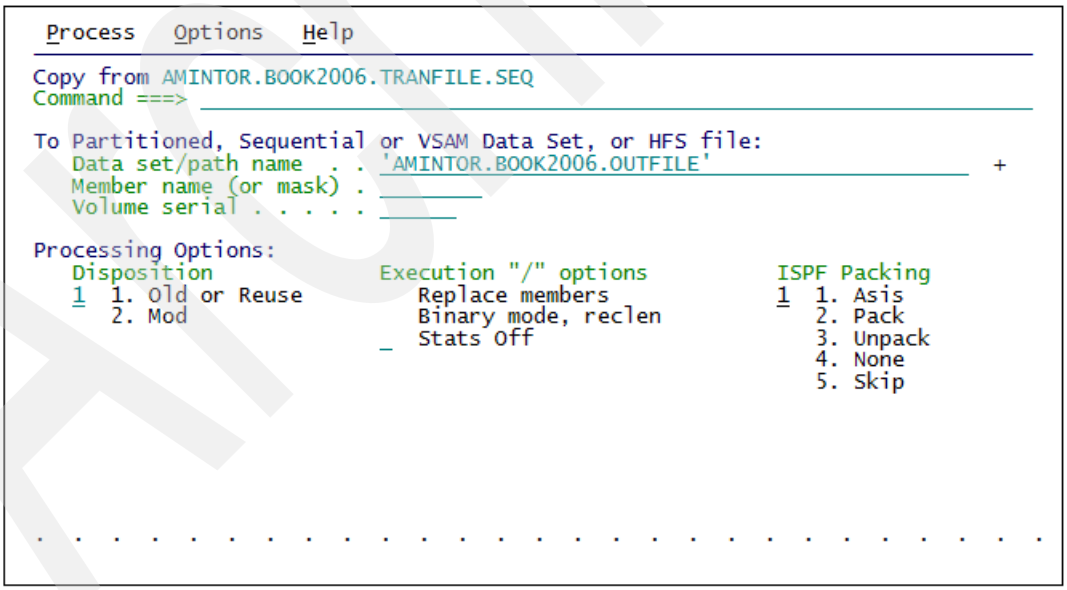


Figure 24-20 Target dataset selection

If the data set does not exist, a panel is automatically displayed in order to create the data set and proceed with the copy. The results is shown in Figure 24-21.

```
  Process    Options    Help
 ------------------------------------------------------------------------------
 File Manager                       VSAM Define
 Command ===> _________________________________________________________________
 Press ENTER to define the catalog entry or EXIT to cancel
 VSAM Catalog Entry:
    Data set name  . . 'AMINTOR.BOOK2006.OUTFILE'
    Catalog ID . . . .
                                                                More:     +
 Basic Information:
    VSAM data type . . KSDS       Expiration date  .
    Data component . .
    Index component  .
 VSAM Cluster Attributes:
    Key length . . . . 81         Key offset . . . . 0
    CI size  . . . . . 4096       size of the data control intervals
    Buffer space . . . 13824      buffer space to be allocated at open time
    Shr cross region . 2          cross system . . . 3         Reuse . . . . N
    Recovery . . . . . Y          Spanned  . . . . . N         Erase . . . . N
    Writecheck . . . . N
 VSAM Data Allocation:
    Allocation unit  . CYL        REC, KB, MB, TRK, or CYL
    Space primary  . . 2          secondary  . . . . 0
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
```

Figure 24-21   target Dataset selection

In this sample, the Input and Output data sets do not have the same definitions. While the Input file is sequential, the Output dataset has been generated as a VSAM file. Since we set the *batch execution*, we receive the following display as shown in Figure 24-22.

```
   File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 -------------------------------------------------------------------------------
 EDIT       SYS06308.T160301.RA000.AMINTOR.R0112217         Columns 00001 00072
 Command ===> ________________________________________________ Scroll ===> PAGE
 000006 //FILEMGR  EXEC PGM=FMNMAIN
 000007 //*TEPLIB  DD DSN=FMN.SFMNMOD1,DISP=SHR
 000008 //*        DD DSN=IGY.SIGYCOMP,DISP=SHR
 000009 //*FMNCOB  DD DUMMY      Uncomment to force use of FM COBOL Compiler
 000010 //SYSPRINT DD SYSOUT=*
 000011 //FMNTSPRT DD SYSOUT=*
 000012 //SYSTERM  DD SYSOUT=*
 000013 //SYSIN    DD *
 000014 $$FILEM SHOWSORT
 000015 $$FILEM DSC DSNIN=AMINTOR.BOOK2006.TRANFILE.SEQ,
 000016 $$FILEM PACK=ASIS,
 000017 $$FILEM POSITION=0,
 000018 $$FILEM RECCOUNTS=YES,
 000019 $$FILEM DISP=MOD,
 000020 $$FILEM DSNOUT=AMINTOR.BOOK2006.OUTFILE,
 000021 $$FILEM PROC=*
 000022 *FASTPROC
 000023  INCLUDE COND=(6,11,CH,EQ,C'ALBANE')
 000024 /+
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
```

Figure 24-22   Sample JCL provided from the FM utility

Here you see the results of the selection of only those records with ALBANE in columns 6 to 11. Only 3224 records have been copied from 346308, as shown in Figure 24-23.

```
  Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY AMINTORB JOB03216  DSID   102 LINE 35      COLS 02- 81
 COMMAND INPUT ===>                                          SCROLL ===> PAGE
ICE156I 0 MAIN STORAGE ABOVE 16MB = (6234096,6234096)
ICE127I 0 OPTIONS: OVFLO=RC0 ,PAD=RC0 ,TRUNC=RC0 ,SPANINC=RC16,VLSCMP=N,SZERO=Y,
ICE128I 0 OPTIONS: SIZE=6291456,MAXLIM=1048576,MINLIM=450560,EQUALS=N,LIST=Y,ERE
ICE129I 0 OPTIONS: VIO=N,RESDNT=ALL ,SMF=NO   ,WRKSEC=Y,OUTSEC=Y,VERIFY=N,CHALT=
ICE130I 0 OPTIONS: RESALL=4096,RESINV=0,SVC=109 ,CHECK=Y,WRKREL=Y,OUTREL=Y,CKPT=
ICE131I 0 OPTIONS: TMAXLIM=6291456,ARESALL=0,ARESINV=0,OVERRGN=16384,CINV=Y,CFW=
ICE132I 0 OPTIONS: VLSHRT=N,ZDPRINT=Y,IEXIT=N,TEXIT=N,LISTX=N,EFS=NONE    ,EXITC
ICE133I 0 OPTIONS: HIPRMAX=OPTIMAL,DSPSIZE=MAX ,ODMAXBF=0,SOLRF=Y,VLLONG=Y,VSAMI
ICE235I 0 OPTIONS: NULLOUT=RC0
ICE084I 0 EXCP ACCESS METHOD USED FOR SYS00002
ICE084I 0 EXCP ACCESS METHOD USED FOR SYS00001
ICE751I 1 EF-K10929 F0-Q84357 E8-K11698
ICE090I 0 OUTPUT LRECL = 80, BLKSIZE = 27920, TYPE = FB
ICE055I 0 INSERT 0, DELETE 343084
ICE054I 0 RECORDS - IN: 346308, OUT: 3224
ICE052I 0 END OF DFSORT
346308 record(s) read
3224 record(s) successfully copied
******************************** BOTTOM OF DATA ********************************
```

Figure 24-23 The SYSOUT with result and information

24.7 How to perform a global find and replace in a PDS

File Manager's Find/Change Utility allows you to search for or change a string in a partitioned data set (PDS), a VSAM data set, or a sequential file.

Scenario: Global update of JCL

This section describes both online and batch approaches to the global update of a JCL deck.

Using File Manager online

This scenario presents the succession of panels to change all references of one user ID to another in all the members of a library. You reach the File/Change Utility panel by selecting option **3.6** on the Main panel of File Manager and entering data in the appropriate fields, as shown below.

This scenario describes how to change all the members (in the Member field, '*' for all the members) of the library specified in Input, panel field: "Data set name".

During the following sample, since all of these members are source JCL, enter a forward slash (/) in the JCL Source format field, as shown in Figure 24-24.

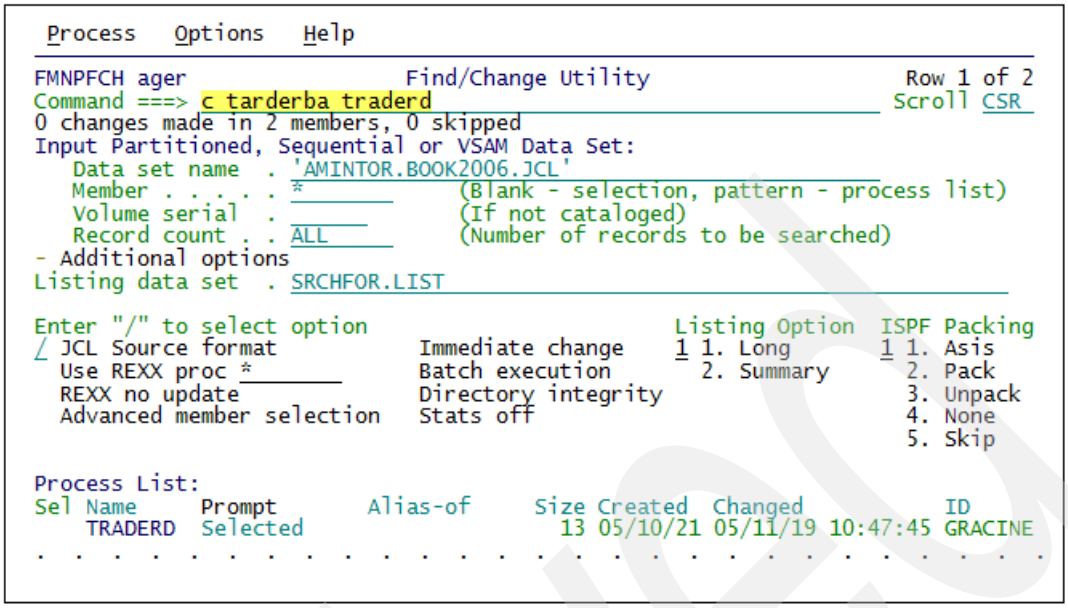


Figure 24-24 Find/change utility panel

3. Enter the change command C TRADERBA TRADERD on the command line and press Enter. As we did not ask for immediate change, we get a confirmation panel displaying all of the proposed modifications, as shown in Figure 24-25.

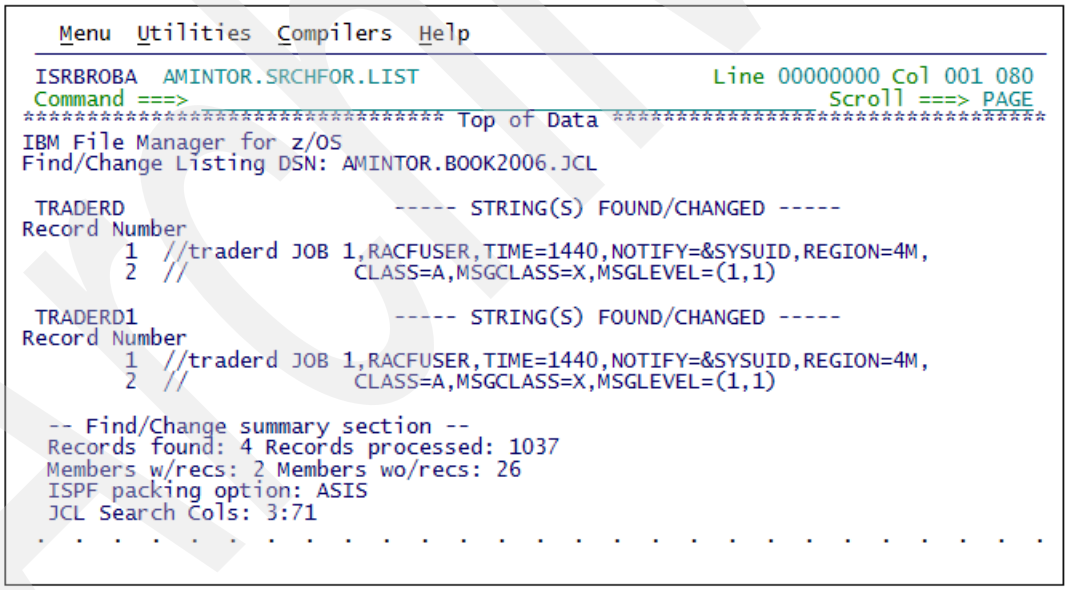


Figure 24-25 Proposed modification

To confirm the changes, press F3, followed by Enter to save, or F3 to cancel as shown in Figure 24-26.

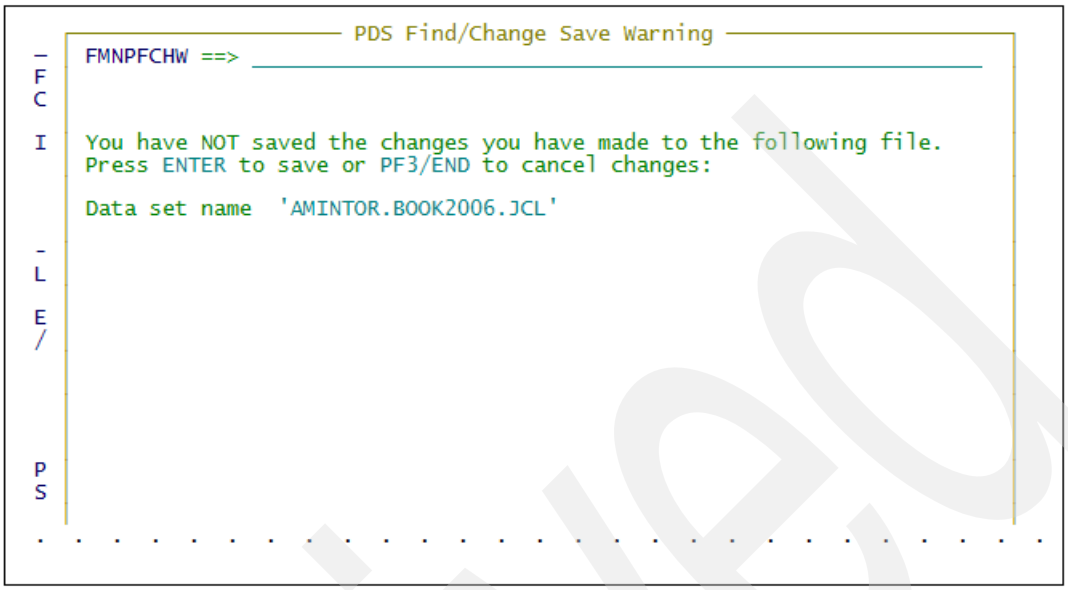


Figure 24-26 Confirmation changes

File Manager completes the job, as shown in Figure 24-27. We see that one change has been made in two members.

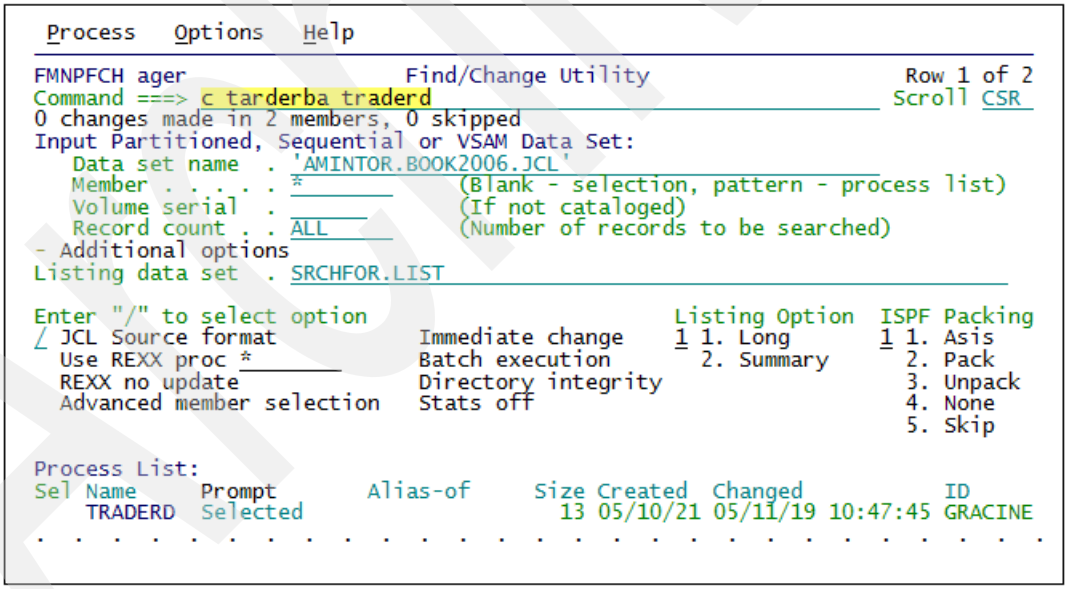


Figure 24-27 Notify Panel if press Enter to save

You can then refer to the listing data set, SRCHFOR.LIST, for the details of the changes.

Tip: The first time you ask File Manager to perform a find/change, request a *batch execution* (insert a slash (/) in this field on the panel referenced in the following figures) and save the displayed JCL in your JCL library. You then can reuse it as required.

Using File Manager in batch

Figure 24-28 presents a JCL batch job generated by File Manager to change job name jobcard to another in the member's wildcard selection of a library.

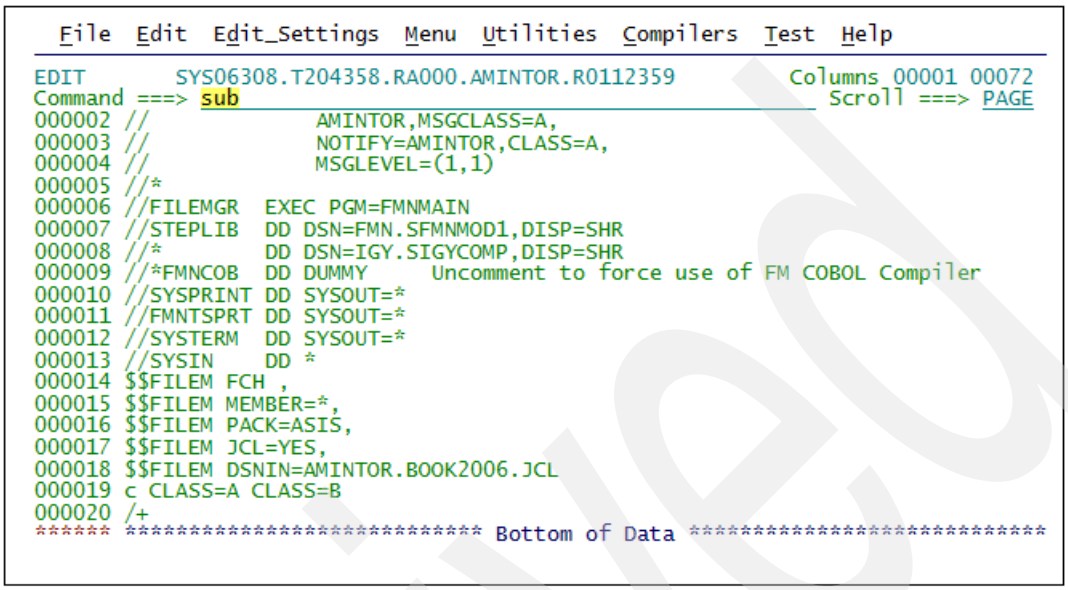


Figure 24-28 Batch job to change Class name to the selective members

More complex scenario

We decided to use the File Manager function Data Set Copy (DSC), along with some simple REXX code, to perform a very selective global find and replace.

Assume the following scenario: As a production support specialist, you must help an application developer set up a portion of a job stream for a User Acceptance Test (UAT). You must take the production job card members, not the procedures, that were created for production, and convert them to UAT standards. The changes identified in Table 24-1 must be made.

Any job that invokes the program FTP must be copied, but must not be changed. These jobs contain the string XMIT2 in the accounting information parameter of the JOB card.

Table 24-1 JCL modifications to make in selected members of a PDS.

Field	From	To
OPC user ID	ZOPCPRD	ZUATUSR
MSGCLASS	S	J
Symbolic	MODEP='P'	MODEP='U'
Member name	Do not copy if it ends in 'T'	

To ensure that no transmission occurs, the program name in the procedure is changed from FTP to IEFBR14. How to make this change is not covered as part of this scenario.

How to set up the batch job

We decided to pre-allocate a separate output file for the changed members. The JCL is shown in Example 24-2.

Example 24-2 Batch job to search for string

```
//FM6SRCH1 JOB 1,RACFUSER,TIME=1440,NOTIFY=&SYSUID,REGION=4M,
//             CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
/*
//* FILE MANAGER BATCH: SEARCH FOR STRING
//*
//STEP01   EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=ADTOOLS.FM610.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//FMNTSPRT DD  SYSOUT=*
//SYSTERM  DD  SYSOUT=*
//DDIN     DD  DISP=SHR,DSN=DAVIN6.SVLSAMP.JCL
//DDOUT    DD  DISP=SHR,DSN=DAVIN6.SVL§UAT.JCL
//SYSIN    DD  *
$$FILEM DSC  INPUT=DDIN,MEMBER=*,
$$FILEM      OUTPUT=DDOUT,REPLACE=YES,
$$FILEM      PROC=*
IF LEFT(INREC,3) <> '//*' THEN DO
   SELECT
      WHEN CO(INREC,' JOB ') & ,
           CO(INREC,'XMIT2') THEN DO
     PRINT('MEMBER NOT CHANGED BECAUSE IT IS FTP','CHAR')
           RETURN
      END
      WHEN CO(INREC,' PROC ') THEN DO
          PRINT('MEMBER NOT COPIED BECAUSE IT IS A PROC','CHAR')
           RETURN 'STOP IMMEDIATE'
      END
      WHEN RIGHT(STRIP(SUBSTR(INREC,3,8)),1) = 'T' THEN DO
           PRINT('MEMBER NOT COPIED BECAUSE IT IS FOR TEST','CHAR')
           RETURN 'STOP IMMEDIATE'
      END
OTHERWISE DO
           OUTREC = CHANGE(INREC,'ZOPCPRD','ZUATUSR')
           OUTREC = CHANGE(OUTREC,'MSGCLASS=S','MSGCLASS=J')
           OUTREC = CHANGE(OUTREC,"MODE='P'","MODE='U'")
           WRITE()
      END
   END
END
/+
/*
0
```

What is happening in this step

The file DDIN is the input file that contains all of the members, which consist of production and test jobs and procedures. While this is not something an application programmer would do in the real world, it does suffice for this example. The default output file has the DD name DDOUT.

The File Manager program keyword DSC is used to invoke the Data Set Copy function. The input and output files are identified, and the keyword PROC is used to indicate that an in-stream REXX routine is being supplied.

The File Manager control cards indicate that all of the members should be selected, and that if any already exist in the output file, they should be replaced. This allows us to run this sample repeatedly.

The first line of the REXX routine selects only non-comment lines for processing.

Then three conditions are applied to the input record:

1. It is searched to see if whether contains the strings JOB and XMIT2. If it does, the member is copied but is not changed.
2. It is searched to see whether it contains the string PROC. If it does, the member is not copied.
3. It is parsed to determine whether the last character of the job name is the letter T. If it is, the member is not copied.

Otherwise, the appropriate changes are made to the JCL and are written to the output file.

Reviewing the report output

The key portion of the batch job's output report is shown in Example 24-3.

Note: Each page in the report starts with the title *IBM File Manager for z/OS and OS/390*. Also, this report has been edited (represented by facing sets of slashes) to fit within the confines of this section.

Example 24-3 Report of global find and replace

```
IBM File Manager for z/OS and OS/390
$$FILEM DSC  INPUT=DDIN,MEMBER=*,
$$FILEM      OUTPUT=DDOUT,REPLACE=YES,
$$FILEM      PROC=*
Member SVLD011P - Copied
12 record(s) copied: 0 truncated: 0 fields truncated
                           MEMBER NOT COPIED BECAUSE IT IS FOR TEST
Member SVLD011T - Copied
0 record(s) copied: 0 truncated: 0 fields truncated
Member SVLD012P - Copied
12 record(s) copied: 0 truncated: 0 fields truncated
                           MEMBER NOT COPIED BECAUSE IT IS FOR TEST
Member SVLD012T - Copied
0 record(s) copied: 0 truncated: 0 fields truncated
Member SVLD021P - Copied
11 record(s) copied: 0 truncated: 0 fields truncated
                           MEMBER NOT COPIED BECAUSE IT IS FOR TEST
Member SVLD021T - Copied
0 record(s) copied: 0 truncated: 0 fields truncated
//\\
\\//
                           MEMBER NOT COPIED BECAUSE IT IS A PROC
Member SVLD104 - Copied
0 record(s) copied: 0 truncated: 0 fields truncated
Member SVLD104C - Copied
16 record(s) copied: 0 truncated: 0 fields truncated
Member SVLD104D - Copied
16 record(s) copied: 0 truncated: 0 fields truncated
                           MEMBER NOT CHANGED BECAUSE IT IS FTP
Member SVLD104E - Copied
16 record(s) copied: 0 truncated: 0 fields truncated
37 member(s) copied: 0 member(s) replaced: 0 member(s) error
```

The first page contains a copy of the input commands. This is followed by a series of status messages that indicate the processing performed during the copy.

The DSC function writes out any of the PRINT statements from the REXX routine before it writes its own statistics. These contain the name of the member and the action taken (copied or replaced), followed by the number of records copied.

We found that when the number of records is zero, the member is not copied, despite what the action indicates.

File Manager external REXX functions used in this routine

A brief explanation of each of the File Manager external REXX functions that were used in this routine follows.

DSC

This copies data from one file to another. The file can be any of the File Manager supported structures (VSAM, QSAM, or PDS).

CONTAIN

If the string being searched for is contained in the input record, then CONTAIN returns 1. Otherwise, CONTAIN returns 0.

PRINT

This prints the string in a specified format to the output report.

WRITE

This writes a record to the specified data sets. If the WRITE function is successful, it returns a value of 0. If the WRITE function is unsuccessful, it raises the REXX syntax error condition.

RETURN

In REXX, you can use the RETURN or EXIT instruction to leave a procedure. RETURN is preferred for performance reasons. You can optionally specify a character string as a parameter on the instruction. This character string is returned to the caller of the procedure.

STOP IMMEDIATE

The character string STOP IMMEDIATE tells File Manager to terminate the current function without writing the current record to the primary output data set. When used with DSC, the member is not copied.

24.8 How to initialize a VSAM file with low-value records

When you create a VSAM file for a CICS application, you usually must initialize it with a low-value record. You probably create a control card (or sequential file) containing binary zeros that matches the record length of the file, so you can REPRO the record into the new file.

In this example, we use File Manager to perform that process, so it does not depend on different control cards for each file size.

To start, you must have an empty VSAM file. You can use the method described previously or IDCAMS control cards.

Start the process by following these steps:

1. Access File Manager in your ISPF session.
2. Go to Data Create Utility (option **3.1**).
3. Enter the name of the new VSAM file.
4. Indicate the number of records to be created.
5. Specify a Fillchar of x'00' (binary zeros).

> **Tip:** Do not make the mistake of selecting a Fillchar of BIN, thinking it creates binary zeros — you *actually* get binary data such as:
>
> `x'0102030405060708090A0B0C0D0E0F10...'`

6. Specify a Disposition of Old.
7. Specify the Copybook or template of None.
8. Select the option for Batch execution.

   When you are finished, your panels should resemble as shown in Figure 24-29 and Figure 24-30.

```
 Process   Options   Help
──────────────────────────────────────────────────────────────────────────────
File Manager                 Data Create Utility
Command ===> ____________________________________________________________
                                                                More:     +
Output Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . 'AMINTOR.BOOK2006.PDHVSAM'                      +
   Member  . . . . . . . . ________     (Blank or pattern for member list)
   Volume serial . . . . . ______       (If not cataloged)
   Record length . . . . . 80 ____      Optional record length for RECFM V
   Records . . . . . . . . 1 _____      number of records
   Fillchar  . . . . . . . x'00'        char or hex value, AN, BIN, or RAND
   Key position  . . . . . ________     if sequence field desired
   Key length  . . . . . . 8            length from 1 to 9
   Key increment . . . . . 10 _____     increment value
   Like data set . . . . . ____________________________________________
Copybook or Template:
   Data set name . . . . . ____________________________________________
   Member  . . . . . . . . _________    (Blank or pattern for member list)
Processing Options:
   Disposition       Copybook or template     Enter "/" to select option
   2  1. Old/Reuse   3  1. Above                 Edit template    Type (1,2,S)
      2. Mod            2. Previous              Batch execution
   ISPF Packing         3. None                  Binary mode, reclen
```

Figure 24-29 Data Create utility panel to initialize VSAM file with binary zeroes

```
 Process   Options   Help
File Manager                Data Create Utility          1 record(s) written
Command ===> _____________________________________________________________
                                                                More:     +
Output Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . 'AMINTOR.BOOK2006.PDHVSAM'                    +
   Member  . . . . . . . . _______     (Blank or pattern for member list)
   Volume serial . . . . . ______      (If not cataloged)
   Record length . . . . . 80          Optional record length for RECFM V
   Records . . . . . . . . 1           number of records
   Fillchar  . . . . . . . x'00'       char or hex value, AN, BIN, or RAND
   Key position  . . . . . _______     if sequence field desired
   Key length  . . . . . . 8           length from 1 to 9
   Key increment . . . . . 10          increment value
   Like data set . . . . . ____________________________________________
Copybook or Template:
   Data set name . . . . . ____________________________________________
   Member  . . . . . . . . _________   (Blank or pattern for member list)
Processing Options:
   Disposition       Copybook or template     Enter "/" to select option
   2  1. Old/Reuse   3  1. Above                Edit template    Type (1,2,S)
      2. Mod            2. Previous             Batch execution
   ISPF Packing         3. None                 Binary mode, reclen
```

Figure 24-30 Result of 1 record written

Note: For a VSAM file, add only *one* record. Otherwise, duplicate keys generate the following error message:

```
VSAM PUT RC X'08', Error Code X'0C'.
```

9. Press Enter. The JCL for the batch job, using Data Set Generate (DSG), is displayed, as shown in Figure 24-31.

:

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       SYS06308.T204358.RA000.AMINTOR.R0112359         Columns 00001 00072
Command ===> ____________________________________________________ Scroll ===> PAGE
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (A),
000002 //             AMINTOR,MSGCLASS=A,
000003 //             NOTIFY=AMINTOR,CLASS=A,
000004 //             MSGLEVEL=(1,1)
000005 //*
000006 //FILEMGR  EXEC PGM=FMNMAIN
000007 //STEPLIB  DD DSN=FMN.SFMNMOD1,DISP=SHR
000008 //*        DD DSN=IGY.SIGYCOMP,DISP=SHR
000009 //*FMNCOB  DD DUMMY     Uncomment to force use of FM COBOL Compiler
000010 //SYSPRINT DD SYSOUT=*
000011 //FMNTSPRT DD SYSOUT=*
000012 //SYSTERM  DD SYSOUT=*
000013 //SYSIN    DD *
000014 $$FILEM DSG DSNOUT=AMINTOR.BOOK2006.PDHVSAM,
000015 $$FILEM FILLCHAR=x'00',
000016 $$FILEM DISP=MOD,
000017 $$FILEM PACK=NONE,
000018 $$FILEM NLRECS=1
****** **************************** Bottom of Data ****************************
```

Figure 24-31 JCL for batch job

10. Submit the batch job.

Save a copy of this JCL.

What is happening in this step

No additional data set is required for this batch job, aside from the standard File Manager load library.

The File Manager program keyword DSG is used to invoke the Data Set Generate function. The output file is identified, along with the keywords, to indicate how the file should be loaded.

The fill character is specified as a hexadecimal zero, and the number of logical records is specified as one.

Reviewing the report output

The key portion of the batch job's output report is listed in Example 24-4.

Example 24-4 Report of DSG low-value record creation

```
IBM FILE MANAGER FOR Z/OS
$$FILEM DSG DSNOUT=AMINTOR.BOOK2006.VSAMPDH,
$$FILEM FILLCHAR=X'00',
$$FILEM DISP=OLD,
$$FILEM NLRECS=1
1 record(s) written
```

Note: Each page in the report starts with the title *IBM File Manager for z/OS*.

Here you can see the input commands listed, followed by a message that states that the requested number of records were written to the output file.

File Manager functions used in this routine

A brief explanation of the File Manager function used in this routine follows.

DSG

Initializes VSAM data sets, sequential data sets, and PDS members.

Specify the output data set name, the disposition, the number of logical records, and the fill character.

To fill each byte of each record with data, specify one of the following:

char	To write a character, such as 0, in each byte
X'cc'	To write a binary character, such as X'04', in each byte
AN	To write alphanumeric characters (A to Z and 0 to 9)
BIN	To write binary characters (X'00' to X'FF')
RAND	To write random binary characters (X'00' to X'FF')

The default is a blank.

Modify the JCL for generic use

To reuse the code from this example, convert the JCL into a procedure. It can then be used to initialize all of your VSAM files. The next example shows the modifications we made.

In Example 24-5, we added the PROC statement and changed the DSG parameter DSNOUT to OUTPUT. This lets you use an override statement in the JCL to point to your file.

Example 24-5 DSG batch step converted to a proc

```
//DSGPROC  PROC
//FILEMAN  EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=ADTOOLS.FM710.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//DDOUT DD DSN=DSNOUT=AMINTOR.BOOK2006.VSAMPDH,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM DSG OUTPUT=DDOUT,
$$FILEM FILLCHAR=x'00',
$$FILEM DISP=OLD,
$$FILEM NLRECS=1
```

Note: We would place the SYSIN statements in a member of a control card library for a production batch job.

24.9 How to populate this newly created file

We can now insert records into the newly created file by copying from an existing file, or by using a template to generate the data.

Copy data from QSAM to VSAM

Perform the following steps to copy data from QSAM to VSAM:

1. Fill the Copy Utility panel (option **3.3**) with the QSAM source file name, as shown Figure 24-32. Press Enter.

```
 Process   Options   Help
File Manager                    Copy Utility
Command ===> ____________________________________________________________

From Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . 'chabert.trader.compmast'                    +
   Member  . . . . . . . . ________      (Blank or pattern for member list)
   Volume serial . . . . . ______
   Start key . . . . . . . ____________________________ key or slot
   Skip count  . . . . . . ________     number of records to be skipped
   Copy count  . . . . . . ALL_____     number of records to be copied
From Copybook or Template:
   Data set name . . . . . ____________________________________________
   Member  . . . . . . . . ________      (Blank or pattern for member)
Processing Options:
   Copybook/template    Enter "/" to select option
   2  1. Above          /  Batch execution          Advanced member selection
      2. None              Use proc *_____          Skip member name list
      3. Create dynamic    Ignore length mismatch   REXX member selection: P
   _  Edit template        JCL Source format        Directory integrity
   _  Type (1,2,S)         Export mode           /  Report PDS record counts
                                                 _  Binary mode, reclen
```

Figure 24-32 Copy Utility Panel

2. Fill the next panel, as shown in Figure 24-33, with the name of the target VSAM file.

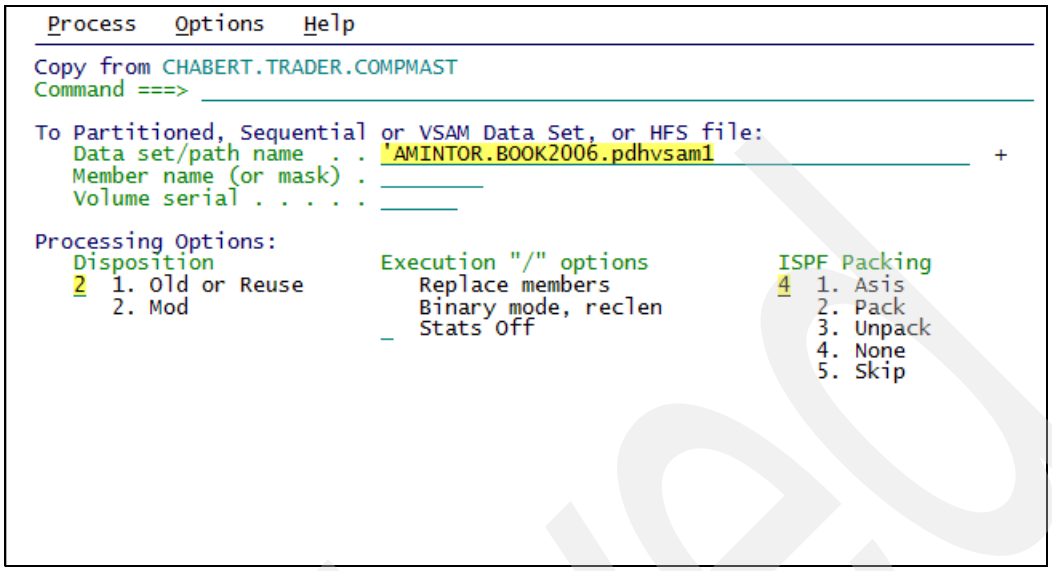


Figure 24-33 Target VSAM file

3. We used the empty file created previously. If you name a file that does not exist, File Manager displays the VSAM Define panel. After selecting KSDS, you see panels as shown Figure 24-34 and Figure 24-35.

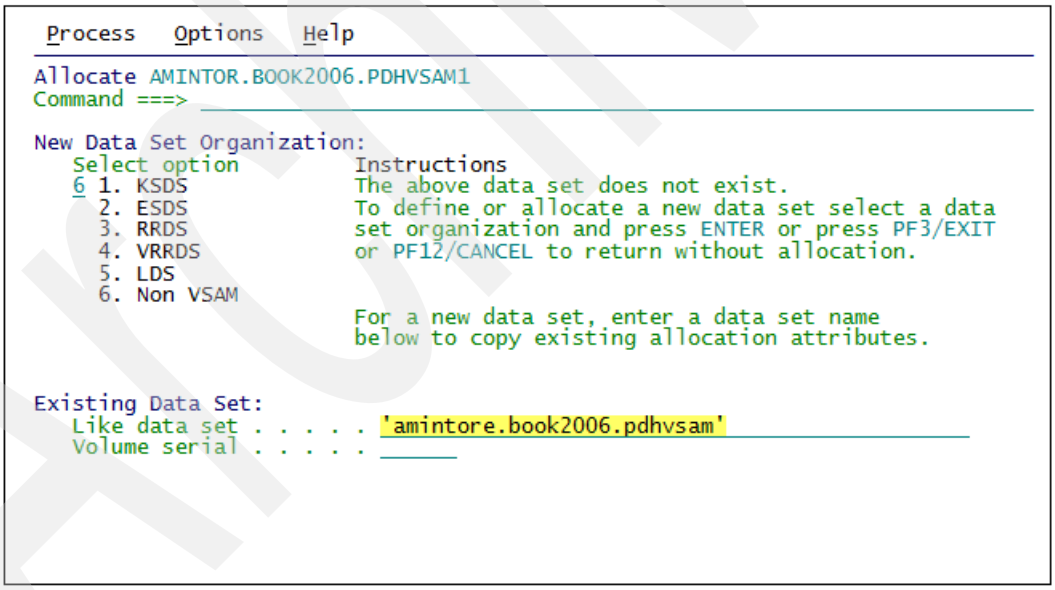


Figure 24-34 VSAM panel, part 1 of 2

```
 Process    Options    Help
------------------------------------------------------------------------------
File Manager                      VSAM Define
Command ===> ________________________________________________________________
Press ENTER to define the catalog entry or EXIT to cancel
VSAM Catalog Entry:
   Data set name  . . 'AMINTOR.BOOK2006.PDHVSAM1'______________________
   Catalog ID . . . .  _____________________________________________
                                                                 More:     +
Basic Information:
   VSAM data type . . KSDS       Expiration date  . ________
   Data component . .  _____________________________________________
   Index component  .  _____________________________________________
VSAM Cluster Attributes:
   Key length . . . . 64         Key offset . . . . 0
   CI size  . . . . . 20480      size of the data control intervals
   Buffer space . . . 41984____  buffer space to be allocated at open time
   Shr cross region . 1          cross system . . . 3         Reuse . . . . N
   Recovery . . . . . Y          Spanned  . . . . . N         Erase . . . . N
   Writecheck . . . . N
VSAM Data Allocation:
   Allocation unit  . CYL        REC, KB, MB, TRK, or CYL
   Space primary  . . 1_______   secondary  . . . . 0_______
   Recsize average  . 4089____   maximum  . . . . . 4089____
```

Figure 24-35   VSAM panel, part 2 of 2

In addition to the online capabilities, you can also use the batch functions of File Manager as shown in Figure 24-36.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       SYS06308.T204358.RA000.AMINTOR.R0112359          Columns 00001 00072
Command ===> ______________________________________________________ Scroll ===> PAGE
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (*),
000002 //             AMINTOR,MSGCLASS=A,
000003 //             NOTIFY=AMINTOR,CLASS=A,
000004 //             MSGLEVEL=(1,1)
000005 //*
000006 //FILEMGR  EXEC PGM=FMNMAIN
000007 //STEPLIB  DD DSN=FMN.SFMNMOD1,DISP=SHR
000008 //*       DD DSN=IGY.SIGYCOMP,DISP=SHR
000009 //*FMNCOB  DD DUMMY     Uncomment to force use of FM COBOL Compiler
000010 //SYSPRINT DD SYSOUT=*
000011 //FMNTSPRT DD SYSOUT=*
000012 //SYSTERM  DD SYSOUT=*
000013 //SYSIN    DD *
000014 $$FILEM showsort
000015 $$FILEM DSC DSNIN=CHABERT.TRADER.COMPMAST,
000016 $$FILEM PACK=NONE,
000017 $$FILEM POSITION=0,
000018 $$FILEM RECCOUNTS=YES,
000019 $$FILEM DISP=MOD,
```

Figure 24-36   Set the batch execution option when you have to copy more than 7 occurrences

4. After pressing Enter in Figure 24-36, you get a confirmation, as shown in Figure 24-37 (if you have not set the Batch execution option before).

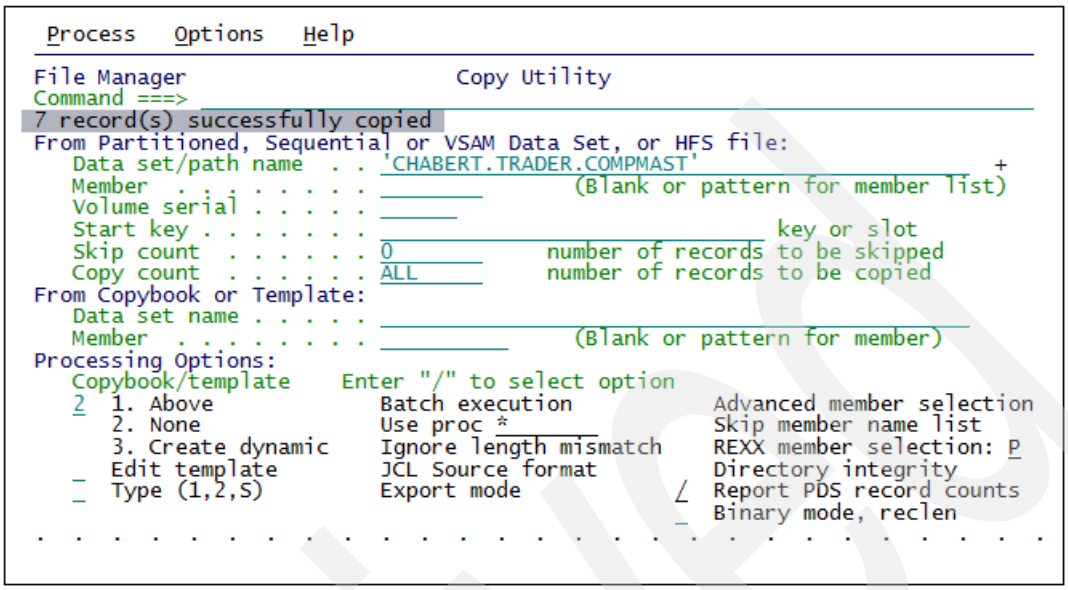


Figure 24-37 Copy completed successfully

Note: File Manager, Version 4.1 and later, automatically uses the DFSORT COPY option for this, and similar simple data set to data set copy actions, provided that DFSORT is available and at the required maintenance level. This is because the input data set is not partitioned and the record format is not undefined.

You can now browse this data set to check that the content has correctly been loaded. Specify option **1** on the Main panel. The capability to browse without any template can be used, as shown in Figure 24-38.

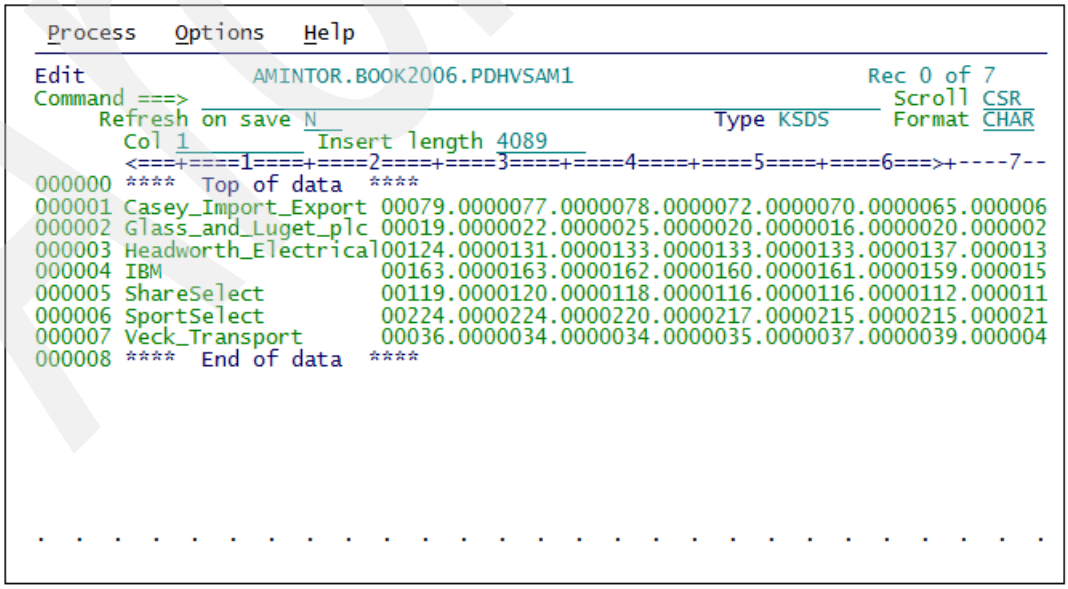


Figure 24-38 Browsing without template

However, it is nicer to look at it using a template, as shown in Figure 24-39 and Figure 24-40.

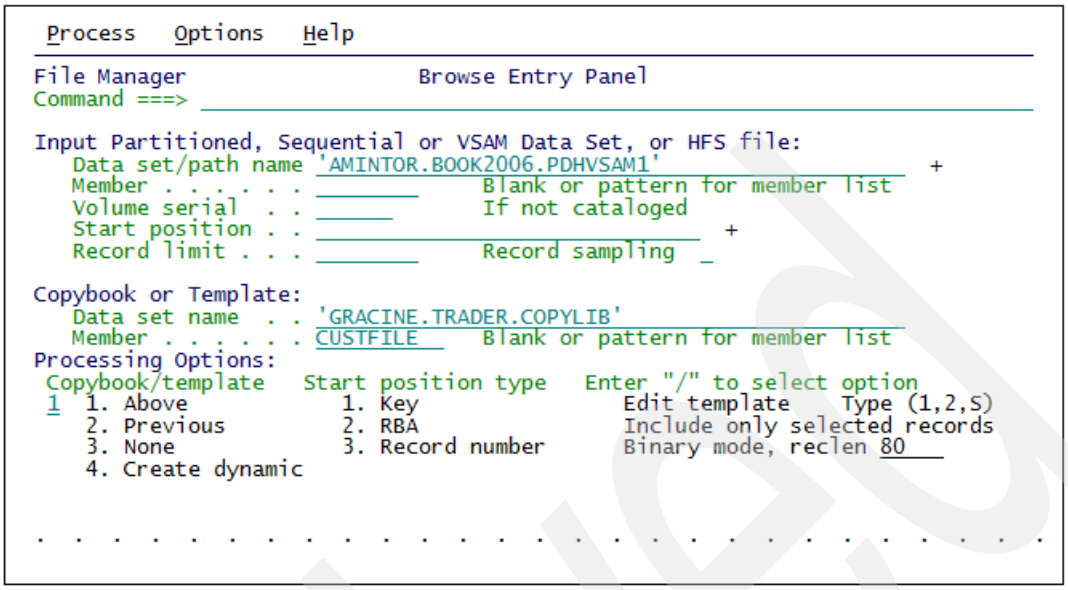


Figure 24-39 The Panel shows how to set the Template or Copybook

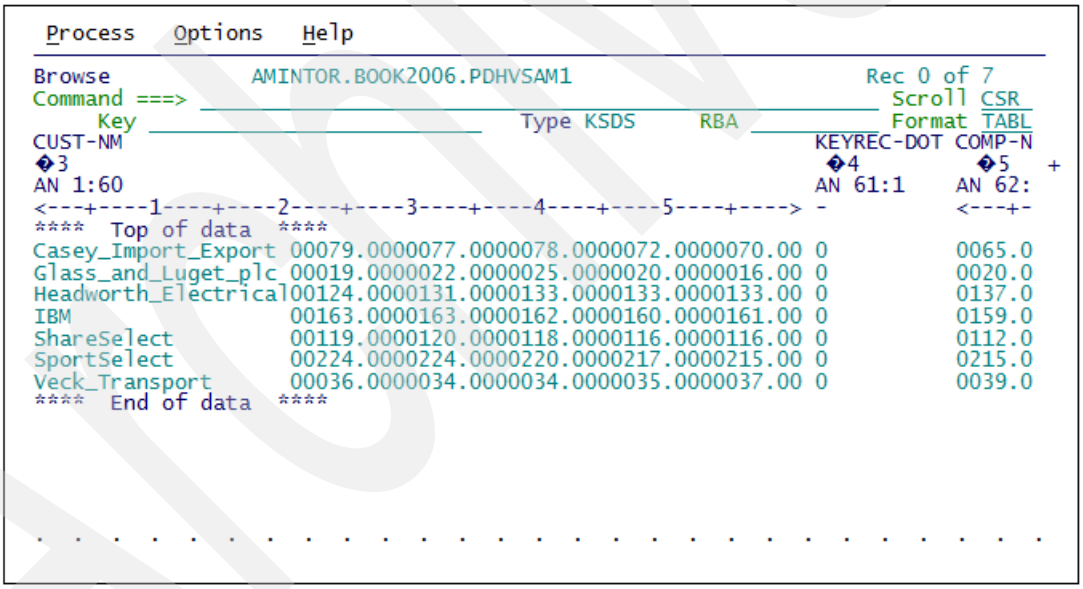


Figure 24-40 Browsing with a template/copybook

When you are browsing or editing a file, you can use commands such as SORT to present the data differently. The default sorting for a VSAM KSDS is the keys in ascending order (SORT KEY). You can sort the records according to the order of the data in a column range or a field if you use a template.

Figure 24-41 presents the result of the SORT #2 D. (*D* stands for *descending*.)

```
 Process    Options    Help
------------------------------------------------------------------------------
Browse          AMINTOR.BOOK2006.PDHVSAM1                      Rec 0 of 7
Command ===> ____________________________________________________ Scroll CSR
     Key ____________________________ Type KSDS     RBA _______ Format TABL
CUST-NM                                                KEYREC-DOT COMP-N
 3                                                      4          5   +
AN 1:60                                                AN 61:1    AN 62:
<---+----1----+----2----+----3----+----4----+----5----+----> -          <---+-
****  Top of data  ****
Casey_Import_Export 00079.0000077.0000078.0000072.0000070.00 0          0065.0
Glass_and_Luget_plc 00019.0000022.0000025.0000020.0000016.00 0          0020.0
Headworth_Electrical00124.0000131.0000133.0000133.0000133.00 0          0137.0
IBM                 00163.0000163.0000162.0000160.0000161.00 0          0159.0
ShareSelect         00119.0000120.0000118.0000116.0000116.00 0          0112.0
SportSelect         00224.0000224.0000220.0000217.0000215.00 0          0215.0
Veck_Transport      00036.0000034.0000034.0000035.0000037.00 0          0039.0
****  End of data  ****
```

Figure 24-41 Sample browse with sort; the data has been sorted by first column descending

Generate data using a template

Instead of using existing data, we can load the new VSAM with computer-generated data, following the rules given in a template such as a copybook.

This time use the Data Creation utility option **3.1** on the Main panel, as shown in Figure 24-42.

```
 Process    Options    Help
------------------------------------------------------------------------------
File Manager                 Data Create Utility
Command ===> ____________________________________________________________
                                                                More:     +
Output Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . 'AMINTOR.BOOK2006.PDpak.custtst1'                +
   Member  . . . . . . . . ________  (Blank or pattern for member list)
   Volume serial . . . . . ______    (If not cataloged)
   Record length . . . . . 80        Optional record length for RECFM V
   Records . . . . . . . . ________  number of records
   Fillchar  . . . . . . . _____     char or hex value, AN, BIN, or RAND
   Key position  . . . . . ________  if sequence field desired
   Key length  . . . . . . 8         length from 1 to 9
   Key increment . . . . . 10        increment value
   Like data set . . . . . ____________________________________________
Copybook or Template:
   Data set name . . . . . 'amintor.book2006.template'
   Member  . . . . . . . . temp01    (Blank or pattern for member list)
Processing Options:
   Disposition       Copybook or template     Enter "/" to select option
   2  1. Old/Reuse   1  1. Above                 Edit template    Type (1,2,S)
      2. Mod            2. Previous              Batch execution
```

Figure 24-42 Data create Panel

You can browse the result using option **1** on the Main panel, which shows that there are default values for data generation in the template. Edit the template if necessary.

The default values for the fields are mostly fixed, as listed in Table 24-2.

Table 24-2 Some default values used in the example

Field	Type	Default value
CUST-NM	AN	FX 'TEST ADT' repeat
KEYREC-DOT	AN	FX ‘x’
COMP-NM	AN	RO 'aaabbbcccdddeeefff'
DEC-NO-SHARES	PD	Init=10, Incr=7
BUY-FROM	AN	FX ‘Unknown’
BUY-FROM-NO	AN	RA '0123456789'

Generating data with SHOWCOB and SHOWSORT

If you select the **/** batch execution option, the following JCL is automatically generated as shown in Example 24-6.

Example 24-6 Load VSAM and generate data

```
IBM File Manager for z/OS and OS/390
DSG CREATE and Write data to a Dataset
------------------------------------------------------------------------------
//AMINTORB JOB (*),
//            AMINTOR,MSGCLASS=A,
//            NOTIFY=AMINTOR,CLASS=A,
//            MSGLEVEL=(1,1)
//*
//FILEMGR  EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=FMN.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//*FMNCOB  DD DUMMY     Uncomment to force use of FM COBOL Compiler
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM SHOWCOB
$$FILEM SHOWSORT
$$FILEM DSG DSNOUT=AMINTOR.BOOK2006.PDPAK.CUSTTST1,
$$FILEM TCOUT=AMINTOR.BOOK2006.TEMPLATE(TEMP01),
$$FILEM DISP=MOD,
$$FILEM PACK=NONE,
$$FILEM NLRECS=10
```

The following control statements have been added manually inside the previous Job (above):

- $$FILEM SHOWCOB (show which COBOL compiler is currently being used)
- $$FILEM SHOWSORT (allow to verify if DFSORT is being used as the sort utility)

The output is written to the SYSPRINT dataset, how shown inside Figure 24-43.

```
  Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY AMINTORB JOB03508  DSID   102 LINE 6       COLS 02- 81
 COMMAND INPUT ===>                                          SCROLL ===> CSR
PRINTOUT=SYSPRINT  HEADERPG=YES  ASCII=BOTH     RECLIMIT=(1,*)
PRINTLEN=132       PAGESKIP=NO   PAD=OFF        EOD=/*
PAGESIZE=60        DATAHDR=YES   DBCSPRT=OFF    LANGUAGE=ENGLISH
PRTTRANS=ON        DUMP=UPDOWN   TAPELBL=SL     CYLHD=ABSOLUTE
SMFNO=000          PRTDISP=MOD   USEIOX=DISABLE IOX=
TEMPHLQ=
ABENDCC=NONE

IBM File Manager for z/OS
$$FILEM SHOWCOB
Using Licensed COBOL
$$FILEM SHOWSORT
SORT debugging is on
$$FILEM DSG DSNOUT=AMINTOR.BOOK2006.PDPAK.CUSTTST1,
$$FILEM TCOUT=AMINTOR.BOOK2006.TEMPLATE(TEMP01),
$$FILEM DISP=MOD,
$$FILEM PACK=NONE,
$$FILEM NLRECS=1
IBM File Manager for z/OS
1 record(s) written
```

Figure 24-43 Output to SYSPRINT database

How to split a single file into constituent record types

There might be times when you must take one or more of the record types in a multi-record file and segregate the records for additional processing. The recommended way to do this is by using a DFSORT procedure. We show batch job streams, using both DFSORT and REXX procedures, that take a file and split it into three record types. All other record types are ignored.

The required DFSORT procedure would be similar to that shown in Example 24-7.

Example 24-7 DFSORT procedure

```
//*
//FILEMGR  EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=ADTOOLS.FM710.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//REC01    DD DSN=GRACINE.BOOK2005.SEQFILEA,DISP=OLD
//REC02    DD DSN=GRACINE.BOOK2004.SEQFILEB,DISP=OLD
//REC03    DD DSN=GRACINE.BOOK2005.SEQFILEC,DISP=OLD
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM DSC DSNIN=CHABERT.BOOK2005.TRANFILE,
$$FILEM POSITION=0,
$$FILEM DISP=MOD,
$$FILEM DSNOUT=GRACINE.BOOK2005.SEQFILEX,
$$FILEM PROC=*
*FASTPROC
 OUTFIL FNAMES=REC01,INCLUDE=(1,2,CH,EQ,C'01')
 OUTFIL FNAMES=REC02,INCLUDE=(1,2,CH,EQ,C'02')
 OUTFIL FNAMES=REC03,INCLUDE=(1,2,CH,EQ,C'03')
/+
```

The same requirement can be met by using an in-stream REXX routine to process the records. The File Manager step of the batch job, shown in Example 24-8, illustrates this.

Example 24-8 File Manager step of batch job

```
//*
//FILEMGR  EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=ADTOOLS.FM710.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//REC01    DD DSN=GRACINE.BOOK2005.SEQFILEA,DISP=OLD
//REC02    DD DSN=GRACINE.BOOK2005.SEQFILEB,DISP=OLD
//REC03    DD DSN=GRACINE.BOOK2005.SEQFILEC,DISP=OLD
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM SHOWCOB
$$FILEM SHOWSORT
$$FILEM DSC DSNIN=CHABERT.BOOK2005.TRANFILE,
$$FILEM POSITION=0,
$$FILEM DISP=MOD,
$$FILEM DSNOUT=GRACINE.BOOK2005.SEQFILEX,
$$FILEM PROC=*
DDNAME = 'REC' || FLD(1,2)
IF NCO(FLD(1,2),1,2,3) THEN DO
   WRITE(DDNAME)
   RETURN 'DROP'
END
/+
```

What is happening in these jobs

The file TRANFILE is the input file that contains multiple record types. The default output file has the DD name SEQFILEX. Each of the record types we are interested in goes into REC01, REC02, or REC03.

The File Manager program keyword DSC is used to invoke the Data Set Copy function. The input and output files are identified, and the keyword PROC=* is used to indicate that an in-stream routine is being supplied.

The result is that all type 01 records end up in REC01, type 02 records go to REC02, type 03 records go to REC03, and all other record types go to the file EXTRA.

Reviewing the report output

The key portion of the batch job's output report is listed in Example 24-9.

Note: Each page in the report starts with the title *IBM File Manager for z/OS and OS/390*.

Example 24-9 Report of DSC multiple record split

```
IBM File Manager for z/OS and OS/390
DSC      WRITE summary report
-------------------------------------------------------------------------------
Total records written to REC01    = 20
Total records written to REC02    = 20
Total records written to REC03    = 15
IBM File Manager for z/OS and OS/390
67 record(s) read
12 record(s) copied: 0 truncated: 0 fields truncated
```

The first part contains the output of the record split operation (a copy). Notice that you do not have to do any extra programming to obtain the number of records sent to each file; File Manager does that automatically.

The last line contains the total number of records processed. In this case, 12 records did not meet any of the selection criteria, and were written to the default file (EXTRA).

File Manager external REXX functions used in this routine

A brief explanation of each of the File Manager external REXX functions that were used in this routine follows.

FLD Returns the value of a field from the current input record (INREC), starting at start_column, of length number_of_bytes, interpreted according to the specified type:

- **B** If the field is binary. If you specify B for the type, the length must be 2, 4, or 8.

  If the field contains characters, the following conventions must be used.
- **C** If the field is packed decimal. This is the default.
- **P** If you specify P for type, length must be between 1 and 16 and the field is zoned decimal.
- **Z** If you specify Z for type, length must be between 1 and 32, or if the field contains a separate sign character, between 1 and 33.

NCO If the numeric value of any of the match arguments is equal to the numeric value of number, then NCONTAIN returns 1. Otherwise, NCONTAIN returns 0.

WRITE Writes a record to the specified data sets. If the WRITE function is successful, it returns a value of 0. If the WRITE function is unsuccessful, it raises the REXX syntax error condition.

RETURN In REXX, you can use the RETURN or EXIT instruction to leave a procedure. RETURN is preferred for performance reasons. You can optionally specify a character string as a parameter on the instruction. This character string is returned to the caller of the procedure.

DROP The character string DROP tells File Manager to not write the current record to the primary output data set.

24.10 Replacing a string in a specific location in a file

If you must unconditionally replace a string in one location of a file, you can use this utility.

The code to perform this function with File Manager is shown in Example 24-10.

Example 24-10 File Manager string replace batch step

```
//*
//* FILE MANAGER BATCH: REPLACE A STRING IN A SPECIFIC LOCATION
//*
//STEP01   EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=ADTOOLS.FM710.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//FMNTSPRT DD  SYSOUT=*
//SYSTERM  DD  SYSOUT=*
//EDITFILE DD  DISP=OLD,DSN=YOUR.FILE.TO.EDIT
//SYSIN    DD  *
$$FILEM DSU INPUT=EDITFILE,
$$FILEM     PROC=*
OUTREC=OVERLAY('vaLUE',INREC,11)
/+
```

What is happening in this step

The File Manager program keyword DSU invokes the Data Set Update utility, which is only available in batch.

The utility reads records sequentially from the input file. When File Manager processes them, it uses two built-in REXX variables, INREC and OUTREC, to refer to the input and output records.

In this case, we use a standard REXX function, OVERLAY, to indicate that a string should be placed at a specific location. This is assigned to the output record that is written to the file, a portion of which is shown in Example 24-11.

Example 24-11 Output from string replace batch job

```
IBM File Manager for z/OS and OS/390
$$FILEM DSU DSNIN=GRACINE.FILE.TO.EDIT,
$$FILEM     PROC=*
13 record(s) read
13 record(s) updated
```

24.11 Copying selected variably blocked records to another file

If you must copy selected records from a production file to a test file, you can use this utility. In this case there are multiple criteria and there is only one output file. This example also demonstrates how File Manager processes new file allocation and variable blocked records.

The steps necessary to perform this function using a procedure with File Manager follow:

1. Select option **3.3** from the main FM menu, producing a panel, as shown in Figure 24-44.

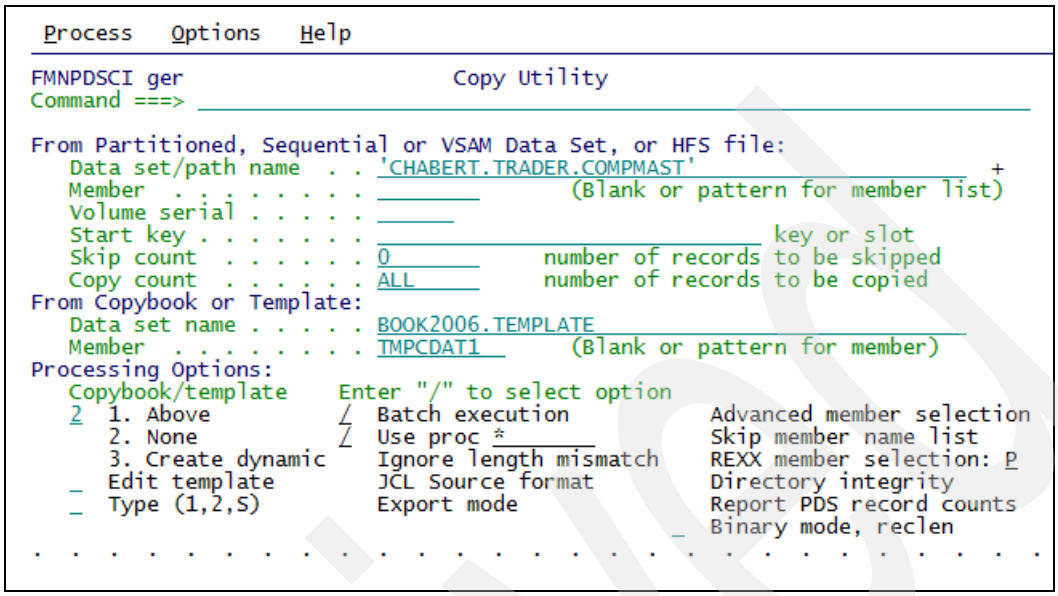


Figure 24-44 Option selection from the main File Manager panel

2. Select **/** Use proc with the generic asterisk (*) and press Enter. You can now enter record selection criteria for the copy. In this example we select only those records with a C, G, L, M, or S in column 3, as shown in Figure 24-45.

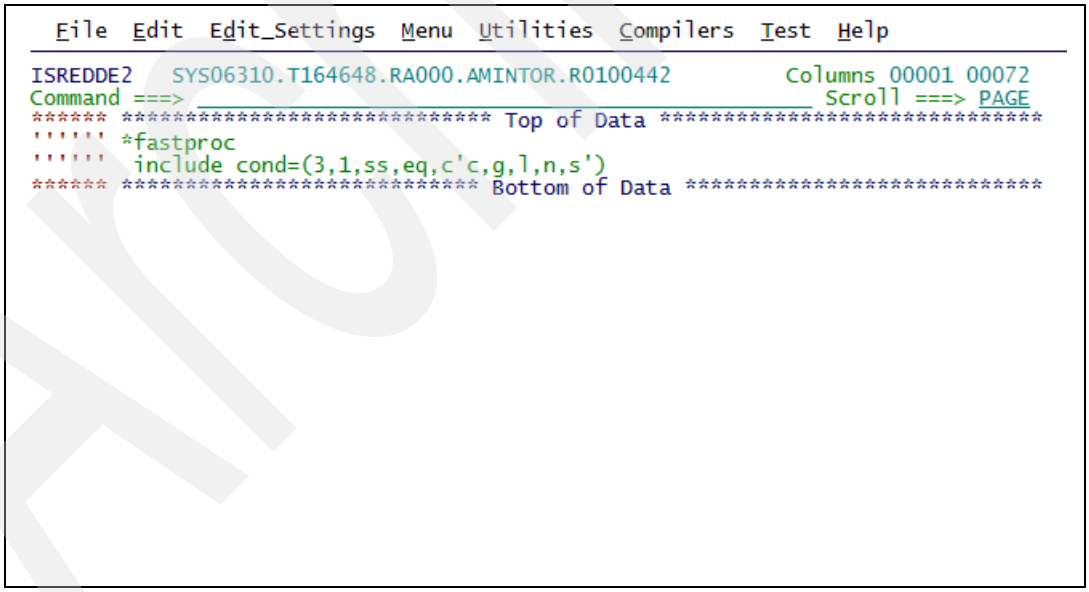


Figure 24-45 Record selection criteria

3. Press Enter, followed by PF3. You can now enter the name of the file to which the data is to be copied. In this example we dynamically define a new file (SEQFILEX) with the same characteristics as the input file. These steps have usually been showed in others samples before. FM/Base shows sequentially the panels to define the dataset according to preferences and definition you require. To make this scenario more simple and to make focus and details on the *Copy Selected* utility, we are omitting these panels.

4. Since we selected batch execution for this copy job, we are presented with the next JCL as shown in Figure 24-46.

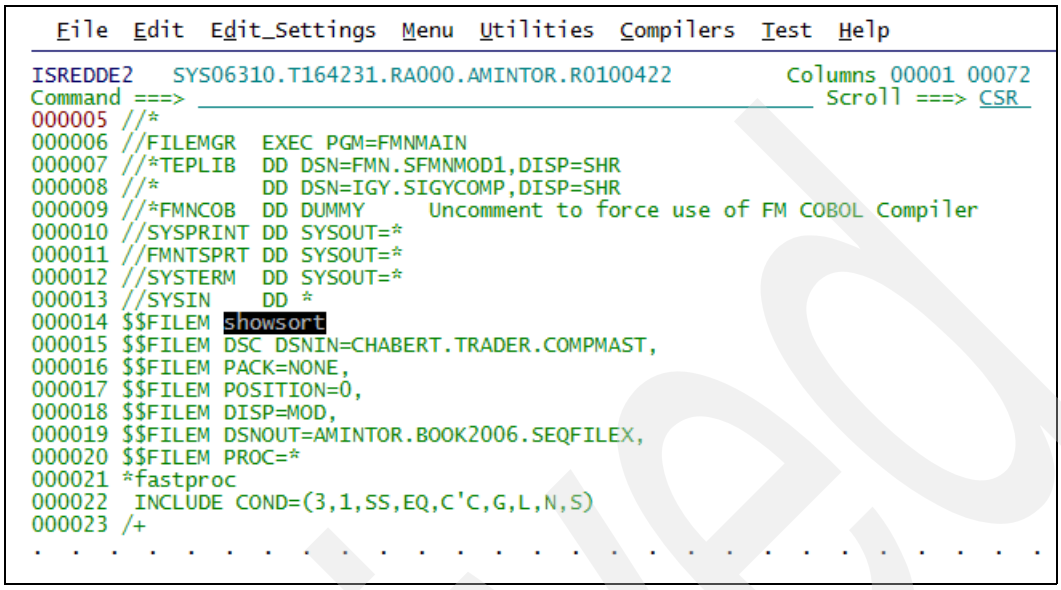


Figure 24-46 JCL for Batch execution

5. Sine we are using FASTPROC, we have manually added the SHOWSORT command to verify that DFSORT is used. Submitting this job, we see the successful execution report, as shown in Example 24-12.

Example 24-12 A part of SYSOUT and DFSORT message

```
//FILEMGR  EXEC PGM=FMNMAIN
$$FILEM SHOWSORT
SORT debugging is on
$$FILEM DSC DSNIN=CHABERT.TRADER.COMPMAST,
$$FILEM PACK=NONE,
$$FILEM POSITION=0,
$$FILEM DISP=MOD,
$$FILEM DSNOUT=AMINTOR.BOOK2006.SEQFILEX,
$$FILEM PROC=*
ICE143I 0 BLOCKSET     COPY  TECHNIQUE SELECTED
ICE250I 0 VISIT http://www.ibm.com/storage/dfsort FOR DFSORT PAPERS, EXAMPLES AN
ICE000I 0 - CONTROL STATEMENTS FOR 5694-A01, Z/OS DFSORT V1R5 - 17:27 ON MON NOV
          OPTION COPY,MSGDDN=FMNSRTP,PAD=RC0,TRUNC=RC0,VLLONG,NOVLSCMP,VLSHRT,SO
                      TIN=SYS00001,SORTOUT=SYS00002
          INCLUDE COND=(3,1,SS,EQ,C'C,G,L,M,S')
ICE201I E RECORD TYPE IS F - DATA STARTS IN POSITION 1
ICE751I 0 C5-K90007 C6-K90007 C7-K90000 C8-K90007 E9-K90007 C9-BASE   E5-K14794
ICE193I 0 ICEAM2 ENVIRONMENT IN EFFECT - ICEAM2 INSTALLATION MODULE SELECTED
ICE088I 0 AMINTORB.FILEMGR .        , INPUT LRECL = 90, BLKSIZE = 27990, TYPE =
ICE093I 0 MAIN STORAGE = (MAX,6291456,6291456)
ICE156I 0 MAIN STORAGE ABOVE 16MB = (6234096,6234096)
ICE127I 0 OPTIONS: OVFLO=RC0 ,PAD=RC0 ,TRUNC=RC0 ,SPANINC=RC16,VLSCMP=N,SZERO=Y,
ICE128I 0 OPTIONS: SIZE=6291456,MAXLIM=1048576,MINLIM=450560,EQUALS=N,LIST=Y,ERE
ICE129I 0 OPTIONS: VIO=N,RESDNT=ALL ,SMF=NO   ,WRKSEC=Y,OUTSEC=Y,VERIFY=N,CHALT=
ICE130I 0 OPTIONS: RESALL=4096,RESINV=0,SVC=109 ,CHECK=Y,WRKREL=Y,OUTREL=Y,CKPT=
ICE131I 0 OPTIONS: TMAXLIM=6291456,ARESALL=0,ARESINV=0,OVERRGN=16384,CINV=Y,CFW=
ICE132I 0 OPTIONS: VLSHRT=N,ZDPRINT=Y,IEXIT=N,TEXIT=N,LISTX=N,EFS=NONE    ,EXITC
ICE133I 0 OPTIONS: HIPRMAX=OPTIMAL,DSPSIZE=MAX ,ODMAXBF=0,SOLRF=Y,VLLONG=Y,VSAMI
```

```
ICE235I 0 OPTIONS: NULLOUT=RC0
ICE084I 0 EXCP ACCESS METHOD USED FOR SYS00002
ICE084I 0 EXCP ACCESS METHOD USED FOR SYS00001
ICE751I 1 EF-K10929 F0-Q84357 E8-K11698
ICE090I 0 OUTPUT LRECL = 90, BLKSIZE = 27990, TYPE = FB
ICE055I 0 INSERT 0, DELETE 6
ICE054I 0 RECORDS - IN: 7, OUT: 1
ICE052I 0 END OF DFSORT
7 record(s) read
1 record(s) successfully copied
******************************** BOTTOM OF DATA ********************************
```

6. We note that the DFSORT has been used for this sort utility.
7. Alternatively, we could have used a REXX procedure to obtain the same result. For this method we coded the procedure, as shown below. Remember, though, that the performance when processing large data sets is significantly better if DFSORT can be used.
8. Following this through, we see JCL created, as shown in Example 24-13.

Example 24-13 JCL created

```
//FILEMGR  EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=ADTOOLS.FM710.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM DSC DSNIN=GRACINE.TEST.SEQFILE,
$$FILEM POSITION=0,
$$FILEM DISP=MOD,
$$FILEM DSNOUT=GRACINE.TEST.SEQFILEX,
$$FILEM PROC=*
IF CO(FLD(3,1),C,G,L,M,S) THEN
   RETURN
ELSE
   RETURN 'DROP'
/+
```

Successful completion of the job is shown Example 24-14, with identical results. Note that for a job such as this, particularly when working with a large file, DFSORT performance is significantly better than that of REXX.

Example 24-14 Successful completion

```
IBM File Manager for z/OS
$$FILEM DSC DSNIN=GRACINE.TEST.SEQFILE,
$$FILEM POSITION=0,
$$FILEM DISP=MOD,
$$FILEM DSNOUT=GRACINE.TEST.SEQFILEX,
$$FILEM PROC=*
67 record(s) read
20 record(s) copied: 0 truncated: 0 fields truncated
```

What is happening in these jobs

The File Manager program keyword DSC is used to invoke the data set copy function. The input and output files are identified, and the keyword PROC is used to indicate that an in-stream routine, either DFSORT or REXX, is being supplied.

DFSORT We know that DFSORT is being used by the presence of the *FASTPROC statement. The routine is built using standard DFSORT program control statements, as described in the *DFSORT R14 Application Programming Guide,* SC26-7050-05. The INCLUDE statement uses a substring comparison test to search for one of the listed values in column 3 of the input records. If found, the record is copied to the output file. All other records are bypassed.

REXX The first line of the routine checks the contents of the record (using the function FLD) starting in position 3 for a length of 1 in order to see if it matches one of the listed values. If it does, the second line of the routine writes out the records to the output file. Otherwise, the fourth line ignores the records.

Notice that in both examples the Data Control Block (DCB) information for the new file is copied from the input file.

24.12 Searching for a string in all members of a PDS

To determine which members of a PDS contain a particular string, you can use this utility.

The code to perform this function with File Manager is shown Example 24-15.

Example 24-15 File Manager string find in a PDS batch step

```
//*
//* FILE MANAGER BATCH: SEARCH FOR STRING
//*
//STEP01   EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=ADTOOLS.FM710.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//FMNTSPRT DD  SYSOUT=*
//SYSTERM  DD  SYSOUT=*
//DDIN     DD  DISP=SHR,DSN=YOUR.CHANGE.MGMT.UAT.JCLLIB
//SYSIN    DD  *
$$FILEM FCH INPUT=DDIN,MEMBER=*,
$$FILEM     PROC=*
   IF CO(INREC,'UNIT=CART') | ,
      CO(INREC,'UNIT=TAPE') THEN
      RETURN
   ELSE
      RETURN 'DROP'
/+
```

What is happening in this step

The File Manager utility FCH is used to invoke the Find/Change function.

The file DDIN (the default input file for the FCH function) is the PDS you want to search.

The first two lines check for one of two strings. The fifth line ignores any records that do not contain the strings.

Reviewing the report output

The key portion of the batch job's output report is shown in Example 24-16.

Example 24-16 Output from string find in a PDS batch job

```
IBM File Manager for z/OS
$$FILEM FCH INPUT=DDIN,MEMBER=*,
$$FILEM     PROC=*
IBM File Manager for z/OS
Record-#  Find/Change Listing DSN:GRACINE.WORK.JCL
FABSERCH                    ---------- STRING(S) FOUND ----------
12s             IF=(1,0,C'UNIT=CART'),
13s           ORIF=(1,0,C'UNIT=TAPE')
FMBSERCH                    ---------- STRING(S) FOUND ----------
14s    IF CO(INREC,'UNIT=CART') | ,
15s       CO(INREC,'UNIT=TAPE') THEN DO
IECD01                      ---------- STRING(S) FOUND ----------
833s //             DISP=(,CATLG,DELETE),UNIT=CART,EXPDT=99000,
862s //             DISP=(,CATLG,DELETE),UNIT=CART,EXPDT=99000,
ISBSERCH                    ---------- STRING(S) FOUND ----------
9s SRCHFOR  'UNIT=CART'
10s SRCHFOR  'UNIT=TAPE'
TESTME                      ----------- member in use -----------
------- Find/Change summary section ------------------
  Records found: 8 Records processed: 2744
  Members w/recs: 5 Members wo/recs: 42
------- Find/Change statement section ----------------
IF CO(INREC,'UNIT=CART') | ,
  CO(INREC,'UNIT=TAPE') THEN
  RETURN
  ELSE
  RETURN 'DROP'
```

Each of the members in which either one of the strings was found is listed. The lines on which the strings were found are displayed.

Notice that our test file is still in use. No search was performed on this member (otherwise, the string would have been found there as well).

The summary statistics appear at the end of the report, along with a display of the search commands.

24.13 Multiple find

The FIND utility command (option **3.6**) allows you to specify more than one string to search for, optionally with a limit on the range of columns to be searched. The command can be entered as shown in Example 24-17.

Example 24-17 FIND utility command

```
FIND string1 AND string2 OR string3 col1 col2
F string1 & string2 | string3 col1 col2
```

Note: This does not apply to the FIND command in Edit/Browse mode.

This is a free-form FIND. If you do not specify a column range, the total length of the record is used. There are some limitations to this utility. Use REXX procedures for more complex processing, such as:

- Searching for a string in one field or another string in another field.
- Combining FIND and CHANGE in the same command.

Restriction: You do not have the option of specifying a template and field reference instead of the starting and ending columns.

When you request the online FIND, you get the results in the SRCHFOR.LIST file specified on the panel, as shown in Example 24-18.

Example 24-18 Result of combined FIND '01' | '03' 1 2

```
******************************** Top of Data ********************************
IBM File Manager for z/OS
Record-#  Find/Change Listing DSN:GRACINE.DEMOS.TESTFILE

 FMNCDATA                   ---------- STRING(S) FOUND ----------

1  01Grant Sutherland    ...-.Í.....................................
      2  01Andrew Astle       .M...Í.....................................
      3  01Graham Purdie      .h.;.Ç.....................................
      4  01Bill Soper         .....Ç.....................................
      5  01Tyrone Dalais      .O...-.....................................
      6  01Rod Turner         .Ö.Ã.....................................
      7  01Clive Nealon       .u.Ã.....................................
      8  01Jim Alexander      .ì.>.....................................
      9  01Silvano Prez       .....-.....................................
     10  01Don Pharoah        .....Ä.....................................
     11  01John Levrington     .è...Å.....................................
     12  01Liz Rushton        .è...Ã.....................................
     13  01Bob McCormack      .µ...Î.....................................
14  01Keith Stewart       .....ø.....................................
     15  01John Nicholls      .è...Ñ.....................................
     16  01Peter van Dyke     .µ...e.....................................
     17  01Anna Waghorn       .d...-.....................................
     18  01Mike Moriarty      .....d.....................................
     19  01Merrill Bani       ...-.ì.....................................
     20  01Angus King         ...-.....................................

  ------- Find/Change summary section -----------------
  Records found: 20 Records processed: 40
  Members w/recs: 1 Members wo/recs: 0
  Search cols: 1:2 Longest line: 80

  Edit options in effect: CAPS:OFF

  Search argument: 01
  Search argument: 03
******************************** Bottom of Data ********************************
```

The FIND corresponding procedure is shown in Example 24-19.

Example 24-19 FIND procedure

```
$$FILEM FCH ,
$$FILEM MEMBER=*,
$$FILEM DSNIN=GRACINE.DEMOS.TESTFILE
BNDS 1 2
CAPS OFF
find '01' | '03' 1 2
/+
```

Result of a batch FIND command

You can see the result of the FIND command with SDSF. The new command is presented in Example 24-20.

Example 24-20 SDSF output of batch FIND command

```
Record Number
1  //TRADERBA  JOB  1, RACFUSER,TIME=1440,NOTIFY=&SYSUID,REGION=4M,

   -- Find /Change  summary section --
   Record found:  23 Record processed: 1037
   Members w/recs: 23 Members wo/recs: 5
    ISPF packing option: NONE
    Search cols:  1:80 Longest  Line: 80
    Edit options in effect : CAPS:OFF

    -                      Count     Value
    Found    :                23     >&SYSUID <

------------------------------------------------------------------------------------------

****************************** BOTTOM OF DATA ****************************************
```

You can specify an output file in the SYSPRINT DD card when you request the batch execution of FIND.

24.14 Template updating and generation using a model

In many cases, when you require a formatted output that does not exist, File Manager helps you fill in the information required in order to manipulate your view of a data set by applying an existing Template, or creating dynamically or from a copybook a new Template. When you supply a COBOL or PL/I copybook, or creating a new dynamic template, File Manager compiles it into a template before use a new or an existing Template.

The following process assumes that you have already created a template (dynamically, from COBOL or PL/I structures), but you want to start again from the same “Mapping” to modify the output of your data without alter the original definitions Template.

Figure 24-47 shows how you find the original copybook and perform a template editing:

1. Select the dataset to display.
2. Choose a member (You can enter a generic member name to display a member list. If you do not remember the name, you can use *. Use **S** to select one from the list) of an existing Template from a PDS Library.
3. Type **/** to use the mapping Template from the *Copybook/template* field.
4. Enter **/** to select the option, *Edit template.*

```
 Process   Options   Help
 File Manager                  Edit Entry Panel
 Command ===> _____________________________________________________________

 Input Partitioned, Sequential or VSAM Data Set, or HFS file:
    Data set/path name 'DENARDI.FMN.DATA'                                  +
    Member . . . . . . FILEAMI     Blank or pattern for member list
    Volume serial  . . ______      If not cataloged
    Start position . . _____________________________  +
    Record limit . . . ________    Record sampling  _
    Inplace edit . . . _           Prevent inserts and deletes
 Copybook or Template:
    Data set name  . . 'DENARDI.FMN.TEMPLATE'
    Member . . . . . . TEMPZZZ3    Blank or pattern for member list
 Processing Options:
  Copybook/template   Start position type   Enter "/" to select option
  1  1. Above         3  1. Key             /  Edit template    Type (1,2,S)
     2. Previous         2. RBA               Include only selected records
     3. None             3. Record number     Binary mode, reclen 80
     4. Create dynamic                        Create audit trail

  F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
  F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 24-47 Edit Entry panel to edit a Template

After pressing Enter, the panel in Figure 24-48 is displayed with all the Records (01 Levels) defined in the original Copybook and used to generate the Template. In the Command Line, type `=7;4` to select the new *Update template(s) utility* function. Press Enter.

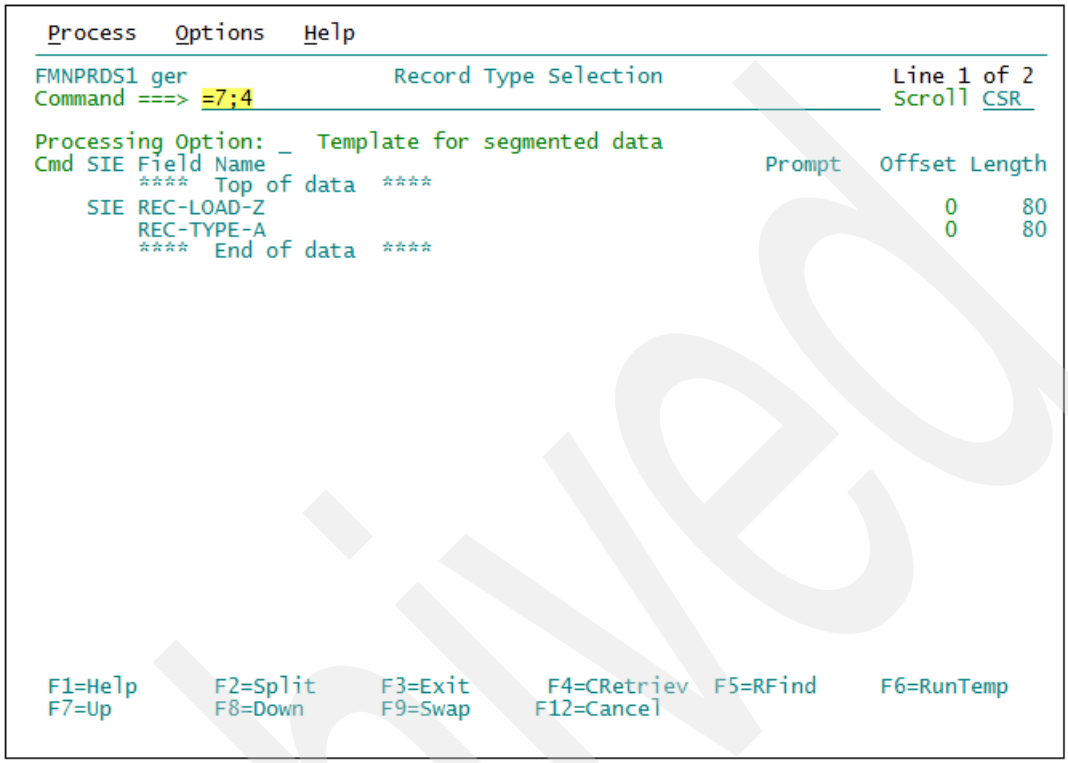


Figure 24-48 The record selection Panel

FM displays the Template Member Selection panel as shown in Figure 24-49. You can select a name from this list and press Enter to overwrite the existing selected member panel.

```
 Process   Options   Help
FMNPMSLU ger              Template Member Selection                Row 1 of 21
Command ===> _____________________________________________________ Scroll CSR

Input data set  DENARDI.FMN.TEMPLATE
Update data set DENARDI.FMN.TEMPLATE

Sel Name     Prompt       Created     Updated              Lang   Ver
    AAA1TEMP              2006/02/15  2006/02/15 09:00:57  COBOL    1
_   AAA4TEMP              2006/02/15  2006/02/15 09:38:49  COBOL    1
_   BATCH1                2004/04/16  2004/04/16 15:42:30  COBOL    1
_   COPYAMI               2004/04/04  2004/04/06 11:56:56  COBOL    1
_   COPYAMI2              2004/04/06  2004/04/06 12:37:48  COBOL    1
_   FILAMX                2005/11/28  2006/07/30 09:18:24  COBOL    1
_   MODEMP01              2004/03/21  2004/03/21 16:48:44  COBOL    1
_   MODEMP02              2004/03/21  2004/03/29 16:46:13  COBOL    1
_   PROVAREC              2004/04/04  2004/04/04 18:22:09  COBOL    1
_   TEMEMP05              2004/03/30  2004/04/04 17:54:40  COBOL    1
_   TEMPCUS1              2006/11/03  2006/11/04 21:24:26  COBOL    2
_   TEMPCUS2              2006/11/04  2006/11/04 21:58:37  COBOL    2
_   TEMPZZZ2              2006/07/30  2006/07/30 10:22:45  COBOL    1
u   TEMPZZZ3              2006/07/30  2006/11/08 09:07:18  COBOL    2
_   TEMP0004              2006/07/29  2006/10/26 00:20:12  COBOL    2
_   TEMP01                2004/04/04  2004/04/04 17:56:58  COBOL    1
_   TEMP1DB2              2005/07/11  2005/07/11 11:33:06  COBOL    1
_   TEMP2DB2              2005/07/11  2005/07/11 12:48:19  COBOL    1
_   TEMP3DB2              2005/07/11  2005/07/11 13:10:32  COBOL    1
_   TEMP3333              2006/07/30  2006/11/08 22:46:01  COBOL    2
_   TTT4                  2006/08/03  2006/08/03 10:45:57  COBOL    1
 F1=Help    F2=Split   F3=Exit    F4=Expand  F5=RFind   F7=Up      F8=Down
 F9=Swap    F10=Left   F11=Right  F12=Cancel
```

Figure 24-49 Previous Template Member Selection panel, to update or create new templates

Figure 24-49 shows the templates list and lets you select the templates that you want to update. Type **U** in the prefix command (*Sel* column) in order to update and verify your template definitions. Press Enter to display the next Copy Selection display as shown in Figure 24-50.

```
 Process   Options   Help
FMNPTWMC ger                 Copybook Selection                 Line 1 of 1
Command ===> _____________________________________________________ Scroll CSR

Cmd R Member    Lib 01 Field Name
*** **** Top of data ****
u    ZZZ3COPY    1 N
*** **** End of data ****
```

Figure 24-50 This panel shows the original copybook

Note: From the previous panel, the column *Member* specifies the copybooks from which the template has been generated the first time.

Press Enter and then PF3 to update and verify if the definition between Copy and Template are correct. If not, the Update Template utility updates the template according the last Copybook definitions (ZZZ3COPY). Type **e** in the Prefix Command line as shown in Figure 24-51 and press Enter.

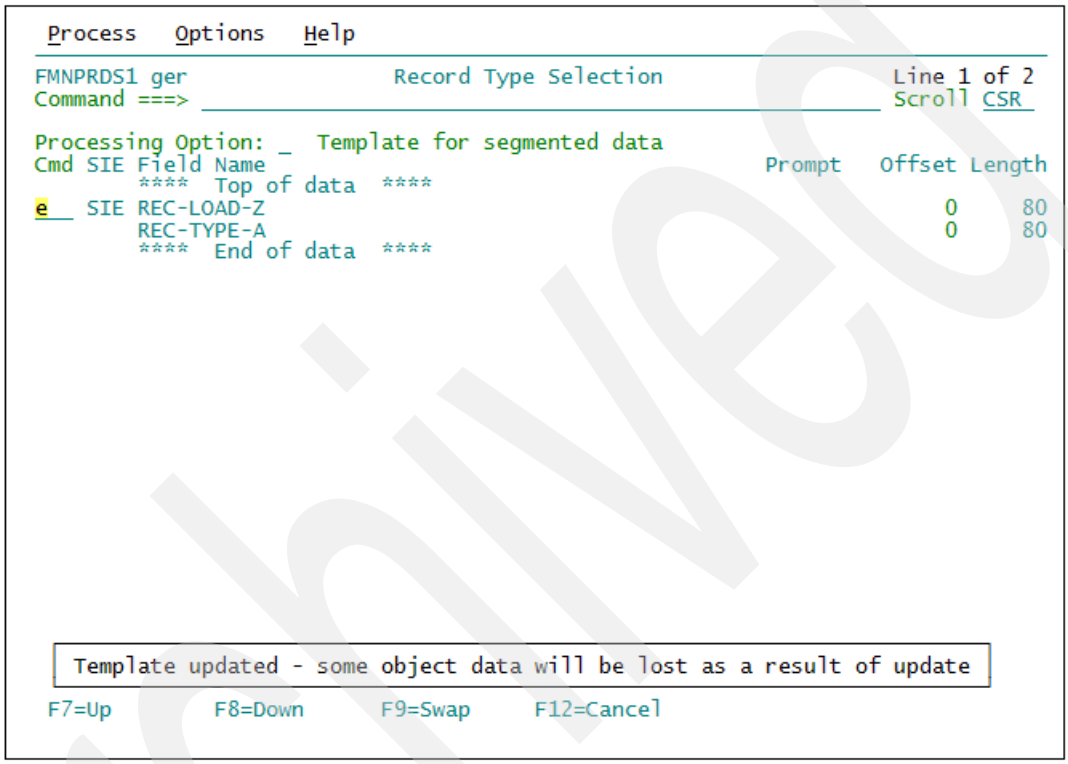


Figure 24-51 Record type selection

The Field Selection/Edit panel is displayed as shown in Figure 24-52. Figure 24-52 shows the template definitions (of TEMPZZZ3) and the alter that we are going to make. In this sample we alter the Output definition (de-select of REC-DATASS instead of GG, MM and AAAA. We alter the selected conditions related to Figure 24-53, Record Selection Criteria, and perform the following tasks as shown in Figure 24-54, Figure 24-55, Figure 24-56, and Figure 24-57:

- Remove parentheses.
- Alter the REXX condition for the field IMP-COGNOME.
- Delete the condition to the SEX field.
- Run (PF6) the template to display the new layout and results.
- Save the new Template as TEMPTST3.

```
 Process   Options   Help
FMNPRDS2 ger                  Field Selection/Edit                 Line 1 of 21
Command ===> 2                                                     Scroll CSR

---------- Criteria - Enter 1 or 2 to specify expression by field ----------
1 Id : ¢2 = 'Z'                                                              +
2 Sel: FLD_CO(22,15,C,'y') ! (FLD_CO(22,15,C,'D') ! ¢19 = ' ')               +
Offset      0
Cmd Seq SHC Ref Field Name                              Picture  Type Start Length
                ****  Top of data  ****
___           1 1 REC-LOAD-Z                                     AN      1     80
___           2  2 REC-TYPE                             X        AN      1      1
___ S         3  2 IMP-NO                               X(04)    AN      2      4
___           4  2 IMP-NO-GRP REDEFINES IMP-NO                   AN      2      4
___ S         5   3 IMP-NO-P                            X(01)    AN      2      1
___           6   3 IMP-NO-KEY                          X(03)    AN      3      3
___ S         7  2 REC-SET                              X(02)    AN      6      2
___ S         8  2 REC-REP                              X(03)    AN      8      3
___ S         9  2 REC-NOME                             X(11)    AN     11     11
___ S        10  2 IMP-COGNOME                          X(15)    AN     22     15
___ S        11  2 REC-DIPART                           X(3)     AN     37      3
___ S        12  2 JOB                                  X(08)    AN     40      8
___ S        13  2 REC-DATASS                                    AN     48     10
___          14   3 GG                                  XX       AN     48      2
___          15   3 FILLER                              X        AN     50      1
___          16   3 MM                                  XX       AN     51      2
___          17   3 FILLER                              X        AN     53      1
___          18   3 AAAA                                XXXX     AN     54      4
___ S        19  2 SEX                                  X        AN     58      1
 F1=Help     F2=Split    F3=Exit     F4=Expand   F5=RFind    F6=RunTemp
 F7=Up       F8=Down     F9=Swap     F10=Left    F11=Right   F12=Cancel
```

Figure 24-52 TEMPZZZ3 definitions. It could be kept if you use SAVEAS and change the Template's name

```
 Process   Options   Help
FMNPDYNT ger              Record Selection Criteria          Cannot delete line
Command ===> run temp                                        Scroll CSR

Cmd Con ( Field Name                          Op  Value                     )
    <-> - <---+----1----+----2----+----3----> <-> <---+----1----+----2----+-- -
*** ****  Top of data  ****
___     _ REC-LOAD-Z                          ___ ____________________________ _
___ AND _ REC-TYPE                            ___ ____________________________ _
___ AND _ IMP-NO                              ___ ____________________________ _
___ AND _ IMP-NO-GRP                          ___ ____________________________ _
___ AND _ IMP-NO-P                            ___ ____________________________ _
___ AND _ IMP-NO-KEY                          ___ ____________________________ _
___ AND _ REC-SET                             ___ ____________________________ _
___ AND _ REC-REP                             ___ ____________________________ _
___ AND _ REC-NOME                            ___ ____________________________ _
___ AND _ IMP-COGNOME                         CO  'y'                        _
___ OR  _ IMP-COGNOME                         CO  'Y'                        _
___ AND _ REC-DIPART                          ___ ____________________________ _
___ AND _ JOB                                 ___ ____________________________ _
___ AND _ REC-DATASS                          ___ ____________________________ _
___ AND _ GG                                  ___ ____________________________ _
___ AND _ FILLER                              ___ ____________________________ _
___ AND _ MM                                  ___ ____________________________ _
___ AND _ FILLER                              ___ ____________________________ _
___ AND _ AAAA                                ___ ____________________________ _
___ AND _ SEX                                 ___ ____________________________ _
___ AND _ REC-NO                              ___ ____________________________ _
___ AND _ FILLER                              ___ ____________________________ _
 F1=Help     F2=Split    F3=Exit     F4=CRetriev F5=RFind    F6=RunTemp
 F7=Up       F8=Down     F9=Swap     F10=Left    F11=Right   F12=Cancel
```

Figure 24-53 PF6 to run the template with the new definitions

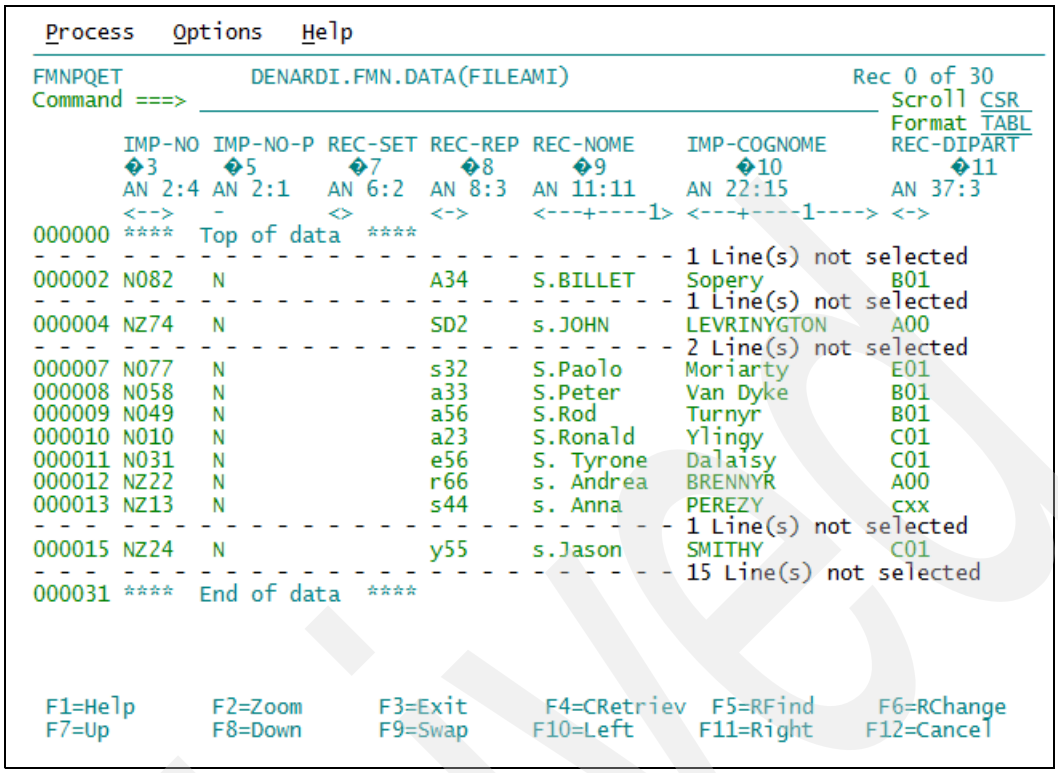


Figure 24-54 Display of the new layout and results

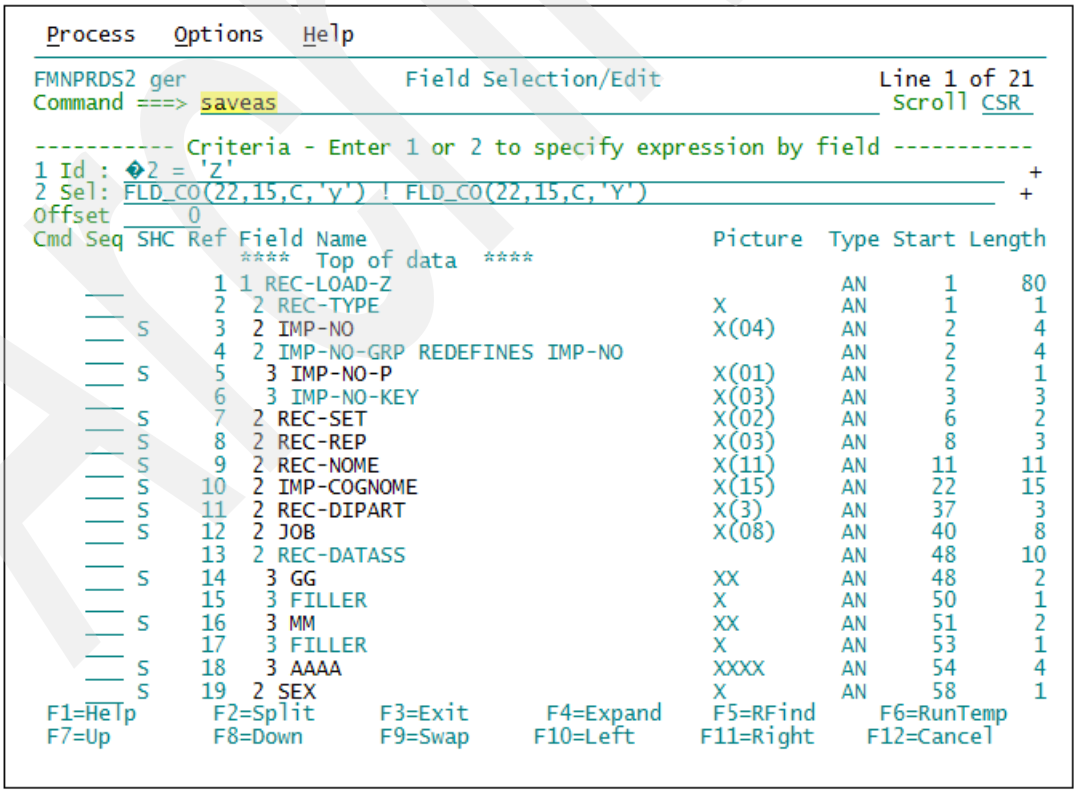


Figure 24-55 Save the new definitions

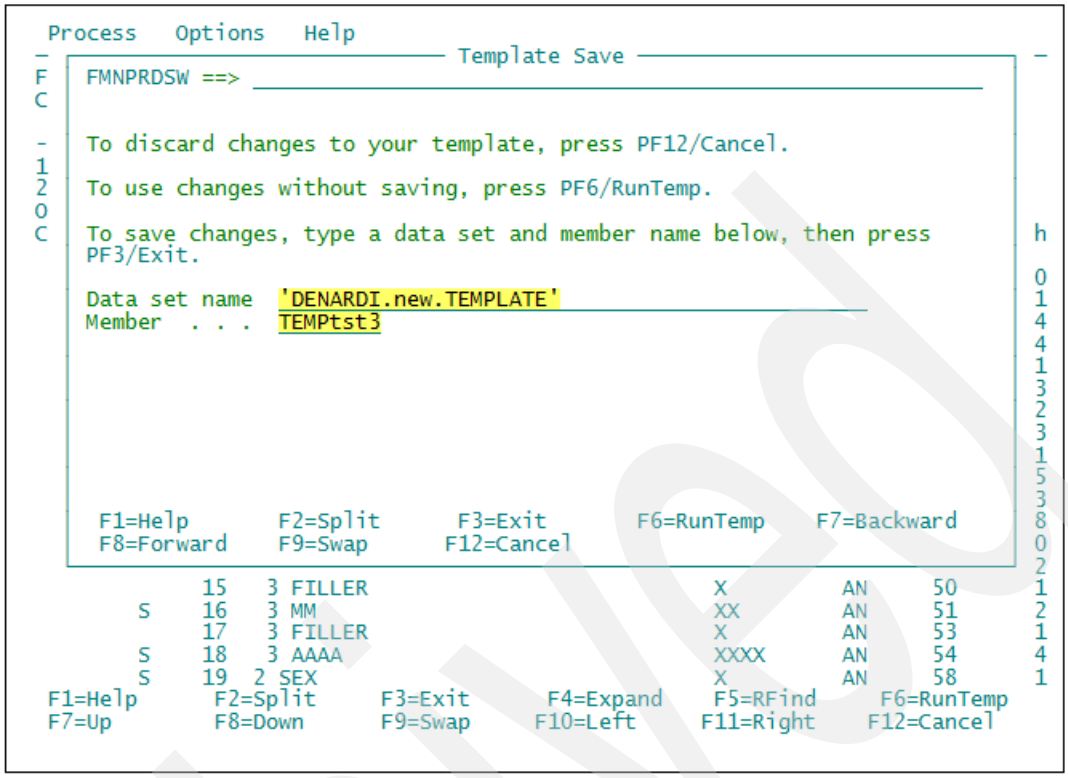


Figure 24-56 Save the new Template as TEMPTST3

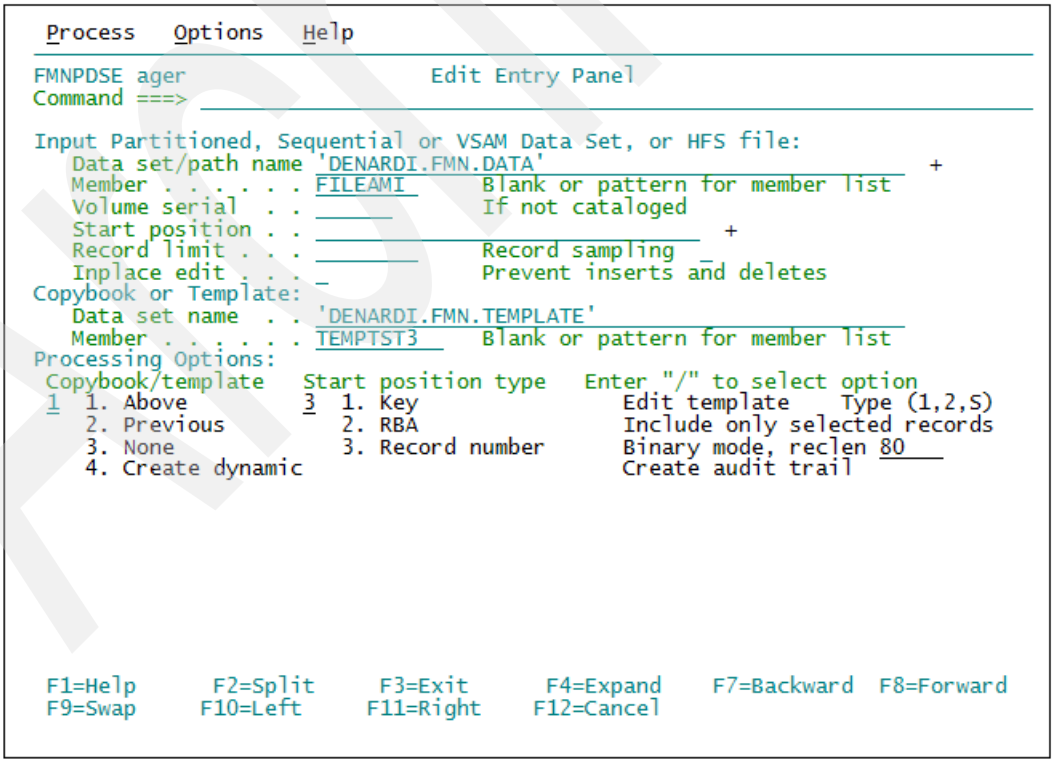


Figure 24-57 A new template has been created and TEMPZZZ3 has not been modified

Following all the previous steps described above, we have just created a new Template (TEMPTST3) with the new update definitions. Typing SAVEAS in the Command Line, before you exit, permits you to create a new Template TEMPTST3 and to close the changed one starting from the TEPZZZ3 template.

24.15 Working with VTOC

Follow these steps to work with VTOC:

1. Select option **3** (Utilities, on the FM main menu), then option **5** (VTOC). This takes you to the Display VTOC panel as shown in Figure 24-58. Figure 24-58 permits you to choose the format display. If you keep “blank” in the Command line, the output is a list of datasets. From here you can choose to display the volumes (option **V**).

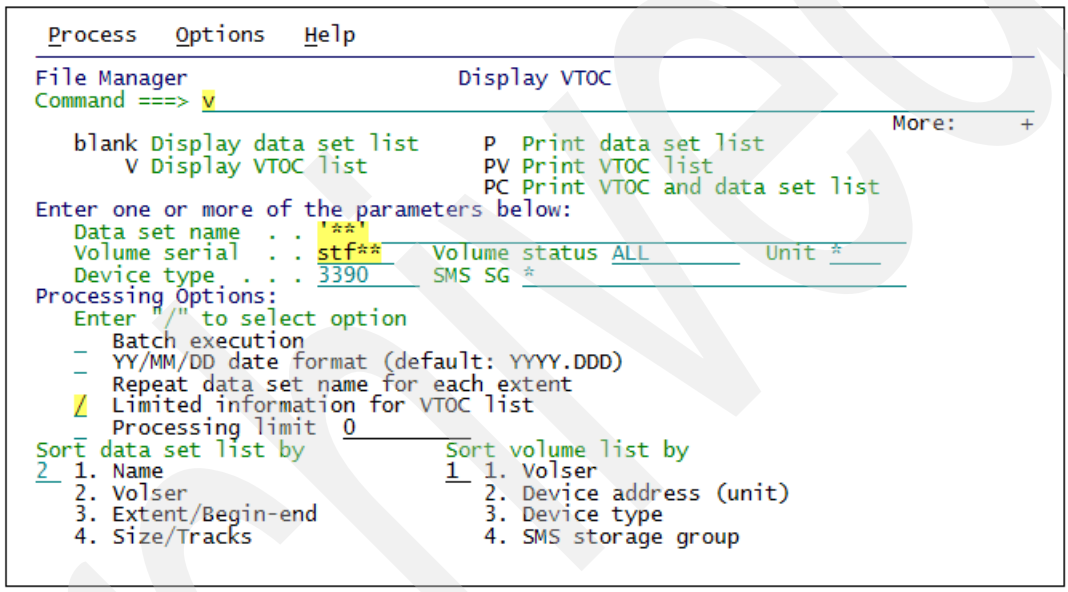


Figure 24-58 List of volumes for using VTOC (part 1 of 3)

You can use fully qualified or generic *data set name* and *volume serial* to limit the output. The volume selection can be further refined by entering additional criteria for the Volume status, Unit, Device type and/or SMS Storage group. There are two new Processing Options to limit the output:

- **Limited information for VTOC list:** Choose "/" to show only limited information for each volume displayed on the volume VTOC list (V command). Selecting the limited information option provides faster response time at the expense of some detailed information about the volumes. This option applies only to online processing.
- **Processing limit:** Allows for a limit of data set entries to be displayed or printed when this option is selected. Default is zero

2. You can select one volume here to view the list of the files it contains as shown in Figure 24-59 and Figure 24-60.

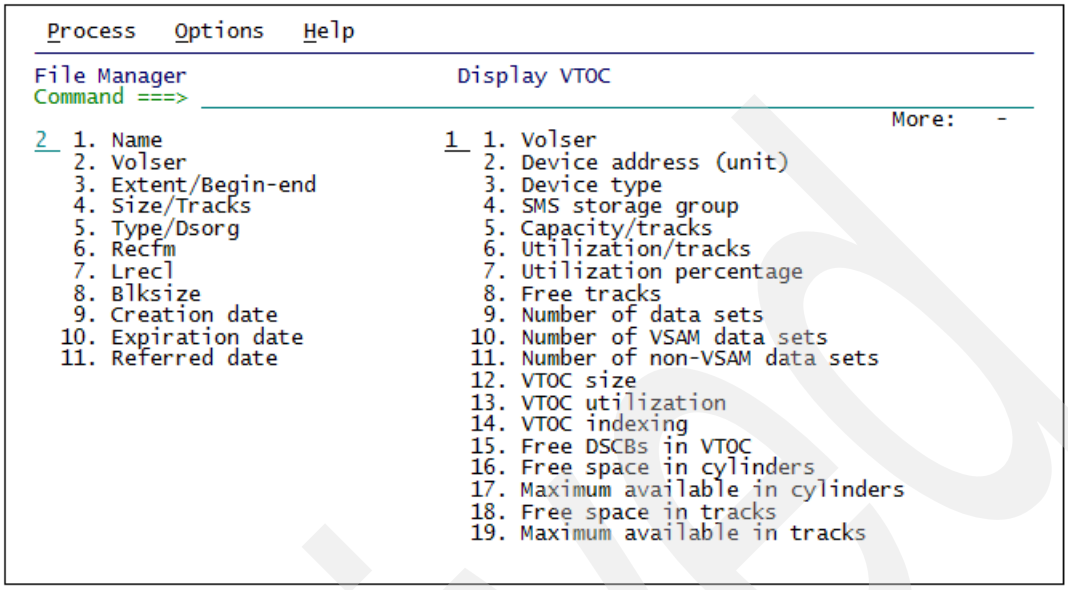


Figure 24-59 List of volumes for using VTOC (part 2 of 3)

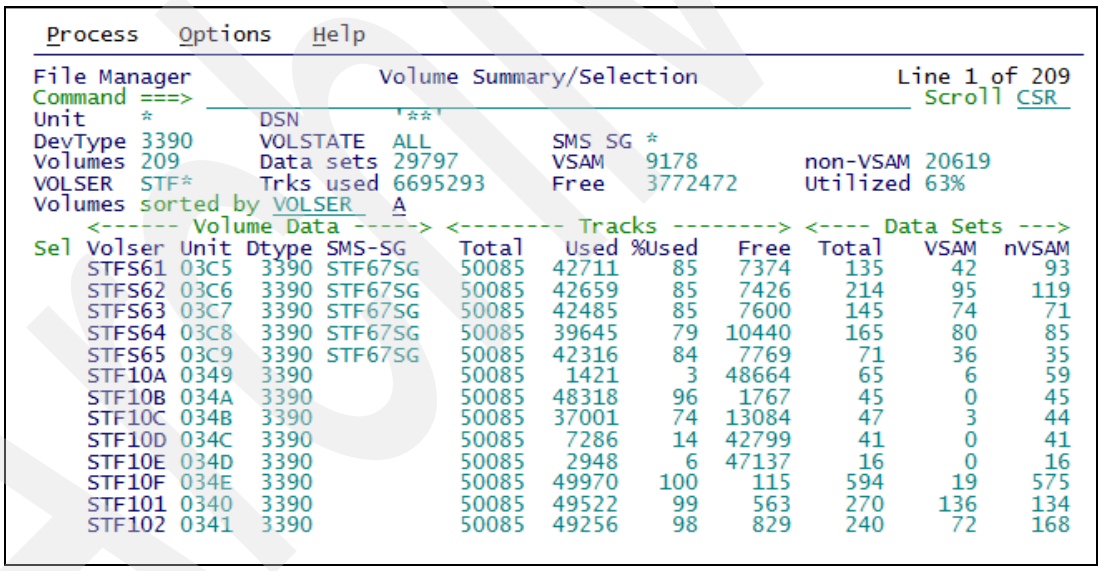


Figure 24-60 List of volumes for using VTOC (part 3 of 3)

3. If you want the list for all volumes, go back to the Display VTOC panel and enter a generic name in the Data Set Name field, as shown in Figure 24-61.

```
 Process   Options   Help
------------------------------------------------------------------------------
File Manager           Display VTOC Data Set List                Line 15 of 28
Command ===>                                                      Scroll CSR
Unit     *        DSN        'AMINTOR.BOOK2006.**'
DevType 3390      VOLSTATE   ALL          SMS SG *
Volumes 14        Data sets  28           VSAM    18          non-VSAM 10
VOLSER  STF**     Trks used  2360         Free    N/A         Utilized N/A
    --- Data Set Name --- sorted by VOLSER   A -              RECFM   BLKSIZE
Cmd 1...5...10...15...20...25...30...35...40.... Tracks DSORG   LRECL
    AMINTOR.BOOK2006.PDHVSAM.INDEX                    1 VS  U        0  4096
    AMINTOR.BOOK2006.OUTFILE                        750 PS  FB      80 27920
    AMINTOR.BOOK2006.CUSTFILE.SEQ.FRO9B2SC           30 VS  U        0  4096
    AMINTOR.BOOK2006.CUSTFILE.SEQ.FY5UM2SC            1 VS  U        0  4096
    AMINTOR.BOOK2006.TRANFILE                        75 PS  FB      80 27920
    AMINTOR.BOOK2006.DATA                           120 PO  FB      80 27920
    AMINTOR.BOOK2006.TRANFILE.VSAM.DATA             300 VS  U        0  4096
ex  AMINTOR.BOOK2006.TRANFILE.VSAM.INDEX              1 VS  U        0  4096
    AMINTOR.BOOK2006.PDHVSAM.AIX1.DATA               15 VS  U        0  4096
    AMINTOR.BOOK2006.PDHVSAM.AIX1.INDEX               1 VS  U        0  4096
    AMINTOR.BOOK2006.VARDATA                        120 POE VB    8000 32760
    AMINTOR.BOOK2006.PDHVSAM1.DATA                   15 VS  U        0  4096
    AMINTOR.BOOK2006.PDHVSAM1.INDEX                   1 VS  U        0  4096
    AMINTOR.BOOK2006.TEMPLATE                         4 PO  FB      80 27920
```

Figure 24-61 Display VTOC dataset List

4. From the Prefix Command line, you can select an Entry to view more details (**v**), edit the Dataset (**e**) or use this function to display the begin and end of the extents of a data set as shown in Figure 24-62.

```
 Process   Options   Help
------------------------------------------------------------------------------
File Manager          VSAM Statistics and Extent Detail
Command ===>

VSAM Catalog Entry:
   Data set name . 'AMINTOR.BOOK2006.TRANFILE.VSAM'
   Catalog ID  . . 'SYS1.ICFCAT.VSTF600'
                                                                      More:   +
VSAM Statistics:
 Component ---------------- Records ------------------ ------ Splits -------
  -------  ---Total-- -Deleted-- -Inserted- -Updated-- ----CI---- ----CA----
  Data             1          0          0          0          0          0
  Index            1          0          0          0          0          0

Data Extents:
Total volumes: 1   Extents: 1   Tracks: 300
 Ext Volume   ---Begin-end--- Reltrk,------  ----------- Kilobytes ----------
 num serial   Cyl-hd   Cyl-hd       numtrks   Low-alloc High-alloc  High-used
     STF64E(1)  Extents: 1  Tracks: 300                    14800        740
   1 --"--    765  0   784 14  11475,300              0      14799

Index Extents:
Total volumes: 1   Extents: 1   Tracks: 1
```

Figure 24-62 Results of EXTENTS command

Absolute disk extents and disk extents relative to the beginning of the data set are displayed. You can use the information for all basic disk functions which prompt you for the location of a disk extent.

Notice the differences in the information given by this data set list display and that supplied by File Manager DSLIST (option **3.4**) or the ISPF DSLIST (option **3.4**), as shown in Figure 24-63.

```
 Process   Options   Help
 ──────────────────────────────────────────────────────────────────────────────
 File Manager                     Data Set List                     Line 1 of 37
 Command ===> ____________________________________________________ Scroll CSR
 Catalog ID ''                                                     Types ALL
          --- Data Set Name --- sorted by NAME       ---    Entry  Prim  M Created
 Command  1...5...10...15...20...25...30...35...40....     type  volume V YYYY.DDD
          AMINTOR.BOOK2006.COPYBOOK                        NVSAM STF60C   2006.308
          AMINTOR.BOOK2006.CUSTFILE                        KSDS           2006.308
            AMINTOR.BOOK2006.CUSTFILE.DATA                 DATA  STF620   2006.308
            AMINTOR.BOOK2006.CUSTFILE.INDEX                INDEX STF620   2006.308
          AMINTOR.BOOK2006.CUSTFILE.SEQ                    KSDS           2006.308
            AMINTOR.BOOK2006.CUSTFILE.SEQ.DATA             DATA  STFS63   2006.308
            AMINTOR.BOOK2006.CUSTFILE.SEQ.INDEX            INDEX STFS63   2006.308
          AMINTOR.BOOK2006.CUSTFILE.SEQ.CLUSTER            AIX            2006.311
            AMINTOR.BOOK2006.CUSTFILE.SEQ.FRO9B2SC         DATA  STF636   2006.311
            AMINTOR.BOOK2006.CUSTFILE.SEQ.FY5UM2SC         INDEX STF636   2006.311
          AMINTOR.BOOK2006.CUSTFIL2                        KSDS           2006.308
            AMINTOR.BOOK2006.CUSTFIL2.DATA                 DATA  STF505   2006.308
            AMINTOR.BOOK2006.CUSTFIL2.INDEX                INDEX STF505   2006.308
          AMINTOR.BOOK2006.DATA                            NVSAM STF64E   2006.310
          AMINTOR.BOOK2006.DATA.SEQ                        NVSAM STF505   2006.307
          AMINTOR.BOOK2006.JCL                             NVSAM STF508   2006.308
          AMINTOR.BOOK2006.OUTFILE                         NVSAM STF632   2006.308
```

Figure 24-63 File Manager DSLIST

24.16 Getting information about a load module

It can be useful to get information about an existing load module. Use option **3** (Utilities) then option **10** (Loadlib) to get to the panel as shown in Figure 24-64.

```
 Process   Options   Help
 ──────────────────────────────────────────────────────────────────────────────
 File Manager                Load Module Information
 Command ===> _________________________________________________________________

 Input:
    Data set name . . . . . . 'adtools.FM710.SFMNMOD1'
    Member  . . . . . . . . . fmnmod16      Blank or pattern for member list
    Volume serial . . . . . . ______        If not cataloged


 Processing Options:

    Order CSECTs by                         Output to
    1    1. Address                         1    1. Display
         2. Name                                 2. Printer

    Enter "/" to select option
    _  YY/MM/DD date format (default: YYYY.DDD)
```

Figure 24-64 Utilities and Loadlib options selected

You can specify the module name in Figure 24-64 or press Enter and select the module you want from the resultant list.

For the module selected, you get the components, their size, and the short name (if available), or the program number of the compiler used, as shown in Figure 24-65.

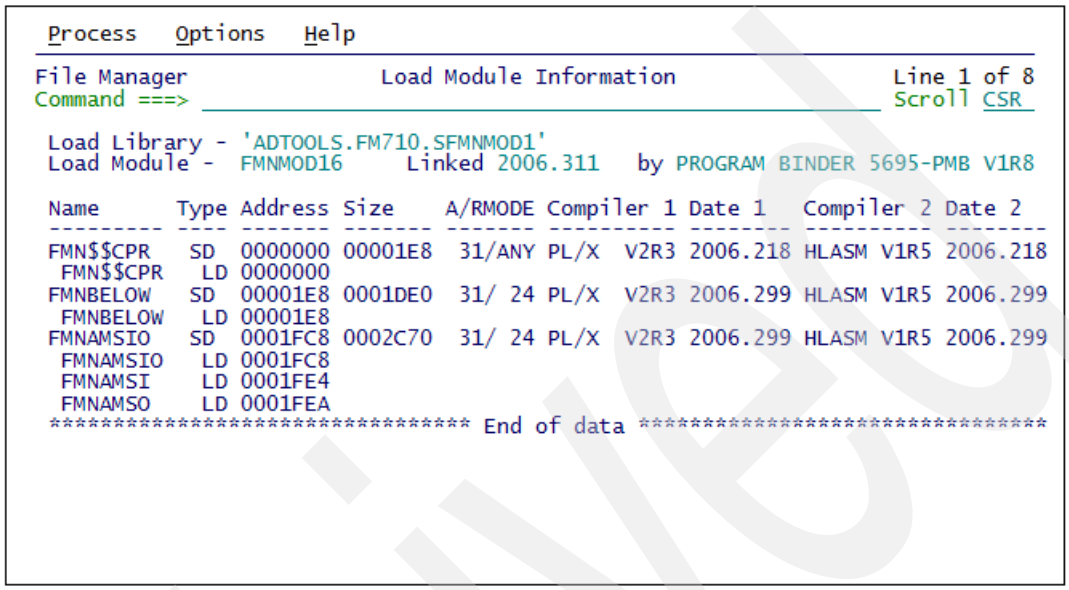


Figure 24-65 Load module information

24.17 Template Build utility

You can use the File Manager Template Build Utility to create corresponding templates from selected copybooks. The utility builds one template per copybook and the template name is derived from the input member name and any output member mask that has been specified. If no template member mask is supplied, then the name is the same as the copybook name. You can run the utility in batch and foreground.

The panel in Figure 24-66 is invoked by the following path starting from the Main panel (FMNSTASK): option **3** (Utilities), then option **7** (Template), and then option **3** (Template build). It is a new functionality provided with the new FM Version.

To use the next panel, you have to specify an input copybook library and an output partitioned template data set. The fields are described below:

- Copybook Data set name: Specify a partitioned data set that contains the copybook. Specify a generic data set name to select from a list.
- Copybook Member: Enter a blank or a pattern for a member selection list. (Note: This is ignored for advanced member selection).
- Template Data set name: Specify a partitioned data set in which the templates created are stored.

- Member mask: Specify a pattern to rename members in the output partitioned data set based upon the member names in the input partitioned data set. A member name pattern can consist of any characters that are valid in a member name and two special pattern characters: the asterisk (*) and the percent sign (%).
  - (*): is a place holding character that means multiple characters with no change. Only one asterisk should appear in the mask. Any subsequent asterisk characters are treated as percent signs. For example, if you enter:
    - ABC* - The renamed members all begin with ABC followed by the remainder of the old member name.
  - (%) is a place holding character that means a single character with no change. As many percent symbols as necessary can appear anywhere in a member name. For example, if you enter:
    - %%%A* - The first 3 characters of the renamed members remain unchanged, the fourth character is replaced with the letter "A" and the remainder of the old member name remains unchanged.

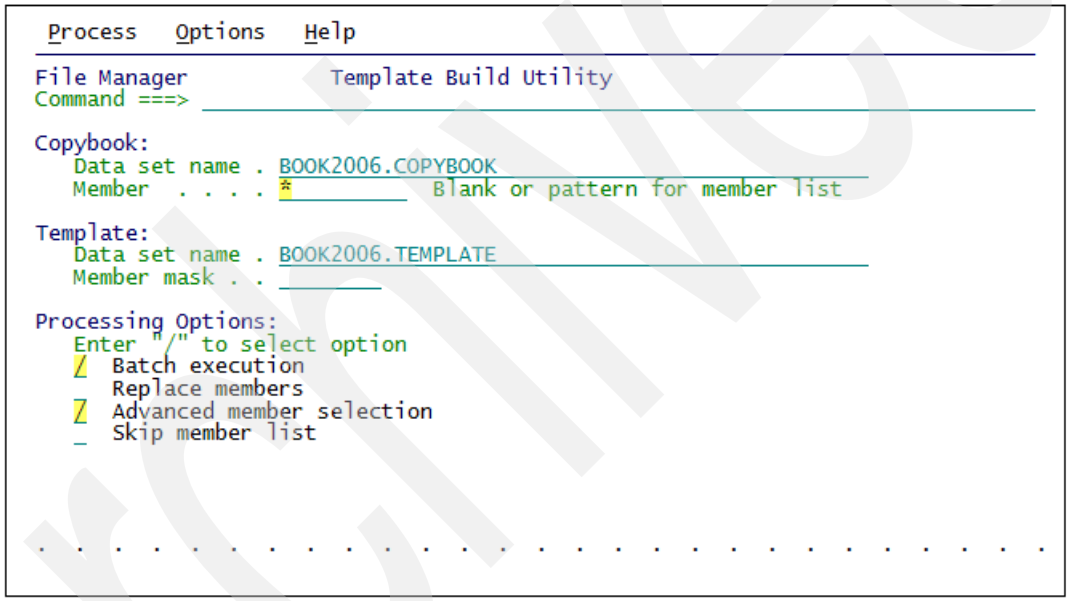


Figure 24-66 Template build utility

You can choose between the following *Process options*:

- **Batch execution:** Type "/" to edit the JCL to run the function in batch.
- **Replace existing members:** Type "/" to replace like named members in the output partitioned data set.
- **Advanced member selection:** Type "/" to specify a range of members to be selected rather than a specific or generic member name. If you specify this option, the copybook member name is ignored.
- **Skip member list:** Type "/" to run without showing the member selection list. This option is ignored if errors are found while processing; for example, duplicate output member names when an output member mask has been used.

Figure 24-67 and Figure 24-68 show the libraries (INPUT and OUTPUT, corresponding to COPYBOOK and TEMPLATE) before running the utility. We run the Function in batch mode (the previous panel shows how) because we chose to type / in the Batch execution field.

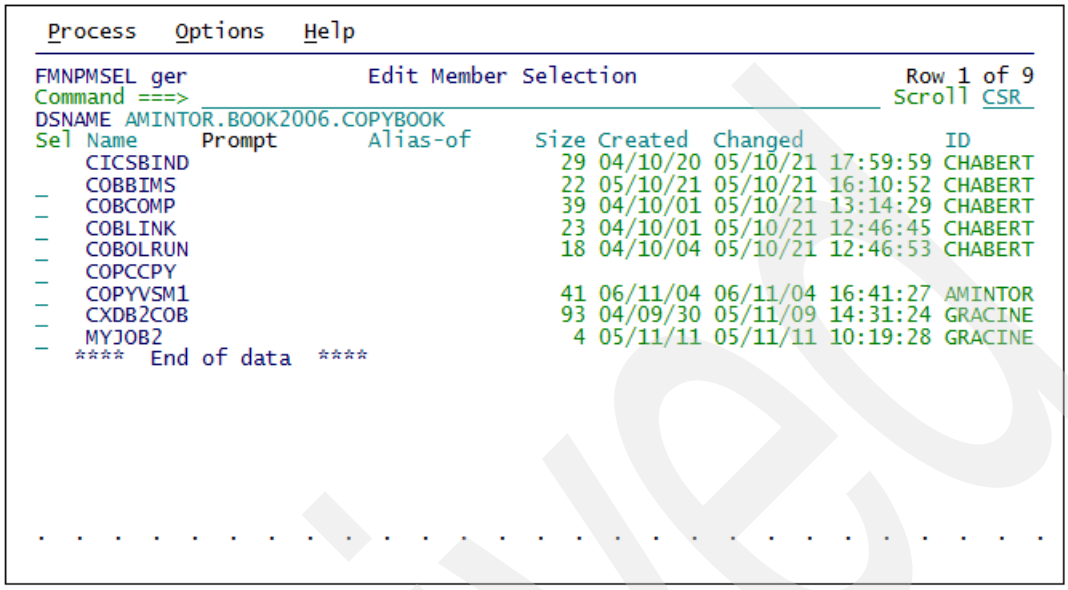


Figure 24-67 INPUT library before the utility execution

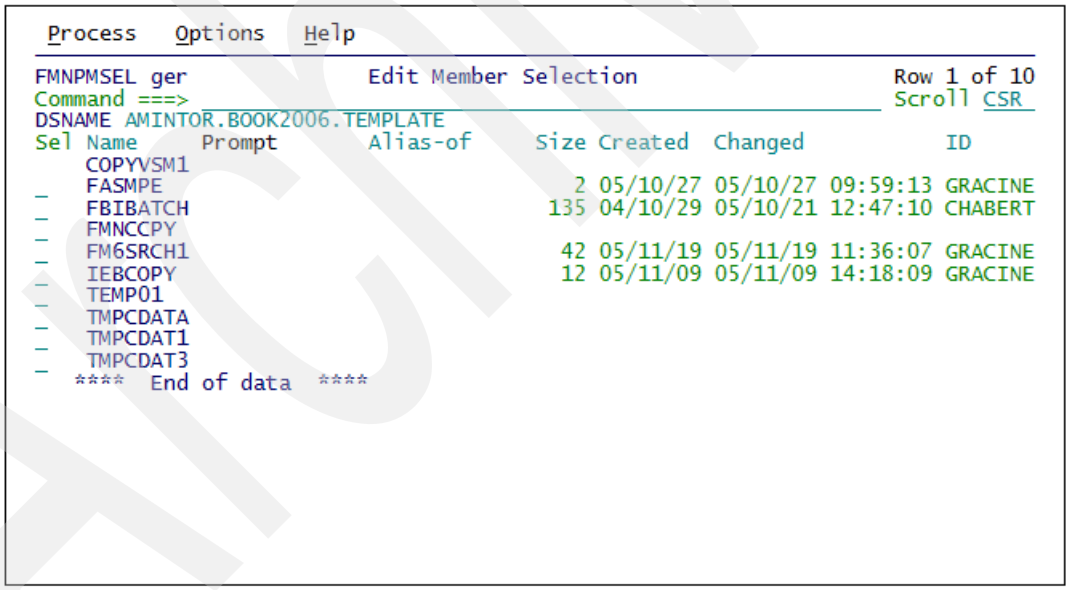


Figure 24-68 Output library before the utility execution

Since we chose *Advanced Member Selection*, a panel similar to Figure 24-69 is shown to set the Member name or mask to restrict the list of members eligible for processing. In this sample we specified the pattern as `COP*`:

:

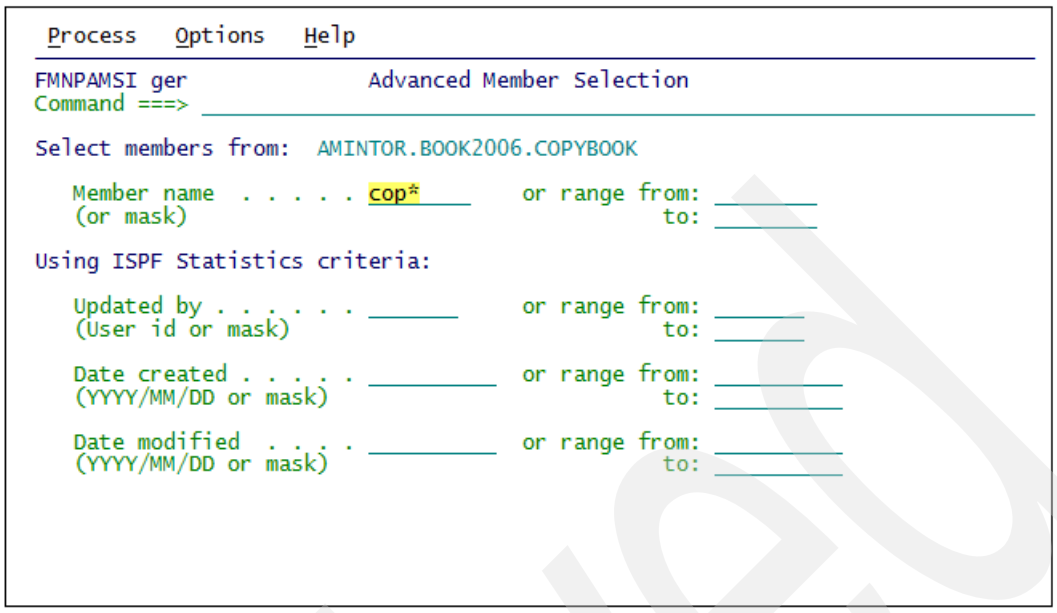


Figure 24-69 Filter using COP to restrict the list of member to process*

You can use File Manager Advanced Member Selection to restrict the list of PDS members eligible for processing. There are four criteria which can be used in this process. For each of the criteria, you have the choice of specifying an exact value or mask on the left of the panel, or a range of values on the right of the panel.

Press Enter to process the member selection list. Figure 24-70 shows how to select the PDS members.

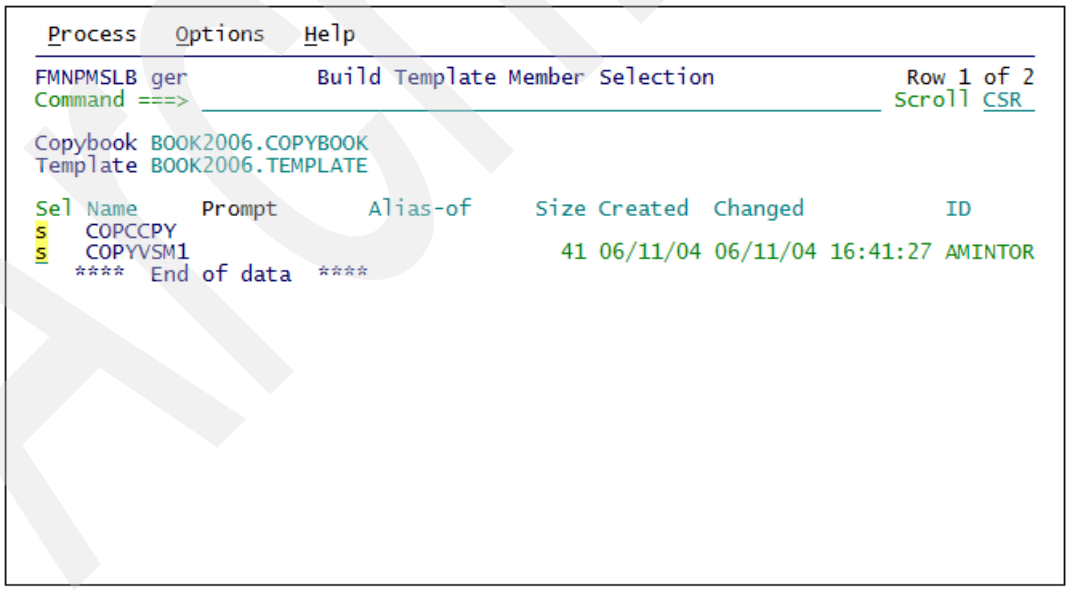


Figure 24-70 Member Selection Panel

Select the members and press Enter to generate the following JCL as shown in Example 24-21.

Example 24-21 Build Template Function: JCL and SYSOUT

```
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (*),
000002 //             AMINTOR,MSGCLASS=A,
000003 //             NOTIFY=AMINTOR,CLASS=A,
000004 //             MSGLEVEL=(1,1)
000005 //*
000006 //FILEMGR  EXEC PGM=FMNMAIN
000007 //*TEPLIB  DD DSN=FMN.SFMNMOD1,DISP=SHR
000008 //*        DD DSN=IGY.SIGYCOMP,DISP=SHR
000009 //*FMNCOB  DD DUMMY     Uncomment to force use of FM COBOL Compiler
000010 //SYSPRINT DD SYSOUT=*
000011 //FMNTSPRT DD SYSOUT=*
000012 //SYSTERM  DD SYSOUT=*
000013 //SYSIN    DD *
000014 $$FILEM BTB DSNIN=AMINTOR.BOOK2006.COPYBOOK,
000015 $$FILEM MEMBER=COP*,
000016 $$FILEM MEMLIST=(COPCCPY,
000017 $$FILEM          COPYVSM1),
000018 $$FILEM DSNOUT=AMINTOR.BOOK2006.TEMPLATE
****** **************************** Bottom of Data ****************************


 SDSF OUTPUT DISPLAY AMINTORB JOB03812  DSID   102 LINE 9       COLS 02- 81
 COMMAND INPUT ===>                                           SCROLL ===> CSR
PRTTRANS=ON        DUMP=UPDOWN   TAPELBL=SL     CYLHD=ABSOLUTE
SMFNO=000          PRTDISP=MOD   USEIOX=DISABLE IOX=
TEMPHLQ=
ABENDCC=NONE       WIDEPRT=YES

IBM File Manager for z/OS
$$FILEM BTB DSNIN=AMINTOR.BOOK2006.COPYBOOK,
$$FILEM MEMBER=COP*,
$$FILEM MEMLIST=(COPCCPY,
$$FILEM          COPYVSM1),
$$FILEM DSNOUT=AMINTOR.BOOK2006.TEMPLATE
IBM File Manager for z/OS
          Template Build Report
Copybook       Template    Status
--------------------------------------------------------------------------------
COPCCPY        COPCCPY     Created
COPYVSM1       COPYVSM1    Not replaced
2 members read : Template : 1 Created  1 Not replaced  0 Errors
******************************** BOTTOM OF DATA ********************************
```

Here we provide the SYSOUT together with the JCL to compare and verify the results according to the original INPUT and OUTPUT libraries. We note that only one member has been added inside the TEMPLATE library because COPYVSM1 was defined before.

24.18 Comparing data sets: Compare Utility

For this example we compare two members of the same data set source to determine their differences, shown in Figure 24-71 and Figure 24-72.

```
 Process   Options   Help
------------------------------------------------------------------------------
FMNPDSMO ger           Compare Utility : Old Data Set              End of data
Command ===> ____________________________________________________________________

"Old" Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . 'CHABERT.TRADER.SOURCE'                            +
   Member  . . . . . . . . mytradmd       (Blank or pattern for member list)
   Volume serial . . . . . ______
   Start key . . . . . . . ________________________________ key or slot
   Skip count  . . . . . . _______      number of records to be skipped
   Compare count . . . . . ALL          number of records to be compared

"Old" Copybook or Template:
   Data set name . . . . . ____________________________________________________
   Member  . . . . . . . . _________

Processing Options:
   Copybook/template usage            Enter "/" to select option
   3  1. Above                           Edit template    Type (1,2,S)
      2. Previous                     /  Batch execution
      3. None                            Binary mode, reclen
      4. Create dynamic
```

Figure 24-71 Compare utility panel (part 1 of 2)

```
 Process   Options   Help
------------------------------------------------------------------------------
FMNPDSMN ger           Compare Utility : New Data Set
Command ===> ____________________________________________________________________

"New" Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . 'CHABERT.TRADER.SOURCE'                            +
   Member  . . . . . . . . mytradmi       (Blank or pattern for member list)
   Volume serial . . . . . ______
   Start key . . . . . . . ________________________________ key or slot
   Skip count  . . . . . . _______      number of records to be skipped
   Compare count . . . . . _______      number of records to be compared

"New" Copybook or Template:
   Data set name . . . . . ____________________________________________________
   Member  . . . . . . . . _________

Processing Options:
   Copybook/template usage            Enter "/" to select option
   3  1. Above                           Edit template    Type (1,2,S)
      2. Previous                        Binary mode, reclen
      3. None
      4. Create dynamic
```

Figure 24-72 Compare utility panel (part 2 of 2)

Select the compare utility: option **3** (Utilities), then option **11** (Compare).

Specify both files names (and templates) that you want to compare, or select them in lists, and then specify the characteristics of the comparison, as shown in Figure 24-73.

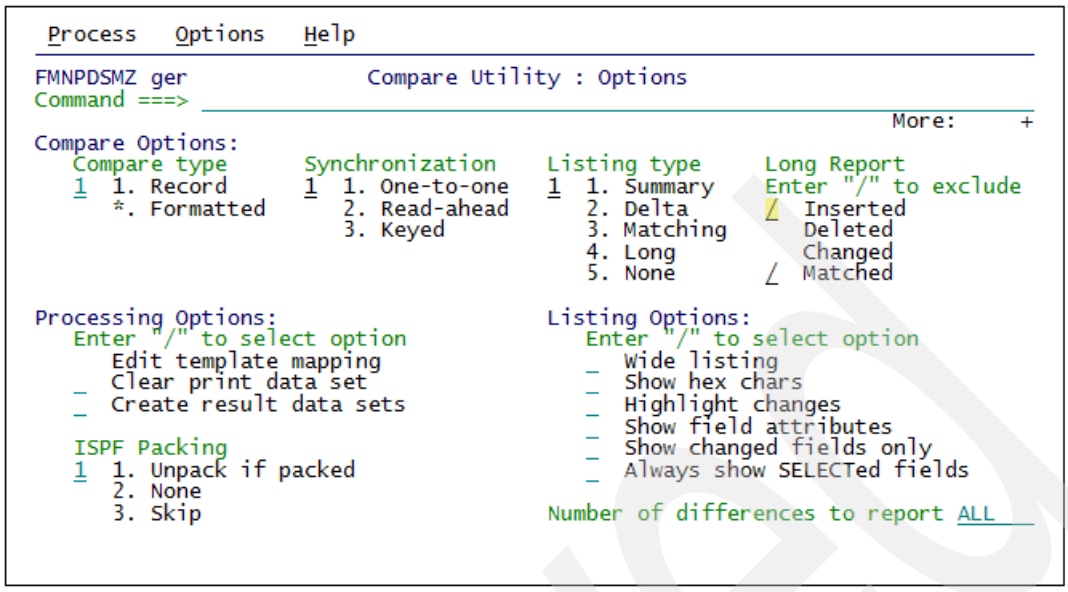


Figure 24-73 Comparison characteristics

Figure 24-73 describes the processing options found on the *Compare utility options* panel. In order to control the type of listing, and the numbers of occurrences produced by the compare utility, we have added the following *Long Report Exclusions*:

Inserted: Exclude inserted records from the report.
Deleted: Exclude deleted records from the report.
Changed: Exclude changed records from the report.
Matched: Exclude matched records from the report.

These options are ignored if other than a Listing type of Long has been selected.

The results of the operation is displayed in Example 24-22.

Example 24-22 Comparison step of batch job and SYSOUT

```
000001 //AMINTORB JOB (A),
000002 //            AMINTOR,MSGCLASS=A,
000003 //            NOTIFY=AMINTOR,CLASS=A,
000004 //            MSGLEVEL=(1,1)
000005 //*
000006 //FILEMGR  EXEC PGM=FMNMAIN
000007 //*TEPLIB  DD DSN=FMN.SFMNMOD1,DISP=SHR
000008 //*        DD DSN=IGY.SIGYCOMP,DISP=SHR
000009 //*FMNCOB  DD DUMMY     Uncomment to force use of FM COBOL Compiler
000010 //SYSPRINT DD SYSOUT=*
000011 //FMNTSPRT DD SYSOUT=*
000012 //SYSTERM  DD SYSOUT=*
000013 //SYSIN    DD *
000014 $$FILEM DSCMP TYPE=RECORD,
000015 $$FILEM PACK=UNPACK,
000016 $$FILEM SYNCH=ONETOONE,
000017 $$FILEM LIST=SUMMARY,
000018 $$FILEM NUMDIFF=ALL,
000019 $$FILEM DSNOLD=CHABERT.TRADER.SOURCE(MYTRADMD),
000020 $$FILEM SKIPOLD=0,
000021 $$FILEM CMPOLD=ALL,
000022 $$FILEM SKIPNEW=0,
```

```
000023 $$FILEM CMPNEW=ALL,
000024 $$FILEM DSNNEW=CHABERT.TRADER.SOURCE(MYTRADMI)
000025 /*
****** ***************************** Bottom of Data ****************************

 SDSF OUTPUT DISPLAY AMINTORB JOB03772  DSID   102 LINE 4       COLS 02- 81
 COMMAND INPUT ===>                                            SCROLL ===> CSR
 DEFAULT SET FUNCTION VALUES:

PRINTOUT=SYSPRINT  HEADERPG=YES  ASCII=BOTH     RECLIMIT=(1,*)
PRINTLEN=132       PAGESKIP=NO   PAD=OFF        EOD=/*
PAGESIZE=60        DATAHDR=YES   DBCSPRT=OFF    LANGUAGE=ENGLISH
PRTTRANS=ON        DUMP=UPDOWN   TAPELBL=SL     CYLHD=ABSOLUTE
SMFNO=000          PRTDISP=MOD   USEIOX=DISABLE IOX=
TEMPHLQ=
ABENDCC=NONE       WIDEPRT=YES

IBM File Manager for z/OS
$$FILEM DSCMP TYPE=RECORD,
$$FILEM PACK=UNPACK,
$$FILEM SYNCH=ONETOONE,
$$FILEM LIST=SUMMARY,
$$FILEM NUMDIFF=ALL,
$$FILEM DSNOLD=CHABERT.TRADER.SOURCE(MYTRADMD),
$$FILEM SKIPOLD=0,
$$FILEM CMPOLD=ALL,
$$FILEM SKIPNEW=0,
$$FILEM CMPNEW=ALL,
$$FILEM DSNNEW=CHABERT.TRADER.SOURCE(MYTRADMI)
IBM File Manager for z/OS
* * * * Record comparison with one-to-one synchronization
* * New data set: CHABERT.TRADER.SOURCE(MYTRADMI)
* * Old data set: CHABERT.TRADER.SOURCE(MYTRADMD)


Comparison summary:

             Old data set records processed:        841
             New data set records processed:        887
                    Matching records found:        166
              Old data set records deleted:        675
             New data set records inserted:        721
                   Total changed records:        721
                          Paired changes:        675
    Non-paired old data set record deletions:          0
   Non-paired new data set record insertions:         46
                         Synchronization: One-to-one
                         Comparison type: Record
                            Listing type: Summary

******************************* BOTTOM OF DATA ********************************
```

24.19 Dynamic template

When a COBOL or PL/I layout is not available, you can use the unformatted template. It allows you to define a record structure you can use to map the data. The following steps describe how to use the template to format your data:

1. Select option **3** (None) on the browse or edit panel. You are presented with unformatted data, as shown in Figure 24-74.

```
 Process   Options   Help
FMNPQED          DENARDI.FMN.DATA(AAA4FILE)                      Rec 0 of 104
Command ===> ____________________________________________________ Scroll CSR
       Col 1         Insert length 80                            Format CHAR
       ----+----1----+----2----+----3----+----4----+----5----+----6----+----7--
000000 ****  Top of data  ****
000001 BAP01..01 ANNA  ....<WAGHORN   ..&.!E01ADMIN   04.11.2004M....<04.11.200
000002 bAP02W@07 BILL  ....<SOPER     ....(B01ARCHITEC25.04.2004F....(25.04.200
000003 bAP03. 07 Bonnet....!MCCORMACK .. .!C01DEVELOPE13.07.2003F....!13.08.200
000004 bAP04  SD2Serena... .Ciproni   ....(A00DEVELOPE19.11.1997F  ..(19.11.199
000005 bAP05..A98KEITH ....<STEWART     ..<B01DEVELOPE09.09.1980F....<09.09.198
000006 bAP06.8B44john  . ..<RUSHTON      .<A00DEVELOPE01.01.2003M. ..<01.01.200
000007 bAP07  S32MIKE  ....<MORIARTY     ..<E01DEVELOPE11.04.2004M....<11.04.200
000008 bAP08  A33PETER ....<VAN DYKE  .. .!B01DEVELOPE25.04.2004M....<25.04.200
000009 bAP09  A56ROD   .....TURNER     .. .(E01DEVELOPE13.08.2003F....(13.08.200
000010 bAP10  23 Ronald.....LING         .<C01DEVELOPE19.11.1997F....<19.11.199
000011 AAP.ZANNA WAGHORN     ADMIN       221 YALE RD          SOUTH PERTH
000012 AAP. BILL SOPER       ARCHITECT   498 OAK GROVE RD     SPRINGFIELD
000013 AAP.@BOB MCCORMACK    DEVELOPER   12  CLAYTON ST       ARMADALE
000014 AAP..JOHN LEVRINGTON  DEVELOPER   22  MONTROSE ST      BOONE
000015 bAP11  E56Tyrone....<DALAIS       ..<C01MANAGER 09.09.1980M....<09.09.198
000016 bIT12  R66ANDREA....<BRENNER      ..<A00MANAGER 01.01.2003F....<01.01.200
000017 bIT13  R65Annon ....<PEREZ         .<B01OPERATOR04.05.2003M....<04.05.200
000018 bIT14  Y44Serena....<SMITHA     ...  .C01OPERATOR05.12.2002M.....05.12.200
000019 bIT15  Y44Serena....<SMITHAM       < C01OPERATOR05.12.2002M...< 05.12.200
000020 bIT96  Y44Serena....<SMITH         < C01OPERATOR05.12.2002M...< 05.12.200
000021 bIT17  Y44Serena....<Ciproni    .....C01OPERATOR05.12.2002M.....05.12.200
000022 bIT18  Y44Serena....<SMITH      ...f.C01OPERATOR05.12.2002M... .05.12.200
 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 24-74 Result of an unformatted browse

2. Select option **4** (Create dynamic) on the Browse or Edit panel. You are presented with Figure 24-75 where you can specify the characteristics of the record layout. To add field names, use PF11 from this panel.

```
 Process   Options   Help
FMNPDYNT ger                 Dynamic Template                        Line 1 of 11
Command ===>                                                          Scroll CSR

Cmd Con (  Start  Length Type  Op  Value                                        )
    <-> -  <---->  <---->  <>  <-> <---+----1----+----2----+----3----+----4-- -
*** ****  Top of data   ****
___      _      1       1  AN  ___ ___________________________________________ _
___ AND  _      2       2  AN  ___ ___________________________________________ _
___ AND  _      4       2  AN  ___ ___________________________________________ _
___ AND  _      6       1  BI  ___ ___________________________________________ _
___ AND  _      8       3  AN  ___ ___________________________________________ _
___ AND  _     11       6  AN  ___ ___________________________________________ _
___ AND  _     17       5  PD  ___ ___________________________________________ _
___ AND  _     22      10  AN  ___ ___________________________________________ _
___ AND  _     32       5  AN  ___ ___________________________________________ _
___ AND  _     32       5  AN  ___ ___________________________________________ _
___ AND  _     37       3  AN  ___ ___________________________________________ _
*** ****  End of data   ****




 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=RunTemp
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 24-75 Dynamic template generation

3. Execute RunTemp (F6) and Figure 24-76 is displayed. It presents the sample data sorted and presented with this dynamic template, including the addition of individual field names. Note that field numbers are also added.

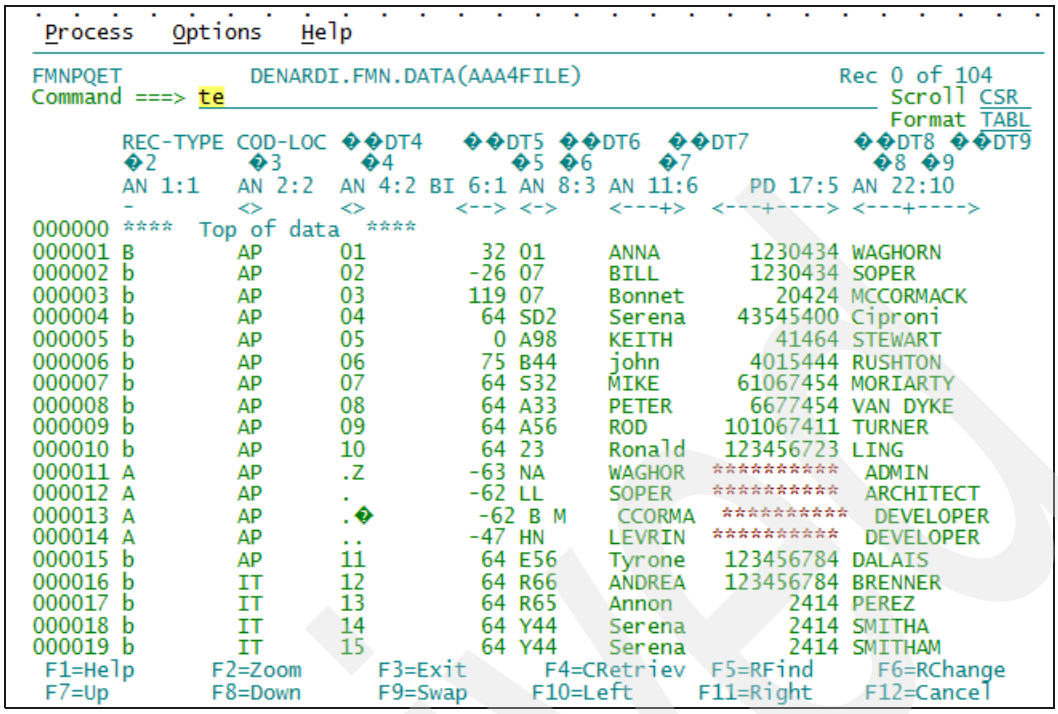


Figure 24-76 Dynamic template usage (part 1 of 2)

Press the F11 key to move to the right of the screen and edit the data as shown in Figure 24-77.

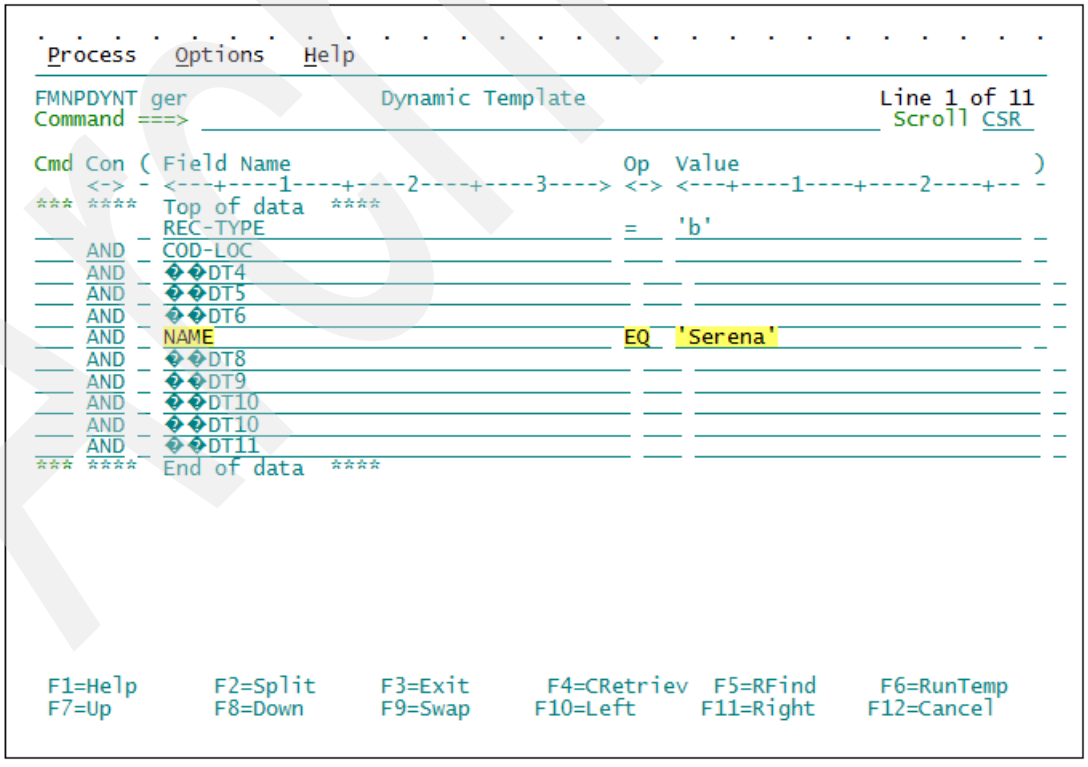


Figure 24-77 Dynamic template usage after typing TE and option 1 on Command line (part 2 of 2)

4. You can edit this template to add a specific condition, or type a 1 on the command line, as shown in Figure 24-78, and press Enter to display again the dynamic template panel, as shown in the previous panels:

   The General information panel allows you to:

   – Select fields for display or printing purposes.
   – Edit field formatting and create attributes.
   – Specify expressions for record identification and selection.
   – Specify and generate a segmented Record template from the data structure
   – Modify the offset value for a layout.
   – View and print the 01 layout.

   The Field Selection/Edit panel is displayed when a template is being edited for purposes other than reformatting (that is, not the TO panel on a COPY function and *not* the mapping panel of the template workbench). If the copybook or template does *not* contain multiple record types, then this is the first panel displayed during an edit template session. If the template contains multiple record types, then this panel follows a record type selection from the record type selection panel.

```
 Process   Options   Help
------------------------------------------------------------------------------
FMNPDYNT ger                Dynamic Template                  Line 1 of 11
Command ===> _____________________________________________________ Scroll CSR

Cmd Con (  Start  Length Type  Op  Value                                    )
    <-> - <---->  <----> <>    <-> <---+----1----+----2----+----3----+----4-- -
*** **** Top of data  ****
___     _     1       1  AN    =   'b'                                      _
___ AND _     2       2  AN        ________________________________________ _
___ AND _     4       2  AN        ________________________________________ _
___ AND _     6       1  BI        ________________________________________ _
___ AND _     8       3  AN        ________________________________________ _
___ AND _    11       6  AN    eq  'Serena'                                 _
___ AND _    17       5  PD        ________________________________________ _
___ AND _    22      10  AN        ________________________________________ _
___ AND _    32       5  AN        ________________________________________ _
___ AND _    32       5  AN        ________________________________________ _
___ AND _    37       3  AN        ________________________________________ _
*** **** End of data  ****


 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=RunTemp
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 24-78 Adding a condition to a dynamic template

5. You can alternatively specify the selection criteria by field name as well getting the same results shown in Figure 24-79. Use the F6 key (RunTemp) to execute.

```
 Process   Options   Help
FMNPQET          DENARDI.FMN.DATA(AAA4FILE)                    Rec 0 of 104
Command ===> sha all off                                       Scroll CSR
SHAD                                                           Format TABL
       REC-TYPE COD-LOC ??DT4   ??DT5 ??DT6  NAME          ??DT8 ??DT9
       ?2       ?3      ?4       ?5 ?6      ?7              ?8 ?9
       AN 1:1   AN 2:2  AN 4:2 BI 6:1 AN 8:3 AN 11:6    PD 17:5 AN 22:10
       -        <>      <>       <--> <->    <---+>  <---+----> <---+---->
000000 ****  Top of data  ****
000004 b        AP      04         64 SD2    Serena    43545400 Ciproni
000018 b        IT      14         64 Y44    Serena        2414 SMITHA
000019 b        IT      15         64 Y44    Serena        2414 SMITHAM
000020 b        IT      96         64 Y44    Serena        2414 SMITH
000021 b        IT      17         64 Y44    Serena        2414 Ciproni
000022 b        IT      18         64 Y44    Serena        2414 SMITH
000105 ****  End of data  ****

 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 24-79 Result of template with selection

24.20 Generating XML data as output

FM/Base uses templates to provide different formatted data, and enabling you to view, edit and manipulate the output data according your requirements. A File Manager template is a collection of information and definition to “map” the original data structure that you can use to generate different data (CSV in FM/DB2) as output.

A new function has been added to generate well-formed XML from an input data set (file) and a template. It is a new functionality provided with FM Version 7.1. The function has an interactive ISPF-based interface and a batch interface.

This is a feature of the Copy Utility. You invoke this function by selecting option **3** (Utilities) from the main panel, then selecting option **3** (Copy).

The following steps describe how to use the template to generate your XML data:

1. Select option **3.3** from the main FM menu. File Manager displays the Copy Utility panel (FMNPDSCI). You are presented with unformatted data. Type the *Data set/path name*, and from the *From Copybook or Template*, supply the data set details for your INPUT File, as well as the other information shown in Figure 24-80.

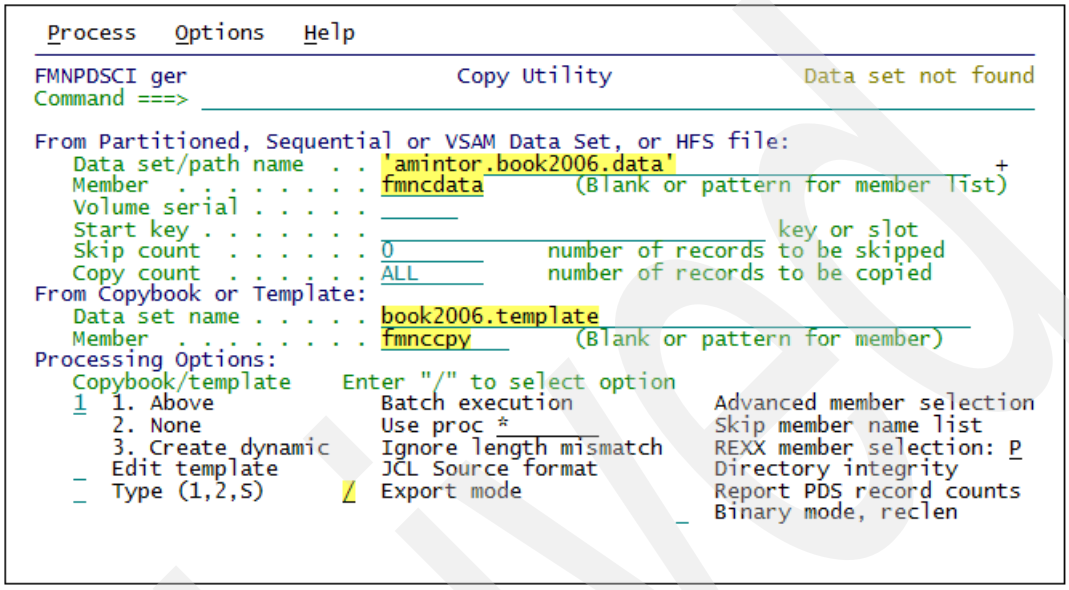


Figure 24-80 XML Generation Invoked from FM 3.3 Copy (DSC) Utility

2. Enter the details for *From copybook or template*. Use a combination of the Data set name and Member entry fields to specify the copybook or template that describes the data in the *Data set name* and *Member* fields. You can select data for copying at either record level or field level:
   a. Select the *Edit template* processing options for record-level selection.You can set the record identification and record selection criteria in your Edit template.
   b. For field-level selection, specify the fields attributes and field mapping in your template.
3. Select **Export mode** to indicate that you want output in an external format.
4. Select Processing Option **1** (above) or **3** for *Create dynamic*, and press Enter.
   a. If you selected option **1**, File Manager displays an extended version of the Copy To panel. This form of the panel allows you to specify the additional options.
   b. If you selected option **3**, then you must create the dynamic template. Once you have done so, File Manager displays the next extended *Copy To* panel (FMNPDSCX) as shown in Figure 24-81.

```
 Process   Options   Help
FMNPDSCX  AMINTOR.BOOK2006.DATA(FMNCDATA)
Command ===> _____________________________________________________________
                                                                More:     +
To Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . 'amintor.book2006.vardata'                    +
   Member name (or mask) . xmltest1
   Volume serial . . . . . ______

Processing Options:
  Disposition           Execution "/" options      Non-print. characters
  1  1. Old or Reuse    /  Replace members         2  1. Asis
     2. Mod               Binary mode, reclen         2. Hex
  ISPF Packing          _  Stats Off                   3. Replace with .____
  1  1. Asis              Include fillers             4. Skip
     2. Pack              Include redefines       Special characters
     3. Unpack            Convert to Unicode      1  1. Escape
     4. None              Split output line           2. CData
     5. Skip                                          3. Hex
  Format                                              4. Replace with _____
  1  1. XML               Indent step 1           Invalid data
                                                  1  1. Hex
                                                     2. Replace with *_____
```

Figure 24-81 New XML output options

Optionally, customize the generation of output. For XML, you can affect the generation and readability by specifying how to represent non-printable characters and invalid data, whether to include fillers and redefines, how to indent when nesting successive levels of XML tag nesting, and so on.

5. Press Enter. File Manager generates the selected data from the selected *Processing Options* and writes it, in XML format, to the Partitioned, Sequential, or VSAM data set or HSF file as shown in Figure 24-82.

```
 Process   Options   Help
FMNPDSCI ger                    Copy Utility
Command ===> ____________________________________________________________
762 record(s) formatted from 40 input record(s)
From Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . 'AMINTOR.BOOK2006.DATA'                       +
   Member  . . . . . FMNCDATA       (Blank or pattern for member list)
   Volume serial . . . . . ______
   Start key . . . . . _____________________________ key or slot
   Skip count  . . . . . 0_______ number of records to be skipped
   Copy count  . . . . . ALL_____ number of records to be copied
From Copybook or Template:
   Data set name . . . . . BOOK2006.TEMPLATE_____________________________
   Member  . . . . . FMNCCPY___     (Blank or pattern for member)
Processing Options:
   Copybook/template    Enter "/" to select option
   1  1. Above             Batch execution          Advanced member selection
      2. None              Use proc *_______        Skip member name list
      3. Create dynamic    Ignore length mismatch   REXX member selection: P
      Edit template        JCL Source format        Directory integrity
   _  Type (1,2,S)      /  Export mode              Report PDS record counts
                                                 _  Binary mode, reclen
```

Figure 24-82 The panel shows that has been created 762 XML formatted records

For this example, we want to generate the XML data format in batch. We used PDS Files as INPUT/OUTPUT datasets, and the sample results (JCL and SYSOUT) are shown Example 24-23.

Example 24-23 FIND procedure

```
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (A),
000002 //             AMINTOR,MSGCLASS=A,
000003 //             NOTIFY=AMINTOR,CLASS=A,
000004 //             MSGLEVEL=(1,1)
000005 //*
000006 //FILEMGR  EXEC PGM=FMNMAIN
000007 //*TEPLIB  DD DSN=FMN.SFMNMOD1,DISP=SHR
000008 //*        DD DSN=IGY.SIGYCOMP,DISP=SHR
000009 //*FMNCOB  DD DUMMY     Uncomment to force use of FM COBOL Compiler
000010 //SYSPRINT DD SYSOUT=*
000011 //FMNTSPRT DD SYSOUT=*
000012 //SYSTERM  DD SYSOUT=*
000013 //SYSIN    DD *
000014 $$FILEM SHOWSORT
000015 $$FILEM DSC DSNIN=AMINTOR.BOOK2006.DATA,
000016 $$FILEM MEMBER=FMNCDATA,
000017 $$FILEM PACK=ASIS,
000018 $$FILEM POSITION=0,
000019 $$FILEM TCIN=AMINTOR.BOOK2006.TEMPLATE(FMNCCPY),
000020 $$FILEM REPLACE=YES,
000021 $$FILEM DISP=OLD,
000022 $$FILEM FORMAT=XML,
000023 $$FILEM FILLERS=NO,
000024 $$FILEM REDEFINES=NO,
000025 $$FILEM UNICODE=NO,
000026 $$FILEM LINESPLIT=NO,
000027 $$FILEM NPRTCHAR=HEX,
000028 $$FILEM SPECCHAR=ESCAPE,
000029 $$FILEM INVDATA=HEX,
000030 $$FILEM INDENT=1,
000031 $$FILEM MEMOUT=XMLTEST1,
000032 $$FILEM DSNOUT=AMINTOR.BOOK2006.VARDATA
****** ***************************** Bottom of Data ****************************

 SDSF OUTPUT DISPLAY AMINTORB JOB03917  DSID   102 LINE 22      COLS 02- 81
 COMMAND INPUT ===>                                            SCROLL ===> CSR
$$FILEM REPLACE=YES,
$$FILEM DISP=OLD,
$$FILEM FORMAT=XML,
$$FILEM FILLERS=NO,
$$FILEM REDEFINES=NO,
$$FILEM UNICODE=NO,
$$FILEM LINESPLIT=NO,
$$FILEM NPRTCHAR=HEX,
$$FILEM SPECCHAR=ESCAPE,
$$FILEM INVDATA=HEX,
$$FILEM INDENT=1,
$$FILEM MEMOUT=XMLTEST1,
$$FILEM DSNOUT=AMINTOR.BOOK2006.VARDATA
IBM File Manager for z/OS
                                                  Member Copy Report
Member         Newname    Alias      Status                          Recs in    Recs out
FMNCDATA       XMLTEST1              Replaced                             40         762
0 member(s) copied; 1 replaced; 0 not copied; 0 in error
****************************** BOTTOM OF DATA ***************************************
```

25

File Manager/DB2

In this chapter we describe some of the capabilities of File Manager for DB2 Data and show you how to use them.

25.1 Overview of FM/DB2

File Manager for DB2 Data (FM/DB2) is a powerful set of utility functions for editing, browsing, printing, copying, and maintaining DB2 data. It also provides utilities for listing DB2 objects, managing DB2 privileges, generating JCL to run DB2 standalone utilities, exporting and importing DB2 tables to or from QSAM or VSAM data sets, creating data to populate DB2 tables, and prototyping SQL SELECT statements.

25.2 Primary option menu

On the Primary Options Menu, Figure 25-1, you can choose to which DB2 SSID to connect. FM/DB2 establishes a connection with a chosen DB2 system. If you do not specify a value, FM/DB2 presents you with the DB2 Subsystem Selection panel. Place a slash (/) or (*) against the required DB2 system in the list displayed.

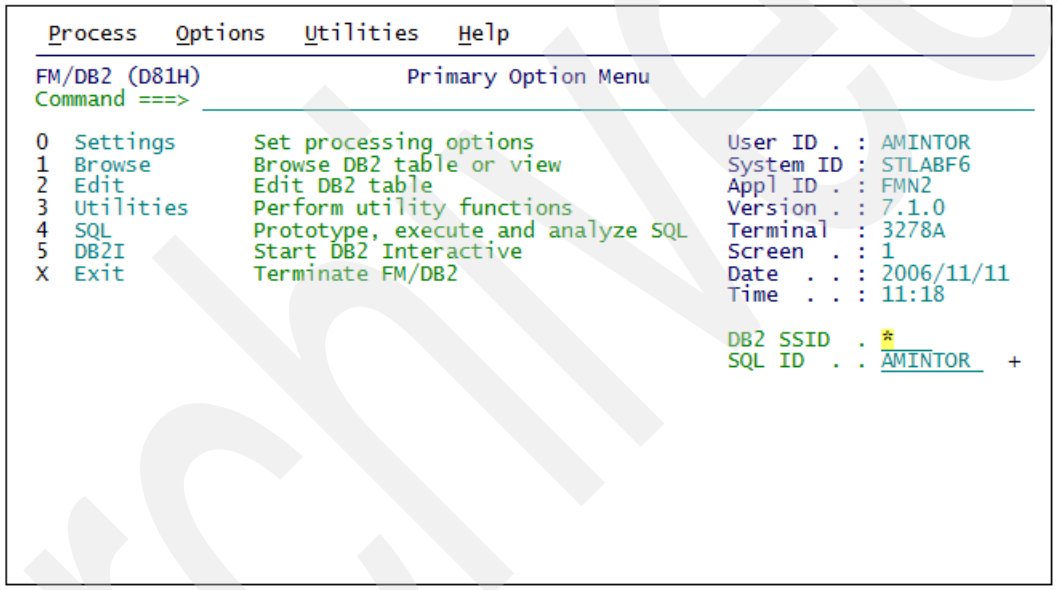


Figure 25-1 Primary Options Menu

The following DB2 Subsystem Selection Panel is displayed *only* if you type * on the field DB2 SSID, as shown in Figure 25-2.

```
 Process   Help
-------------------------------------------------------------------------------
FM/DB2 (D81H)              DB2 Subsystem Selection                 Row 1 of 16
Command ===> _____________________________________________________ Scroll PAGE

Sel SSID Status    Description                                     Prefix
-------------------------------------------------------------------------------
    DB1E ACTIVE    DB2 VERSION 7 LPAR F6                           -DB1E
    D72F ACTIVE    DB2 VERSION 7 LPAR F6                           -D72F
 s  D81H ACTIVE    DB2 VERSION 8 LPAR F6                           -D81H
 _  D82G ACTIVE                                                    -D82G
 _  D91F ACTIVE    DB2 VERSION 9 LPAR F6                           -D91F
 _  DB0E GROUP
    D60F INACTIVE
 _  D70F GROUP     DB2 V7 DATA SHARING GROUP
 _  D80G GROUP
 _  D80H GROUP
 _  D90F GROUP
    D80G INACTIVE
    D70F INACTIVE  DB2 V7 DATA SHARING GROUP
    D61F INACTIVE                                                  -D61F
    D71F INACTIVE                                                  -D71F
    D81G INACTIVE                                                  -D81G
    ****  End of data  ****
```

Figure 25-2 Panel to select and cross between the DB2 SSID

The DB2 Subsystem Selection panel shown here displays some inactive DB2 subsystems. Your system administrator might have customized the list so that, for example, only active DB2 subsystems appear. For more information, see your system administrator or refer to the *File Manager User Guide and Reference,* SC19-1037-00. You can use this panel to select the DB2 subsystem to which you want FM/DB2 to connect.

25.3 Browsing or editing a table

FM/DB2 uses *templates* to provide a formatted view of your data, enabling you to view, edit, and manipulate data according to the columns and data types in the table you are working with.

A FM/DB2 template is a collection of information that you can use to select and format tables and columns in a DB2 object. If you use a FM/DB2 function that interfaces with non-DB2 data, the corresponding template describes the records and fields in the data set.

25.3.1 Editing a table with template generated from the table

You can edit a DB2 table by selecting option **2** in the Primary Option Menu and pressing Enter. The DB2 Edit panel is now displayed, as shown in Figure 25-3.

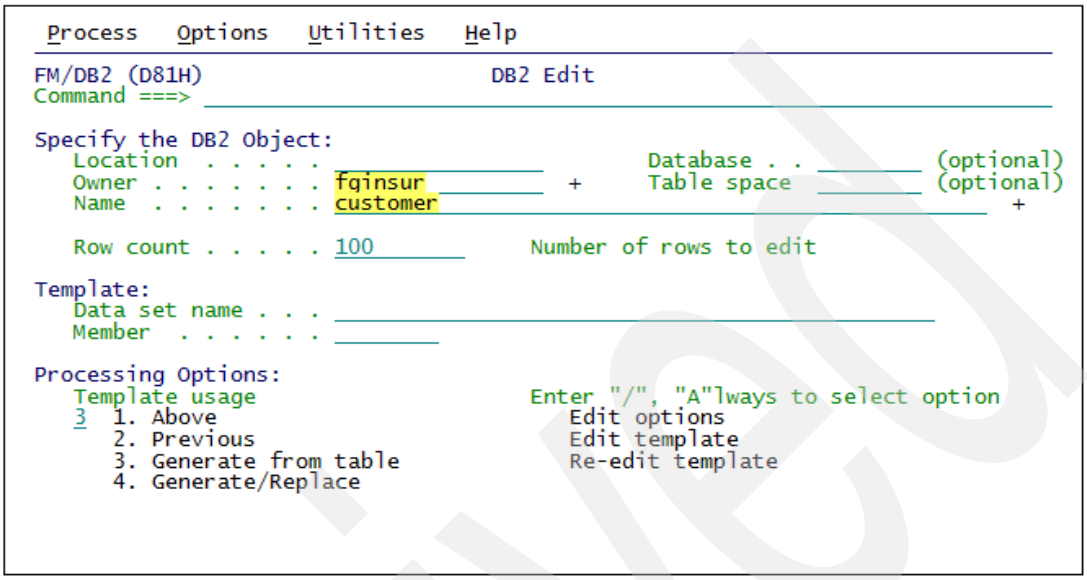


Figure 25-3 This is the Edit Panel to display DB2 objects

To edit the object, enter the owner of the DB2 object and the name of the object. Wildcards are allowed to display a selection list from which the object can be selected.

- Owner FGINSUR
- Name CUSTOMER
- Rowcount 100

  FM/DB2 only reads the first 100 rows into memory. If you specify 0, then all rows are read into memory.
- Template usage

  Select option **3**, which means that FM/DB2 generates the template.

After pressing Enter, the Table Edit Panel is displayed as shown in Figure 25-4. FM/DB presents the formatted data. On this panel, the following information is displayed:

- Upon entering the Entry Panel for the first time, the message `xx rows have been fetched` is displayed when using templates.
- Data is displayed in TABL format.
- In the header:
  - Column names for the table
  - Field reference number for the column
  - Data type
- To the left is the Prefix area where Prefix commands can be entered.

```
 Process    Options    Utilities    Help
FM/DB2 (D81H)                      Table Edit                          0 of 100
Command ===> ____________________________________________________ Scroll PAGE
TABLE FGINSUR.CUSTOMER                                          Format TABL
       CUSTOMERNUMBER FIRSTNAME  LASTNAME            DATEOFBIRTH HOUSENAME
                    ¢1 ¢2         ¢3                   ¢4          ¢5          +
              INTEGER CH(10)     CHARACTER(20)        DATE        CHARACTER(20
          PU--+----1> <---+----> <---+----1----+----> <---+----> <---+----1--
000000 ****  Top of data  ****
000001              37 BPLP       SZWQA                1960-12-15  LRHU WHMED U
000002              33 HOWM       EADOPCYB             1943-06-15  LRHU WHMED U
000003              39 SSME       KNTLL                1983-05-23  NQVV CCCJHPY
000004              38 SSME       LBWTZFXC             1974-09-06  LRHU WHMED U
000005              31 EMIMQ      SZWQA                1957-08-25  YXDI NHQV
000006              36 WMET       EADOPCYB             1959-10-06  WMV APZYHJ
000007              32 WMET       KNTLL                1959-10-15  NQVV CCCJHPY
000008              35 ESKWX      YWXHXUQE             1952-02-09  YXDI NHQV
000009              43 HOX        EADOPCYB             1957-09-19  WMV APZYHJ
000010              44 WYTWN      LBWTZFXC             1960-01-15  NQVV CCCJHPY
000011              49 SSME       FFFLA                1974-09-17  LRHU WHMED U
000012              48 HOWM       BXUZO                1959-08-15  NQVV CCCJHPY
000013              46 EMIMQ      WMEHVL               1945-04-28  YXDI NHQV
000014              40 BPLP       LBWTZFXC             1942-12-20  WMV APZYHJ
```

Figure 25-4 FM/DB2 Table Edit pane

Type directly over the data that you want to change if you want to modify the table's rows. As shown above, the LASTNAME value has been changed and the corresponding row begins to highlight in yellow until the *DB2 commit* is performed.

After finishing the changes, we press F3. The DB2 Edit panels are displayed, and in the upper left corner we see that commit issued, as shown in Figure 25-5.

```
 Process    Options    Utilities    Help
FM/DB2 (D81H)                       DB2 Edit                    Commit issued
Command ===> ____________________________________________________________

Specify the DB2 Object:
   Location  . . . . . ____________          Database . . ________ (optional)
   Owner . . . . . . . FGINSUR          +     Table space  ________ (optional)
   Name  . . . . . . . CUSTOMER                                              +

   Row count . . . . . 100          Number of rows to edit

Template:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                     Enter "/", "A"lways to select option
   3  1. Above                           Edit options
      2. Previous                        Edit template
      3. Generate from table             Re-edit template
      4. Generate/Replace
```

Figure 25-5 Commit issued

25.3.2 Editing a table with its own generated template

Sometimes you might want to view or edit some of the columns in the table. Create a template that only shows the following columns:

- EMPNO
- LASTNAME
- WORKDEPT
- HIREDATE

To edit the template enter a slash (/) in the field Edit Template, as shown in Figure 25-6, and press Enter.

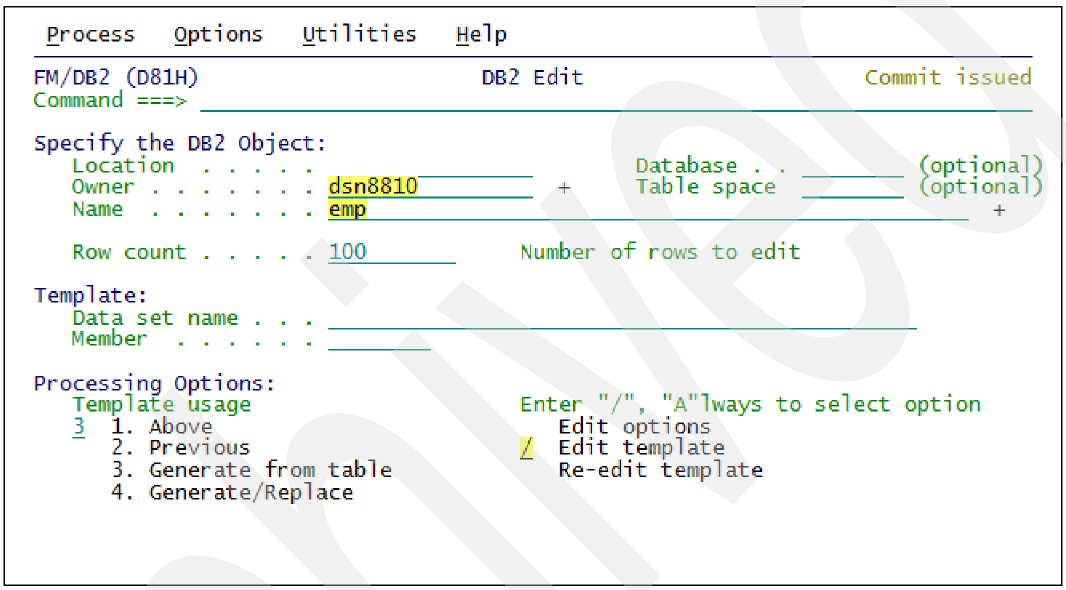


Figure 25-6 Select Edit Template

After selecting Edit Template, the Column Selection/Edit panel is displayed, as shown in Figure 25-7. Above the line, `Top of data`, there is a line with column headers. Some of these header functions are:

- **Cmd:** This is where you can enter prefix commands:

  S Select/deselect a column from processing.
  H Set hold status for a column.

  These commands can also be block commands such as SS, HH, and S4. If the SHC column shows an S for a column, the **S** command deselects the column from processing.

- **Seq:** This is where you can rearrange the order in which the columns are displayed.
- **SHC:** (Select/Hold/Criteria) shows the column status:
  - The S column is selected for processing.
  - The H column has been selected to be held on the display.
  - C (create data) has been defined for the column.

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)              Column Selection/Edit                Line 1 of 14
Command ===> _____________________________________________________ Scroll PAGE
TABLE DSN8810.EMP
------- Row Selection Criteria ---- (Use SQL/PF4 for full screen edit) -------
1 Sel: _____________________________________________________________________ +

Cmd Seq SHC CL? Column name          Data type(length)  Null Default  Order A/D
              ****  Top of data  ****
s*  ___ S     1 EMPNO               CHARACTER(6)             None           _
___ ___ S     2 FIRSTNME            VARCHAR(12)              None           _
___ ___ S     3 MIDINIT             CHARACTER(1)             None           _
___ ___ S     4 LASTNAME            VARCHAR(15)              None           _
___ ___ S     5 WORKDEPT            CHARACTER(3)       Y     Null           _
___ ___ S     6 PHONENO             CHARACTER(4)       Y     Null           _
___ ___ S     7 HIREDATE            DATE               Y     Null           _
___ ___ S     8 JOB                 CHARACTER(8)       Y     Null           _
___ ___ S     9 EDLEVEL             SMALLINT           Y     Null           _
___ ___ S    10 SEX                 CHARACTER(1)       Y     Null           _
___ ___ S    11 BIRTHDATE           DATE               Y     Null           _
___ ___ S    12 SALARY              DECIMAL(9,2)       Y     Null           _
___ ___ S    13 BONUS               DECIMAL(9,2)       Y     Null           _
___ ___ S    14 COMM                DECIMAL(9,2)       Y     Null           _
```

Figure 25-7 Column Selection/Edit panel

In the example above, all columns have been selected for processing, but the only columns we want to process are:

- EMPNO
- LASTNAME
- WORKDEPT
- HIREDATE

Deselect all columns using the prefix command S*. After the deselect command, all of the **S** entries under the SHC header have now been removed, as shown in Figure 25-8.

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)              Column Selection/Edit                Line 1 of 14
Command ===> _____________________________________________________ Scroll PAGE
TABLE DSN8810.EMP
------- Row Selection Criteria ---- (Use SQL/PF4 for full screen edit) -------
1 Sel: _____________________________________________________________________ +

Cmd Seq SHC CL? Column name          Data type(length)  Null Default  Order A/D
              ****  Top of data  ****
___ ___       1 EMPNO               CHARACTER(6)             None           _
___ ___       2 FIRSTNME            VARCHAR(12)              None           _
___ ___       3 MIDINIT             CHARACTER(1)             None           _
___ ___       4 LASTNAME            VARCHAR(15)              None           _
___ ___       5 WORKDEPT            CHARACTER(3)       Y     Null           _
___ ___       6 PHONENO             CHARACTER(4)       Y     Null           _
___ ___       7 HIREDATE            DATE               Y     Null           _
___ ___       8 JOB                 CHARACTER(8)       Y     Null           _
___ ___       9 EDLEVEL             SMALLINT           Y     Null           _
___ ___      10 SEX                 CHARACTER(1)       Y     Null           _
___ ___      11 BIRTHDATE           DATE               Y     Null           _
___ ___      12 SALARY              DECIMAL(9,2)       Y     Null           _
___ ___      13 BONUS               DECIMAL(9,2)       Y     Null           _
___ ___      14 COMM                DECIMAL(9,2)       Y     Null           _
```

Figure 25-8 Deselecting all columns

Now select only those columns you want to process by entering the prefix command **S**, as shown in Figure 25-9.

```
 Process    Options    Utilities    Help
FM/DB2 (D81H)                Column Selection/Edit                 Line 1 of 14
Command ===>                                                       Scroll PAGE
TABLE DSN8810.EMP
------- Row Selection Criteria ---- (Use SQL/PF4 for full screen edit) -------
1 Sel:                                                                      +

Cmd Seq SHC CL# Column name          Data type(length)  Null Default  Order A/D
              ****  Top of data  ****
s   ___         1 EMPNO              CHARACTER(6)            None           _
___ ___         2 FIRSTNME           VARCHAR(12)             None           _
___ ___         3 MIDINIT            CHARACTER(1)            None           _
s   ___         4 LASTNAME           VARCHAR(15)             None           _
s   ___         5 WORKDEPT           CHARACTER(3)       Y    Null           _
___ ___         6 PHONENO            CHARACTER(4)       Y    Null           _
s   ___         7 HIREDATE           DATE               Y    Null           _
___ ___         8 JOB                CHARACTER(8)       Y    Null           _
___ ___         9 EDLEVEL            SMALLINT           Y    Null           _
___ ___        10 SEX                CHARACTER(1)       Y    Null           _
___ ___        11 BIRTHDATE          DATE               Y    Null           _
___ ___        12 SALARY             DECIMAL(9,2)       Y    Null           _
___ ___        13 BONUS              DECIMAL(9,2)       Y    Null           _
___ ___        14 COMM               DECIMAL(9,2)       Y    Null           _
```

Figure 25-9 Selection of columns to process

You can then reorder the columns by entering sequence numbers. After reordering the sequence of the columns, the panel is displayed, as shown in Figure 25-10.

```
 Process    Options    Utilities    Help
FM/DB2 (D81H)                Column Selection/Edit                 Line 1 of 14
Command ===>                                                       Scroll PAGE
TABLE DSN8810.EMP
------- Row Selection Criteria ---- (Use SQL/PF4 for full screen edit) -------
1 Sel:                                                                      +

Cmd Seq SHC CL# Column name          Data type(length)  Null Default  Order A/D
              ****  Top of data  ****
    1   S       1 EMPNO              CHARACTER(6)            None           _
___ ___         2 FIRSTNME           VARCHAR(12)             None           _
___ ___         3 MIDINIT            CHARACTER(1)            None           _
___ 3   S       4 LASTNAME           VARCHAR(15)             None           _
___ 4   S       5 WORKDEPT           CHARACTER(3)       Y    Null           _
___ ___         6 PHONENO            CHARACTER(4)       Y    Null           _
___ 2   S       7 HIREDATE           DATE               Y    Null           _
___ ___         8 JOB                CHARACTER(8)       Y    Null           _
___ ___         9 EDLEVEL            SMALLINT           Y    Null           _
___ ___        10 SEX                CHARACTER(1)       Y    Null           _
___ ___        11 BIRTHDATE          DATE               Y    Null           _
___ ___        12 SALARY             DECIMAL(9,2)       Y    Null           _
___ ___        13 BONUS              DECIMAL(9,2)       Y    Null           _
___ ___        14 COMM               DECIMAL(9,2)       Y    Null           _
```

Figure 25-10 Reorder the sequence of the columns.

When you have specified all definitions for the view, we press F3 and the Template Save panel is displayed, as shown in Figure 25-11. Here you have the following options:

- Discard the changes.
- Use the changes without saving.
- Save the changes.

Save the template as TEMPL1.

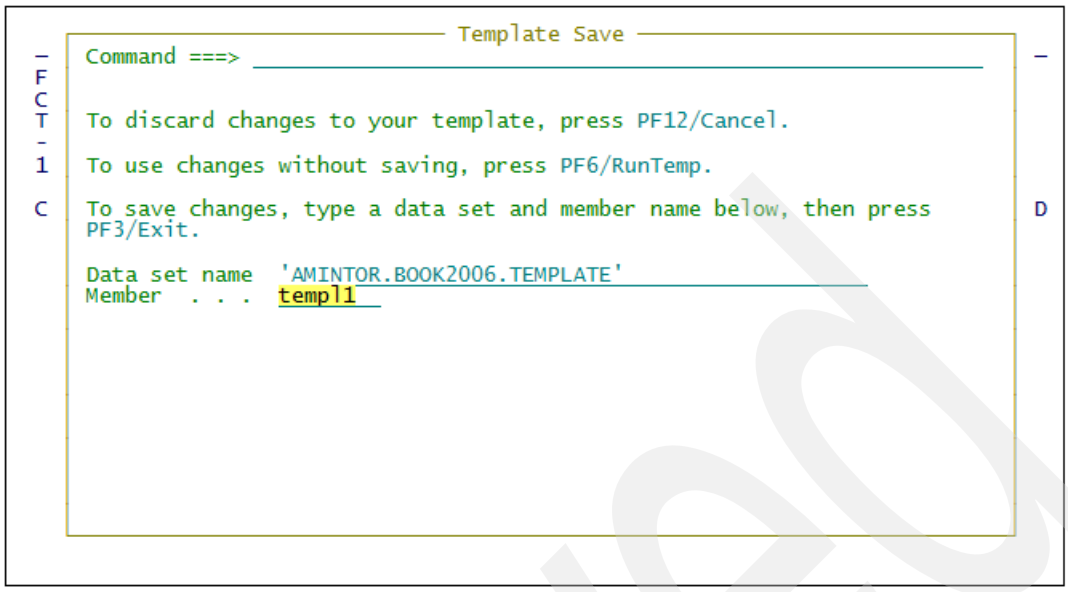


Figure 25-11 Template Save panel.

After saving the template, the Table Edit panel is displayed, as shown in Figure 25-12, and the columns that were selected for the template (`100 rows fetched`) are returned.

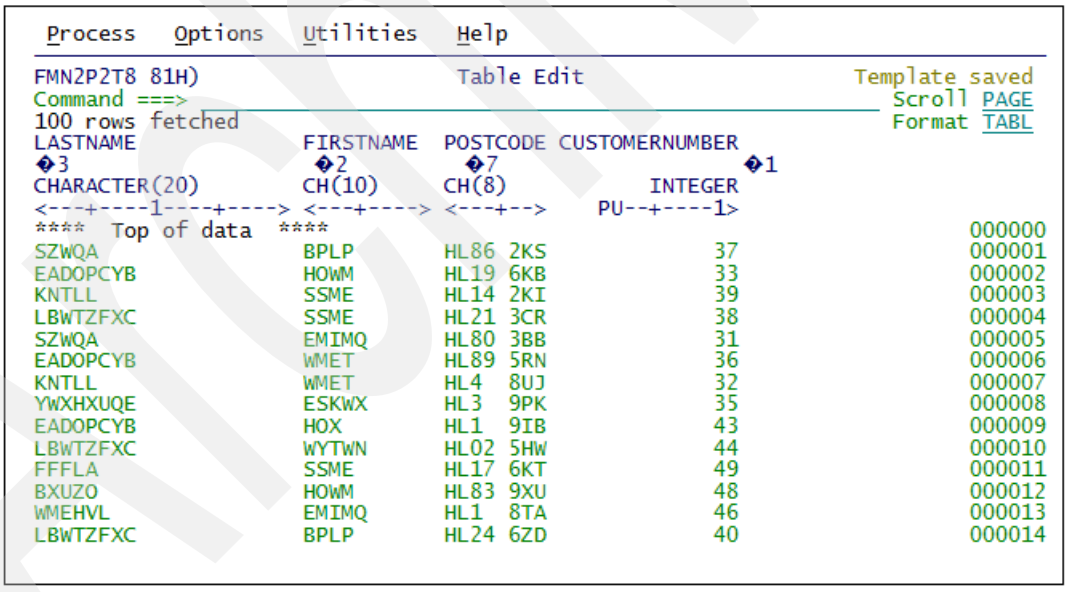


Figure 25-12 Table Edit panel

After exiting the Table Edit panel by pressing PF3, the DB2 Edit panel is displayed, as shown in Figure 25-13. You can see the fields that the template has now updated. If you want to use this template in the future, you can enter option **1** in the field for Template usage.

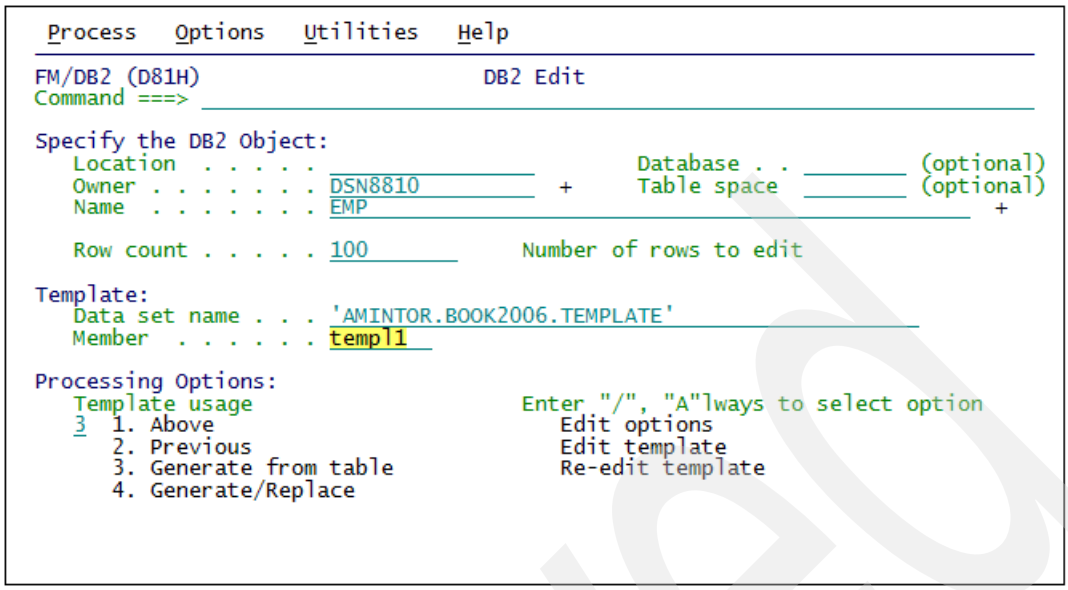


Figure 25-13 Fields for the template are now updated

25.4 Displaying table in different formats

You can display a table in two formats:

- Table format, or TABL format
- Single format, or SNGL format

The data can also be displayed in hex format.

25.4.1 Table format

To display the data in TABL format, type **T** in the Format field, or use the primary command **FT** to display the TABL format as shown in Figure 25-14.

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)                    Table Edit              FM/DB2 template loaded
Command ===>                                                    Scroll PAGE
100 rows fetched                                                Format sABL
       CUSTOMERNUMBER FIRSTNAME  LASTNAME             DATEOFBIRTH HOUSENAME
                   ¢1 ¢2          ¢3                    ¢4          ¢5          +
              INTEGER CH(10)     CHARACTER(20)         DATE        CHARACTER(20
          PU--+----1> <---+----> <---+----1----+----> <---+----> <---+----1--
000000 ****  Top of data  ****
000001              378 ESKWX      EADOPCYB              1949-03-06  YXDI NHQV
000002              371 HOX        OJKZIR                1969-06-13  YXDI NHQV
000003              376 BBXOK      FFFLA                 1975-06-24  YXDI NHQV
000004              372 BPLP       HOWSLQNT              1979-05-28  WMV APZYHJ
000005              370 SZCFJNO    BXUZO                 1956-01-08  NQVV CCCJHPY
000006              375 BBXOK      SZWQA                 1953-03-12  YXDI NHQV
000007              377 ESKWX      HOWSLQNT              1980-03-26  NQVV CCCJHPY
000008              373 HOWM       HOWSLQNT              1966-08-02  NQVV CCCJHPY
000009              374 EMIMQ      BXUZO                 1963-03-08  YXDI NHQV
000010              379 WMET       WMEHVL                1944-05-28  WMV APZYHJ
000011              337 SSME       EADOPCYB              1951-03-09  NQVV CCCJHPY
000012              333 HOWM       SZWQA                 1952-07-06  YXDI NHQV
000013              334 EMIMQ      HOWSLQNT              1940-02-02  LRHU WHMED U
000014              339 BPLP       YWXHXUQE              1972-04-25  YXDI NHQV
```

Figure 25-14 TABL format

25.4.2 Single format

To display the data in SNGL format, type **S** in the Format field or use the primary command **FS**. Figure 25-15 shows the SNGL format.

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)                    Table Edit                      1 of 100
Command ===>                                                    Scroll PAGE
TABLE FGINSUR.CUSTOMER                                          Format SNGL
                                        Top Line is 1    of 7     in Row 1
Key     Column           Data
PU      CUSTOMERNUMBER          378
        FIRSTNAME        ESKWX
        LASTNAME         EADOPCYB
        DATEOFBIRTH      1949-03-06
        HOUSENAME        YXDI NHQV
        HOUSENUMBER       435
        POSTCODE         HL81 8UX
***  End of record  ***
```

Figure 25-15 SNGL format

You can add or change the information for a fields' definitions on the following panel depending on your settings. For example, if you specify a slash (**/**) in the Picture Clause field, the format **PIC** is displayed near the column field as shown in Figure 25-16.

```
 Process   Options   Help
File Manager                 Edit/Browse Options
Command ===> _____________________________________________________________

Related command if applicable, shown in ()
Miscellaneous:            Enter "/" to select options          Initial Display
/  Prefix on   (PRE)              Prefix on right (PRE RIGHT)  1  1. Previous
/  Recognize and interpret ISPF packed data                        2. Table
   Show RBA and Length when browsing VSAM          (RBALEN)       3. Single
   CAPS initially ON - translate changed data to uppercase (CAPS) 4. Character
Expose (do not group) records of types: (SHOW)                    5. Hex
   Not selected     Suppressed  /  Length error                    6. LHex
See shadow lines (deselect to hide) for groups of: (SHADOW)
/  Not selected  /  Suppressed  /  Excluded
Record formatting options for SNGL display or print:
_  Field reference number (REF)    Field type and length values (TYPE)
   Picture clause         (PIC)    Start location               (SLOC)
   Structure              (STR)    Left justify numeric fields  (JUST)
/  Redefined fields       (RDF) (also affects TABL format)

Auxiliary Data Set Allocation Defaults
   High Level Qualifier                          leave blank for default
   Data class                                    leave blank for default
   Storage class                                 leave blank for default
   Management class                              leave blank for default
Model Data Set Name
_____________________________________________________________________
Auxiliary Data Set Name
_____________________________________________________________________
```

Figure 25-16 Edit/Browse options

25.4.3 Hex format

The data can also be displayed in hex format using the primary command HEX ON, as shown in Figure 25-17.

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)                Table Browse                    1 of 100
Command ===> hex on                                        Scroll CSR
TABLE FGINSUR.CUSTOMER                                     Format TABL
LASTNAME             FIRSTNAME  POSTCODE CUSTOMERNUMBER
 3                    2          7                    1
CHARACTER(20)        CH(10)     CH(8)          INTEGER
<---+----1----+----> <---+----> <---+--->    PU--+----1>
SZWQA                BPLP       HL86 2KS             37
EADOPCYB             HOWM       HL19 6KB             33
KNTLL                SSME       HL14 2KI             39
LBWTZFXC             SSME       HL21 3CR             38
SZWQA                EMIMQ      HL80 3BB             31
EADOPCYB             WMET       HL89 5RN             36
KNTLL                WMET       HL4  8UJ             32
YWXHXUQE             ESKWX      HL3  9PK             35
EADOPCYB             HOX        HL1  9IB             43
LBWTZFXC             WYTWN      HL02 5HW             44
FFFLA                SSME       HL17 6KT             49
BXUZO                HOWM       HL83 9XU             48
WMEHVL               EMIMQ      HL1  8TA             46
LBWTZFXC             BPLP       HL24 6ZD             40
SZWQA                HOX        HL05 6OT             45
```

Figure 25-17 HEX ON command

After entering the HEX ON command, the data is displayed as shown in Figure 25-18.

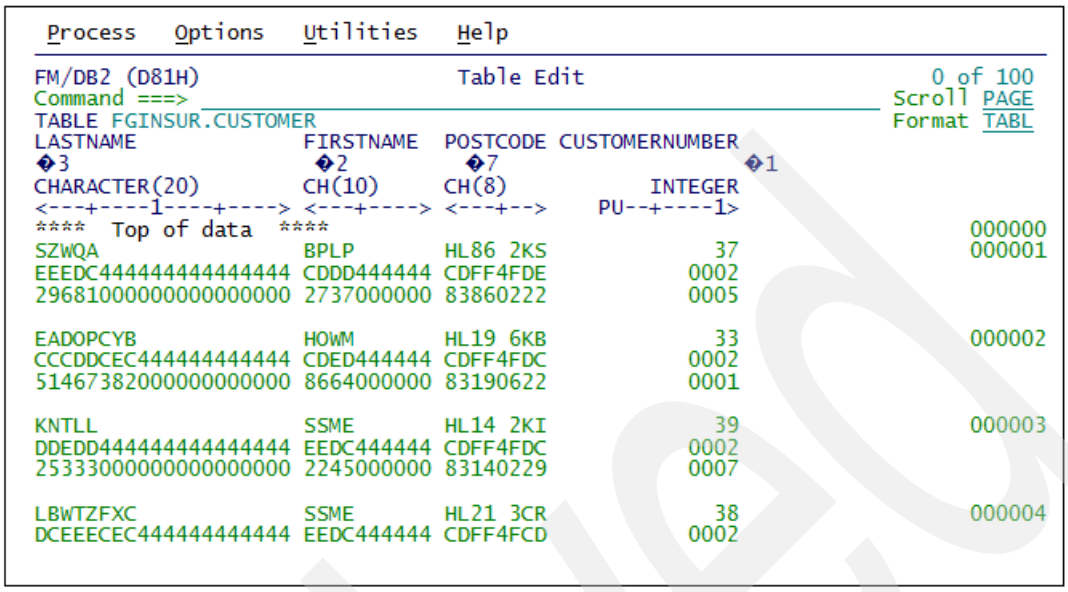


Figure 25-18 Data in hex

Displaying the prefix area

In Edit, the prefix area is used to specify prefix commands. The PREFIX primary command allows you to specify if you want FM/DB2 to display the prefix area and, if so, whether FM/DB2 displays it on the right or left side of the screen. In Edit, for TABL display format only, the PREFIX primary command sets the position and display state of the prefix area. Type `PR LEFT` on the Command line to display the prefix area on the Left as shown in Figure 25-19 and Figure 25-20.

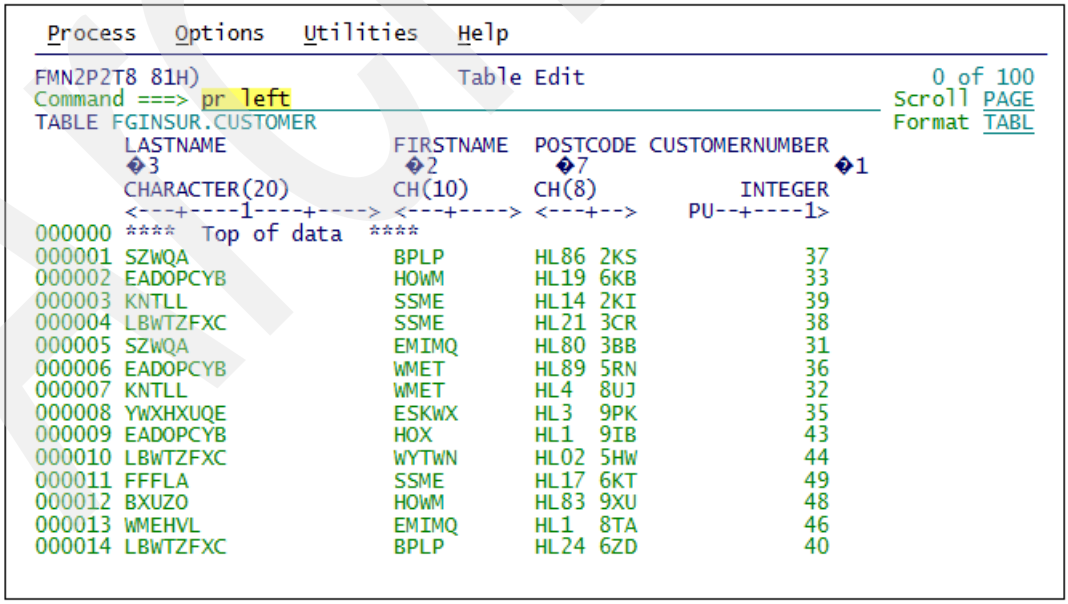


Figure 25-19 How to move the Prefix area to left

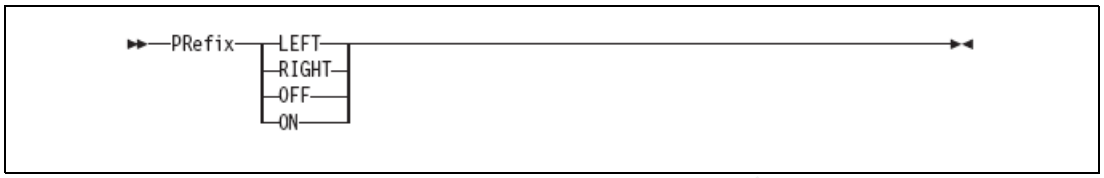


Figure 25-20 Syntax of Prefix Command

- LEFT: Displays the prefix area on the left side.
- RIGHT: Displays the prefix area on the right side.
- OFF: Does not display the prefix area.
- ON: Displays the prefix area at the position last set.

25.5 Inserting rows in a table

To insert a new row in the table, type the letter **I** in the prefix area, as shown in Figure 25-21.

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)                    Table Edit                          1 of 100
Command ===>                                                      Scroll PAGE
TABLE FGINSUR.CUSTOMER                                            Format TABL
       FIRSTNAME  LASTNAME            CUSTOMERNUMBER POSTCODE
       ?2         ?3                        ?1 ?7
       CH(10)     CHARACTER(20)          INTEGER CH(8)
       <---+----> <---+----1----+----> PU--+----1> <---+-->
000001 ESKWX      EADOPCYB                    378 HL81 8UX
000002 HOX        OJKZIR                      371 HL60 4UJ
I      BBXOK      FFFLA                       376 HL68 6HO
000004 BPLP       HOWSLQNT                    372 HL10 1JX
000005 SZCFJNO    BXUZO                       370 HL4  9HY
000006 BBXOK      SZWQA                       375 HL06 7NQ
000007 ESKWX      HOWSLQNT                    377 HL12 7QI
000008 HOWM       HOWSLQNT                    373 HL26 5GA
000009 EMIMQ      BXUZO                       374 HL66 5DP
000010 WMET       WMEHVL                      379 HL8  4LQ
000011 SSME       EADOPCYB                    337 HL85 4GW
000012 HOWM       SZWQA                       333 HL14 5QI
000013 EMIMQ      HOWSLQNT                    334 HL12 6RE
000014 BPLP       YWXHXUQE                    339 HL8  7LJ
000015 SSME       SZWQA                       338 HL05 4XI
```

Figure 25-21 How insert a new row

After pressing Enter, a new line opens, as shown in Figure 25-22.

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)                    Table Edit                          0 of 101
Command ===> ____________________________________________________ Scroll PAGE
TABLE FGINSUR.CUSTOMER                                           Format TABL
       FIRSTNAME  LASTNAME            CUSTOMERNUMBER POSTCODE
       ►2          ►3                        ►1 ►7
       CH(10)     CHARACTER(20)           INTEGER CH(8)
       <---+----> <---+----1----+----> PU--+----1> <---+-->
000000 ****  Top of data  ****
000001 ESKWX      EADOPCYB                   378 HL81 8UX
000002 HOX        OJKZIR                     371 HL60 4UJ
000003 BBXOK      FFFLA                      376 HL68 6HO
000004 _          _                            0 _
000005 BPLP       HOWSLQNT                   372 HL10 1JX
000006 SZCFJNO    BXUZO                      370 HL4  9HY
000007 BBXOK      SZWQA                      375 HL06 7NQ
000008 ESKWX      HOWSLQNT                   377 HL12 7QI
000009 HOWM       HOWSLQNT                   373 HL26 5GA
000010 EMIMQ      BXUZO                      374 HL66 5DP
000011 WMET       WMEHVL                     379 HL8  4LQ
000012 SSME       EADOPCYB                   337 HL85 4GW
000013 HOWM       SZWQA                      333 HL14 5QI
000014 EMIMQ      HOWSLQNT                   334 HL12 6RE
```

Figure 25-22 New line opened

You can now enter information on the new line, as shown in Figure 25-23, and press Enter.

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)                    Table Edit                          0 of 101
Command ===> ____________________________________________________ Scroll PAGE
TABLE FGINSUR.CUSTOMER                                           Format TABL
       FIRSTNAME  LASTNAME            CUSTOMERNUMBER POSTCODE
       ►2          ►3                        ►1 ►7
       CH(10)     CHARACTER(20)           INTEGER CH(8)
       <---+----> <---+----1----+----> PU--+----1> <---+-->
000000 ****  Top of data  ****
000001 ESKWX      EADOPCYB                   378 HL81 8UX
000002 HOX        OJKZIR                     371 HL60 4UJ
000003 BBXOK      FFFLA                      376 HL68 6HO
000004 Cinzia     Margherita                 a01 IT04 100
000005 BPLP       HOWSLQNT                   372 HL10 1JX
000006 SZCFJNO    BXUZO                      370 HL4  9HY
000007 BBXOK      SZWQA                      375 HL06 7NQ
000008 ESKWX      HOWSLQNT                   377 HL12 7QI
000009 HOWM       HOWSLQNT                   373 HL26 5GA
000010 EMIMQ      BXUZO                      374 HL66 5DP
000011 WMET       WMEHVL                     379 HL8  4LQ
000012 SSME       EADOPCYB                   337 HL85 4GW
000013 HOWM       SZWQA                      333 HL14 5QI
000014 EMIMQ      HOWSLQNT                   334 HL12 6RE
```

Figure 25-23 Entering informations

After pressing Enter, FM/DB2 validates the data and gives an error message if the data is “invalid numeric”. As shown in Figure 25-24, the wrong data format was entered for the CUSTOMER NUMBER column.

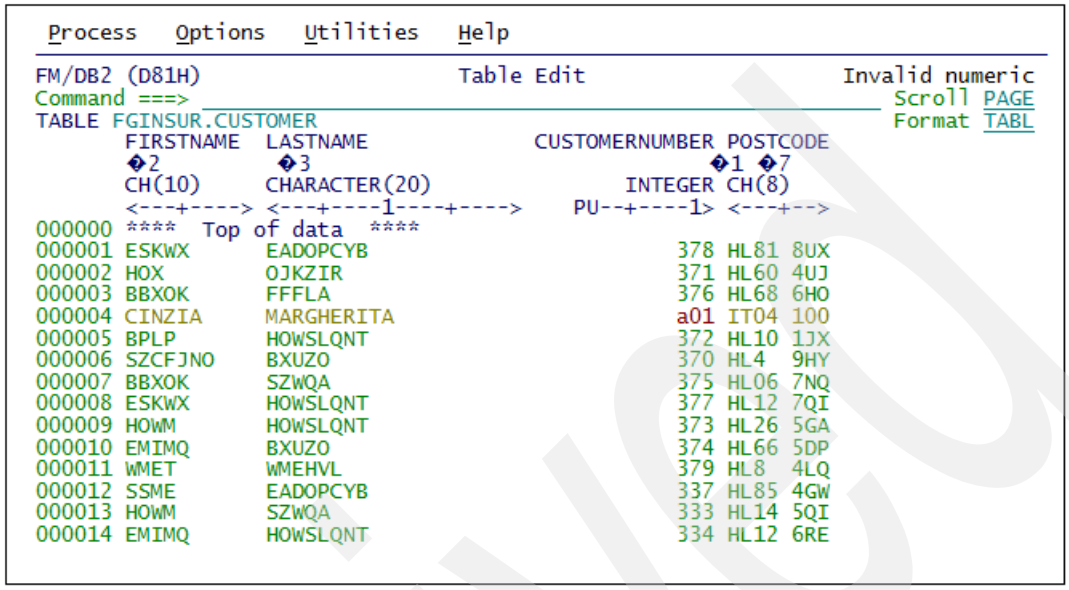


Figure 25-24 Error Message

Correct the CUSTOMER NUMBER with a numeric value (such as 876), and type the command SAVE on the Command line to confirm the new row as shown in Figure 25-25.

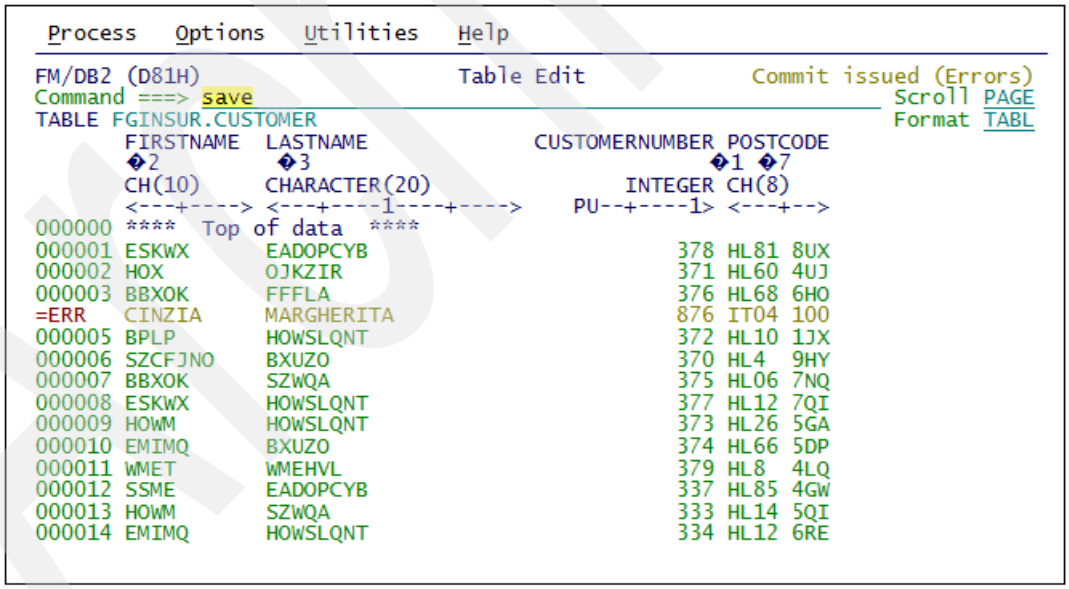


Figure 25-25 Numeric value corrected

Then press Enter to save your changes. The DB2 Edit panel displays a new error message, `Commit issued (Errors)`, at the top right of the panel, and FM marks the incorrect row with the sign=ERR as shown in Figure 25-25. You can receive more information regarding error messages if you type **E** in the Prefix area, as shown in Figure 25-26.

```
 Process    Options    Utilities    Help
FMN2P2T8 81H)                    Table Edit                          0 of 101
Command ===>                                                      Scroll PAGE
TABLE FGINSUR.CUSTOMER                                            Format TABL
       LASTNAME             FIRSTNAME  POSTCODE CUSTOMERNUMBER
       ?3                    ?2         ?7                ?1
       CHARACTER(20)        CH(10)     CH(8)          INTEGER
       <---+----1----+----> <---+----> <---+-->    PU--+----1>
000000 ****  Top of data  ****
000001 SZWQA                BPLP       HL86 2KS             37
000002 EADOPCYB             HOWM       HL19 6KB             33
e      MARGHERITA           CINZIA     IT04 100             98
000004 KNTLL                SSME       HL14 2KI             39
000005 LBWTZFXC             SSME       HL21 3CR             38
000006 SZWQA                EMIMQ      HL80 3BB             31
000007 EADOPCYB             WMET       HL89 5RN             36
000008 KNTLL                WMET       HL4  8UJ             32
000009 YWXHXUQE             ESKWX      HL3  9PK             35
000010 EADOPCYB             HOX        HL1  9IB             43
000011 LBWTZFXC             WYTWN      HL02 5HW             44
000012 FFFLA                SSME       HL17 6KT             49
000013 BXUZO                HOWM       HL83 9XU             48
000014 WMEHVL               EMIMQ      HL1  8TA             46
```

Figure 25-26 You can type “e” on the Prefix area to receive more information regarding the error

Note: In this example of researching an error message, you see that FM/DB2 automatically displays the wrong `CUSTOMER NUMBER` column.

Type **e** on the Prefix Area (related to the wrong row) and press Enter. You receive the next panel with the following instruction as shown in Figure 25-27.

```
                    DB2 Save Error Action
-  Command ===>
F
C  DB2 reported an error while attempting to save this row.
T
   Duplicate Row Error

   The insert or update operation on this line would have
   resulted in two rows with the same index key, and the
0  index is defined as unique.
0
0  Instructions:
0
E     Press ENTER or enter EXIT to return to the edit
0     session and correct the error in this line.
0
0     Enter CANCEL to terminate the edit session.
0     Any changes made since the last commit point
0     will be lost.
0
0
0
0
0
```

Figure 25-27 DB2 Instructions related to the row with “=ERR” in the prefix Area

The DB2 Save Error Action describes the reason for this error (Duplicate Row), and gives instructions to correct the error in this line. Press PF3 (Exit) to return to the Edit panel and change the value related to the Column (PU Primary Key Unique) CUSTOMER NUMBER from 876 to 877. Then press EXIT (or ENTER again) to commit the Insert as shown in Figure 25-28.

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)                    Table Edit                        0 of 101
Command ===> exit                                                  Scroll PAGE
TABLE FGINSUR.CUSTOMER                                             Format TABL
       FIRSTNAME  LASTNAME            CUSTOMERNUMBER POSTCODE
       ►2         ►3                        ►1 ►7
       CH(10)     CHARACTER(20)          INTEGER CH(8)
       <---+----> <---+----1----+---->   PU--+----1> <---+-->
000000 ****  Top of data  ****
000001 ESKWX      EADOPCYB                    378 HL81 8UX
000002 HOX        OJKZIR                      371 HL60 4UJ
000003 BBXOK      FFFLA                       376 HL68 6HO
=ERR   CINZIA     MARGHERITA                  877 IT04 100
000005 BPLP       HOWSLQNT                    372 HL10 1JX
000006 SZCFJNO    BXUZO                       370 HL4  9HY
000007 BBXOK      SZWQA                       375 HL06 7NQ
000008 ESKWX      HOWSLQNT                    377 HL12 7QI
000009 HOWM       HOWSLQNT                    373 HL26 5GA
000010 EMIMQ      BXUZO                       374 HL66 5DP
000011 WMET       WMEHVL                      379 HL8  4LQ
000012 SSME       EADOPCYB                    337 HL85 4GW
000013 HOWM       SZWQA                       333 HL14 5QI
000014 EMIMQ      HOWSLQNT                    334 HL12 6RE
```

Figure 25-28 Table Edit panel

When the Insert Key value is correct, you receive the next panel. In the upper right corner the message Commit issued is returned to confirm that the new row has been inserted as shown in Figure 25-29.

```
 Process   Options   Utilities   Help
FMN2PD2E 81H)                    DB2 Edit                    Commit issued
Command ===>

Specify the DB2 Object:
   Location  . . . . .                    Database . .  ________ (optional)
   Owner . . . . . . . FGINSUR       +    Table space   ________ (optional)
   Name  . . . . . . . CUSTOMER                                        +

   Row count . . . . . 100          Number of rows to edit

Template:
   Data set name . . . 'AMINTOR.BOOK2006.TEMPLATE'
   Member  . . . . . . DB2TEMP1

Processing Options:
   Template usage                   Enter "/", "A"lways to select option
   1  1. Above                         Edit options
      2. Previous                      Edit template
      3. Generate from table           Re-edit template
      4. Generate/Replace
```

Figure 25-29 Insert completed

25.6 Deleting a row from a table

To delete a row from a table, enter the letter **D** in the prefix area, as shown in Figure 25-30 and press Enter.

```
 Process    Options    Utilities    Help
FM/DB2 (D81H)                    Table Edit                          0 of 100
Command ===> sort 1                                                  Scroll PAGE
TABLE FGINSUR.CUSTOMER                                               Format TABL
       FIRSTNAME  LASTNAME             CUSTOMERNUMBER POSTCODE
       2          3                          1 7
       CH(10)     CHARACTER(20)          INTEGER CH(8)
       <---+----> <---+----1----+----> PU--+----1> <---+-->
000000 ****  Top of data  ****
d00001 CINZIA     MARGHERITA                  877 IT04 1OO
000002 WYTWNZZSJ  BXUZO                       881 HL06 4GI
000003 WYTWNZZSJ  HOWSLQNT                    890 HL3  1JC
000004 WYTWNZZSJ  FFFLA                       891 HL7  4JW
000005 BPLP       HOWSLQNT                    895 HL10 2ML
000006 WYTWNZZSJ  HOWSLQNT                    896 HL23 9ZG
000007 WYTWNZZSJ  WMEHVL                      904 HL18 2AP
000008 WYTWNZZSJ  HOWSLQNT                    906 HL63 2LR
000009 WYTWNZZSJ  KNTLL                       907 HL66 5CW
000010 WYTWNZZSJ  OJKZIR                      909 HL06 5BQ
000011 WYTWN      BXUZO                       911 HL18 7UO
000012 WYTWNZZSJ  SZWQA                       921 HL28 2MP
000013 WYTWNZZSJ  KNTLL                       923 HL22 4SO
000014 BPLP       OJKZIR                      926 HL28 4XQ
```

Figure 25-30 D in Prefix Area to delete the row

After pressing Enter, the row is removed, as shown in Figure 25-31.

```
 Process    Options    Utilities    Help
FM/DB2 (D81H)                    Table Edit                          0 of 100
Command ===> exit                                                    Scroll PAGE
TABLE FGINSUR.CUSTOMER                                               Format TABL
       FIRSTNAME  LASTNAME             CUSTOMERNUMBER POSTCODE
       2          3                          1 7
       CH(10)     CHARACTER(20)          INTEGER CH(8)
       <---+----> <---+----1----+----> PU--+----1> <---+-->
000000 ****  Top of data  ****
000001 WYTWNZZSJ  BXUZO                       881 HL06 4GI
000002 WYTWNZZSJ  HOWSLQNT                    890 HL3  1JC
000003 WYTWNZZSJ  FFFLA                       891 HL7  4JW
000004 BPLP       HOWSLQNT                    895 HL10 2ML
000005 WYTWNZZSJ  HOWSLQNT                    896 HL23 9ZG
000006 WYTWNZZSJ  WMEHVL                      904 HL18 2AP
000007 WYTWNZZSJ  HOWSLQNT                    906 HL63 2LR
000008 WYTWNZZSJ  KNTLL                       907 HL66 5CW
000009 WYTWNZZSJ  OJKZIR                      909 HL06 5BQ
000010 WYTWN      BXUZO                       911 HL18 7UO
000011 WYTWNZZSJ  SZWQA                       921 HL28 2MP
000012 WYTWNZZSJ  KNTLL                       923 HL22 4SO
000013 BPLP       OJKZIR                      926 HL28 4XQ
000014 WYTWNZZSJ  OJKZIR                      928 HL24 1HE
```

Figure 25-31 Row removed, type exit on the Command line to update the DB2 Database

When exiting (or pressing F3), a commit is issued, as shown in Figure 25-32.

```
 Process   Options   Utilities   Help
FMN2PD2E 81H)                    DB2 Edit                        Commit issued
Command ===>

Specify the DB2 Object:
   Location  . . . . .                        Database . .  _________ (optional)
   Owner . . . . . . . FGINSUR          +     Table space   _________ (optional)
   Name  . . . . . . . CUSTOMER                                       +

   Row count . . . . . 100          Number of rows to edit

Template:
   Data set name . . . 'AMINTOR.BOOK2006.TEMPLATE'
   Member  . . . . . . DB2TEMP1

Processing Options:
   Template usage                   Enter "/", "A"lways to select option
   1  1. Above                         Edit options
      2. Previous                      Edit template
      3. Generate from table           Re-edit template
      4. Generate/Replace
```

Figure 25-32 Commit issued

You can also delete multiple rows, either with the **DD** block command or **D22** in the Prefix Area (**D22** deletes 22 rows shown in the panel). If you must back out your changes, follow the next steps as shown in Figure 25-33.

1. Choose the lines that you have to delete and press Enter (more than once, if necessary)

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)                    Table Edit                       0 of 101
Command ===>                                                   Scroll PAGE
TABLE FGINSUR.CUSTOMER                                         Format TABL
       FIRSTNAME  LASTNAME            CUSTOMERNUMBER POSTCODE
       ▯2         ▯3                        ▯1 ▯7
       CH(10)     CHARACTER(20)          INTEGER CH(8)
       <---+----> <---+----1----+---->  PU--+----1> <---+-->
000000 ****  Top of data  ****
000001 WYTWNZZSJ  WMEHVL                      969 HL2  0SR
000002 WYTWNZZSJ  YWXHXUQE                    962 HL11 8WI
000003 WYTWNZZSJ  KNTLL                       960 HL84 4KI
000004 WYTWNZZSJ  BXUZO                       963 HL62 6UF
000005 WYTWNZZSJ  SZWQA                       921 HL28 2MP
000006 BPLP       OJKZIR                      926 HL28 4XQ
d      MARIANNA   REBECCA                     877 IT04 100
000008 WYTWNZZSJ  KNTLL                       923 HL22 4SO
000009 WYTWNZZSJ  YWXHXUQE                    929 HL20 1BG
000010 WYTWNZZSJ  OJKZIR                      928 HL24 1HE
000011 WYTWNZZSJ  OJKZIR                      909 HL06 5BQ
000012 WYTWNZZSJ  HOWSLQNT                    906 HL63 2LR
000013 WYTWNZZSJ  KNTLL                       907 HL66 5CW
000014 WYTWNZZSJ  WMEHVL                      904 HL18 2AP
```

Figure 25-33 Choose the rows to delete

2. Enter the CAN command in the Command Line before processing any other command as shown in Figure 25-34.

```
 Process    Options    Utilities    Help
FM/DB2 (D81H)                      Table Edit                          0 of 100
Command ===> can                                                       Scroll PAGE
TABLE FGINSUR.CUSTOMER                                                 Format TABL
       FIRSTNAME  LASTNAME             CUSTOMERNUMBER POSTCODE
       ▯2         ▯3                              ▯1 ▯7
       CH(10)     CHARACTER(20)             INTEGER CH(8)
       <---+----> <---+----1----+---->    PU--+----1> <---+-->
000000 ****  Top of data  ****
000001 WYTWNZZSJ  WMEHVL                        969 HL2  0SR
000002 WYTWNZZSJ  YWXHXUQE                      962 HL11 8WI
000003 WYTWNZZSJ  KNTLL                         960 HL84 4KI
000004 WYTWNZZSJ  BXUZO                         963 HL62 6UF
000005 WYTWNZZSJ  SZWQA                         921 HL28 2MP
000006 BPLP       OJKZIR                        926 HL28 4XQ
000007 WYTWNZZSJ  KNTLL                         923 HL22 4SO
000008 WYTWNZZSJ  YWXHXUQE                      929 HL20 1BG
000009 WYTWNZZSJ  OJKZIR                        928 HL24 1HE
000010 WYTWNZZSJ  OJKZIR                        909 HL06 5BQ
000011 WYTWNZZSJ  HOWSLQNT                      906 HL63 2LR
000012 WYTWNZZSJ  KNTLL                         907 HL66 5CW
000013 WYTWNZZSJ  WMEHVL                        904 HL18 2AP
000014 WYTWNZZSJ  BXUZO                         959 HL5  0KY
```

Figure 25-34 Can command

3. After entering the CAN command, a Confirmation Cancel pop-up panel is displayed, as shown in Figure 25-35.

```
                          Confirm Cancel
 Command ===> ENTER                                    0 of 100
 Data has been changed.                                Scroll PAGE
                                                       Format TABL
 Instructions:

    Press ENTER key to confirm cancel request.
    (All changes will be discarded)

    Enter END or EXIT command to return to edit or view
    session.
```

Figure 25-35 Confirmation panel

4. Press Enter to Rollback your changes (End or exit returns to the previous Edit panel and any changes are saved). After pressing Enter, the DB2 Edit panel is displayed, giving the message `Rollback issued`, as shown in Figure 25-36.

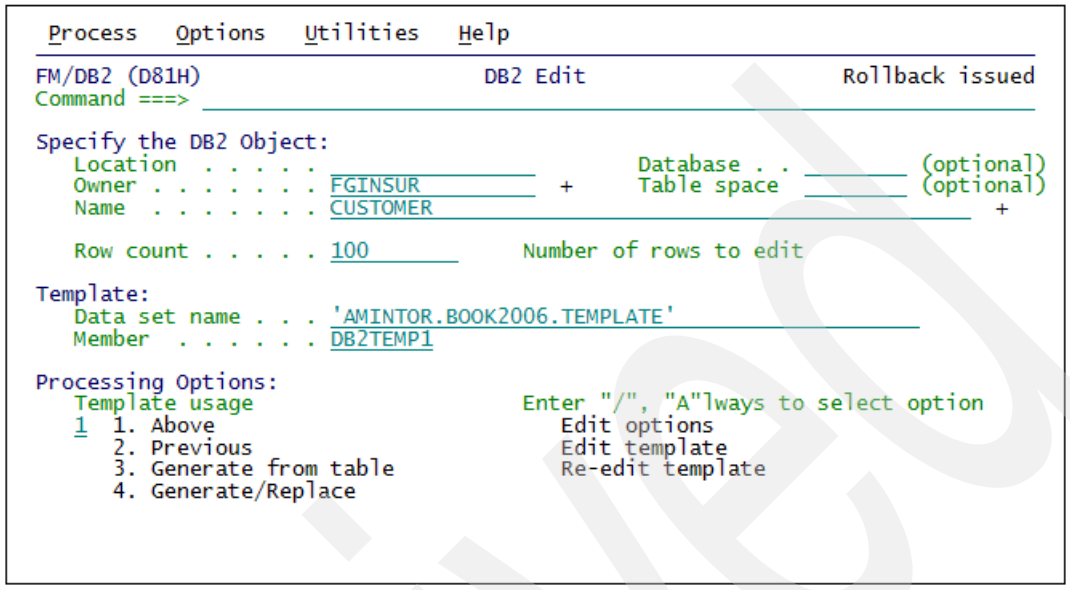


Figure 25-36 Rollback

If you press Enter again to go back to the Table Edit Panel, you can see the “undeleted” rows still displayed. The Table occurrences are committed to DB2 only when you type `SAVE` in the command line, or press PF3 to exit.

25.7 Finding/changing data

In this section we discuss the commands for finding and changing data.

25.7.1 Finding data

The FIND primary command searches the data for a character string or a numeric field with a particular numeric value. The syntax structure could be different, as it depends on the utility you are using:

- In Browse, Edit, or the Object List utility:
  - If the string or numeric value is found, then the FIND command positions the cursor at the beginning of the string or numeric field and, if necessary, automatically scrolls the found data into view.
  - The FIND command highlights all occurrences of the search string or numeric value (even when the commands FIRST, LAST, PREV, NEXT, EX, NX, or X are specified). To turn off the highlighting, enter the RESET FIND command.
- In Print Browse:
  - If the character string is found, the record containing the string is displayed at the top panel.

Note:

FIND * does not repeat the previous FIND command with the same parameters. The command FIND * repeats the previous FIND command with the same string argument, but all other parameters revert to their default values unless specified.

National characters: The symbols “£” or “#” can be used as a “Field reference number” to refer to a Template column. The symbol you should use depends on your Host code-page. Verify your setting before processing commands.

Example: Use “#” instead of “£” if your (IBM Personal Communication) Session Parameters Customization is set as: **037 United States** instead of **1144 Italy Euro**.

In this chapter we are using the following Host code-page: 1144 Italy Euro.

To find the next occurrence of the same string, press the RFind function key (F5), enter `FIND *`, or enter the FIND command with no parameters. A message is displayed if the string cannot be found. The FIND command syntax is shown in Figure 25-37.

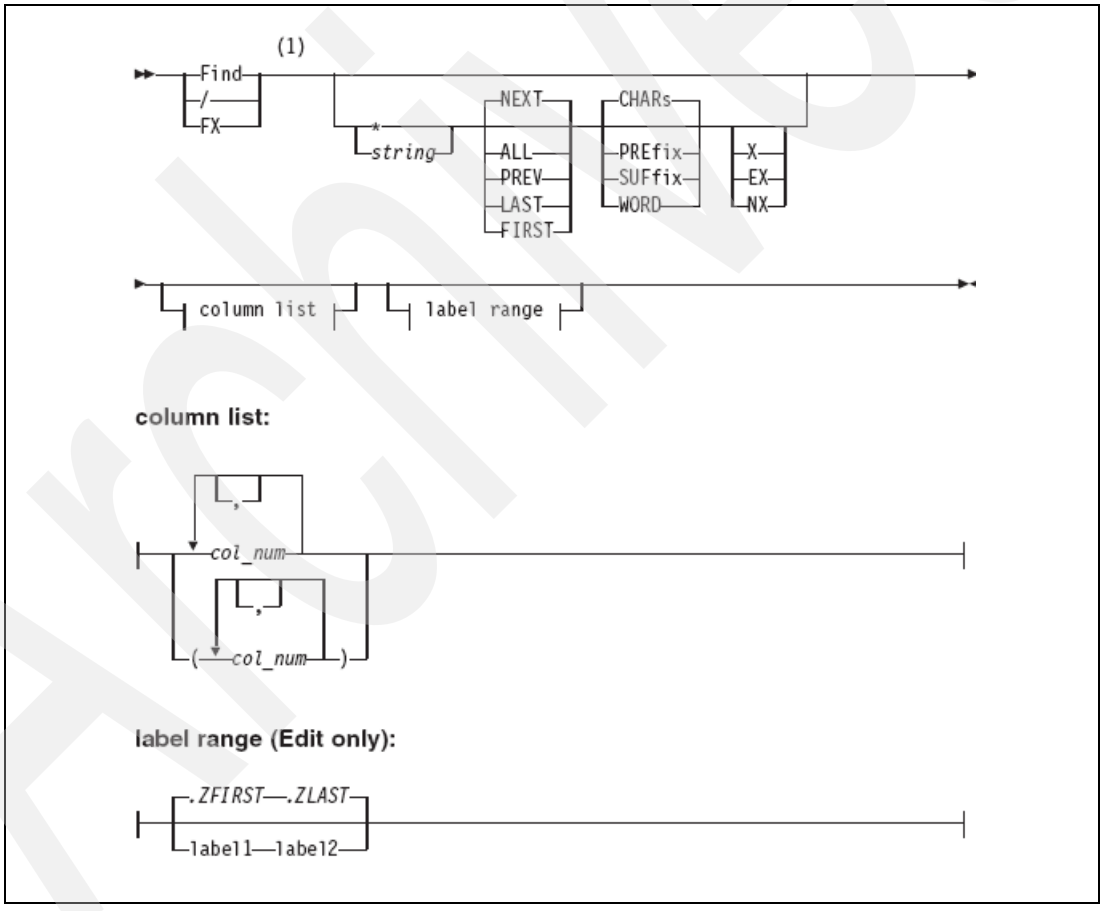


Figure 25-37 The figure shows the main parameters for the FIND Command Syntax

Tip: You can now specify the parameters in any order you prefer, and you obtain the same results. This means that you can type the command (on the command line in the Browse, Edit, or Object List utility) using either of the following syntax examples:

```
FIND “alba” £3 ALL (ISPF- like)
FIND ALL “alba” £3“
```

In this example, as shown in Figure 25-38, the FIND command is used to display only the rows in the table where WORKDEPT = C01.

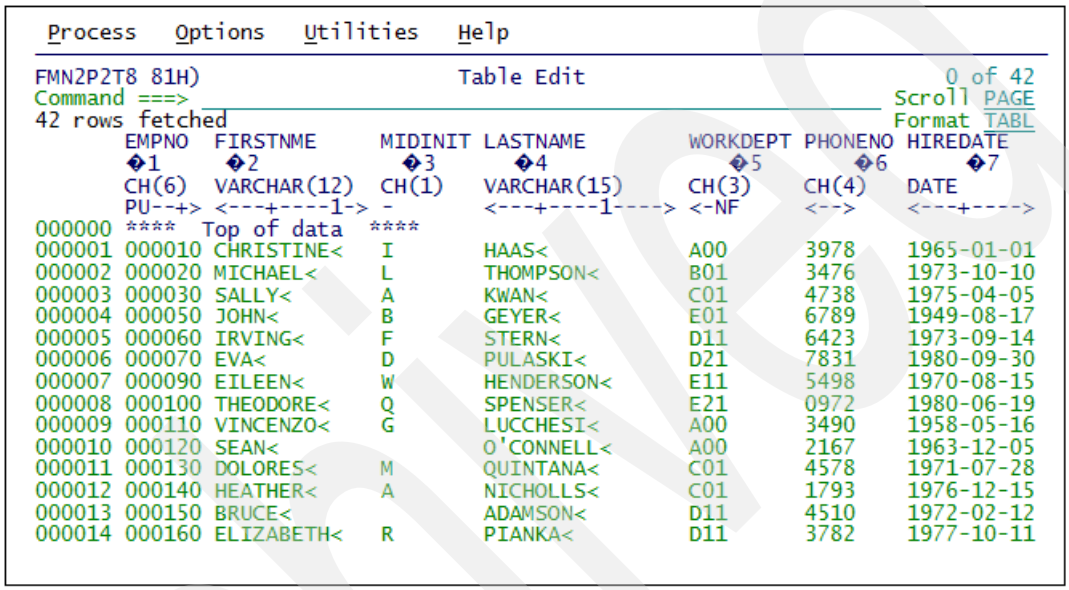


Figure 25-38 Table Edit panel

After displaying all rows, use the primary command EXCLUDE to exclude all rows from the display, as shown in Figure 25-39. This command can be abbreviated to **X** or **EX**.

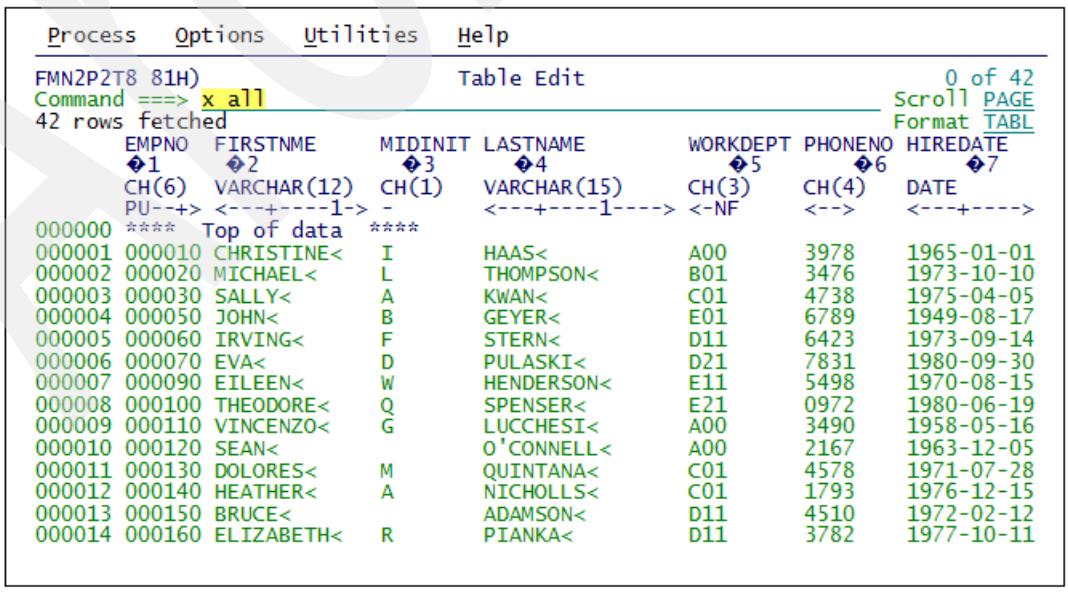


Figure 25-39 About to exclude all rows from display

We can *concatenate* (using Semicolon ";" character) the commands of ISPF. Type the following command: "X ALL; F C01 £5 ALL" and the results are displayed in the next Table Edit panel, as shown in Figure 25-40.

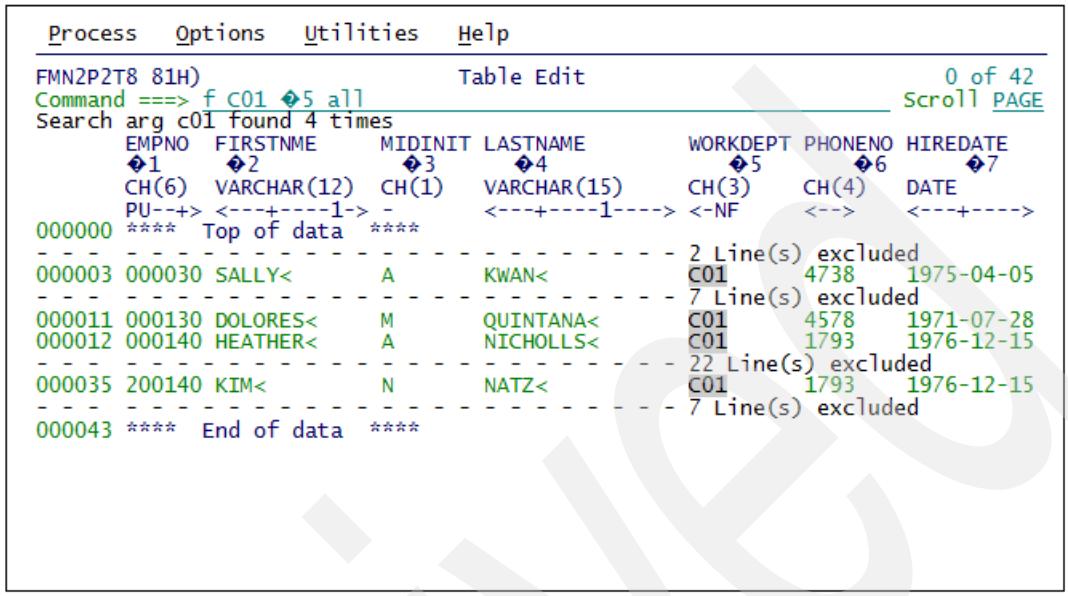


Figure 25-40 After the following Commands; EX ALL; F C01 £5 ALL

We have already executed the FIND command (FIND can be abbreviated to F) in order to display all rows that have the value of C01 in the WORKDEPT column.

In the FIND command, we have substituted WORKDEPT with the Field reference number £5, as shown in Command Line above. You can shadow your *lines excluded* or *not selected* from the panel using the following command, as shown in Figure 25-41.

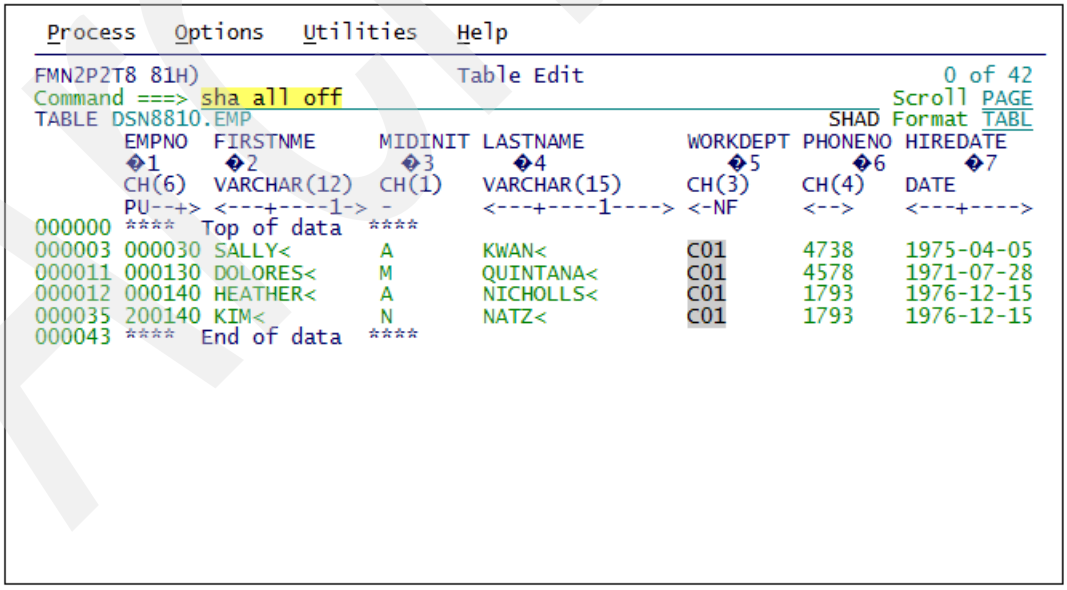


Figure 25-41 Selected rows displayed without comments

25.7.2 Changing data

In Edit mode, use the primary command **CHANGE** to locate and replace one or more occurrences of character or numeric data.

Figure 25-42 shows the main parameters for the syntax of the change command.

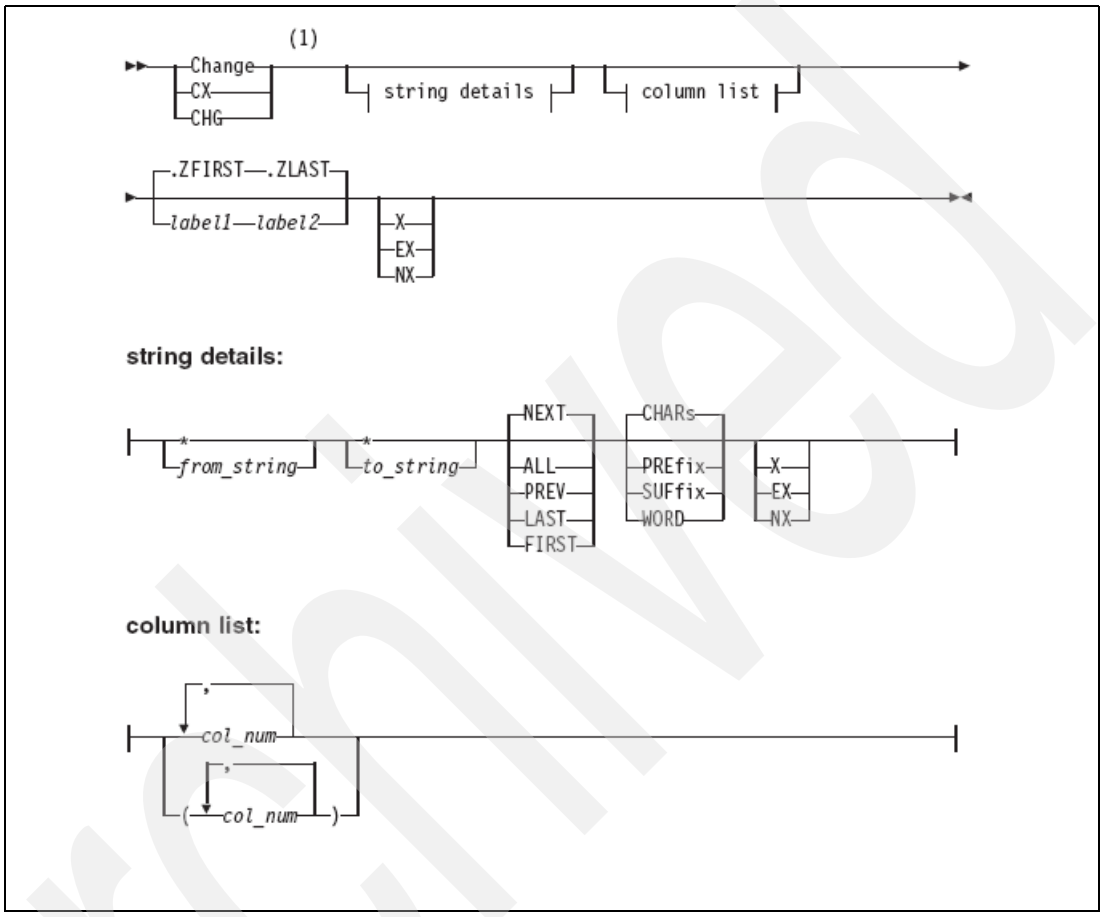


Figure 25-42 The CHANGE command syntax

Note: If you have zoomed in on a row using Single display, the CHANGE command affects only that row. If you limit the number of rows loaded into the FM/DB2 editor by either specifying a row limit or a WHERE clause, the CHANGE command applies only to the data in the FM/DB2 Edit session.

Add same details regarding the new CHANGE Syntax:

- **ASTERISK** When used in place of the search string, FM\DB2 uses the search string specified on the previous CHANGE command. When used in place of the replacement string, FM\DB2 uses the replacement string specified on the previous CHANGE command as the replacement string. The position of the * is important for the CHANGE command. If it is positioned before a string, it indicates the previous search argument; that is, it is treated as the first string. If a string is found prior to the *, then it is treated as the second string (the change argument), taking the change argument from the previous CHANGE command. To use both the previous search string and the previous change string, specify CHANGE * *.
- **CHARS** Matches the search string anywhere in the data.

- **PREFIX** Matches the search string wherever it appears as a prefix in the data. A prefix is defined as being preceded by a non-alphanumeric character, or being the start of a line or column and followed by an alphanumeric character.
- **SUFFIX** Matches the search string wherever it appears as a suffix in the data. A suffix is defined as being preceded by an alphanumeric character and followed by a non-alphanumeric character or the end of a line or column.
- **WORD** Matches the search string wherever it appears as a word in the data. A word is defined as being preceded and followed by a non-alphanumeric character or the start or end of a line or column.

Tip: You can now specify the parameters in any order you prefer, and you obtain the same results. This means that you can type the command (on the command line in the Browse, Edit, or Object List utility) using either of the following syntax examples:

```
CHANGE “alba” “tramonto” £3 ALL (ISPF- like)
CHANGE ALL “alba” “tramonto” £3“
```

In the example below use the CHANGE command to change the content in the WORKDEPT column from C01 to C03 for all rows in the table.

In the CHANGE command, substitute WORKDEPT with the Field reference number Ä5, as shown in Figure 25-43.

```
 Process   Options   Utilities   Help
FMN2P2T8 81H)                        Table Edit                          0 of 42
Command ===> c c01 c03 ❖5 all                                        Scroll PAGE
42 rows fetched                                                      Format TABL
       EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE
       ❖1     ❖2           ❖3      ❖4              ❖5       ❖6      ❖7
       CH(6)  VARCHAR(12)  CH(1)   VARCHAR(15)     CH(3)    CH(4)   DATE
       PU--+> <---+----1-> -       <---+----1----> <-NF     <-->    <---+---->
000000 ****  Top of data  ****
000001 000010 CHRISTINE<  I       HAAS<           A00      3978    1965-01-01
000002 000020 MICHAEL<    L       THOMPSON<       B01      3476    1973-10-10
000003 000030 SALLY<      A       KWAN<           C01      4738    1975-04-05
000004 000050 JOHN<       B       GEYER<          E01      6789    1949-08-17
000005 000060 IRVING<     F       STERN<          D11      6423    1973-09-14
000006 000070 EVA<        D       PULASKI<        D21      7831    1980-09-30
000007 000090 EILEEN<     W       HENDERSON<      E11      5498    1970-08-15
000008 000100 THEODORE<   Q       SPENSER<        E21      0972    1980-06-19
000009 000110 VINCENZO<   G       LUCCHESI<       A00      3490    1958-05-16
000010 000120 SEAN<               O'CONNELL<      A00      2167    1963-12-05
000011 000130 DOLORES<    M       QUINTANA<       C01      4578    1971-07-28
000012 000140 HEATHER<    A       NICHOLLS<       C01      1793    1976-12-15
000013 000150 BRUCE<              ADAMSON<        D11      4510    1972-02-12
000014 000160 ELIZABETH<  R       PIANKA<         D11      3782    1977-10-11
```

Figure 25-43 The CHANGE command

After executing the command the Table Edit panel is displayed, as shown in Figure 25-44. You are informed of how many strings have been changed.

```
 Process   Options   Utilities   Help
FMN2P2T8 81H)                    Table Edit                  4 string(s) changed
Command ===> _____________________________________________________ Scroll PAGE
TABLE DSN8810.EMP                                                  Format TABL
       EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE
       ➊1     ➊2           ➊3      ➊4              ➊5       ➊6      ➊7
       CH(6)  VARCHAR(12)  CH(1)   VARCHAR(15)     CH(3)    CH(4)   DATE
       PU--+> <---+----1-> -       <---+----1----> <-NF     <-->    <---+---->
000000 ****  Top of data  ****
000001 000010 CHRISTINE<   I       HAAS<           A00      3978    1965-01-01
000002 000020 MICHAEL<     L       THOMPSON<       B01      3476    1973-10-10
000003 000030 SALLY<       A       KWAN<           C03      4738    1975-04-05
000004 000050 JOHN<        B       GEYER<          E01      6789    1949-08-17
000005 000060 IRVING<      F       STERN<          D11      6423    1973-09-14
000006 000070 EVA<         D       PULASKI<        D21      7831    1980-09-30
000007 000090 EILEEN<      W       HENDERSON<      E11      5498    1970-08-15
000008 000100 THEODORE<    Q       SPENSER<        E21      0972    1980-06-19
000009 000110 VINCENZO<    G       LUCCHESI<       A00      3490    1958-05-16
000010 000120 SEAN<                O'CONNELL<      A00      2167    1963-12-05
000011 000130 DOLORES<     M       QUINTANA<       C03      4578    1971-07-28
000012 000140 HEATHER<     A       NICHOLLS<       C03      1793    1976-12-15
000013 000150 BRUCE<               ADAMSON<        D11      4510    1972-02-12
000014 000160 ELIZABETH<   R       PIANKA<         D11      3782    1977-10-11
```

Figure 25-44 Change complete

Press PF3 (or the SAVE command and then PF3) to receive the confirm at the top-right (*Commit issued*)

25.7.3 Sorting data

In Browse or Edt use the primary command SORT to change the order of the data in Table display format.

FM/DB2 sorts the data according to its hexadecimal representation using a hierarchy of keys, starting with the first specified column as the primary key, and subsequent specified columns as less significant keys. You specify the hierarchy left to right.

When you use the Object List utility (3.4) to display a list of objects, you can sort a column in the current object list by issuing the SORT primary command with the cursor positioned on the column you want to sort.

Figure 25-45 shows the parameters for the SORTcommand

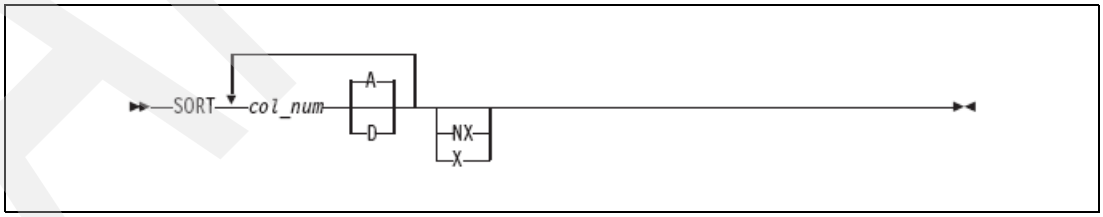


Figure 25-45 Sort primary command Syntax

For example, use the SORT command to sort the display of the table, as shown in the next Figure. Sort on the columns CUSTOMER NUMBER and LASTNAME, and the sort occurs with the first specified column. You can specify up to five columns. Figure 25-46 shows the occurrences before the sort command is processed.

```
 Process    Options    Utilities    Help
FM/DB2 (D81H)                      Table Edit                      Commit issued
Command ===>                                                        Scroll PAGE
TABLE FGINSUR.CUSTOMER                                              Format TABL
       FIRSTNAME  LASTNAME            CUSTOMERNUMBER POSTCODE
       ▯2         ▯3                        ▯1 ▯7
       CH(10)     CHARACTER(20)          INTEGER CH(8)
       <---+----> <---+----1----+---->  PU--+----1> <---+-->
000000 ****  Top of data  ****
000001 SSME       LBWTZFXC                      7 HL22 3HQ
000002 HOX        BXUZO                        10 HL01 8ER
000003 SSME       YWXHXUQE                     11 HL1  8TW
000004 HOX        LBWTZFXC                     14 HL5  1QE
000005 MARIANNA   REBECCA                       2 IT04 100
000006 WMET       BXUZO                        16 HL27 9QQ
000007 ESKWX      OJKZIR                       18 HL66 3DM
000008 BPLP       SZWQA                        19 HL05 0HJ
000009 HOX        LBWTZFXC                     20 HL80 1ER
000010 BBXOK      BXUZO                        21 HL83 4XO
000011 HOX        WMEHVL                       22 HL83 6MH
000012 BBXOK      KNTLL                        23 HL27 7RL
000013 WMET       LBWTZFXC                     24 HL85 1XZ
000014 WYTWN      OJKZIR                       25 HL7  4JH
```

Figure 25-46 Occurrences before the execution of the Sort primary command

The result after the sort is processed is shown in Figure 25-47.

```
 Process    Options    Utilities    Help
FM/DB2 (D81H)                      Table Edit                       0 of 101
Command ===> sort ▯1 ▯3                                             Scroll PAGE
TABLE FGINSUR.CUSTOMER                                              Format TABL
       FIRSTNAME  LASTNAME            CUSTOMERNUMBER POSTCODE
       ▯2         ▯3                        ▯1 ▯7
       CH(10)     CHARACTER(20)          INTEGER CH(8)
       <---+----> <---+----1----+---->  PU--+----1> <---+-->
000000 ****  Top of data  ****
000001 MARIANNA   REBECCA                       2 IT04 100
000002 SSME       LBWTZFXC                      7 HL22 3HQ
000003 HOX        BXUZO                        10 HL01 8ER
000004 SSME       YWXHXUQE                     11 HL1  8TW
000005 HOX        LBWTZFXC                     14 HL5  1QE
000006 WMET       BXUZO                        16 HL27 9QQ
000007 ESKWX      OJKZIR                       18 HL66 3DM
000008 BPLP       SZWQA                        19 HL05 0HJ
000009 HOX        LBWTZFXC                     20 HL80 1ER
000010 BBXOK      BXUZO                        21 HL83 4XO
000011 HOX        WMEHVL                       22 HL83 6MH
000012 BBXOK      KNTLL                        23 HL27 7RL
000013 WMET       LBWTZFXC                     24 HL85 1XZ
000014 WYTWN      OJKZIR                       25 HL7  4JH
```

Figure 25-47 Occurrences after the execution of the Sort primary command

25.8 Utility functions

To perform the utility functions select 3 in the Primary Options Menu. Then press Enter to display the Utility Functions panel as shown below in Figure 25-48.

```
 Process   Options   Utilities   Help
FMN2STUT 81H)                Utility Functions
Command ===>

1  Print          Print DB2 table or view
2  Objects        Create and drop DB2 objects
3  Copy           Copy data within DB2
4  Object List    Display and process DB2 object lists
5  Privileges     Manage DB2 privileges
6  Import         Import sequential or VSAM data into DB2
7  Export         Export DB2 data to sequential or VSAM data set
8  Create         Create DB2 test data
9  Utilities      DB2 utility job generation
10 Audit trail    Print audit trail report
11 Print browse   Browse FM/DB2 print data set
```

Figure 25-48 Utility Functions panel

25.8.1 Creating a table

The Utility Functions panel is displayed when you select 3 (Utilities) from the Primary Option Menu. Figure 25-49 lists the utilities you can use to manipulate DB2 data. To create a new table select option **2** (Objects) as shown in Figure 25-49.

```
 Process   Options   Utilities   Help
FMN2STUT 81H)                Utility Functions
Command ===> 2

1  Print          Print DB2 table or view
2  Objects        Create and drop DB2 objects
3  Copy           Copy data within DB2
4  Object List    Display and process DB2 object lists
5  Privileges     Manage DB2 privileges
6  Import         Import sequential or VSAM data into DB2
7  Export         Export DB2 data to sequential or VSAM data set
8  Create         Create DB2 test data
9  Utilities      DB2 utility job generation
10 Audit trail    Print audit trail report
11 Print browse   Browse FM/DB2 print data set
```

Figure 25-49 DB2 Object Functions create table

The DB2 Object Functions panel is displayed, as shown in Figure 25-50. You can create or delete objects. Select **C** to create an object and select **3** for the object type Table:

```
 Process    Options    Utilities    Help
FMN2POP D81H)                DB2 Object Functions
Command ===> C

       C Create object                        D Drop object

Processing Options:
   Object Type
   3   1. Database        9.  Function
       2. Table space     10. Stored procedure
       3. Table           11. Trigger
       4. View            12. Auxiliary table
       5. Alias           13. Storage group
       6. Index           14. Global temp. table
       7. Synonym         15. Sequence
       8. Distinct type

   Enter "/" to select option
   /  Confirm object drop
```

Figure 25-50 DB2 Object Function panel to create different object types

The Create Table panel is now displayed, as shown in Figure 25-51.

```
 Process    Options    Utilities    Help
FMN2PCT0 81H)                  Create Table
Command ===>
                                                              More:     +
New Table:
   Owner . . . . . . . AMINTOR          + (optional)
   Name  . . . . . . . EMPNEW                                        +
   Database  . . . . . _______            (optional)
   Table Space . . . . _______            (optional)

Model Table/View:  (optional)
   Owner . . . . . . . DSN8810          +
   Name  . . . . . . . emp                                           +
   Usage:
      1  1. Generate LIKE clause
         2. Load table information

Table Creation:
   Creation Options                 (Use option 1 before options 2, 4-9)
   9   1. Columns                   (normally required, invalid with model LIKE)
       2. Nulls/default values      (optional)
       3. Table options             (editproc,validproc,more; optional)
       4. Unique constraints        (primary/unique keys; optional)
```

Figure 25-51 Create Table panel

As shown on the panel above, data for the new table has been entered:

- Owner - AMINTOR
- Name - EMPNEW

As shown on this panel, information for the model table has been entered:

- Owner - DSN8810
- Name - EMP

Enter 9 in the field Creation Options of the Create Table panel shown in the next Figure and press Enter. This processes the statement and the table is created.

By pressing F8 you can scroll in this panel to see all Create Options, as shown in Figure 25-52.

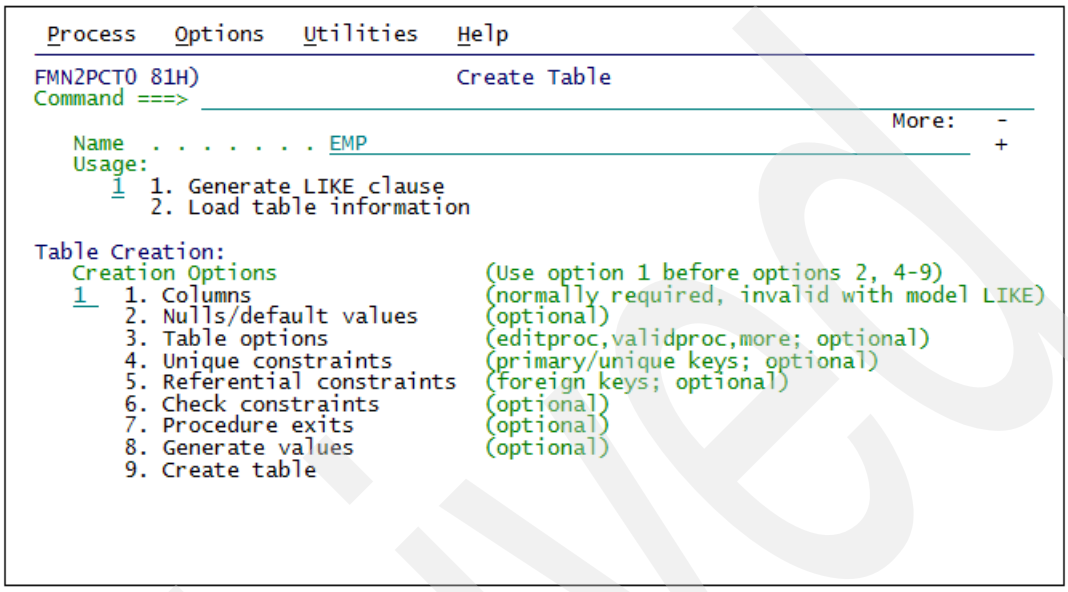


Figure 25-52 Create Table panel scrolled with F8 to see the Options

After pressing Enter, the table is created and the Create Table panel is displayed, as shown in Figure 25-53.

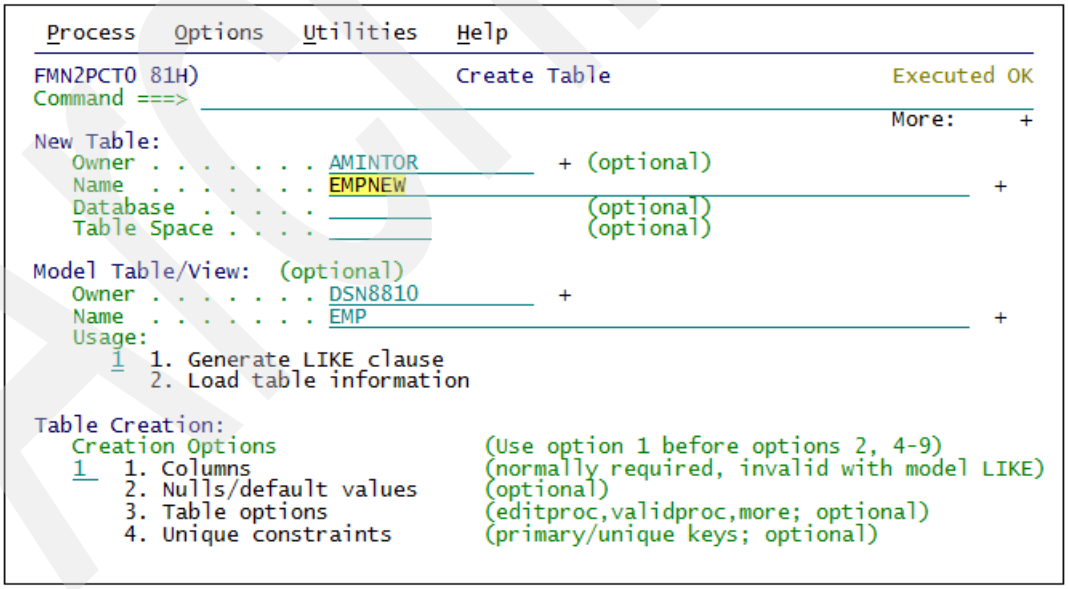


Figure 25-53 Create statement executed OK

25.8.2 Copying a table

Now copy data into the table created in the previous step.

To copy data select 3 from the Utility Functions panel shown in Figure 25-54.

```
 Process   Options   Utilities   Help
FMN2PCPI 81H)                    Copy Utility
Command ===> ____________________________________________________________

From DB2 Object:
   Location  . . . . . _____________          Database . . ________ (optional)
   Owner . . . . . . . DSN8810_____  +       Table space  ________ (optional)
   Name  . . . . . . . emp_____________________________________________ +

   Copy count  . . . . ALL_____        Number of rows to copy

From Template:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                    Enter "/", "A"lways to select option
   3  1. Above                          Edit template
      2. Previous                       Copy "From" values to "To" panel
      3. Generate from table
      4. Generate/Replace
```

Figure 25-54 Utility Functions panel

Provide information about the owner and name of the object to be copied. You can also use a template for selecting which records to copy. Select the number of rows to be copied. When you have completed the information on this panel, press Enter. The Copy Utility panel is displayed again as shown in Figure 25-55.

```
 Process   Options   Utilities   Help
FMN2PCPI 81H)                    Copy Utility                42 row(s) copied
Command ===> ____________________________________________________________

From DB2 Object:
   Location  . . . . . _____________          Database . . ________ (optional)
   Owner . . . . . . . DSN8810_____  +       Table space  ________ (optional)
   Name  . . . . . . . EMP_____________________________________________ +

   Copy count  . . . . ALL_____        Number of rows to copy

From Template:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                    Enter "/", "A"lways to select option
   3  1. Above                          Edit template
      2. Previous                       Copy "From" values to "To" panel
      3. Generate from table
      4. Generate/Replace
```

Figure 25-55 Copy complete

After pressing Enter, the Copy Utility panel is displayed with information about how many rows have been copied.

25.8.3 Objects

To display and process the DB2 object list select 4 from the Utility Functions panel as shown in Figure 25-56.

:

```
 Process   Options   Utilities   Help
FMN2STUT 81H)                Utility Functions
Command ===> 4

1  Print          Print DB2 table or view
2  Objects        Create and drop DB2 objects
3  Copy           Copy data within DB2
4  Object List    Display and process DB2 object lists
5  Privileges     Manage DB2 privileges
6  Import         Import sequential or VSAM data into DB2
7  Export         Export DB2 data to sequential or VSAM data set
8  Create         Create DB2 test data
9  Utilities      DB2 utility job generation
10 Audit trail    Print audit trail report
11 Print browse   Browse FM/DB2 print data set
```

Figure 25-56 Select Object List

The *Object List Utility* panel is now displayed, as shown in Figure 25-57.

```
 Process   Options   Utilities   Help
FMN2POL D81H)              Object List Utility
Command ===>
                                                              More:     +
   blank Display object list                P Print object list

Object Identification Criteria:
   Location  . . . . . . . _____________    Enter * for list
   Owner . . . . . . . . . _____________  +
   Name  . . . . . . . . . ____________________________________________ +
   Database/collect/schema ____________________________________________ +

Additional Selection Criteria:
 . . . . . . . . . . . . . . . . . . . . . . Enter * to list catalog columns
   Column  . . . . . . . . ____________________________
   Operator  . . . . . . . ___________
   Value . . . . . . . . . ____________________________________________ +

Processing Options:
   Object Type                                 Enter ? to list all object types
   ?    1. Database          8.  Package           15. Trigger
        2. Table space       9.  Collection        16. Storage group
        3. Table/view/alias  10. DBRM              17. Sequence
```

Figure 25-57 Object List Utility panel

Provide the following information:

- The name of the object to list or print. Wildcards are allowed to get a list of objects.
- The type of the object. The default object type is 3 (table/view/alias). By entering a question mark (?) you can get a scrollable list of valid object types.

List all tables (object type 3) starting with EMP* for owner AMINTOR, as shown in Figure 25-58.

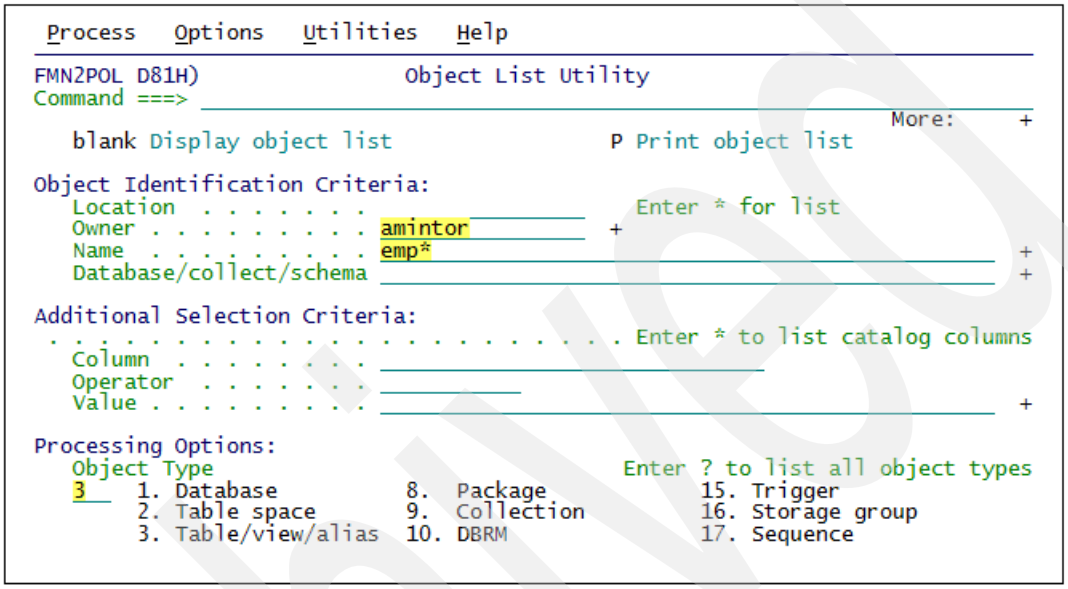


Figure 25-58 List all tables starting with EMP for owner AMINTOR*

As shown in Figure 25-59, there are five tables found for the criteria. You can scroll to the right to see more information about the objects.

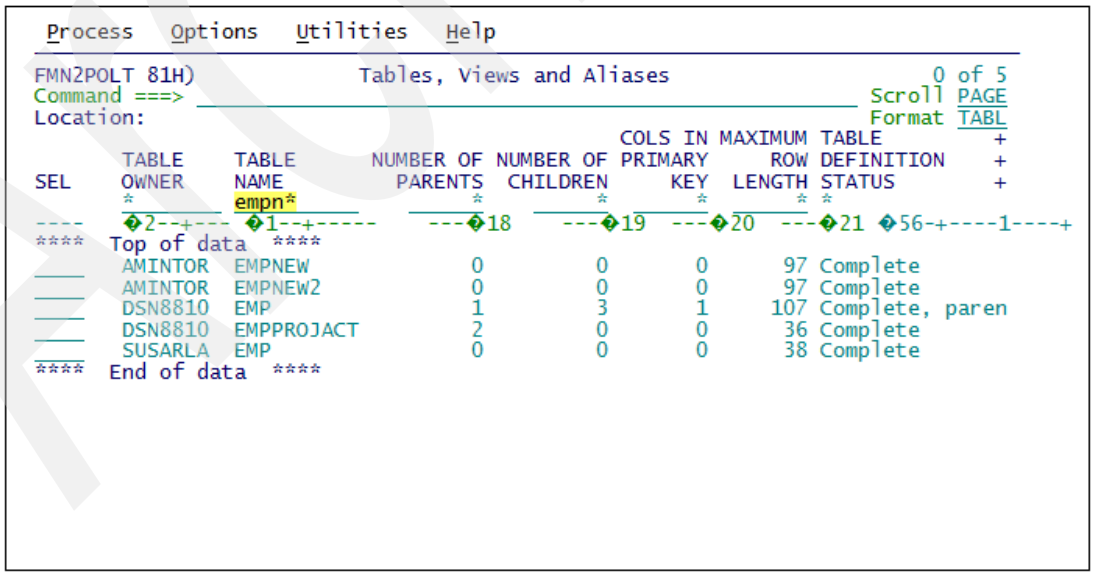


Figure 25-59 Five tables found for the criteria. We can use wildcard to filter the list

If there are a lot of objects displayed, you can use a wildcard to receive a new list, as shown above. Under the header TABLE NAME, enter `EMPN*`.

After entering the wildcard pattern the panel displays, as shown in Figure 25-60.

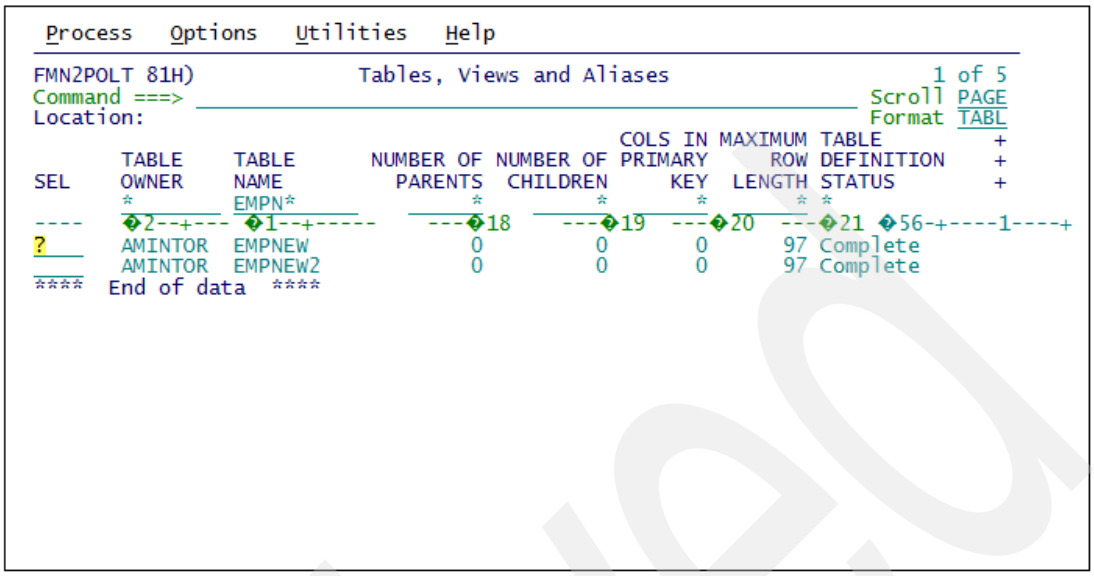


Figure 25-60 After using the wildcard

As we can see above, there is a Prefix Command Area (SEL column) were we can enter the object list line commands. By entering a question mark (?) in the Prefix Area you can get to the Object Type List utility, as shown in Figure 25-61.

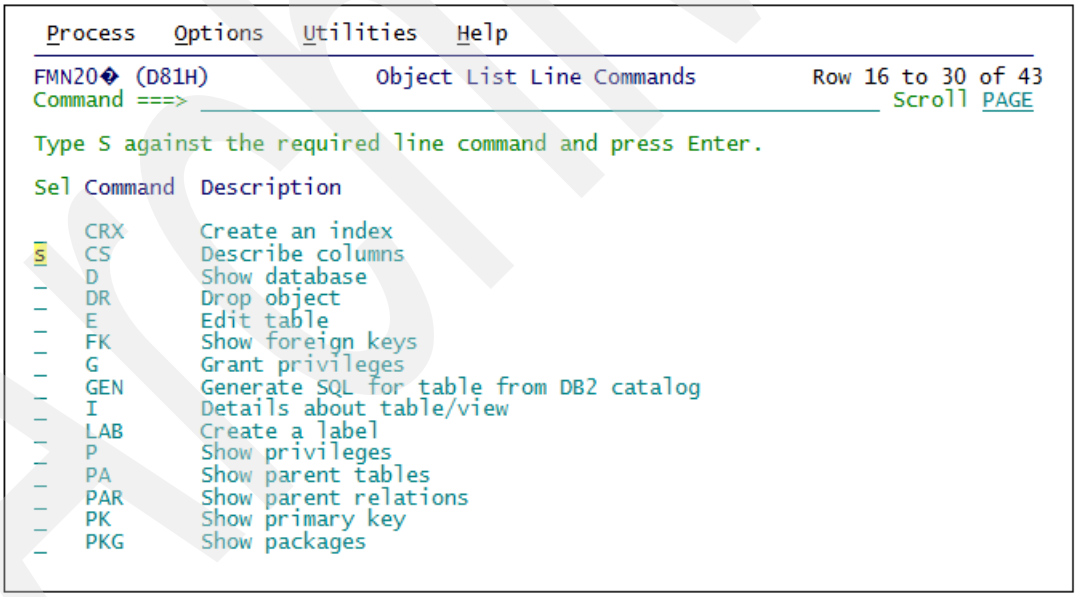


Figure 25-61 Prefix Command area were we can enter the object list line commands

As shown above, select the option Describe Columns and press Enter. The columns for AMINTOR.EMPNEW is displayed, as shown in Figure 25-62.

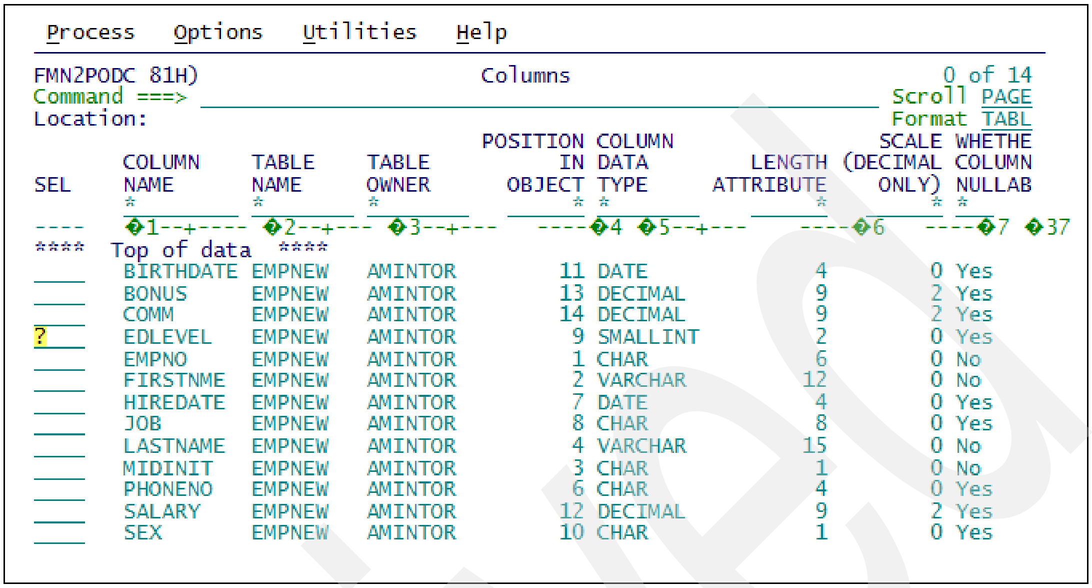

Process Options Utilities Help
FMN2PODC 81H) Columns 0 of 14
Command ===> Scroll PAGE
Location: Format TABL
SEL COLUMN NAME TABLE NAME TABLE OWNER POSITION IN OBJECT COLUMN DATA TYPE LENGTH ATTRIBUTE SCALE (DECIMAL ONLY) WHETHE COLUMN NULLAB
**** Top of data ****
BIRTHDATE EMPNEW AMINTOR 11 DATE 4 0 Yes
BONUS EMPNEW AMINTOR 13 DECIMAL 9 2 Yes
COMM EMPNEW AMINTOR 14 DECIMAL 9 2 Yes
? EDLEVEL EMPNEW AMINTOR 9 SMALLINT 2 0 Yes
EMPNO EMPNEW AMINTOR 1 CHAR 6 0 No
FIRSTNME EMPNEW AMINTOR 2 VARCHAR 12 0 No
HIREDATE EMPNEW AMINTOR 7 DATE 4 0 Yes
JOB EMPNEW AMINTOR 8 CHAR 8 0 Yes
LASTNAME EMPNEW AMINTOR 4 VARCHAR 15 0 No
MIDINIT EMPNEW AMINTOR 3 CHAR 1 0 No
PHONENO EMPNEW AMINTOR 6 CHAR 4 0 Yes
SALARY EMPNEW AMINTOR 12 DECIMAL 9 2 Yes
SEX EMPNEW AMINTOR 10 CHAR 1 0 Yes

Figure 25-62 Column display

25.8.4 Generating DDL Statement

When you select GEN (in Prefix Area) from the previous Object List Line Command panel, you receive the following panel as shown in Figure 25-63.

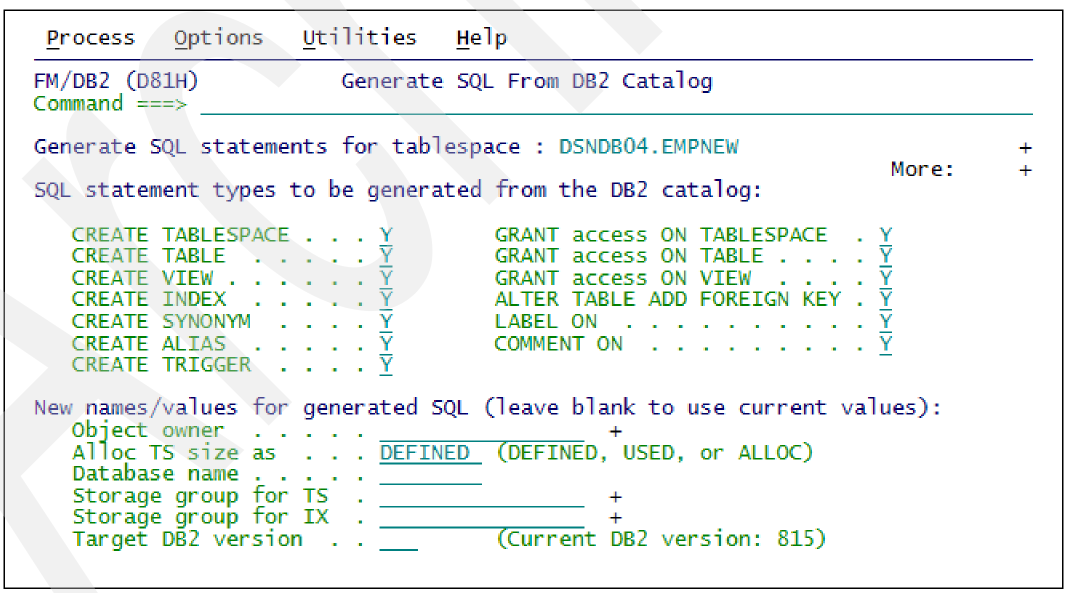

Process Options Utilities Help
FM/DB2 (D81H) Generate SQL From DB2 Catalog
Command ===>
Generate SQL statements for tablespace : DSNDB04.EMPNEW +
More: +
SQL statement types to be generated from the DB2 catalog:
CREATE TABLESPACE . . . Y GRANT access ON TABLESPACE . Y
CREATE TABLE . . . . . . Y GRANT access ON TABLE . . . . Y
CREATE VIEW . . . . . . Y GRANT access ON VIEW . . . . Y
CREATE INDEX . . . . . . Y ALTER TABLE ADD FOREIGN KEY . Y
CREATE SYNONYM . . . . . Y LABEL ON . . . . Y
CREATE ALIAS . . . . . . Y COMMENT ON . . . . Y
CREATE TRIGGER . . . . . Y
New names/values for generated SQL (leave blank to use current values):
Object owner . . . . . . +
Alloc TS size as . . . DEFINED (DEFINED, USED, or ALLOC)
Database name . . . . . .
Storage group for TS . +
Storage group for IX . +
Target DB2 version . . (Current DB2 version: 815)

Figure 25-63 Generate SQL from DB2 Catalog panel

On this panel, you specify how the generation of SQL from the DB2 catalog should work. You can specify which SQL statement types FM/DB2 should generate, and optionally which names and values should be changed. Press enter and the following JCL is generated to create and export your DDL definitions for another DB2 Database as shown in Example 25-1.

Example 25-1 DDL statement for the DB2 object selected via previous Panel

```
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (A),
000002 //             AMINTOR,MSGCLASS=A,
000003 //             NOTIFY=AMINTOR,CLASS=A,
000004 //             MSGLEVEL=(1,1)
000005 //*
000006 //*********************************************************************
000007 //*
000008 //* FILE MANAGER DB2 GENERATED JOB
000009 //*
000010 //*********************************************************************
000011 //*
000012 //*********************************************************************
000013 //* STEP GENSQL: GENERATE SQL FROM DB2 CATALOG
000014 //*********************************************************************
000015 //GENSQL EXEC PGM=IKJEFT01,DYNAMNBR=100,REGION=8M
000016 //STEPLIB  DD DSN=SYS1.D8OH.SDSNEXIT,DISP=SHR
000017 //         DD DSN=SYS1.D81H.SDSNLOAD,DISP=SHR
000018 //SYSTSPRT DD SYSOUT=*
000019 //SYSTSIN  DD *
000020   DSN SYSTEM(D81H)
000021   RUN PROG(FMN2GEN) PLAN(FMN2GEN)
000022   END
000023 //SYSPRINT DD SYSOUT=*
000024 //SQL      DD SYSOUT=*,DCB=(RECFM=FB,LRECL=80)
000025 //IN       DD *
000026 db2sys   = 'D81H',
000027 db2aloc  = '',
000028 db2serv  = '',
000029 db2auth  = 'AMINTOR',
000030 db2rel   = '815',
000031 gendb    = '',
000032 gents    = 'Y',
000033 gentable = 'Y',
000034 genview  = 'Y',
000035 genindex = 'Y',
000036 gensyn   = 'Y',
000037 genalias = 'Y',
000038 genlabel = 'Y',
000039 gencomm  = 'Y',
000040 genrels  = 'Y',
000041 gentrig  = 'N',
000042 grantdb  = '',
000043 grantts  = 'Y',
000044 granttab = 'Y',
000045 grantvw  = 'Y',
000046 newdb    = '',
000047 newtssg  =
000048 '',
000049 newixsg  =
000050 '',
000051 newsqlid  =
000052 '',
000053 spcalloc = 'DEFINED',
000054 tgtdb2   = '';
000055 db='DSNDB04',
000056 ts='EMPNEW',
000057 own=
000058 '',
```

```
000059 tb=
000060 '';
000061 /*
000062 //*
000063 //* This JCL created on 06/11/14 at 17:46 by AMINTOR at STLABF6
000064 //*
****** *************************** Bottom of Data ****************************
```

25.8.5 Printing a DB2 table

To print a table or part of a table, use the Print function, as shown in Figure 25-64.

```
 Process   Options   Utilities   Help
FMN2STUT 81H)                Utility Functions
Command ===> 1

1  Print          Print DB2 table or view
2  Objects        Create and drop DB2 objects
3  Copy           Copy data within DB2
4  Object List    Display and process DB2 object lists
5  Privileges     Manage DB2 privileges
6  Import         Import sequential or VSAM data into DB2
7  Export         Export DB2 data to sequential or VSAM data set
8  Create         Create DB2 test data
9  Utilities      DB2 utility job generation
10 Audit trail    Print audit trail report
11 Print browse   Browse FM/DB2 print data set
```

Figure 25-64 Utility Functions panel

After selecting option **1**, the Print Utility panel is displayed as shown in Figure 25-65.

```
 Process   Options   Utilities   Help
FMN2PDBP 81H)                  Print Utility
Command ===>

DB2 Object:
   Location  . . . . . fqinsur _________       Database . . _________ (optional)
   Owner . . . . . . . customer _________ +    Table space  _________ (optional)
   Name  . . . . . . . ____________________________________________ +

   Row count . . . . . ALL_____       Number of rows to print

DB2 Template:
   Data set name . . . amintor.book2006.template_____________________
   Member  . . . . . . db2temp1

Processing Options:
   Print Mode       Template usage              Enter "/", "A"lways to select
   1  1. Table      1  1. Above                    Edit template
      2. Single        2. Previous                 Batch execution
                       3. Generate from table      Use uncommitted read
                       4. Generate/Replace         Print HEX representation
```

Figure 25-65 Print Utility Panel

For this example, you do the following operations:

- Print the CUSTOMER table.
- Print all rows.
- Use the template TDB2TEMP1 to print some of the columns as defined in the template.

After pressing Enter, the table is printed and a message appears displaying how many rows were printed, as shown in Figure 25-66.

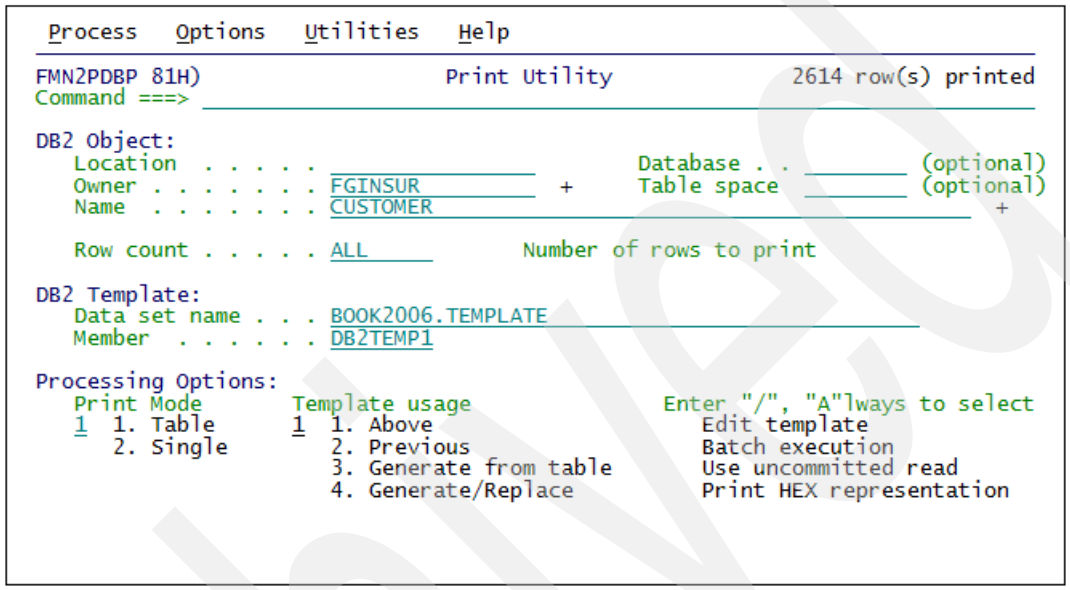


Figure 25-66 Print finished

The printout dataset is shown in Example 25-2.

Example 25-2 JCL Printout generated from "Print utility" panel

```
. . . . . .
 SDSF EDIT    AMINTORC (JOB03960) JCLEDIT                  Columns 00001 00072
 Command ===>                                                Scroll ===> CSR
 ****** ***************************** Top of Data ******************************
 000001 //AMINTORC  JOB '*',
 000002 //            AMINTOR,       **JOB STATEMENT GENERATED BY SUBMIT**
 000003 //            NOTIFY=AMINTOR,
 000004 //            MSGLEVEL=(1,1)
 000005 //* JCL TO PRINT AND DELETE ISPF LIST DATASET
 000006 //PRINTLST EXEC PGM=IEBGENER
 000007 //SYSUT1   DD DSN=AMINTOR.SPF1.LIST,
 000008 //            DISP=(OLD,DELETE,KEEP)
 000009 //SYSUT2   DD SYSOUT=(A),
 000010 //            DCB=(AMINTOR.SPF1.LIST)
 000011 //SYSPRINT DD DUMMY
 000012 //SYSIN    DD DUMMY
 ****** **************************** Bottom of Data ****************************

******************************** TOP OF DATA *********************************
IBM File Manager for z/OS DB2 Component
DB2 SSID: D81H    SQL ID: AMINTOR     Location:         Table/View: "FGINSUR"."C
CUSTOMERNUMBER FIRSTNAME  LASTNAME             POSTCODE
       INTEGER CH(10)     CHARACTER(20)        CH(8)
   PU--+----1> <---+----> <---+----1----+----> <---+-->
           378 ESKWX      EADOPCYB             HL81 8UX
```

```
  371 HOX        OJKZIR             HL60 4UJ
  376 BBXOK      FFFLA              HL68 6HO
  372 BPLP       HOWSLQNT           HL10 1JX
  370 SZCFJNO    BXUZO              HL4  9HY
  375 BBXOK      SZWQA              HL06 7NQ
  377 ESKWX      HOWSLQNT           HL12 7QI
  373 HOWM       HOWSLQNT           HL26 5GA
  374 EMIMQ      BXUZO              HL66 5DP
  379 WMET       WMEHVL             HL8  4LQ
  337 SSME       EADOPCYB           HL85 4GW
  333 HOWM       SZWQA              HL14 5QI
  334 EMIMQ      HOWSLQNT           HL12 6RE
 9004 BPLP       SZWQA              1971-07-12  WMV APZYHJ
 9009 WMET       WMEHVL             1974-03-14  LRHU WHMED UXKB
 9055 HOWM       YWXHXUQE           1970-06-26  WMV APZYHJ
 9057 SZCFJNO    EADOPCYB           1954-04-08  YXDI NHQV
 9053 WYTWN      YWXHXUQE           1958-01-21  WMV APZYHJ
 9054 BPLP       FFFLA              1969-03-22  YXDI NHQV
 9059 HOX        EADOPCYB           1976-06-23  NQVV CCCJHPY
 9058 SSME       WMEHVL             1970-08-23  WMV APZYHJ
 9051 SSME       KNTLL              1958-02-01  YXDI NHQV
 9056 HOWM       BXUZO              1947-03-16  LRHU WHMED UXKB
 9052 BPLP       KNTLL              1969-08-03  NQVV CCCJHPY
 9050 WMET       BXUZO              1949-11-01  WMV APZYHJ
 9073 ESKWX      FFFLA              1941-11-25  LRHU WHMED UXKB
 9074 ESKWX      BXUZO              1940-02-18  YXDI NHQV
 9079 BPLP       FFFLA              1941-08-06  LRHU WHMED UXKB
 9078 HOWM       WMEHVL             1949-06-04  WMV APZYHJ
 9071 HOX        EADOPCYB           1943-06-16  WMV APZYHJ
2687 row(s) printed
******************************** BOTTOM OF DATA ********************************
```

26

File Manager/IMS

In this chapter we describe how to use File Manager to manipulate IMS data.

This chapter covers the following functions of File Manager/IMS:

- Setting the processing options
- Editing/browsing data with and without views
- Creating templates for DBD
- Creating and editing the view
- Examining primary commands and line commands that can be used in various panels

26.1 Starting with FM/IMS

This section demonstrates basic tasks that we can do to customize our own settings for FM/IMS. We cover the following items:

- System settings
- Compiler language specifications
- Edit and browse options
- DLI mode settings
- DLI mode datasets

26.2 Primary Option Menu

The Primary Option Menu, as shown in Figure 26-1, shows all options we can perform within FM/IMS. On the command line we type in which function we want to do.

On the Primary Option Menu we choose option **0** and press Enter, to get to the Set Processing Options.

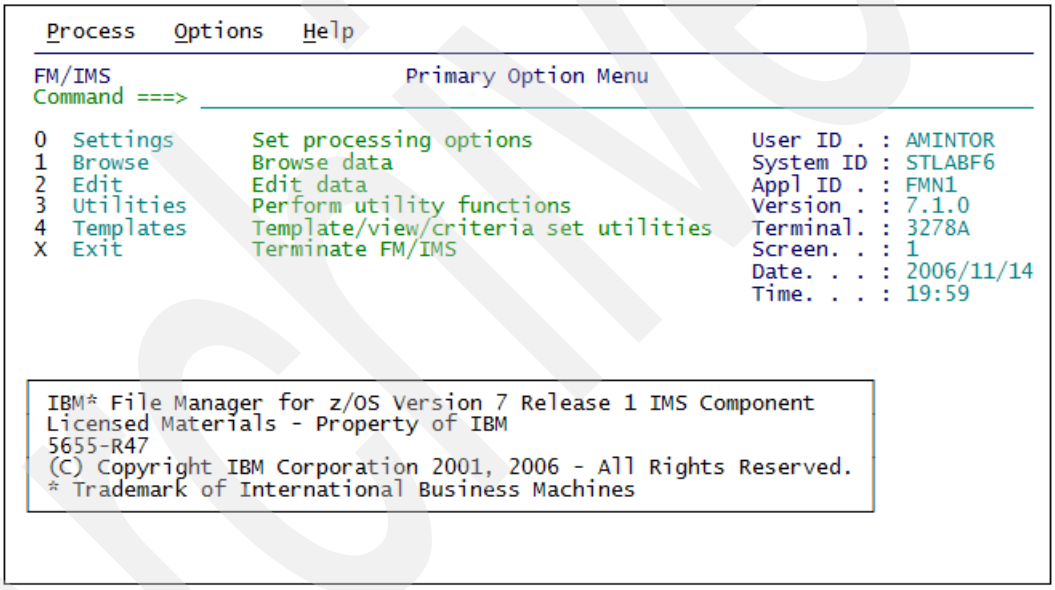


Figure 26-1 IMS Primary Options Menu

This takes us to the panel where we can set the processing options. The Set Processing Options menu appears, as shown in Figure 26-2. From here we choose option **2** (System settings) and press Enter.

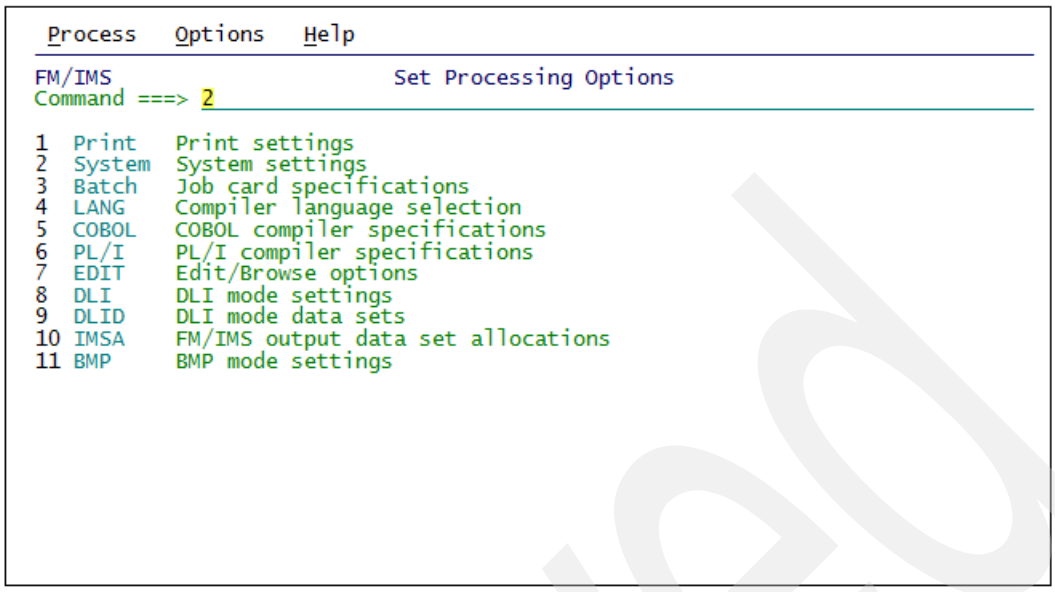


Figure 26-2 Set Processing Options panel

26.2.1 System settings

The System Settings panel, shown in Figure 26-3, allows us to set system-wide settings for:

- PSB processing
- Checkpoint frequency for both BMP and DLI mode
- Pad character
- High Level Qualifier of datasets created by FM/IMS

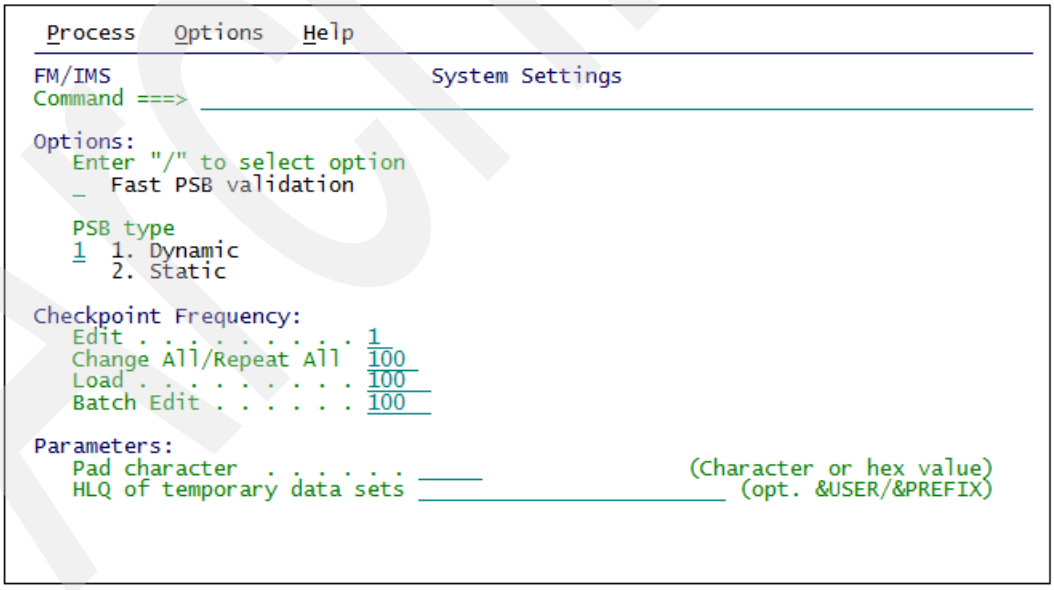


Figure 26-3 The system settings panel

Figure 26-4 shows that:

- We are using a dynamic PSB. FM/IMS generates the PSB when editing/browsing. FM/IMS can also use a static PSB, and when using that, the rules of the PSB control the access to the segments.
- Checkpoint frequency when editing is set to 1.

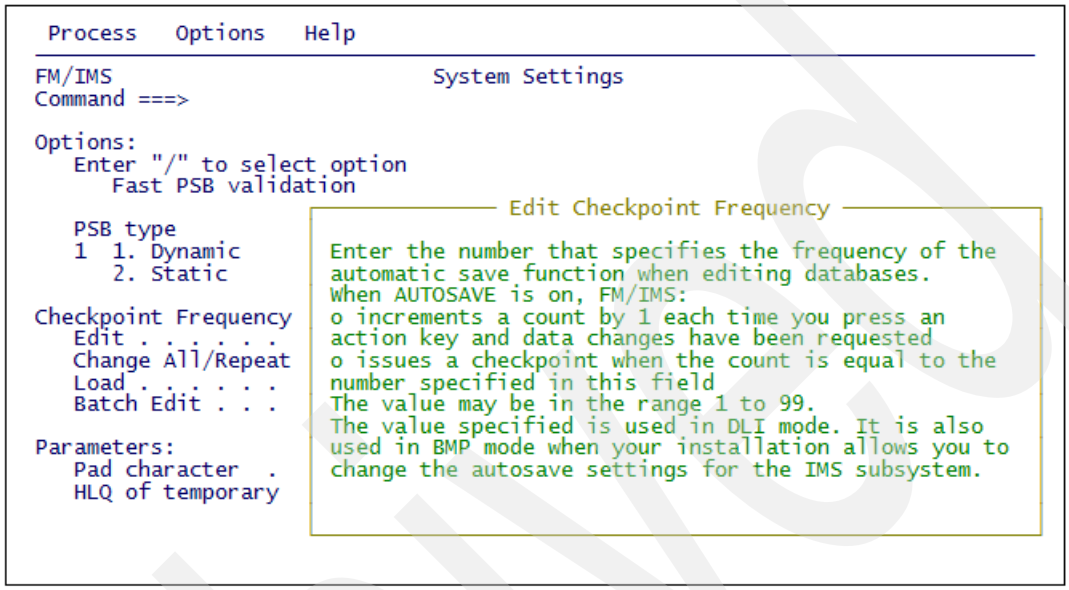


Figure 26-4 IMS help (PF1) - System Settings panel

In the area regarding the Frequency, the panel adds the field *Batch Edit Checkpoint Frequency*. Use this field to specify the following values:

- Enter the number that specifies the frequency of the automatic save function when running an FM/IMS Batch Edit job. FM/IMS increments a count by 1 each time your Batch Edit job inserts, deletes, or replaces a segment and issues a checkpoint when the count is equal to the number specified in this field.
- The value can be in the range 1 to 99,999.

To get help about individual fields in any panel, just place the cursor on any input field and press F1 and you get information from the built-in help function, as shown in Figure 26-4.

26.2.2 Compiler language selection

When creating templates, FM/IMS uses either the COBOL complier or the PL/I compiler.

To navigate to the Compiler Language Selection, we choose option **4** from the Menu Bar *Options* (or by using the Set Processing Options panel) shown in Figure 26-5.

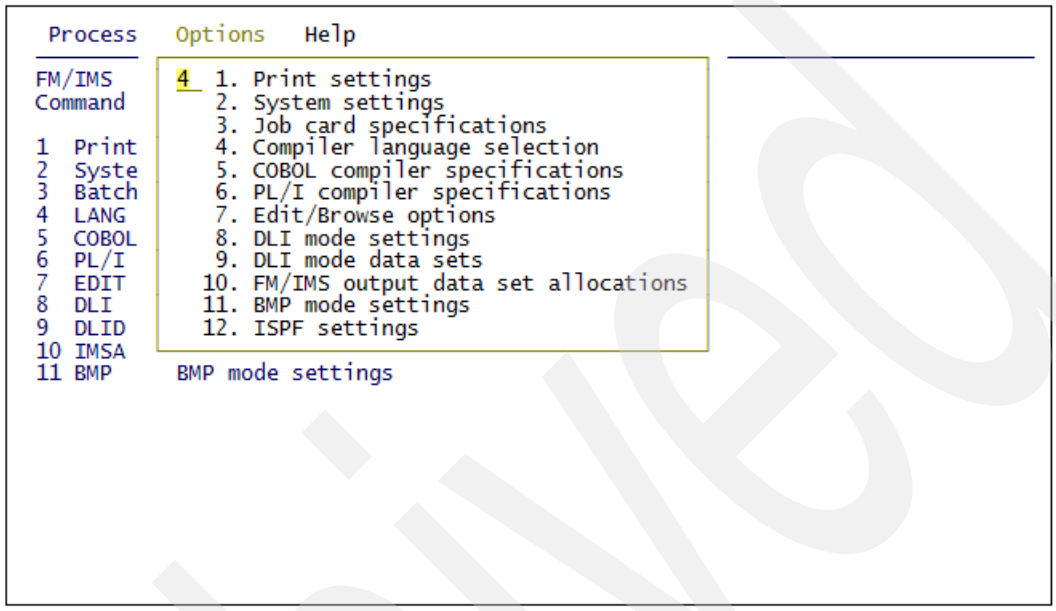


Figure 26-5 Use the menu bar to navigate through the panels

Figure 26-6 shows the Compiler Selection panel. FM/IMS use the COBOL compiler as the default. In this case we let FM/IMS detect whether it is a COBOL or PL/I copybook that is going to be used.

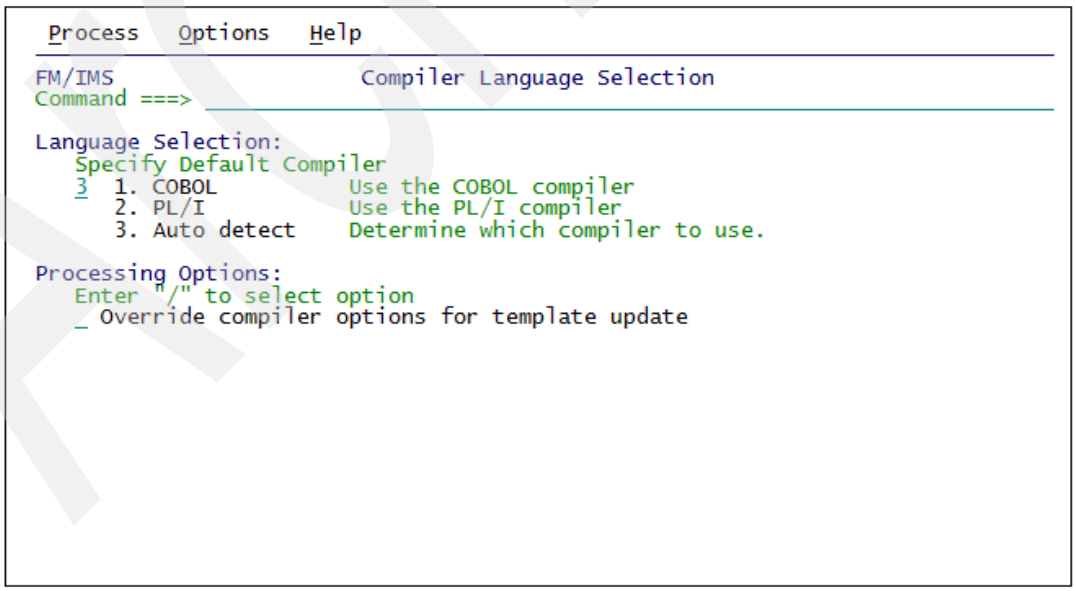


Figure 26-6 IMS Compiler Language Selection pane

Under *Processing Options*, if you use a / to set the field, *Override compiler options for template update*, the compiler options found in a Template that is being updated with the current compiler options are overridden.

Note: Foreground processing always overrides the compiler options for older templates that do not contain compiler options. This option should be flagged for batch template update of older Templates to generate compiler options in the JCL.

Older templates do not contain compiler options, but after the update is performed, the template is upgraded to contain compiler options to be used in subsequent updates if this option is not selected.

26.2.3 Edit/Browse Options

To navigate to the Edit/Browse Options panel, we choose option **7** on the Set Processing Options panel. See Figure 26-7.

```
 Process   Options   Help
FM/IMS                      Edit/Browse Options
Command ===> _____________________________________________________________

Related command if applicable, shown in ()                    Initial Display
Miscellaneous:           Enter "/" to select options          1  1. Previous
   CAPS initially ON - translate changed data to uppercase (CAPS)  2. Table
Expose (do not group) records of types: (SHOW)                    3. Single
   Suppressed                                                     4. Character
                                                                  5. Hex
                                                                  6. LHex
Record formatting options for SNGL display or print:
_  Field reference number (REF)    Field type and length values (TYPE)
   Picture clause         (PIC)    Start location               (SLOC)
   Structure              (STR)    Left justify numeric fields  (JUST)
/  Redefined fields       (RDF) (also affects TABL format)
Other:
/  Use * (any segment type) as default on NEXT and PREVIOUS commands
   Set SCOPE REC, limiting segments to within a database record
   Set AUTOSAVE ON, issuing a SAVE after checkpoint frequency in options 0.2
```

Figure 26-7 IMS Edit/Browse Options panel

You can use the Set Processing Options panel to control processing of the current FM/IMS session, t,o get detailed help for a specific processing option,

26.2.4 DLI mode settings

To navigate to the DLI mode settings we choose option **8** on the Set Processing Options panel, shown in Figure 26-8.

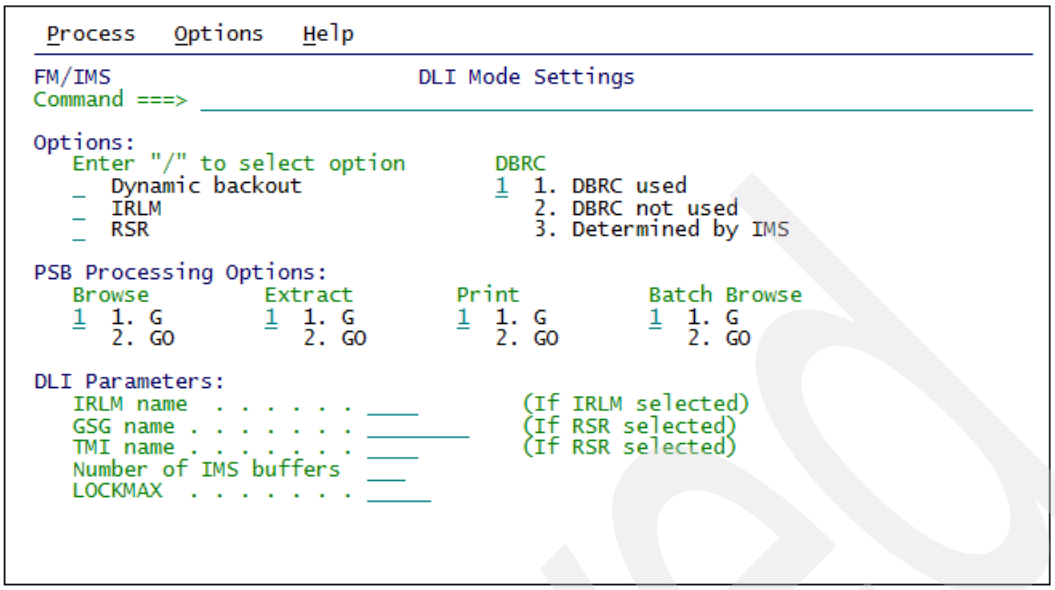


Figure 26-8 DLI Mode Settings panel

The panel above sets the parameters and processing options that become active when you are running your utility of DLI functions.

Note: The Dynamic back-out option only takes effect if you are running the FM/IMS function with an IMS log data set.

26.2.5 DLI mode datasets

To navigate to the DLI mode datasets, we choose option **9** on the Set Processing Options panel. As shown in Figure 26-9 here, we see the DLI Mode Data Sets panel. This panel displays and sets data set default names. FM/IMS uses the data sets specified on this panel when editing, browsing, extracting, and loading databases in DLI mode.

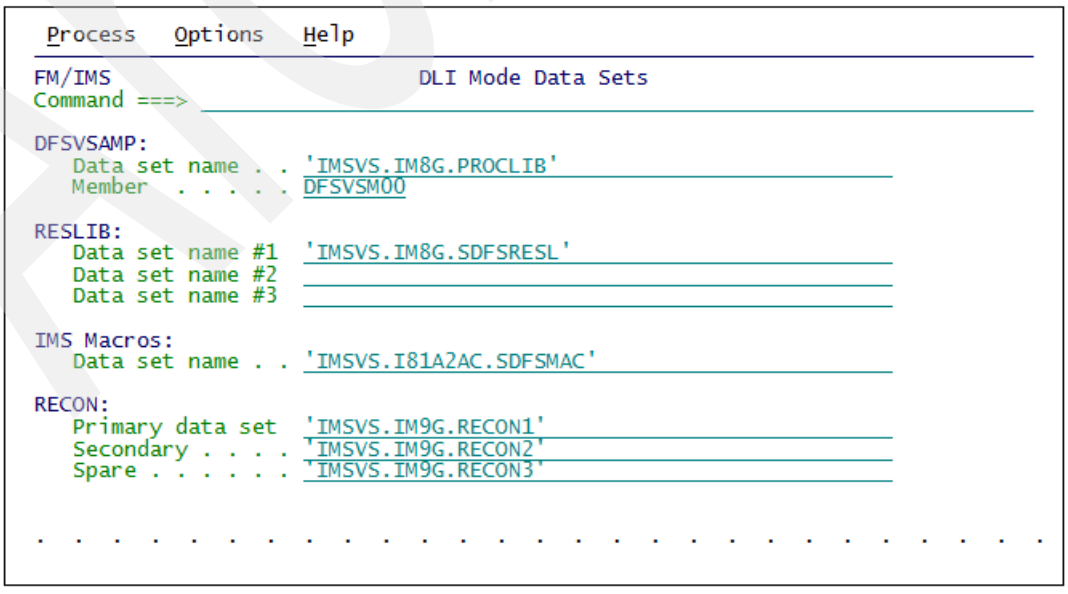


Figure 26-9 DLI Mode Data Sets panel

26.3 Edit/browse without using a view

Next we discuss editing and browsing without using a view.

26.3.1 Preparation

From the Primary Option Menu shown in Figure 26-10, we select option **1** (Browse data) and press Enter.

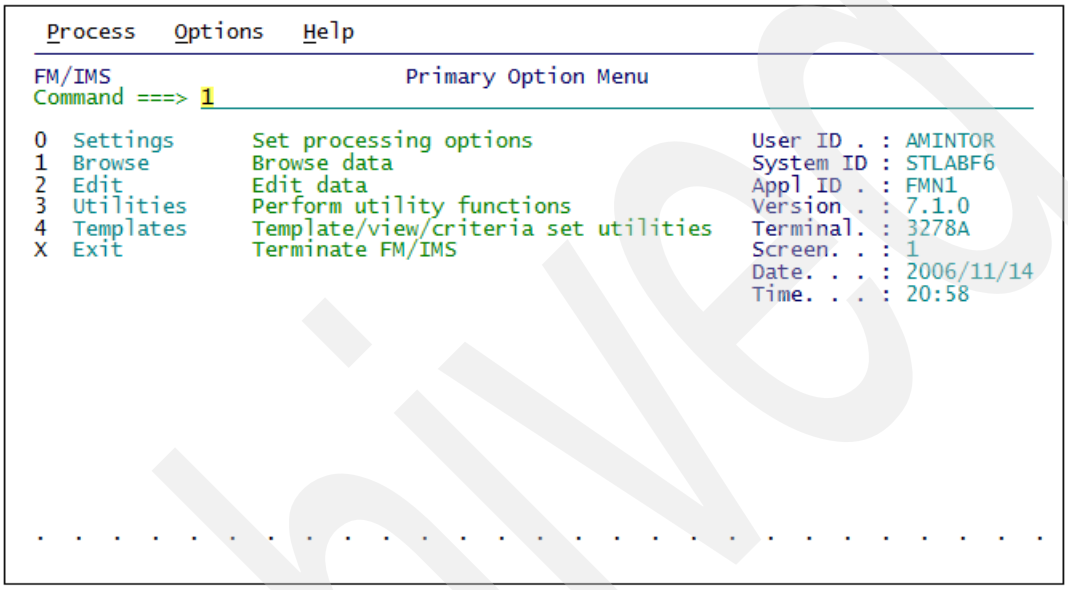


Figure 26-10 Primary Options Menu

FM/IMS now shows the Browse Entry Panel as shown in Figure 26-11. In this next panel, we can see that a dynamic PSB is to be used. We defined this in a previous step.

On this panel we specify:

- For the DBD:
  - Data set name 1: the DBD library (or libraries) that the database definitions are to be obtained from (in this example, CHABERT.IMS.TRADER.DBDLIB).
  - Member: the name of the database (in this example, DR1E). If we leave the field blank or enter a pattern, a member selection list is displayed. We can select a member from this list.
- Processing options:

  When accessing the database, we can do that in two modes:

  - DL/I mode

    When the database is offline, we access it in DLI mode. In DLI mode, we must specify the data sets that we want to update, and security is controlled at the data set level (that is, if we have access to the data sets via ISPF, we can access them via FM/IMS in DLI mode). These data sets are allocated to the TSO address space and we manually control the environment by using the Set Options panels. This type of access is typically used by developers. While DLI mode usually provides single user access, we can share databases with other users by working with the IRLM option turned on (depending on your IMS subsystems definitions).

– BMP mode:

  When the database is allocated to an online system, we access it in BMP mode. BMP mode provides strict controls that are suited to live production environments. For example, we can only access those database data sets that have been defined for the IMS subsystem. Any changes that we make are logged in the IMS online log data set and IMS controls such things as multi-user access to the segment data and automatic backout if necessary.

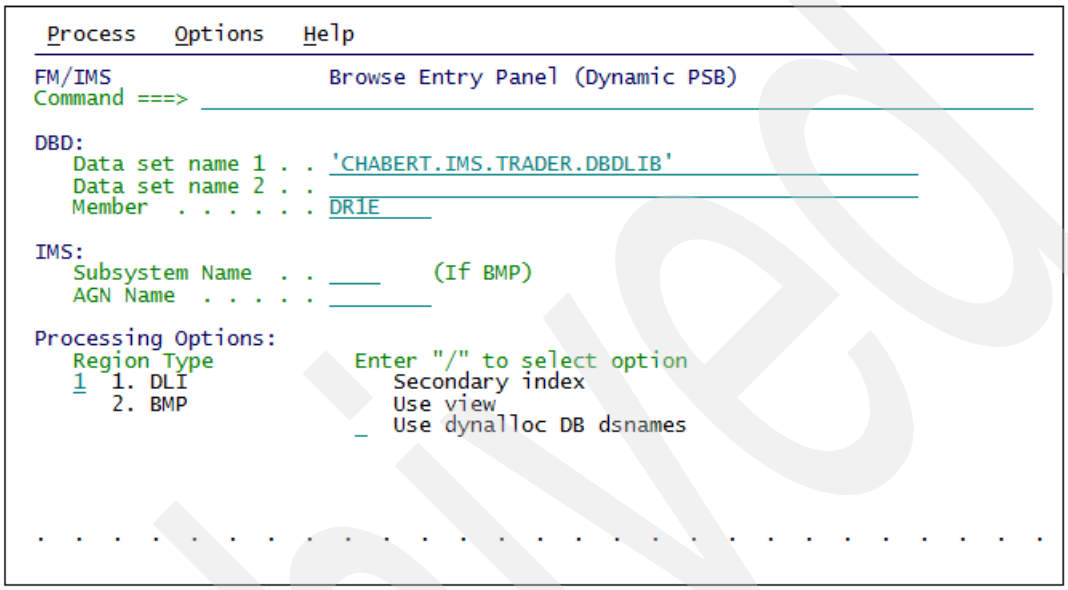


Figure 26-11 Browse Entry Panel

After completing the information in the Browse Entry Panel, we press Enter, and the Database Data Set Specification panel, as shown in Figure 26-12, is shown. Here we enter the data set names for the DBD. We can also enter a slash (*/*) to choose to save these names in our profile to be reused next time we access this DBD.

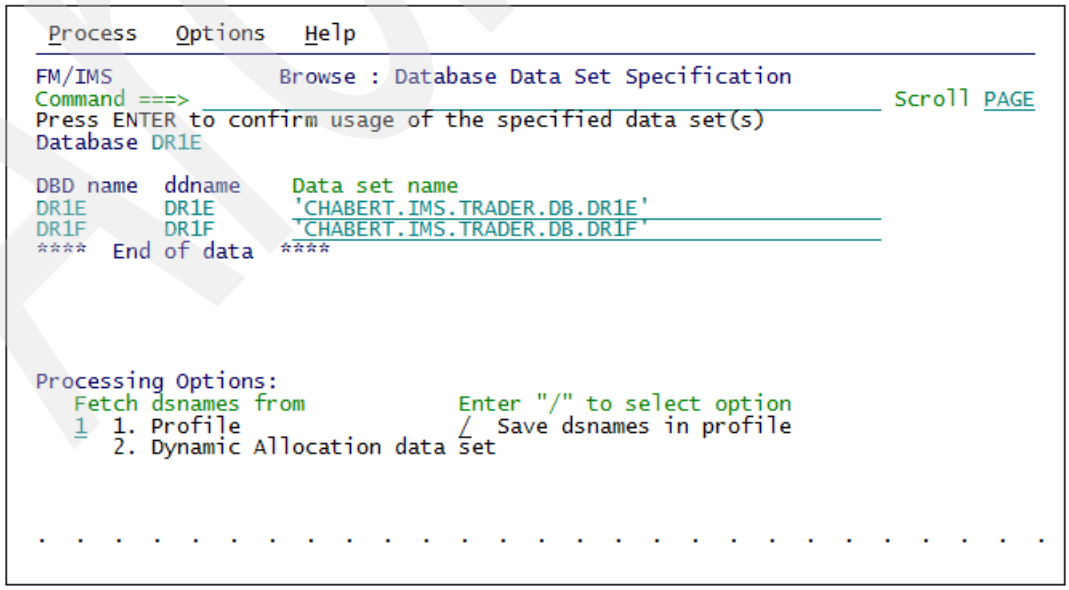


Figure 26-12 Database Data Set Specification panel

26.3.2 Database positioning

When we have finished the settings as just described, press Enter, and the Database Positioning panel is displayed, as shown in Figure 26-13.

```
 Process   Options   Help
FM/IMS                  Browse : Database Positioning
Command ===> _____________________________________________ Scroll PAGE

Database DR1E                        Key sequence                Format CHAR
View     None                              Key
Cmd SXE Level     Segment   Description     len  Key value
___ SX  1         CUSTOMER                   25  AAB
___  X   2        COMPANY                    20  ....................
___ ****  End of data  ****
```

Figure 26-13 Database Positioning panel

Figure 26-14 through Figure 26-19 show the tutorial describing all of the fields on this panel. Put the cursor on the Prefix Area (Cmd field) and press PF11 to display the FM/IMS Tutorial.

```
                        FM/IMS Tutorial - Browse
Command ===> ____________________________________________________________
                                                              More:     +
Database Positioning

This panel is used to specify the starting point of the data to display and
the display format to start with.

For a dynamic PSB Edit/Browse dialog, the Database Positioning panel displays
a list of all the segment types in the selected DBD.

For a static PSB Edit/Browse dialog, the Database Positioning panel displays a
list of all the segment types in the selected PCB.

Below is an overview of the fields displayed or required on this panel, and
the line commands available.

   Database
      This displays the name of the DBD entered on the Edit entry or Browse
      entry panel. This line will also contain Key sequence for the following
      types of database:
      HSAM, SHSAM, HISAM, SHISAM, HIDAM, GSAM, INDEX, MSDB.
```

Figure 26-14 Tutorial (part 1 of 6)

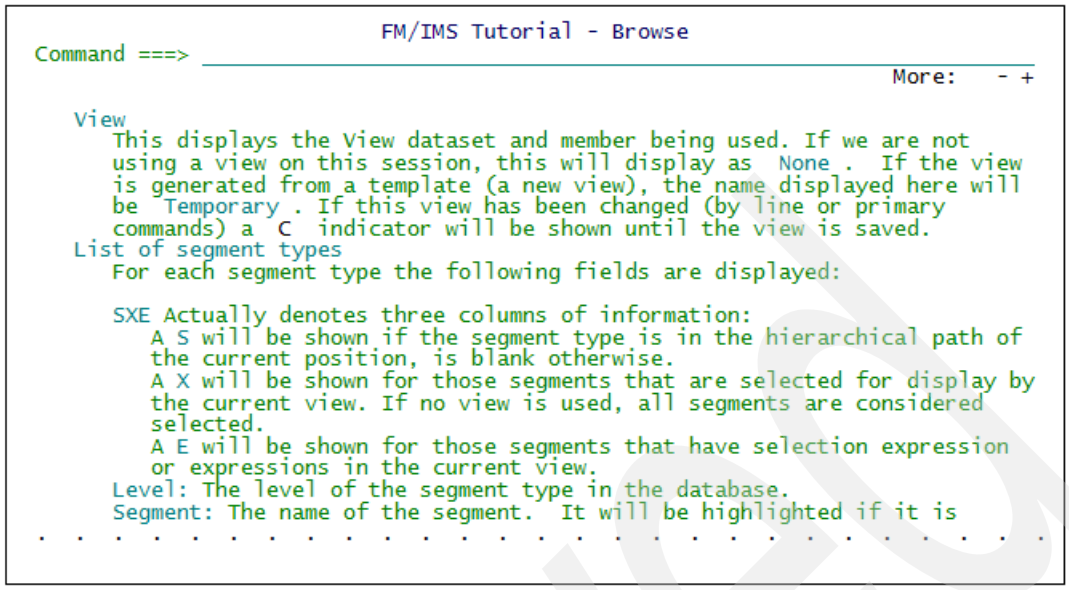


Figure 26-15 Tutorial (part 2 of 6)

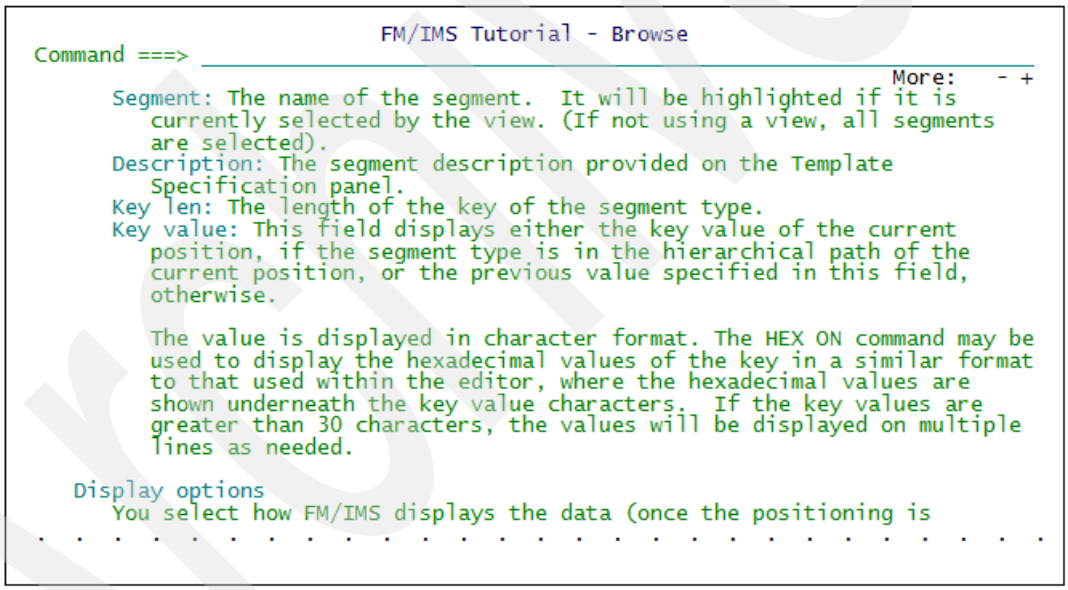


Figure 26-16 Tutorial (part 3 of 6)

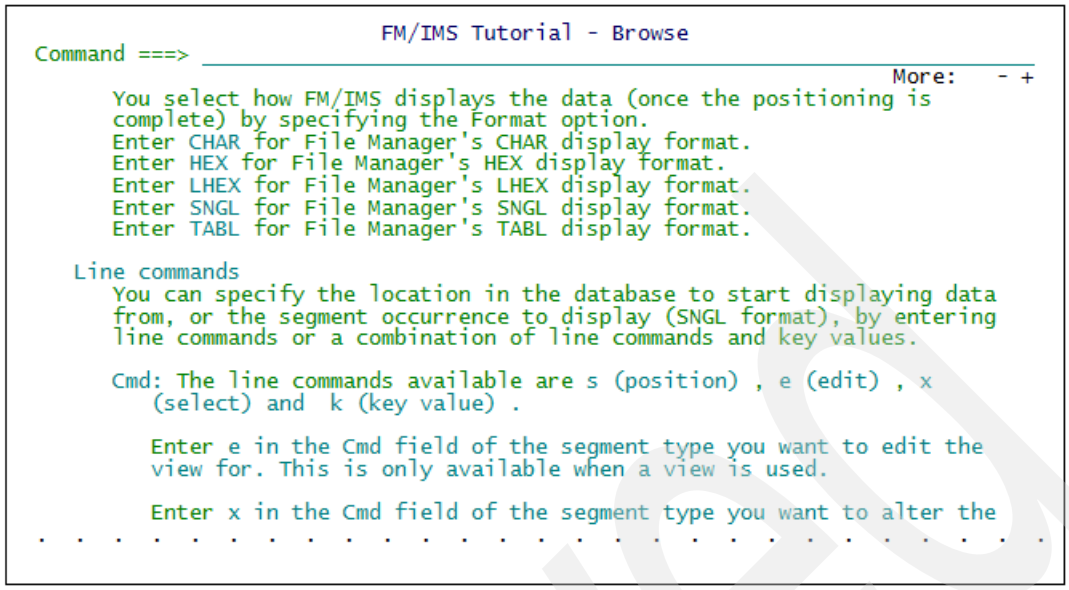


Figure 26-17 Tutorial (part 4 of 6)

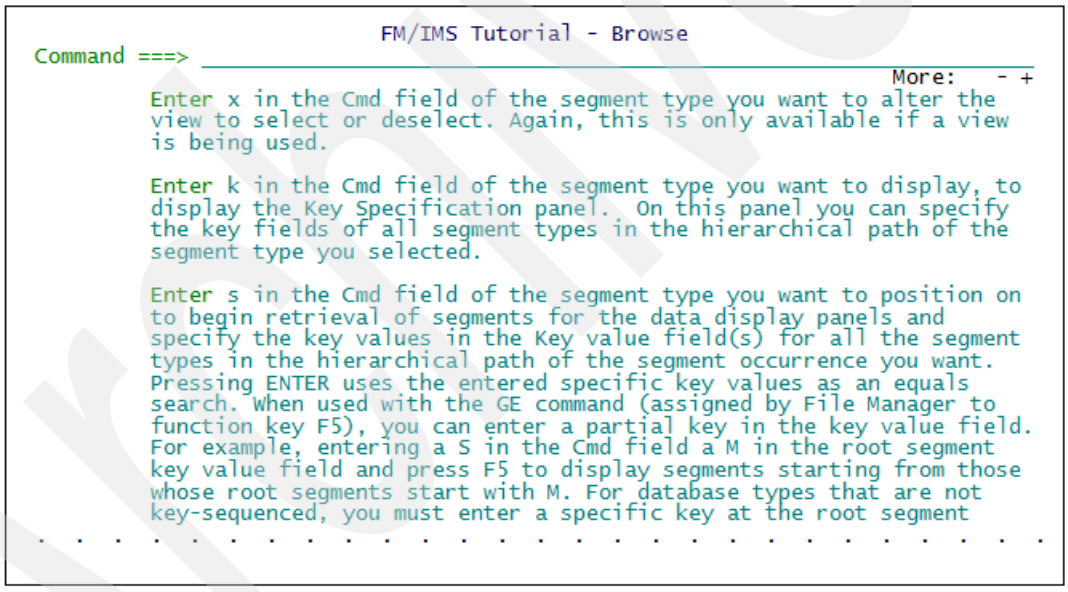


Figure 26-18 Tutorial (part 5 of 6)

```
                        FM/IMS Tutorial - Browse
Command ===> ____________________________________________________________
                                                                More:    -
         display the Key Specification panel.  On this panel you can specify
         the key fields of all segment types in the hierarchical path of the
         segment type you selected.

         Enter s in the Cmd field of the segment type you want to position on
         to begin retrieval of segments for the data display panels and
         specify the key values in the Key value field(s) for all the segment
         types in the hierarchical path of the segment occurrence you want.
         Pressing ENTER uses the entered specific key values as an equals
         search. When used with the GE command (assigned by File Manager to
         function key F5), you can enter a partial key in the key value field.
         For example, entering a S in the Cmd field a M in the root segment
         key value field and press F5 to display segments starting from those
         whose root segments start with M. For database types that are not
         key-sequenced, you must enter a specific key at the root segment
         level (even when using the GE command), but you may enter a partial
         key at lower levels.

         If any of the key values fields in the hierarchical path are left
         blank, FM/IMS formats an unqualified SSA at that level.
. . . . . .
```

Figure 26-19 Tutorial (part 6 of 6)

We now enter an **S** in the Cmd field in front of the segment type on which we want to position, to begin retrieval of segments, as shown in Figure 26-20.

```
 Process   Options   Help
 FM/IMS                 Browse : Database Positioning
 Command ===> ____________________________________________ Scroll PAGE

 Database DR1E                        Key sequence              Format CHAR
 View     None                              Key
 Cmd SXE Level     Segment   Description    len  Key value
 s__  SX  1        CUSTOMER                 25  AAB_____________________
 ___   X  2        COMPANY                  20  ....................
 ___ ****  End of data  ****

 . . . . . .
```

Figure 26-20 IMS Database Positioning panel

26.3.3 Browsing a database

After we have specified a starting position within the database on the Database Positioning or Key Specification panel, the IMS Database panel is displayed, as shown in Figure 26-21.

As shown, we get a lot of information, not just the actual data:

- Cmd: Where we can type different line commands
- Level: The hierarchical level of the segment
- Scope: The current settings of the scope option, either DB or REC
- Col: The starting column of the segment data displayed, which can be overtyped to start showing the data from any position in the record
- Format: CHAR HEX LHEX

```
 Process    Options    Help
FM/IMS                        Browse : IMS Database DR1E
Command ===> _____________________________________________ Scroll PAGE
                                          Scope DB  Col 1_____ Format CHAR
Cmd Level Segment    ----+----1----+----2----+
                     ****   Top of window   ****
___ 1     CUSTOMER   AAB
___  2    COMPANY                      122
___  2    COMPANY    AAB               313B
___  2    COMPANY    IBM               114A
___ 1     CUSTOMER   ABBA CANNING COMP
___  2    COMPANY                      X02
___  2    COMPANY    KAVIAR            3A3B
___  2    COMPANY    NEW COMPANY       123B
___ 1     CUSTOMER   ALBANE
___  2    COMPANY    Casey_Import_Export ....
___  2    COMPANY    Glass_and_Luget_plc ....
___  2    COMPANY    Headworth_Electrical....
___  2    COMPANY    IBM                 ....
___  2    COMPANY    ShareSelect         ....
___  2    COMPANY    Veck_Transport      ...C
___ 1     CUSTOMER   B NEW
. . . . . .
```

Figure 26-21 IMS Database panel

Changing the display format of the data

By typing over the CHAR in the Format field with **HEX** or just an **H**, the data is displayed as shown in Figure 26-22.

```
 Process    Options    Help
FM/IMS                      Browse : IMS Database DR1E
Command ===> _____________________________________________ Scroll PAGE
                                        Scope DB  Col 1     Format hHAR
Cmd Level Segment    ----+----1----+----2----+
                     ****   Top of window   ****
___ 1     CUSTOMER   AAB
___  2    COMPANY                      122
___  2    COMPANY    AAB               313B
___  2    COMPANY    IBM               114A
___ 1     CUSTOMER   ABBA CANNING COMP
___  2    COMPANY                      X02
___  2    COMPANY    KAVIAR            3A3B
___  2    COMPANY    NEW COMPANY       123B
___ 1     CUSTOMER   ALBANE
___  2    COMPANY    Casey_Import_Export ....
___  2    COMPANY    Glass_and_Luget_plc ....
___  2    COMPANY    Headworth_Electrical....
___  2    COMPANY    IBM                 ....
___  2    COMPANY    ShareSelect         ....
___  2    COMPANY    Veck_Transport      ...C
___ 1     CUSTOMER   B NEW
. . . . . .
```

Figure 26-22 Browse Panel IMS Database

Changing current position in the database

If we wish to change the current position in the database, we enter a **K** in the Cmd field, as shown in Figure 26-23.

.

```
 Process    Options    Help
FM/IMS                      Browse : IMS Database DR1E
Command ===> _____________________________________________ Scroll PAGE
                                        Scope DB  Col 1     Format 1___
Cmd Level Segment     - - - - + - - - - 1 - - - - + - - - - 2 - - - - +
                     ****   Top of window   ****
k__ 1     CUSTOMER   C1C1C2404040404040404040404040404040404040404040
___  2    COMPANY    404040404040404040404040404040404040404040F1F2F240
___  2    COMPANY    C1C1C2404040404040404040404040404040404040F3F1F3C2
___  2    COMPANY    C9C2D4404040404040404040404040404040404040F1F1F4C1
___ 1     CUSTOMER   C1C2C2C140C3C1D5D5C9D5C740C3D6D4D74040404040404040
___  2    COMPANY    404040404040404040404040404040404040404040E7F0F240
___  2    COMPANY    D2C1E5C9C1D9404040404040404040404040404040F3C1F3C2
___  2    COMPANY    D5C5E640C3D6D4D7C1D5E84040404040404040404040F1F2F3C2
___ 1     CUSTOMER   C1D3C2C1D5C5404040404040404040404040404040404040404040
___  2    COMPANY    C381A285A86DC994979699A36DC5A7979699A34000000000
___  2    COMPANY    C79381A2A26D8195846DD3A48785A36D9793834000000000
___  2    COMPANY    C8858184A69699A3886DC5938583A39989838193000000000
___  2    COMPANY    C9C2D4404040404040404040404040404040404040000000000
___  2    COMPANY    E288819985E285938583A34040404040404040404000000000
___  2    COMPANY    E58583926DE3998195A2979699A34040404040400000001BC3
___ 1     CUSTOMER   C240D5C5E64040404040404040404040404040404040404040
. . . . . .
```

Figure 26-23 IMS data panel in HEX forma

After pressing Enter, the Key Specification panel is displayed, as shown in Figure 26-24.

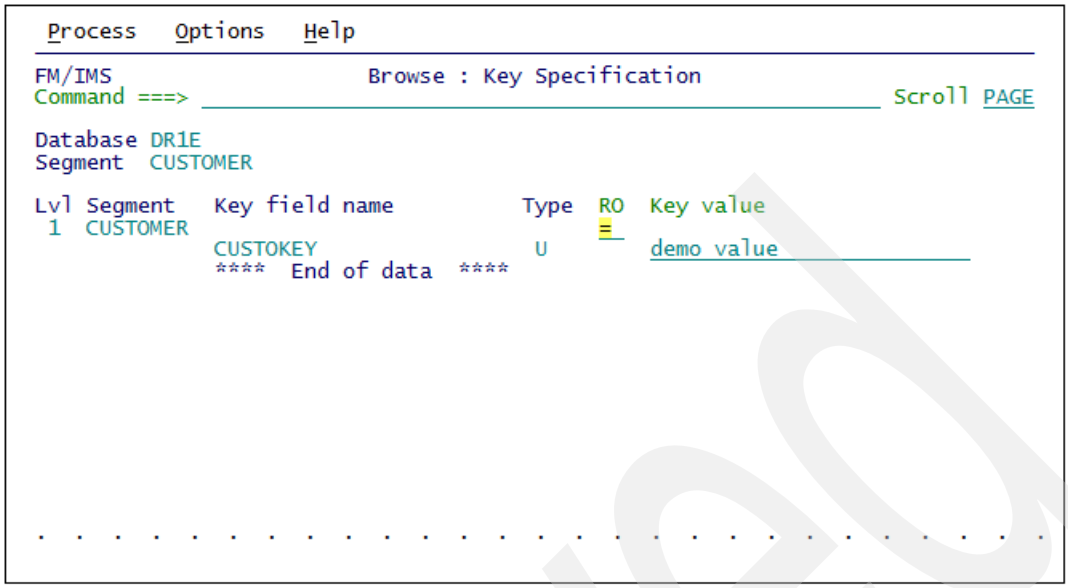


Figure 26-24 Key Specification panel

In the Key Specification panel, we can change:

- The value of the RO field (relational operator):

  Possible values are:

  - > or GT: Greater than
  - >=, =>,or GE: Greater than or equal to
  - = or EQ: Equal to
  - < or LT: Less than
  - <=, =>, or LE: Less than or equal to
  - ¨=, =¨ or NE: Not equal to
  - UQ FM/IMS uses an unqualified SSA at this level (If the Key value field does contain a value, it is ignored.)

- The key value:

  We enter the value of the key field. FM/IMS starts the display from the first segment occurrence that matches the expression formed by the relational operator and the key value.

As shown in Figure 26-25, we have changed the RO field from = to >.

```
 Process   Options   Help
FM/IMS                    Browse : Key Specification
Command ===> _____________________________________________ Scroll PAGE

Database DR1E
Segment  CUSTOMER

Lvl Segment   Key field name          Type  RO  Key value
 1  CUSTOMER                               >
               CUSTOKEY                 U     demo value_____________
               ****  End of data  ****
```

Figure 26-25 Key Specification panel

After pressing Enter, the next IMS Data panel is displayed, as shown in Figure 26-26, and we can see that the current position in the database has been changed.

```
 Process   Options   Help
FM/IMS                    Browse : IMS Database DR1E
Command ===> _____________________________________________ Scroll PAGE
                                        Scope DB  Col 1_____ Format CHAR
Cmd Level Segment   ----+----1----+----2----+
                    ****   Top of window   ****
___ 1     CUSTOMER  ABBA CANNING COMP
___  2    COMPANY                      X02
___  2    COMPANY   KAVIAR             3A3B
___  2    COMPANY   NEW COMPANY        123B
___ 1     CUSTOMER  ALBANE
___  2    COMPANY   Casey_Import_Export ....
___  2    COMPANY   Glass_and_Luget_plc ....
___  2    COMPANY   Headworth_Electrical....
___  2    COMPANY   IBM                ....
___  2    COMPANY   ShareSelect        ....
___  2    COMPANY   Veck_Transport     ...C
___ 1     CUSTOMER  B NEW
___ 1     CUSTOMER  BJORN
___  2    COMPANY   Headworth_Electrical....
___ 1     CUSTOMER  CHARLENE
___  2    COMPANY   Casey_Import_Export ....
```

Figure 26-26 IMS Database Panel

26.4 Templates/views

When browsing or editing a database in FM/IMS, we can use a view to format segments into their individual fields and to select the segments and fields we want to display. To create a view, we first must create a template from the copybooks that define the layouts of the database segments

26.4.1 Creating a template from a DBD

Before we create a template, we must know the following information:

- The DBD load library and the member name of the database
- The name of the partitioned data set where we want to save the template
- The names of the partitioned data sets containing the copybooks that describe the segment layouts
- Which copybook member corresponds to each segment type

To create or update a template for a DBD, we select option **4** on the primary option menu, as shown in Figure 26-27. The result is shown in Figure 26-28.

```
 Process   Options   Help
FM/IMS                      Primary Option Menu
Command ===> 4

0  Settings      Set processing options              User ID . : AMINTOR
1  Browse        Browse data                         System ID : STLABF6
2  Edit          Edit data                           Appl ID . : FMN1
3  Utilities     Perform utility functions           Version . : 7.1.0
4  Templates     Template/view/criteria set utilities Terminal. : 3278A
X  Exit          Terminate FM/IMS                    Screen. . : 1
                                                     Date. . . : 2006/11/14
                                                     Time. . . : 22:17
```

Figure 26-27 Primary Option Menu

```
 Process   Options   Help
FMN1PTUT           Template/View/Criteria Set Utilities
Command ===>

1  Templates        Create or update template
2  Views            Create, edit, or update view
3  Criteria Sets    Create, edit or update criteria set
4  Template update  Update template(s) utility
5  View update      Update view(s) utility
6  Criteria update  Update criteria set(s) utility
```

Figure 26-28 Template/View/Criteria set Utilities menu

This menu regroups all the activities and processes that could be done using FM/IMS Templates. A new function has been added to improve the management and maintenance of the templates: the *Template Update Utility* (option **4** above). This utility updates one or more templates in either foreground or batch.

After selecting option **4**, the *Template Entry Panel* is displayed, as shown in Figure 26-29. On this panel, we specify:

- For the DBD:
  - Data set name 1: The DBD library (or libraries) that the database definitions are to be obtained from (in this example, CHABERT.IMS.TRADER.DBDLIB).
  - Member: The name of the database the template is for (in this example, DR1E). If we leave the field blank or enter a pattern, a member selection list is displayed. We can select a member from this list.
- For the template:

  The name of the partitioned data set where we intend to store the new template. The template member name is always the name of the database.

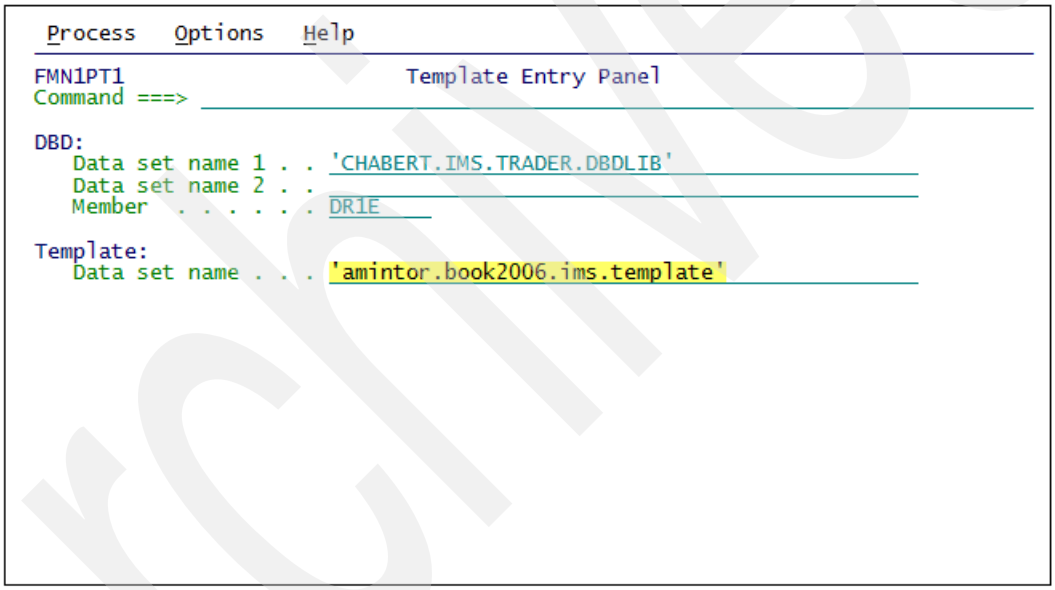


Figure 26-29 Template Entry Panel

After completing the information required for the Template Entry Panel and pressing Enter, the Library List panel is displayed, as shown in Figure 26-30, if a template does not exist for the referenced DBD.

Use this panel to add, remove, or modify the names of partitioned data sets or Panvalet libraries containing the COBOL or PL/I copybooks that describe the segment layouts of the specified DBD.

In this scenario, as shown in the next Figure (Library List), the copybooks are stored in the following concatenated datasets:

- AMINTOR:BOOK2006.IMS.COPYLIB
- SE65273.TRADER.COPYLIB.

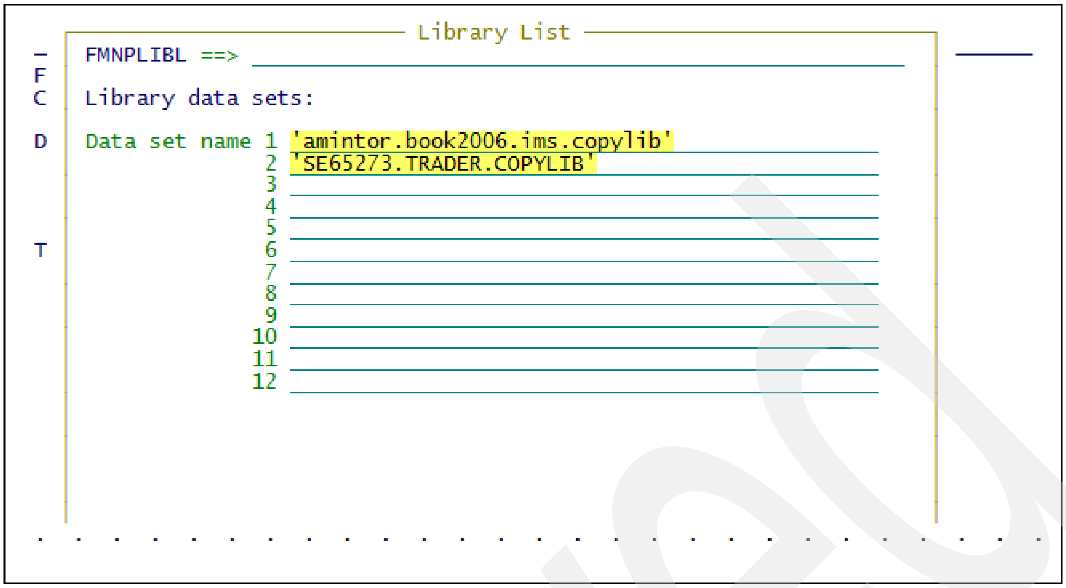


Figure 26-30 Library list panel

Example 26-1 is a sample definition of Copybook that we expect to find in the Library List.

Example 26-1 Copybook or COBOL Structure definition of CUSTFILE file

```
ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
EDIT       AMINTOR.BOOK2006.IMS.COPYLIB(CUSTFILE) - 01.00  Columns 00001 00072
Command ===>                                                 Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 *******01 CUSTOMER-IO-BUFFER.
000002           03 KEYREC.
000003              05 CUST-NM                   PIC X(60).
000004              05 KEYREC-DOT                PIC X(1).
000005              05 COMP-NM                   PIC X(20).
000006           03 DEC-NO-SHARES                PIC S9(7) COMP-3.
000007           03 BUY-FROM                     PIC X(8).
000008           03 BUY-FROM-NO                  PIC X(4).
000009           03 BUY-TO                       PIC X(8).
000010           03 BUY-TO-NO                    PIC X(4).
000011           03 SELL-FROM                    PIC X(8).
000012           03 SELL-FROM-NO                 PIC X(4).
000013           03 SELL-TO                      PIC X(8).
000014           03 SELL-TO-NO                   PIC X(4).
000015           03 ALARM-PERCENT                PIC X(3).
****** **************************** Bottom of Data ****************************
```

After completing the information in the Library List panel and pressing Enter, the Template Specification panel is displayed, as shown in Figure 26-31.

The initial information we see on this panel is:

- Current template name
- Segments in the DBD

If you want to return to the *Template Entry Panel* without confirming the changes that you make on the next panels, type **CAN** on the command line as shown in Figure 26-31.

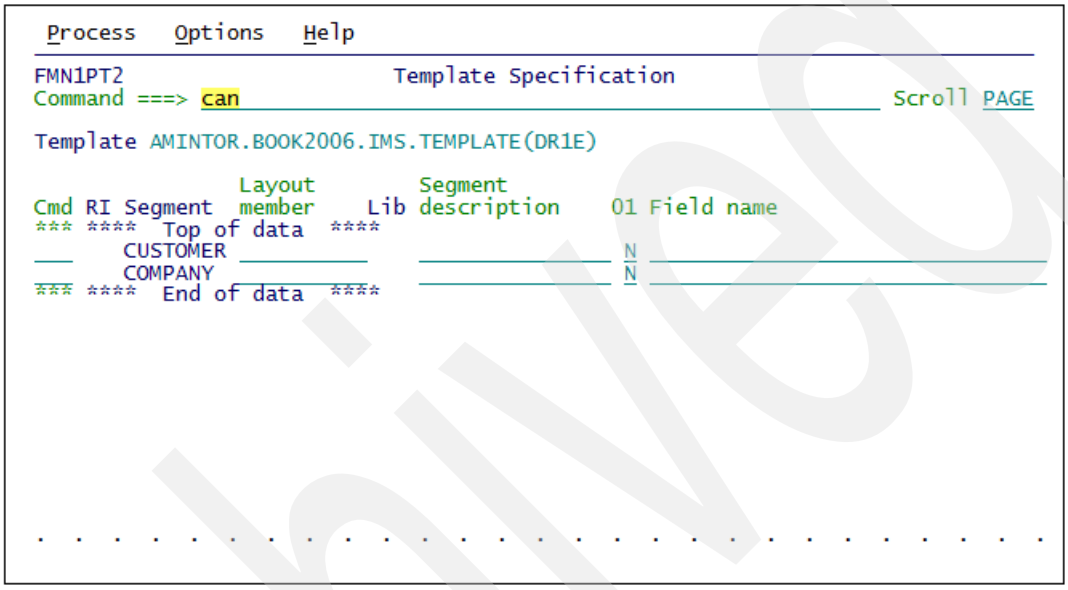


Figure 26-31 Template Specification panel

We now map copybook to the database segment. If we know the name of the copybook we can simply type the name of the copybook on the line to the right of the segment name, *Layout member*.

Note: If the COBOL copybook does not contain a 01 level statement, FM/IMS generates that statement if the N (no) under the 01 header is changed to a Y (yes).

FM/IMS generates a name to that 01 level. If we do not give it a name under the header Field name, FM/IMS also generates a name.

To complete the Template definition, we have to associate all the PSB segments to the respective layouts or copybooks.

Some useful information shown in the next panel includes:

- The name of the segment; we select copybook, which is shown to the left of the header Member selection
- The name of the copybook dataset

If you do not remember the association between copybook and segment, FM/IMS can help you in the following way.

Type asterisk (*) on the *Layout member* field near the CUSTOMER segment, as shown in Figure 26-32, or more simply, you can place the cursor on this field, and press Enter.

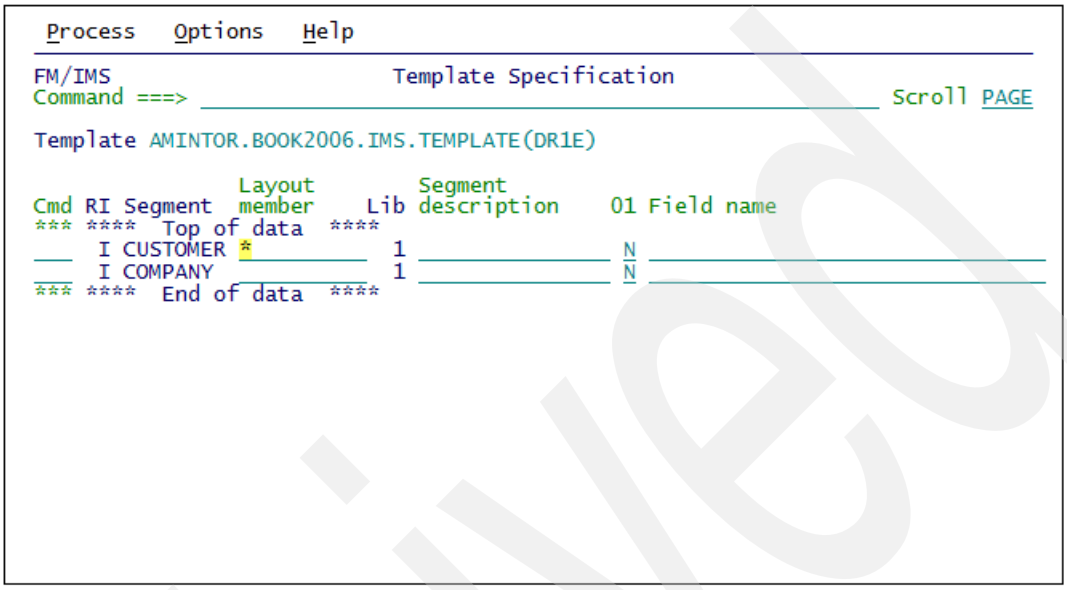


Figure 26-32 Template specification panel

After pressing Enter, the panel in Figure 26-33 is displayed.

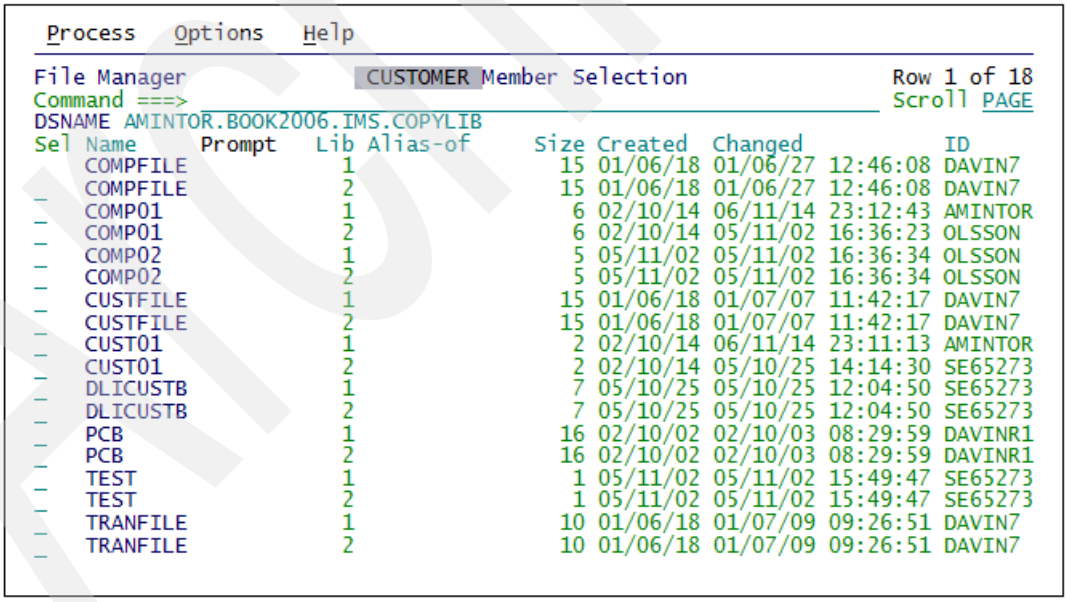


Figure 26-33 Member selection panel

The Panel above lists all the concatenated members that you have inserted in the previous *Library List* panel (FMNPLIOL). Since the cursor was at the CUSTOMER *Member layout* level, you receive the members list related to this segment. Select the copybook (COMP01) related to this segment and put an **S** in front of the corresponding member, as shown in Figure 26-34.

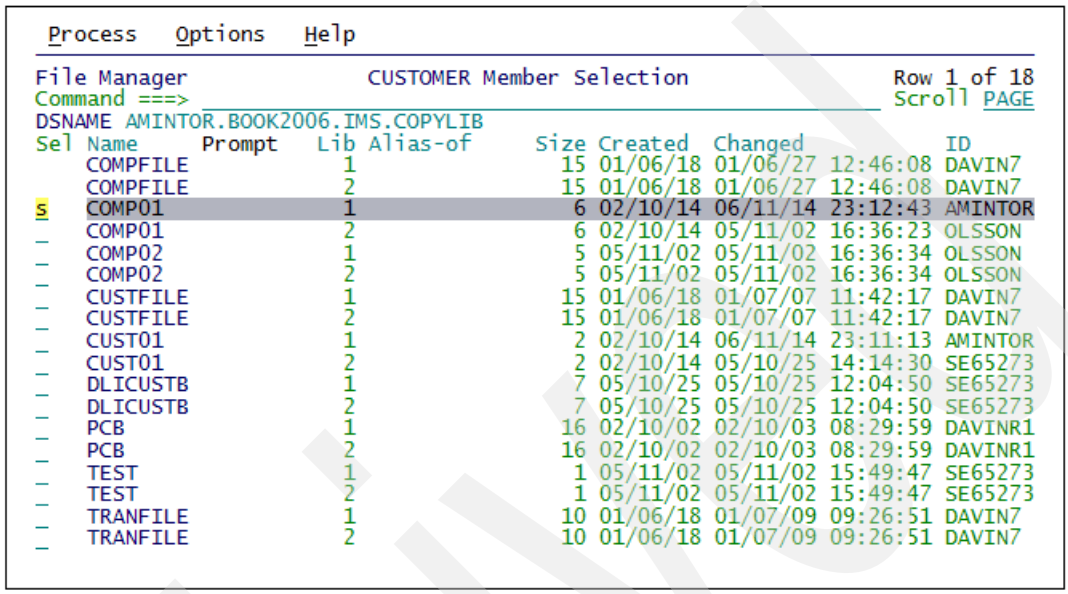


Figure 26-34 Member selection panel related to CUSTOMER

You have just associated COMP01 to the CUSTOMER segment. Press Enter, and the panel in Figure 26-35 lists the next association between the COMPANY segment and copybooks.

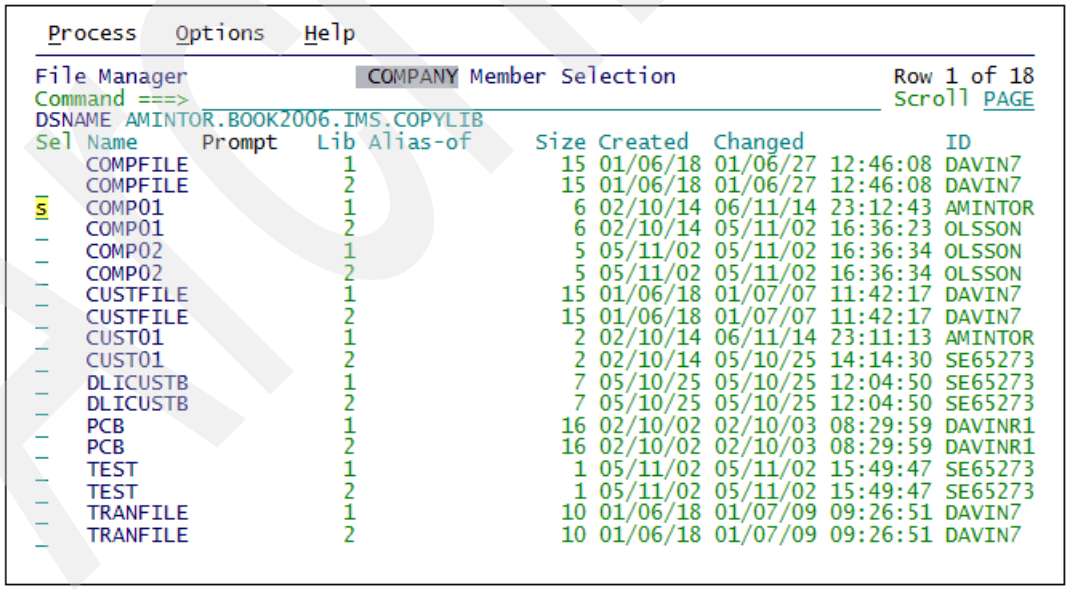


Figure 26-35 Member selection panel related to COMPANY

The *Member Selection* panel related to the COMPANY has been redisplayed. Select the copybook related to the COMPANY segment and press Enter. When all then PSB segments have been selected and associated, the *Template Specification* panel is redisplayed as shown in Figure 26-36.

```
 Process   Options   Help
FMN1PT2                    Template Specification
Command ===> _____________________________________________ Scroll PAGE

Template AMINTOR.BOOK2006.IMS.TEMPLATE(DR1E)

                  Layout          Segment
Cmd RI Segment  member    Lib description    01 Field name
*** **** Top of data  ****
___    CUSTOMER CUST01     1 _____________ N _____________________________
___    COMPANY  COMP01     1 _____________ N _____________________________
*** **** End of data  ****
```

Figure 26-36 template Specification panel

You note that the Layout member has been associated according your selection. Press Enter to confirm the mapping and FM/IMS displays a new panel, as shown in Figure 26-37.

The Template DR1E from DBD has been created and the *Template Entry* panel is displayed with the information `Template saved` in the upper right corner.

:

```
 Process   Options   Help
FMN1PT1                    Template Entry Panel          Template saved
Command ===> _____________________________________________

DBD:
   Data set name 1 . . 'CHABERT.IMS.TRADER.DBDLIB'_____________
   Data set name 2 . . _____________________________________
   Member  . . . . . . DR1E____

Template:
   Data set name . . . 'AMINTOR.BOOK2006.IMS.TEMPLATE'_________
```

Figure 26-37 Template Entry panel

Tip: If you do not know the name of the copybook that corresponds to the segment, press Enter on the *Template Specification* panel, so that the *Member Selection* panel is displayed.

26.4.2 Creating a view from a template

To create a view from the template, we select first option **4** on the Primary option menu, and option **2** from the *Template/View/Criteria Set Utilities* Menu, as shown in Figure 26-38.

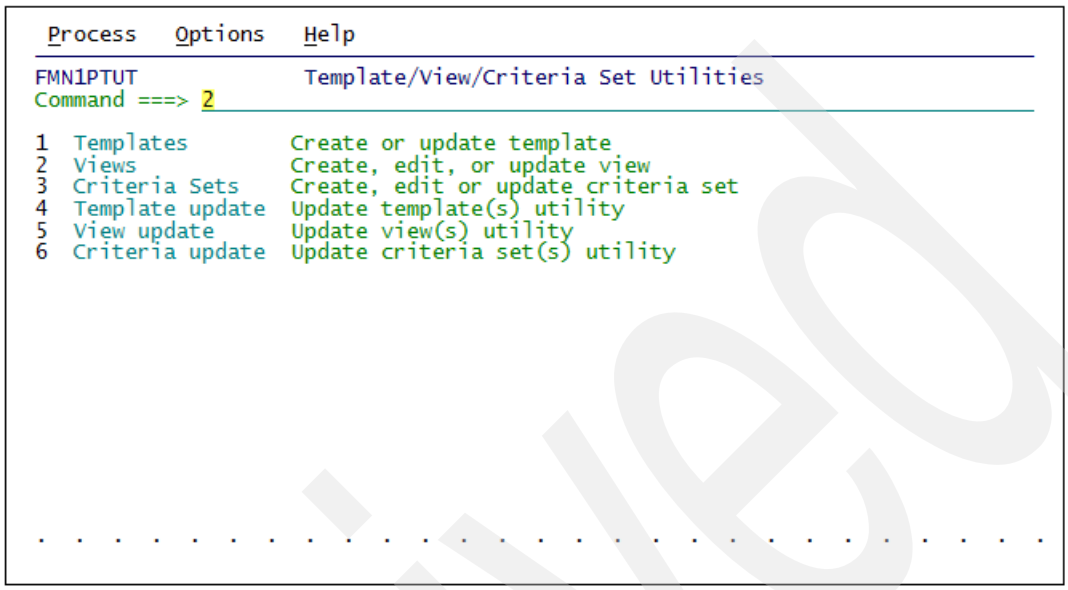


Figure 26-38 Template/View/Criteria Set Utilities Panel

After selecting option **2**, the View Entry Panel is displayed, as shown in Figure 26-39.

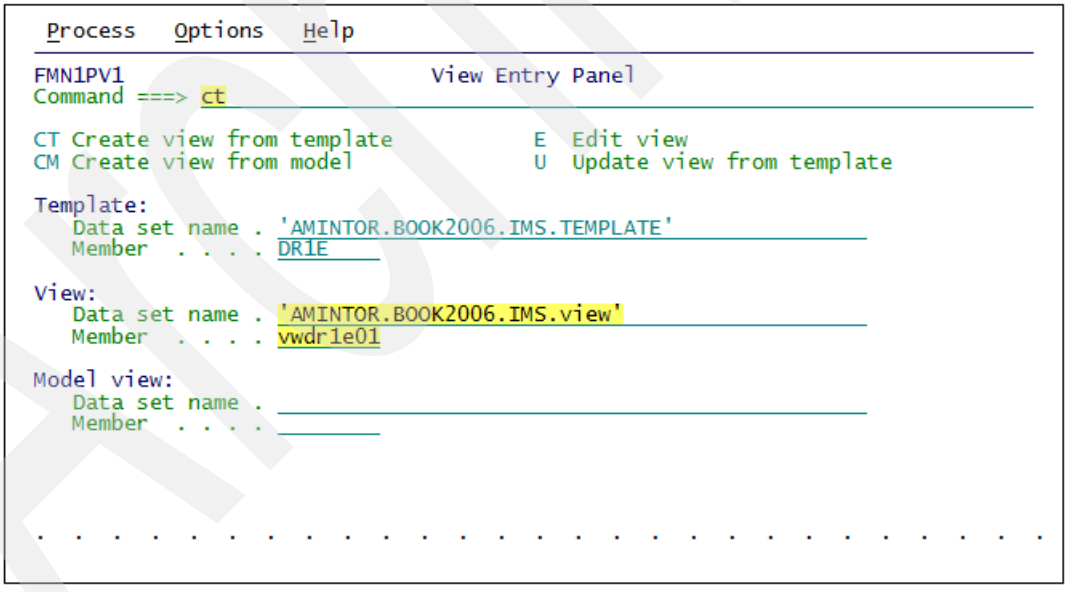


Figure 26-39 View Entry Panel

On the View Entry Panel, we specify:

- For the template:
  – Data set name: The dataset where the template is stored (in this example, AMINTOR.BOOK2006.IMS.TEMPLATE
  – Member: The name of the template (in this example, DR1E)
- For the view:
  – Data set name: The dataset where the view is to be stored (in this example, AMINTOR.BOOK2006.IMS.VIEW
  – Member: The name of the view is VWDR1E

When all information has been obtained, we give the command **`CT`** on the command line and press Enter.

The view is now created and the View Entry Panel is displayed, as shown in Figure 26-40. In the upper right corner, the message `View created` is displayed.

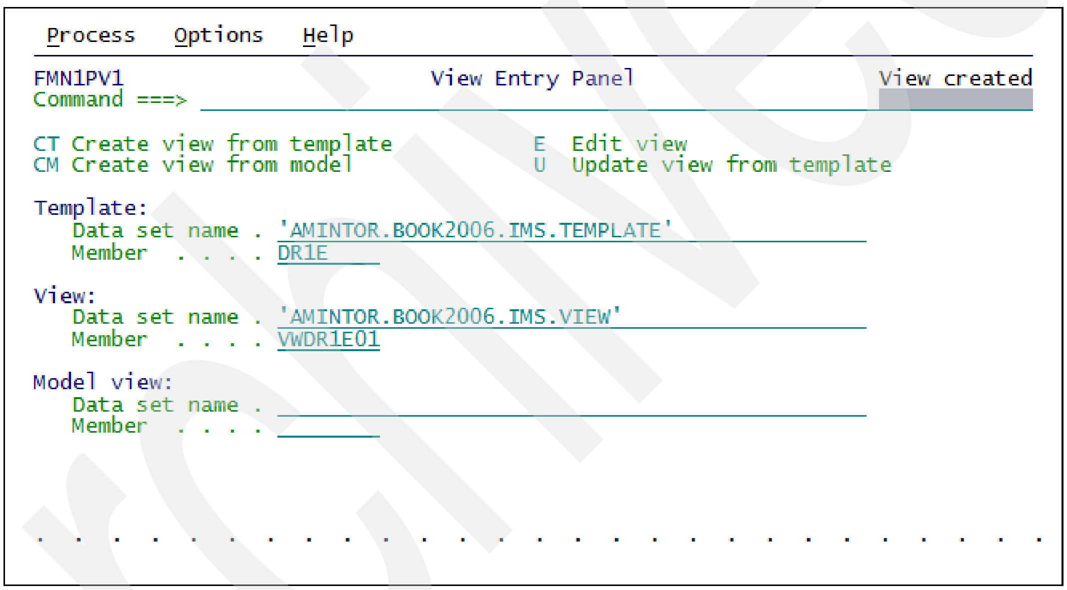


Figure 26-40 View Entry Panel

26.5 Edit/browse using view

Next we discuss editing and browsing using the view.

26.5.1 Preparation

From the Primary Option Menu shown in Figure 26-41, we select option **2** (Edit) and press Enter.

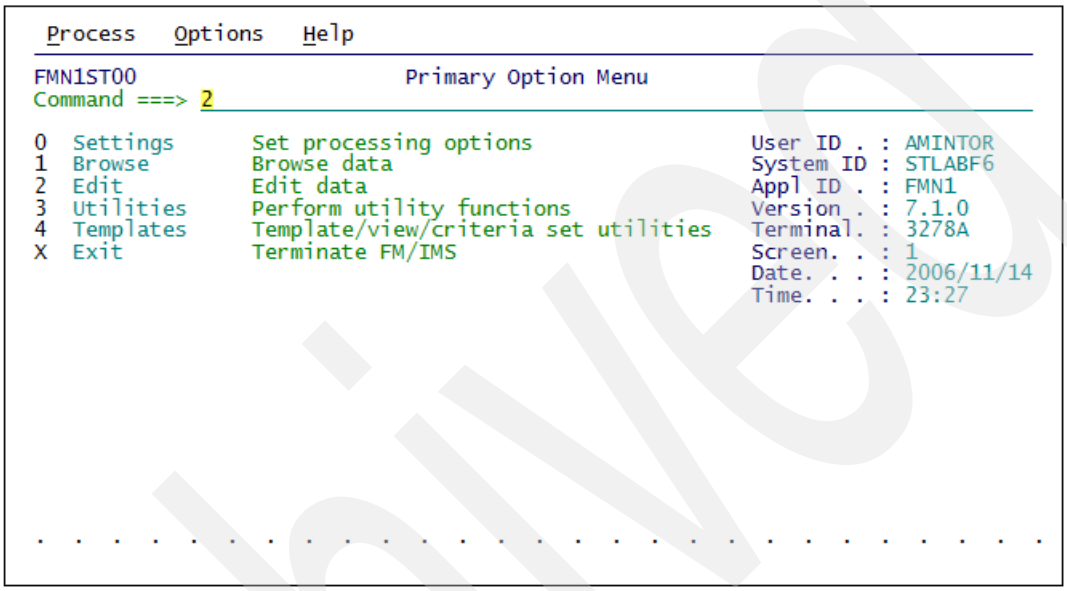


Figure 26-41 Primary Option Menu

FM/IMS now shows the Edit Entry Panel shown in the next Figure. In this panel we can see that a dynamic PSB is used. We defined this in a previous step (System Settings, as shown in Figure 26-42.

On this panel we specify:

- For the DBD:
  - Data set name 1: The DBD library (or libraries) that the database definitions are to be obtained from (in this example, CHABERT.IMS.TRADER.DBDLIB).
  - Member: The name of the database the template is for (in this example, DR1E). If we leave the field blank or enter a pattern, a member selection list is displayed. We can select a member from this list.
- Processing options:
  - We have chosen region type 1 (DLI).
  - We have set the option/use view.

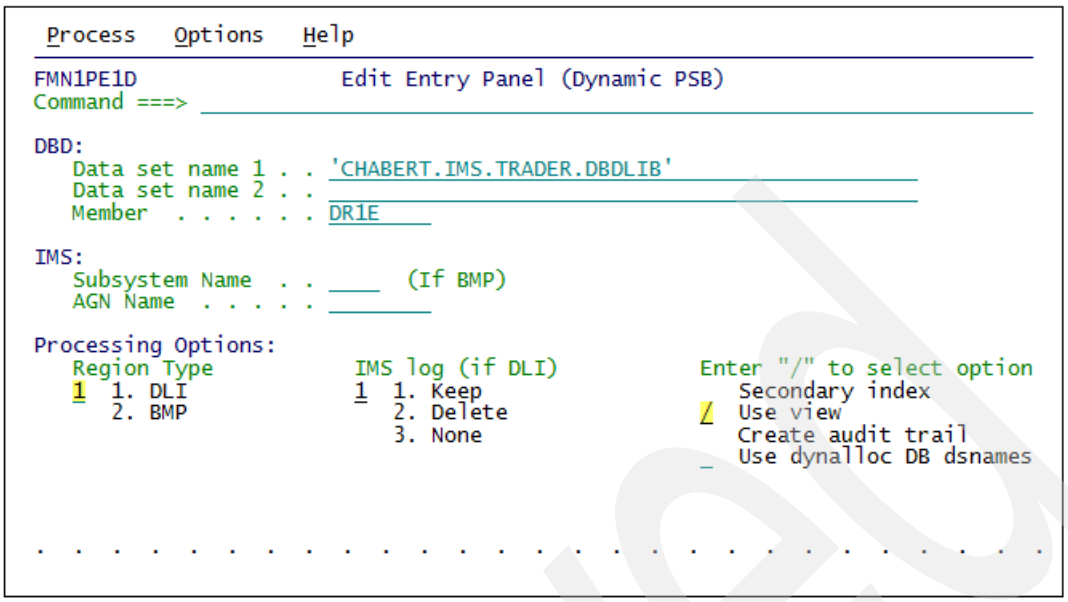


Figure 26-42 Edit Entry Panel

After completing the information in the Edit Entry Panel, we press Enter and the *Database Data Set Specification* panel, as shown in Figure 26-43, is displayed. Here we enter the data set names for the DBD. We can also enter a slash (/) to specify that we want to save these names in our profile to be reused the next time we access this DBD.

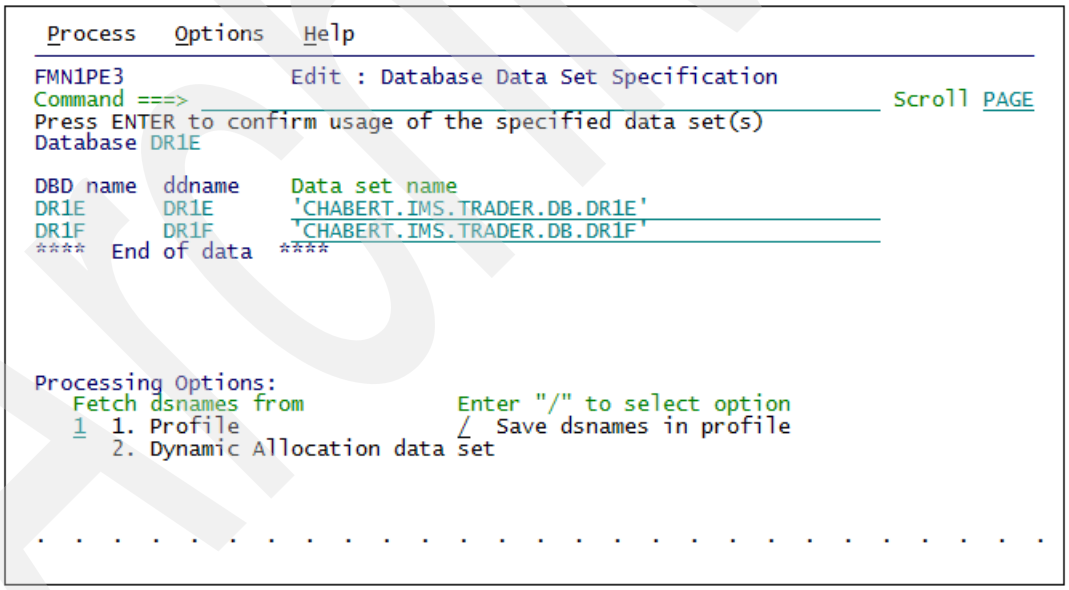


Figure 26-43 Database Data Set Specification panel

After completing the information in the previous panel, we press Enter and the View Specification panel in Figure 26-44 is shown.

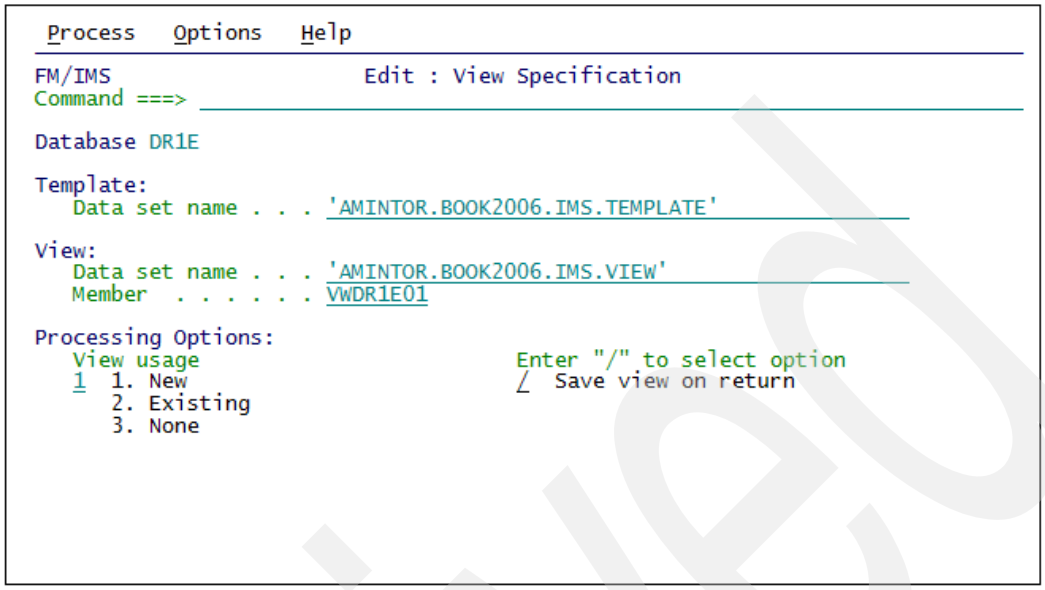


Figure 26-44 View Specification panel

On the View Specification panel, we specify:

- For the template:

  Data set name: The dataset where the template is stored (n this example, AMINTOR:BOOK2006.IMS.TEMPLATE)

- For the view:
  - Data set name: The dataset where the view is stored (in this example, AMINTOR.BOOK2006.IMS.VIEW)
  - Member: The name of the view :(in this example, VWDR1C01

- For the processing options:

  We enter a **1** for using a new view and a **2** for an existing view.

26.5.2 Database positioning

When we have finished the settings, as described under Preparation, we press Enter and the Database Positioning panel is displayed, as shown in Figure 26-45.

```
 Process   Options   Help
FM/IMS                     Edit : Database Positioning
Command ===>                                                       Scroll PAGE

Database DR1E                        Key sequence                 Format TABL
View     AMINTOR.BOOK2006.IMS.VIEW(VWDR1E01 Key
Cmd SXE Level     Segment   Description      len  Key value
s__  SX  1        CUSTOMER                   25   AmiRoot1
___   X   2       COMPANY                    20   ....................
   ****  End of data  ****
```

Figure 26-45 Database Positioning panel

We now enter an `S` in the Cmd field in front of the segment type we want to position on to begin retrieval of segments, and after pressing Enter, the database displays, as shown in Figure 26-46.

```
 Process   Options   Help
FM/IMS                          Edit : IMS Database DR1E
Command ===>                                                      Scroll PAGE
                 CHKPID FM000001 Autosave OFF SHOW SUP OFF Scope DB Format sing
                    COMPANY-NAME
Cmd Level Segment   #2
                    AN 1:25
                    <---+----1----+----2---->
                    ****   Top of window   ****
___ 1     CUSTOMER  AmiRoot1
___  2    COMPANY   2 segments
___ 1     CUSTOMER  AAB
___  2    COMPANY   3 segments
___ 1     CUSTOMER  ABBA CANNING COMP
___  2    COMPANY   4 segments
___ 1     CUSTOMER  ALBANE
___  2    COMPANY   6 segments
___ 1     CUSTOMER  B NEW
___ 1     CUSTOMER  BJORN
___  2    COMPANY   1 segment
___ 1     CUSTOMER  CHARLENE
___  2    COMPANY   6 segments
___ 1     CUSTOMER  ERIC
```

Figure 26-46 Table data display format

As we can see in Figure 26-46, more information is added when we use a template:

- CHKPID
- Autosave
- Field information of current segment

By typing over the format field, when using a view, we can now use a new format to see the data (the SNGL format). We type over the **T** in the Format field with an **S** and press Enter, and the panel in Figure 26-47 is displayed.

```
 Process   Options   Help
FM/IMS                        Edit : IMS Database DR1E
Command ===> _____________________________________________ Scroll PAGE
        CHKPID FM000001 Autosave OFF SHOW SUP OFF Scope DB  Format SNGL
Segment  COMPANY                      Level: 2
Concatenated key value:  AmiRoot1                  NEW COMPANY
                                                Top Line is 1    of 5
Current 01: CUSTCOMP-SEG-IN
Field              Data
COMPANY-CUST     K NEW COMPANY
NO-SHARES-1        3
NO-SHARES-2        5
NO-SHARES-3        3
CUST-LVL           A
***  End of record  ***




 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Previous F11=Next     F12=Cancel
```

Figure 26-47 Changing to format SNGL

After pressing Enter, the current database record is shown in single mode, as shown in the Figure above, and the presentation is more detailed, depending on how we set up the edit/browse options. After pressing F11, the next record is displayed, as shown in Figure 26-48.

```
 Process   Options   Help
FMN1PES                       Edit : IMS Database DR1E
Command ===> =0;7_____________________________________________ Scroll CSR
        CHKPID FM000002 Autosave ON  SHOW SUP ON  Scope DB  Format SNGL
Segment  COMPANY                      Level: 2
Concatenated key value:  AmiRoot1                  AmiSeg1
                                                Top Line is 1    of 5
Current 01: CUSTCOMP-SEG-IN
Field              Data
COMPANY-CUST     K AmiSeg1
NO-SHARES-1        2
NO-SHARES-2        7
NO-SHARES-3        2
CUST-LVL           C
***  End of record  ***




 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Previous F11=Next     F12=Cancel
```

Figure 26-48 Presentation in of SNGL format

As we can see in the foregoing figure, the position is at the root segment. By pressing F11, we retrieve the next segment in the database record. On the command line of any panel, type **=0;7** to display the *Edit /Browse Options* panel, as shown in Figure 26-49.

```
 Process   Options   Help
FM/IMS                      Edit/Browse Options
Command ===> _____________________________________________________________
                                                                More:     +
Related command if applicable, shown in ()                    Initial Display
Miscellaneous:           Enter "/" to select options          1  1. Previous
   CAPS initially ON - translate changed data to uppercase (CAPS)  2. Table
Expose (do not group) records of types: (SHOW)                    3. Single
   Suppressed                                                     4. Character
                                                                  5. Hex
                                                                  6. LHex
Record formatting options for SNGL display or print:
_  Field reference number (REF)    Field type and length values (TYPE)
   Picture clause         (PIC)    Start location               (SLOC)
   Structure              (STR)    Left justify numeric fields  (JUST)
/  Redefined fields       (RDF) (also affects TABL format)
Other:
/  Use * (any segment type) as default on NEXT and PREVIOUS commands
   Set SCOPE REC, limiting segments to within a database record
   Set AUTOSAVE ON, issuing a SAVE after checkpoint frequency in options 0.2

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 26-49 The figure show the actual Edit/Browse Options

Depending on the options you set, the panel displays more details. Set the options as shown in Figure 26-50.

```
 Process   Options   Help
FMN1SETE                    Edit/Browse Options       IMS log data set kept
Command ===> _____________________________________________________________
                                                                More:     +
Related command if applicable, shown in ()                    Initial Display
Miscellaneous:           Enter "/" to select options          1  1. Previous
/  CAPS initially ON - translate changed data to uppercase (CAPS)  2. Table
Expose (do not group) records of types: (SHOW)                    3. Single
/  Suppressed                                                     4. Character
                                                                  5. Hex
                                                                  6. LHex
Record formatting options for SNGL display or print:
   Field reference number (REF) /  Field type and length values (TYPE)
/  Picture clause         (PIC) /  Start location              (SLOC)
/  Structure              (STR) /  Left justify numeric fields (JUST)
/  Redefined fields       (RDF) (also affects TABL format)
Other:
/  Use * (any segment type) as default on NEXT and PREVIOUS commands
/  Set SCOPE REC, limiting segments to within a database record
/  Set AUTOSAVE ON, issuing a SAVE after checkpoint frequency in options 0.2

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 26-50 Set more options to display more details in Edit/Browse

26.5.3 Updating the selected record

We now update the record, as shown in Figure 26-51.

```
 Process    Options    Help
FMN1PES                        Edit : IMS Database DR1E
Command ===> _____________________________________________ Scroll CSR
             CHKPID FM000001 Autosave ON  SHOW SUP ON  Scope DB  Format SNGL
Segment  COMPANY                      Level: 2
Concatenated key value:  AmiRoot1                  AmiSeg1
                                                Top Line is 1    of 6

Current 01: CUSTCOMP-SEG-IN
Field              Typ Start   Len   Data
1 CUSTCOMP-SEG-IN   AN      1    24
 2 COMPANY-CUST     AN      1    20 K AmiSeg1
 2 NO-SHARES-1      ZD     21     1    2
 2 NO-SHARES-2      ZD     22     1    7
 2 NO-SHARES-3      ZD     23     1    2
 2 CUST-LVL         AN     24     1   C
***  End of record  ***



 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Previous F11=Next     F12=Cancel
```

Figure 26-51 Record to be updated

The field we want to update are NO-SHARES-3 and CUST-LVL They have the values ‘2’ and ‘C’ respectively. We change the value of NO-SHARES-3 as ‘9’ and CUST-LVL as ‘B’

To do this, type over these fields with a **9** and **B**, as shown in Figure 26-52. Notice the CHkPID, which has a value of FM000001 before the update.

```
 Process    Options    Help
FMN1PES                        Edit : IMS Database DR1E
Command ===> _____________________________________________ Scroll CSR
             CHKPID FM000001 Autosave ON  SHOW SUP ON  Scope DB  Format SNGL
Segment  COMPANY                      Level: 2
Concatenated key value:  AmiRoot1                  AmiSeg1
                                                Top Line is 1    of 6

Current 01: CUSTCOMP-SEG-IN
Field              Typ Start   Len   Data
1 CUSTCOMP-SEG-IN   AN      1    24
 2 COMPANY-CUST     AN      1    20 K AmiSeg1
 2 NO-SHARES-1      ZD     21     1    2
 2 NO-SHARES-2      ZD     22     1    7
 2 NO-SHARES-3      ZD     23     1    9
 2 CUST-LVL         AN     24     1   B
***  End of record  ***



 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Previous F11=Next     F12=Cancel
```

Figure 26-52 Update record

After pressing Enter, the database record has been updated and the CHKPID has been changed to FM000002, as shown in Figure 26-53.

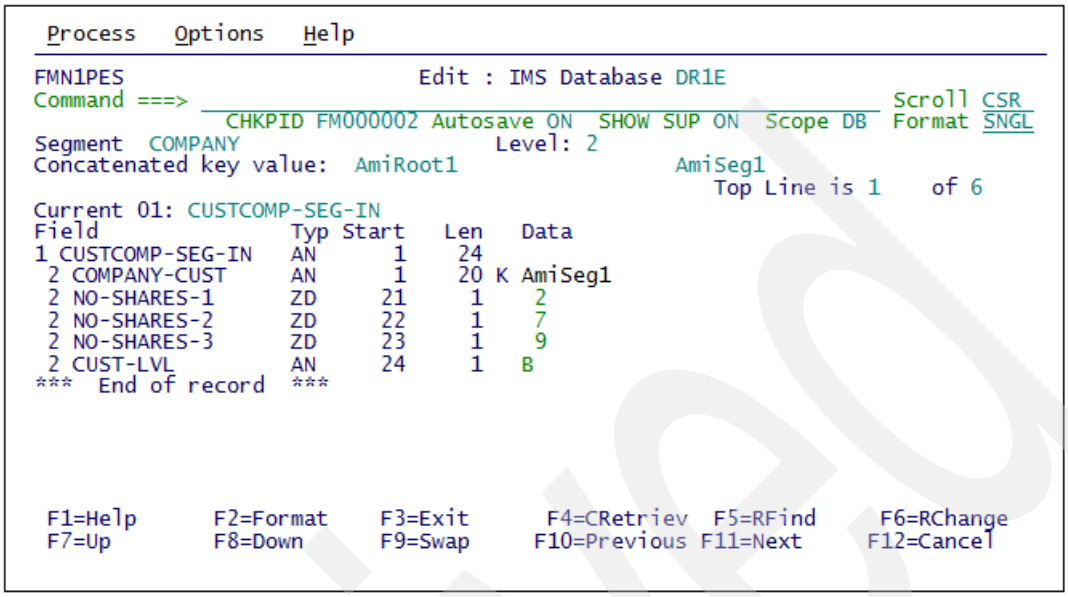


Figure 26-53 After Update

26.6 Inserting segments

Inserting segments can be performed in two ways:

- Using the line command **I**
- Using the primary command INSERT

26.6.1 Inserting a child segment using the line command I

In the following sequence we explain how to do an insert of a segment using the line command **I**.

As shown in Figure 26-54, we insert a new COMPANY segment. We type an **I** on the Cmd field. As we can see, this is a level 2 segment and its parent segment has a key value of AAB.

After typing **I** on the Cmd field and pressing Enter, the panel is displayed as shown in Figure 26-54.

```
 Process   Options   Help
FM/IMS                        Edit : IMS Database DR1E
Command ===> _____________________________________________ Scroll PAGE
              CHKPID FM000002 Autosave OFF SHOW SUP OFF Scope DB  Format TABL
                     COMPANY-NAME
Cmd Level Segment    #2
                     AN 1:25
                     <---+----1----+----2---->
                     ****   Top of window   ****
___ 1     CUSTOMER   AmiRoot1
___  2    COMPANY    2 segments
___ 1     CUSTOMER   AAB
_i_  2    COMPANY    3 segments
___ 1     CUSTOMER   ABBA CANNING COMP
___  2    COMPANY    4 segments
___ 1     CUSTOMER   ALBANE
___  2    COMPANY    6 segments
___ 1     CUSTOMER   B NEW
___ 1     CUSTOMER   BJORN
___  2    COMPANY    1 segment
___ 1     CUSTOMER   CHARLENE
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 26-54 Insert panel 1 of 5

After pressing Enter, the Insert Segment Panel is displayed as shown in Figure 26-55. This panel is used to insert a non-root segment. The panel has three areas:

- Header area:

  Displays the database we edit and the segment type we insert.

- Parent keys area:

  Key to parent segment.

- Data area:

This is where we specify the data we are going to insert. This is a scrollable area. To scroll, we place the cursor in the area and use F7 or F8 to scroll, or use the command UP or DOWN.

```
 Process   Options   Help
FM/IMS                        Insert Segment
Command ===> _____________________________________________ Scroll PAGE
Database DR1E     Segment  COMPANY                    Insert Format SNGL
--------------------------------------------------------------------------
Parent Keys                                                Line  1  of  3
Lvl  Segment  Key field name          Type  Key value                Occur
 1  CUSTOMER
    #1        1 COMPANY-SEG-IN         AN
    #2         2 COMPANY-NAME          AN   AAB_____________________


--------------------------------------------------------------------------
Data                                              Top Field is 1    of 6
Field                  Data
1 CUSTCOMP-SEG-IN
 2 COMPANY-CUST       K
 2 NO-SHARES-1          0
 2 NO-SHARES-2          0
 2 NO-SHARES-3          0
 F1=Help      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange   F7=Up
 F8=Down      F9=Swap     F12=Cancel
```

Figure 26-55 Insert Segment panel (insert 2 of 5)

As shown in Figure 26-56, we have scrolled down in the data area.

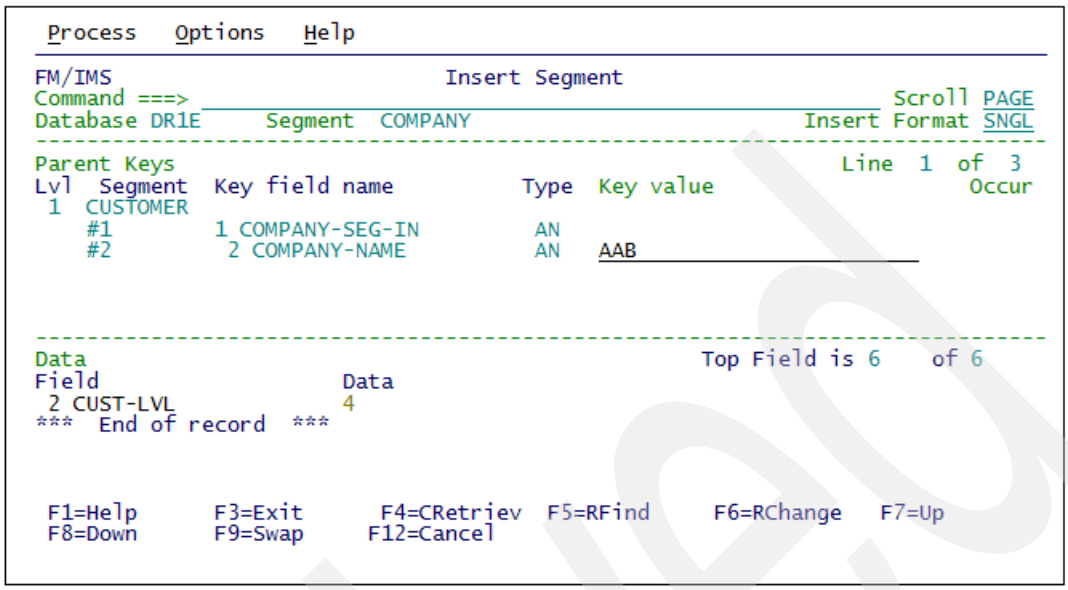


Figure 26-56 Insert Segment panel after scrolling (insert 3 of 5)

As shown in Figure 26-56, we have entered data in the fields NO-SHARES-1, NO-SHARES-2, and CUST-LVL.

When we are ready and want to insert the segment, we press F3.

If we do not want to make the insert, we press F12 (cancel).

After we have pressed PF3, we receive the following message, as shown in Figure 26-57, because we are attempting to insert a segment that has the same Key as an existing one.

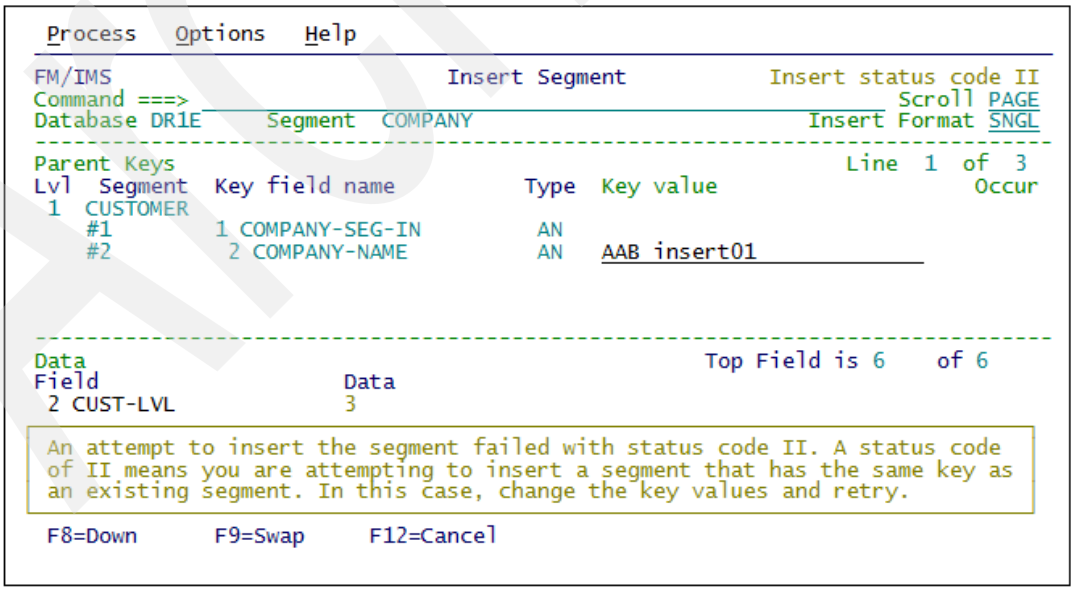


Figure 26-57 Duplicate Key message during the Insert

Type **8** as Key to complete the insert, and the panel in Figure 26-58 is displayed.

```
 Process   Options   Help
FM/IMS                          Insert Segment
Command ===>                                                  Scroll PAGE
Database DR1E     Segment  COMPANY                        Insert Format SNGL
--------------------------------------------------------------------------------
Parent Keys                                                  Line  1  of  3
Lvl  Segment  Key field name          Type  Key value                  Occur
 1  CUSTOMER
    #1        1 COMPANY-SEG-IN         AN
    #2         2 COMPANY-NAME          AN   AAB


--------------------------------------------------------------------------------
Data                                                Top Field is 1    of 6
Field                    Data
1 CUSTCOMP-SEG-IN
 2 COMPANY-CUST       K 08
 2 NO-SHARES-1          2
 2 NO-SHARES-2          9
 2 NO-SHARES-3          5
 F1=Help      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange   F7=Up
 F8=Down      F9=Swap     F12=Cancel
```

Figure 26-58 Insert Segment panel, 4 of 5

As shown in Figure 26-59, the segment is now inserted. The CHKPID has been changed to FM000002.

```
 Process   Options   Help
FM/IMS                      Edit : IMS Database DR1E        INSERT completed
Command ===>                                                  Scroll PAGE
          CHKPID FM000002 Autosave OFF SHOW SUP OFF Scope DB  Format TABL
                 COMPANY-NAME
Cmd Level Segment   #2
                    AN 1:25
                    <---+----1----+----2---->
                    ****   Top of window   ****
___ 1     CUSTOMER  AmiRoot1
___  2    COMPANY   2 segments
___ 1     CUSTOMER  AAB
___  2    COMPANY   4 segments
___ 1     CUSTOMER  ABBA CANNING COMP
___  2    COMPANY   4 segments
___ 1     CUSTOMER  ALBANE
___  2    COMPANY   6 segments
___ 1     CUSTOMER  B NEW
___ 1     CUSTOMER  BJORN
___  2    COMPANY   1 segment
___ 1     CUSTOMER  CHARLENE
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 26-59 Insert Segment panel, complete (5 of 5)

26.6.2 Inserting child segment using line command R

In the following sequence, shown in the next panels, we complete an insert of a segment using the line command **R**. As shown in Figure 26-60, we enter an **R** on the Cmd line to the left of the segment COMPANY and press Enter.

```
 Process   Options   Help
FM/IMS                        Edit : IMS Database DR1E        INSERT completed
Command ===> _____________________________________________________ Scroll PAGE
              CHKPID FM000002 Autosave OFF SHOW SUP OFF Scope DB  Format TABL
                   COMPANY-NAME
Cmd Level Segment  #2
                   AN 1:25
                   <---+----1----+----2---->
                   ****   Top of window   ****
___ 1     CUSTOMER AmiRoot1
r__  2    COMPANY  2 segments
___ 1     CUSTOMER AAB
___  2    COMPANY  4 segments
___ 1     CUSTOMER ABBA CANNING COMP
___  2    COMPANY  4 segments
___ 1     CUSTOMER ALBANE
___  2    COMPANY  6 segments
___ 1     CUSTOMER B NEW
___ 1     CUSTOMER BJORN
___  2    COMPANY  1 segment
___ 1     CUSTOMER CHARLENE
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-60 Repeat 1 of 4

After pressing Enter, the Insert Segment panel is displayed, as shown in Figure 26-61, with parent-key information and also all values from the COMPANY segment.

```
 Process   Options   Help
FM/IMS                        Insert Segment
Command ===> _____________________________________________________ Scroll PAGE
Database DR1E     Segment  COMPANY                        Insert Format SNGL
--------------------------------------------------------------------------------
Parent Keys                                                 Line  1  of  3
Lvl  Segment  Key field name          Type  Key value              Occur
 1  CUSTOMER
    #1        1 COMPANY-SEG-IN         AN
    #2         2 COMPANY-NAME          AN   AmiRoot1_____________


--------------------------------------------------------------------------------
Data                                                Top Field is 1    of 6
Field                    Data
1 CUSTCOMP-SEG-IN
 2 COMPANY-CUST        K NEW COMPANY
 2 NO-SHARES-1           2
 2 NO-SHARES-2           1
 2 NO-SHARES-3           2
 F1=Help      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange   F7=Up
 F8=Down      F9=Swap     F12=Cancel
```

Figure 26-61 Repeat 2 of 4

We now change the values for the fields in the COMPANY segment. To make the INSERT of the segment, we press F3, as shown in Figure 26-62.

```
 Process   Options   Help
FM/IMS                          Insert Segment
Command ===>                                                  Scroll PAGE
Database DR1E     Segment  COMPANY                    Insert Format SNGL
--------------------------------------------------------------------------------
Parent Keys                                                  Line  1  of  3
Lvl  Segment  Key field name          Type  Key value                  Occur
 1  CUSTOMER
    #1        1 COMPANY-SEG-IN         AN
    #2         2 COMPANY-NAME          AN   AmiRoot1


--------------------------------------------------------------------------------
Data                                                Top Field is 1    of 6
Field                    Data
1 CUSTCOMP-SEG-IN
 2 COMPANY-CUST       K NE2 COMPANY
 2 NO-SHARES-1           2
 2 NO-SHARES-2           1
 2 NO-SHARES-3           2
 F1=Help      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange   F7=Up
 F8=Down      F9=Swap     F12=Cancel
```

Figure 26-62 Repeat 3 of 4

After pressing F3, the IMS data panel is displayed, as shown in Figure 26-63, and in the upper right corner we get the information, `REPEAT completed`. CHKPID is changed and the new segment is displayed.

```
 Process   Options   Help
FM/IMS                      Edit : IMS Database DR1E        REPEAT completed
Command ===>                                                  Scroll PAGE
   CHKPID FM000002 Autosave OFF SHOW SUP OFF Scope DB  Format TABL
                COMPANY-NAME
Cmd Level Segment   #2
                    AN 1:25
                    <---+----1----+----2---->
                    ****   Top of window   ****
___ 1     CUSTOMER  AmiRoot1
___  2    COMPANY   3 segments
___ 1     CUSTOMER  AAB
___  2    COMPANY   4 segments
___ 1     CUSTOMER  ABBA CANNING COMP
___  2    COMPANY   4 segments
___ 1     CUSTOMER  ALBANE
___  2    COMPANY   6 segments
___ 1     CUSTOMER  B NEW
___ 1     CUSTOMER  BJORN
___  2    COMPANY   1 segment
___ 1     CUSTOMER  CHARLENE
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 26-63 Repeat 4 of 4

Note: If the `Repeat completed` information does not appear on the top right corner, you have to set `/` on the option *CAPS initially ON* in the *Edit Browse* panel, as shown in Figure 26-64.

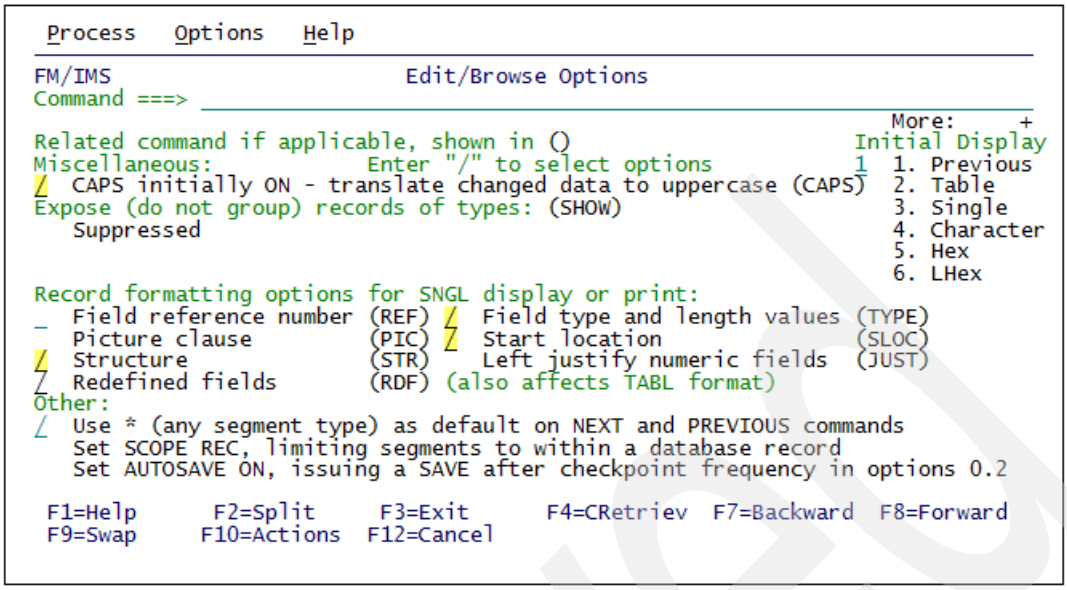


Figure 26-64 Edit Browse Option Panel

26.6.3 Inserting child segment using primary command

The following sequence, shown in Figure 26-65, illustrates how to insert a child segment using the primary command INSERT.

The reason for using this command is that when the root segment does not have any dependent segments of the type we want to insert, we have to use the INSERT command.

As shown in Figure 26-65, there is a CUSTOMER segment, B NEW.

Our current position in the database is at the root segment AmiRoot1.

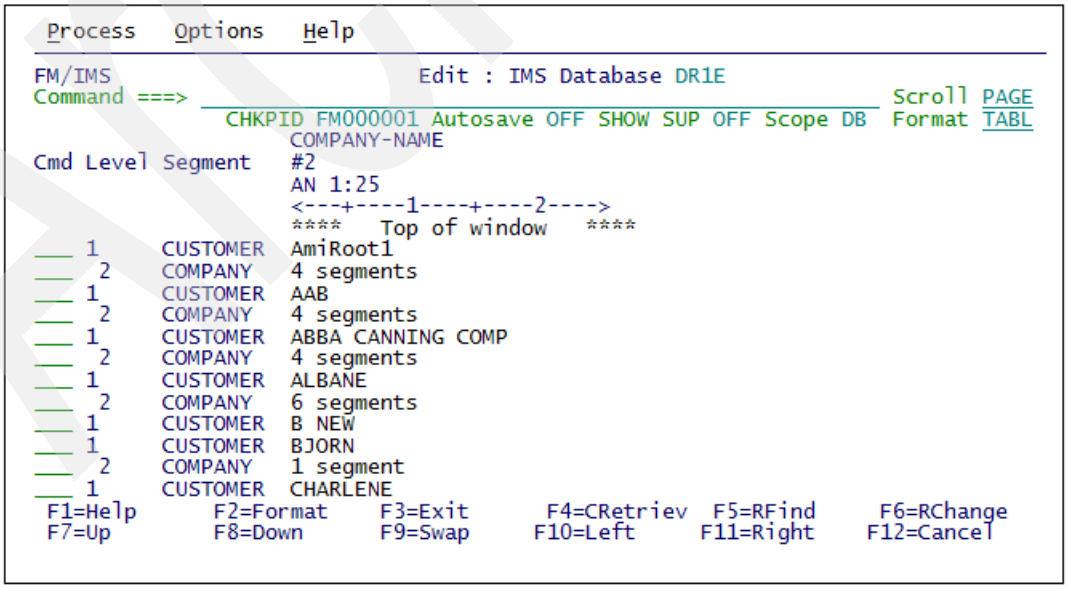


Figure 26-65 CUSTOMER segment NEW CUST

We now reposition in the database so that the current position is at the root segment AmiRoot1. This is done as shown in Figure 26-66 by entering a **K** in the Prefix area field and pressing Enter.

```
 Process   Options   Help
FM/IMS                        Edit : IMS Database DR1E
Command ===> _____________________________________________ Scroll CSR
              CHKPID FM000004 Autosave ON  SHOW SUP OFF Scope DB  Format TABL
                     COMPANY-NAME
Cmd Level Segment    #2
                     AN 1:25
                     <---+----1----+----2---->
                     ****   Top of window   ****
___ 1     CUSTOMER   AmiRoot1
___  2    COMPANY    6 segments
___ 1     CUSTOMER   AAB
___  2    COMPANY    4 segments
___ 1     CUSTOMER   ABBA CANNING COMP
___  2    COMPANY    5 segments
___ 1     CUSTOMER   ALBANE
___  2    COMPANY    7 segments
k__ 1     CUSTOMER   B NEW
___ 1     CUSTOMER   BJORN
___  2    COMPANY    1 segment
___ 1     CUSTOMER   CHARLENE
    elp       F2=Format     F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down       F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-66 Entering a K in Prefix area and pressing Enter

After pressing Enter the Key Specification panel, as shown in Figure 26-67, has the key value of B NEW. This is where we want to be positioned, so we press Enter again.

```
 Process   Options   Help
FM/IMS                      Edit : Key Specification
Command ===> _____________________________________________ Scroll CSR

Database DR1E
Segment  CUSTOMER

Lvl Segment   Key field name          Type  RO  Key value
 1  CUSTOMER                                =_
    #1         1 COMPANY-SEG-IN        AN
    #2          2 COMPANY-NAME         AN       B NEW_____________________
               ****  End of data  ****




    elp      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 26-67 Key Specification panel

After pressing Enter, we come back to the Edit Panel, and we can see that we are positioned on the root segment B NEW, as shown in Figure 26-68.

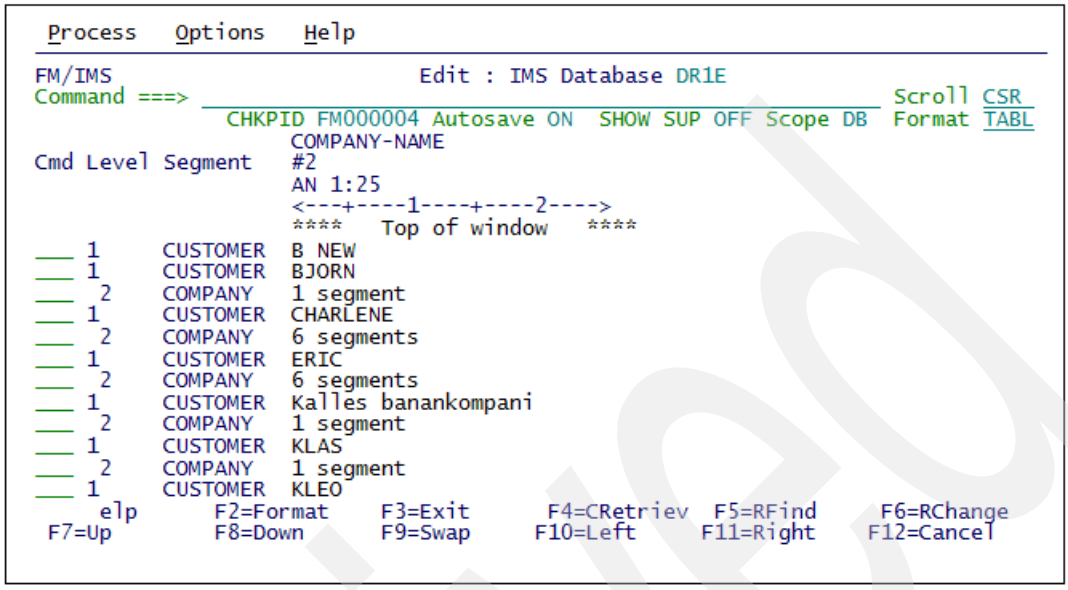


Figure 26-68 Repositioned

On the command line, we enter the INSERT command, as shown in Figure 26-69, and press Enter.

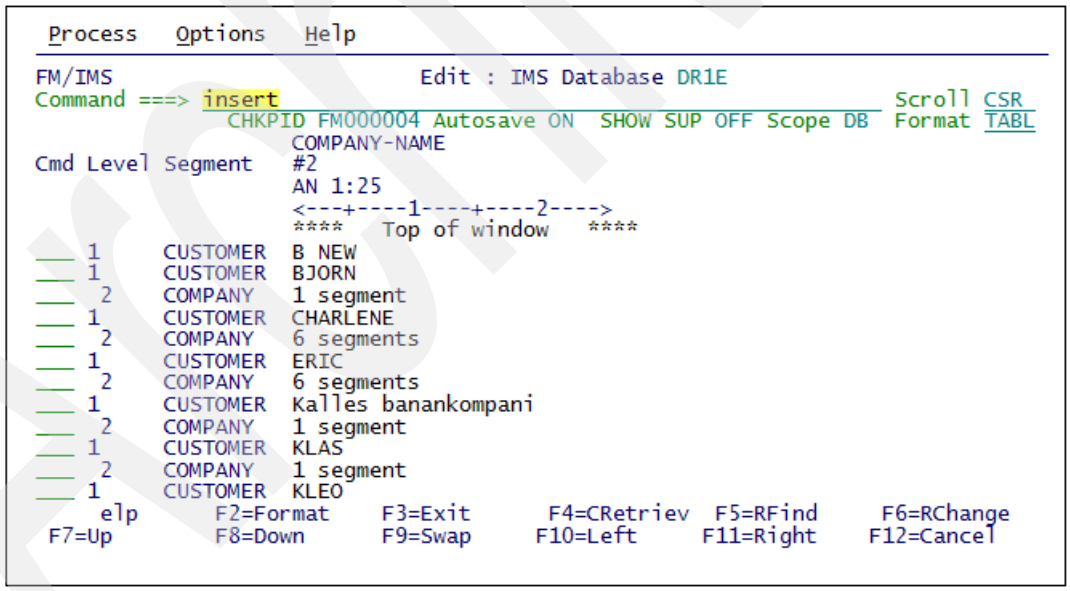


Figure 26-69 Insert command

The insert Segment Selection panel now appears, as shown in Figure 26-70. Here we:

1. Place an **S** in the Cmd field to the left of segment COMPANY. This is the segment type we wish to insert.
2. Press Enter.

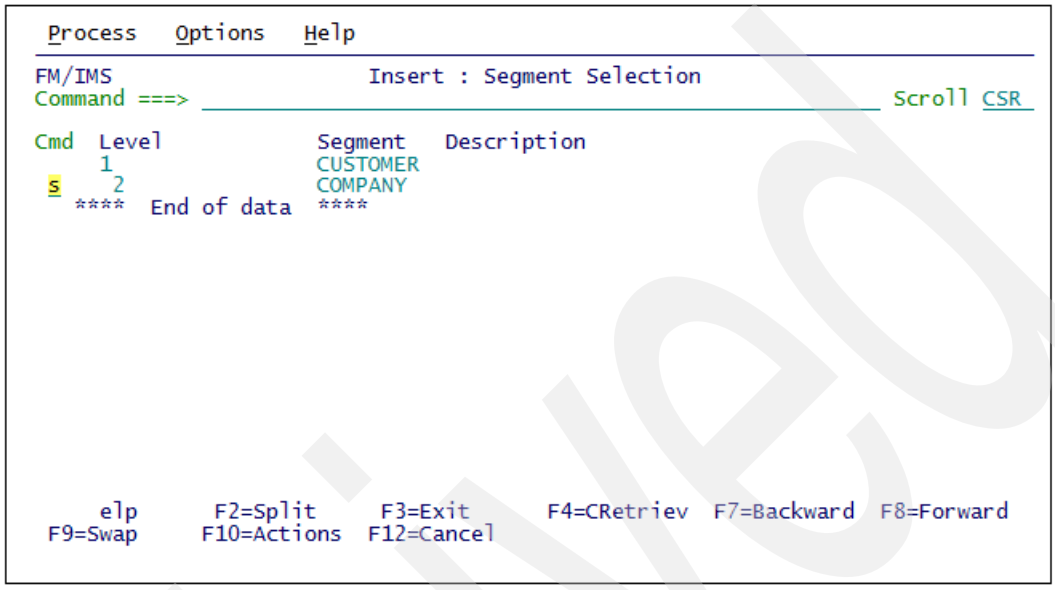


Figure 26-70 Segment Selection pane

The Insert Segment Panel is now displayed, as shown in Figure 26-71.

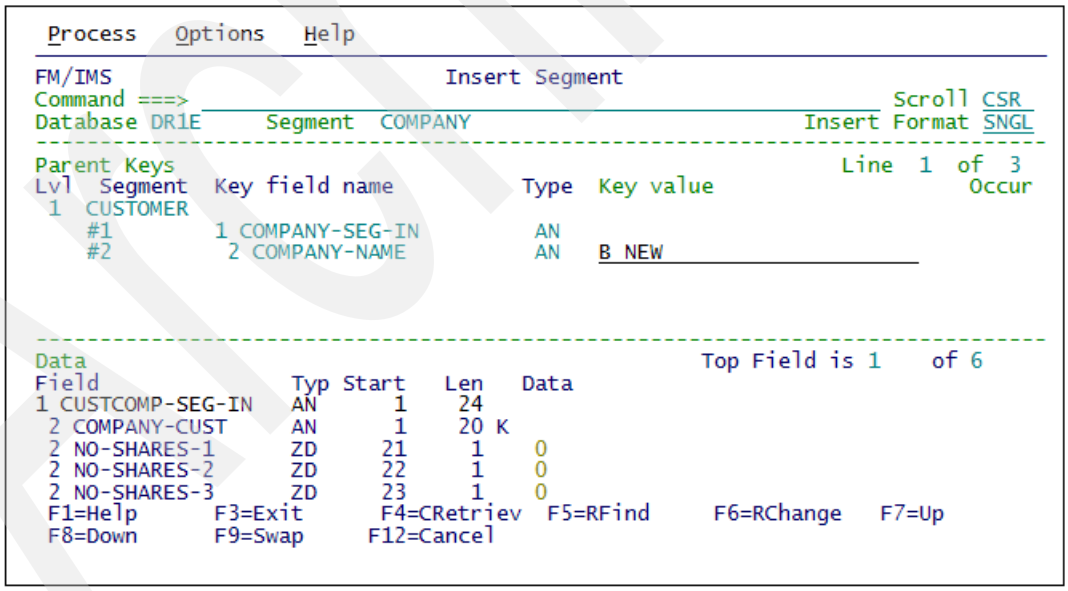


Figure 26-71 Insert Segment pane

We enter the data for the fields, as shown in Figure 26-72.

```
COMPANY-CUST I= MARIANNA CINZIA
```

After completing all fields, we enter F3 and the new segment is inserted.

```
 Process   Options   Help
-------------------------------------------------------------------------------
FM/IMS                          Insert Segment
Command ===>                                                  Scroll CSR
Database DR1E     Segment  COMPANY                    Insert Format SNGL
-------------------------------------------------------------------------------
Parent Keys                                                  Line  1  of  3
Lvl  Segment  Key field name          Type  Key value              Occur
 1  CUSTOMER
    #1        1 COMPANY-SEG-IN         AN
    #2         2 COMPANY-NAME          AN   B NEW


-------------------------------------------------------------------------------
Data                                                 Top Field is 1    of 6
Field               Typ Start   Len   Data
1 CUSTCOMP-SEG-IN   AN      1    24
 2 COMPANY-CUST     AN      1    20 K MARIANNA Cinzia
 2 NO-SHARES-1      ZD     21     1    1
 2 NO-SHARES-2      ZD     22     1    2
 2 NO-SHARES-3      ZD     23     1    3
 F1=Help      F3=Exit     F4=CRetriev  F5=RFind     F6=RChange   F7=Up
 F8=Down      F9=Swap     F12=Cancel
```

Figure 26-72 insert Segment panel

IAfter pressing F3, the Edit screen appears, as shown in Figure 26-73. We can observe the following results:

- `INSERT completed` is shown in the upper right corner.
- CHLKPID has changed.
- The child segment to the root NEW CUST has been inserted.

```
 Process   Options   Help
-------------------------------------------------------------------------------
FM/IMS                    Edit : IMS Database DR1E        INSERT completed
Command ===>                                                  Scroll CSR
          CHKPID FM000002 Autosave ON  SHOW SUP ON  Scope DB  Format TABL
                 COMPANY-NAME
Cmd Level Segment   #2
                    AN 1:25
                    <---+----1----+----2---->
                    ****   Top of window   ****
___ 1     CUSTOMER  AmiRoot1
___  2    COMPANY   Key=AmiSeg1
___  2    COMPANY   Key=Ami01 company
___  2    COMPANY   Key=MARIANNA CINZIA
___  2    COMPANY   Key=NEW COMPANY
___  2    COMPANY   Key=NE2 COMPANY
___  2    COMPANY   Key=NE3 COMPANY
___  2    COMPANY   Key=NE4 COMPANY
___ 1     CUSTOMER  AAB
___  2    COMPANY   Key=
___  2    COMPANY   Key=AAB
___  2    COMPANY   Key=IBM
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-73 Insert completed

26.7 Deleting segments

To delete a segment, we enter a D on the Cmd line to the left of the segment to delete, as shown in Figure 26-74.

```
 Process    Options    Help
FM/IMS                          Edit : IMS Database DR1E
Command ===> _____________________________________________ Scroll CSR
              CHKPID FM000002 Autosave ON  SHOW SUP ON  Scope DB  Format TABL
                     COMPANY-NAME
Cmd Level Segment    #2
                     AN 1:25
                     <---+----1----+----2---->
                     ****   Top of window   ****
___ 1     CUSTOMER   AmiRoot1
___  2    COMPANY    Key=AmiSeg1
___  2    COMPANY    Key=Ami01 company
___  2    COMPANY    Key=MARIANNA CINZIA
___  2    COMPANY    Key=NEW COMPANY
___  2    COMPANY    Key=NE2 COMPANY
___  2    COMPANY    Key=NE3 COMPANY
d__  2    COMPANY    Key=NE4 COMPANY
___ 1     CUSTOMER   AAB
___  2    COMPANY    Key=
___  2    COMPANY    Key=AAB
___  2    COMPANY    Key=IBM
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-74 Delete a segment

After pressing Enter, the segment is deleted. If we enter a `D` command on a `1 Level` segment that has child segments, the segment and all of its child segments are deleted, as shown in Figure 26-75.

```
 Process    Options    Help
FM/IMS                          Edit : IMS Database DR1E
Command ===> _____________________________________________ Scroll CSR
              CHKPID FM000003 Autosave ON  SHOW SUP ON  Scope DB  Format TABL
                     COMPANY-NAME
Cmd Level Segment    #2
                     AN 1:25
                     <---+----1----+----2---->
                     ****   Top of window   ****
___ 1     CUSTOMER   AmiRoot1
___  2    COMPANY    Key=AmiSeg1
___  2    COMPANY    Key=Ami01 company
___  2    COMPANY    Key=MARIANNA CINZIA
___  2    COMPANY    Key=NEW COMPANY
___  2    COMPANY    Key=NE2 COMPANY
___  2    COMPANY    Key=NE3 COMPANY
___ 1     CUSTOMER   AAB
___  2    COMPANY    Key=
___  2    COMPANY    Key=AAB
___  2    COMPANY    Key=IBM
___  2    COMPANY    Key=08
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-75 After delete

26.8 More about views

A view begins as a copy of a template. You can then add information to:

- Select the segments you want displayed.
- Select the fields you want displayed.
- Select the order in which you want the fields displayed.
- Change the display width of a field.
- Display leading zeros on numeric fields.
- Specify a field heading to use instead of the field name in the copybook.

26.8.1 Creating a view from a template

Here, we show how to create a view from a template. On the command line of any panel, type **=4.2** to display the View Entry Panel panel, as shown in Figure 26-76.

```
 Process   Options   Help
------------------------------------------------------------------------------
FM/IMS                          View Entry Panel
Command ===> ct

CT Create view from template           E  Edit view
CM Create view from model              U  Update view from template

Template:
   Data set name . 'AMINTOR.BOOK2006.IMS.TEMPLATE'
   Member  . . . . DR1E

View:
   Data set name . 'AMINTOR.BOOK2006.IMS.VIEW'
   Member  . . . . VWDR1E03

Model view:
   Data set name .
   Member  . . . .


 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 26-76 How to create a view from a template

Type the library names for the Template and View, then the member names (DR1F and VWDR1E03), as shown in Figure 26-77. Type **CT** on the command line and press Enter.

```
 Process   Options   Help
FM/IMS                    Edit : View Specification
Command ===> ____________________________________________________________

Database DR1E

Template:
   Data set name . . . 'AMINTOR.BOOK2006.IMS.TEMPLATE'_____________

View:
   Data set name . . . 'AMINTOR.BOOK2006.IMS.VIEW'_________________
   Member  . . . . . . vwdr1e03

Processing Options:
   View usage                          Enter "/" to select option
   2  1. New                           /  Save view on return
      2. Existing
      3. None



 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 26-77 Specify the view

Now, we have create the “VWDR1E03” view, and we can use it to display and print the data using the *Edit: View Specification* panel, as shown in Figure 26-77. After pressing Enter, the Database Positioning panel is displayed, as shown in Figure 26-78.

```
 Process   Options   Help
FM/IMS                  Edit : Database Positioning      Chkpt FM000001 taken
Command ===> ____________________________________________________ Scroll CSR

Database DR1E                        Key sequence                 Format TABL
View     AMINTOR.BOOK2006.IMS.VIEW(VWDR1E03 Key
Cmd SXE Level     Segment   Description     len  Key value
x   SX  1         CUSTOMER                  25  AmiRoot1_____________
___   X   2       COMPANY                   20  ....................
   ****  End of data  ****




 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=Key >=    F7=Backward
 F8=Forward   F9=Swap     F10=Actions  F12=Cancel
```

Figure 26-78 Database Positioning panel

We can see the following results:

- A header SXE
- To the left of segment CUSTOMER, a combination of SX
- To the left of segment COMPANY, an X

SXE has the following meaning:

S This is shown when the segment type is in the hierarchical path of the current position. It is blank otherwise.

X This is shown for those segments that are selected for display by the current view. If no view is used, all segments are considered selected.

E This is shown for those segments that have selection expression or expressions in the current view.

We type an **X** on the Cmd line to the left, as shown in the previous Figure, and press Enter.

After pressing Enter the X to the left of the customer segment disappears. We have deselected the CUSTOMER segment from display, as shown in Figure 26-79.

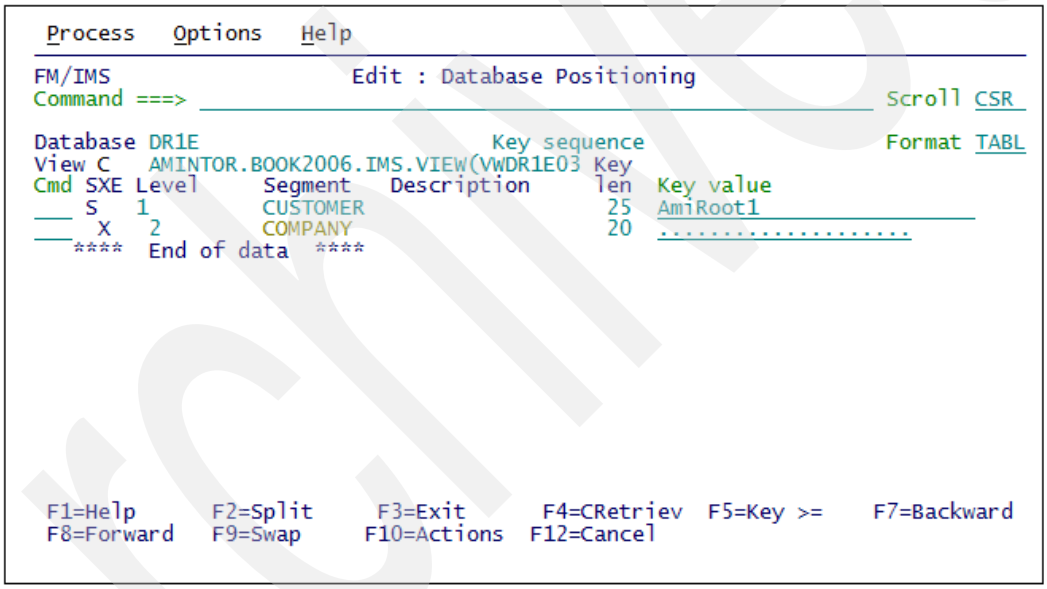


Figure 26-79 Now CUSTOMER segment deselected

We now enter an **S** on the Cmd line, as shown in Figure 26-80, and press Enter.

```
 Process   Options   Help
FM/IMS                    Edit : Database Positioning
Command ===> _____________________________________________ Scroll CSR

Database DR1E                          Key sequence              Format TABL
View C   AMINTOR.BOOK2006.IMS.VIEW(VWDR1E03 Key
Cmd SXE Level     Segment   Description     len  Key value
s   S   1         CUSTOMER                 25  AmiRoot1_____________
___ X   2         COMPANY                  20  ....................
   ****  End of data  ****

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=Key >=    F7=Backward
 F8=Forward   F9=Swap      F10=Actions  F12=Cancel
```

Figure 26-80 Database Positioning panel

Now only the COMPANY segments are displayed, as shown in Figure 26-81.

We are in edit mode, so all functions for editing are viable.

```
 Process   Options   Help
FM/IMS                        Edit : IMS Database DR1E
Command ===> _____________________________________________ Scroll CSR
          CHKPID FM000001 Autosave ON  SHOW SUP ON  Scope DB  Format TABL
                 COMPANY-CUST          NO-SHARES-1 NO-SHARES-2 NO-SHARES-3
Cmd Level Segment #2                          #3          #4          #5
                  AN 1:20                  ZD 21:1     ZD 22:1     ZD 23:1
                  <---+----1----+---->          <>          <>          <>
                  ****   Top of window   ****
___  2    COMPANY   AmiSeg1                      2           7           2
___  2    COMPANY   Ami01 company                2           2           2
___  2    COMPANY   MARIANNA CINZIA              1           3           2
___  2    COMPANY   NEW COMPANY                  3           5           3
___  2    COMPANY   NE2 COMPANY                  2           1           2
___  2    COMPANY   NE3 COMPANY                  2           7           2
___  2    COMPANY                               1           2           2
___  2    COMPANY   AAB                          3           1           9
___  2    COMPANY   IBM                          7           7           7
___  2    COMPANY   08                           2           9           5
___  2    COMPANY                               7           0           2
___  2    COMPANY    001                         8           8           8
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-81 COMPANY segments

We leave this panel by pressing F3. The panel shown in Figure 26-82 is displayed.

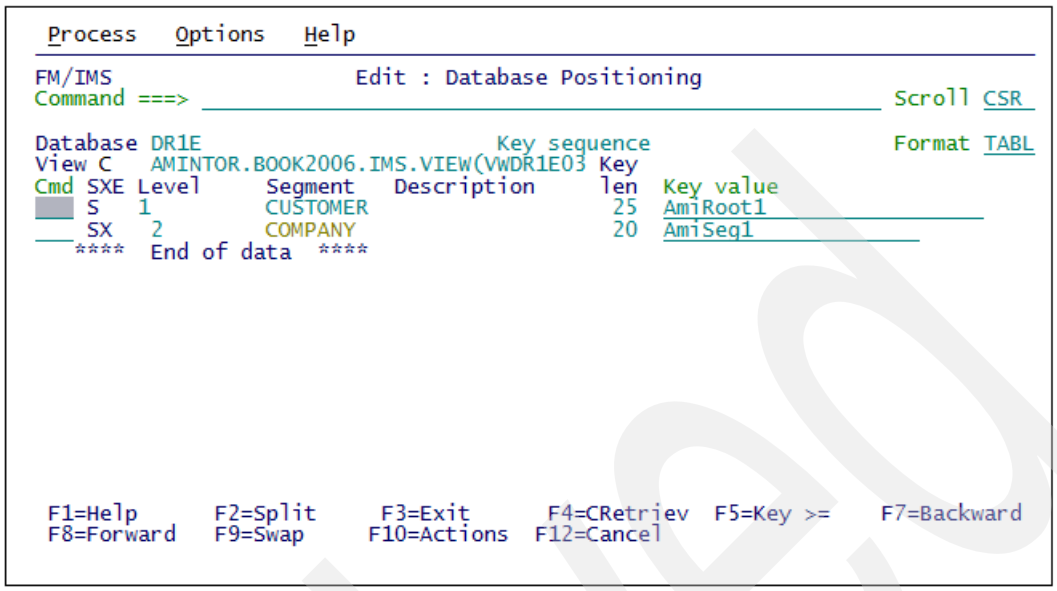


Figure 26-82 Database Positioning panel

When we have exited the Database Positioning panel by pressing F3, the panel shown in Figure 26-83 is displayed.

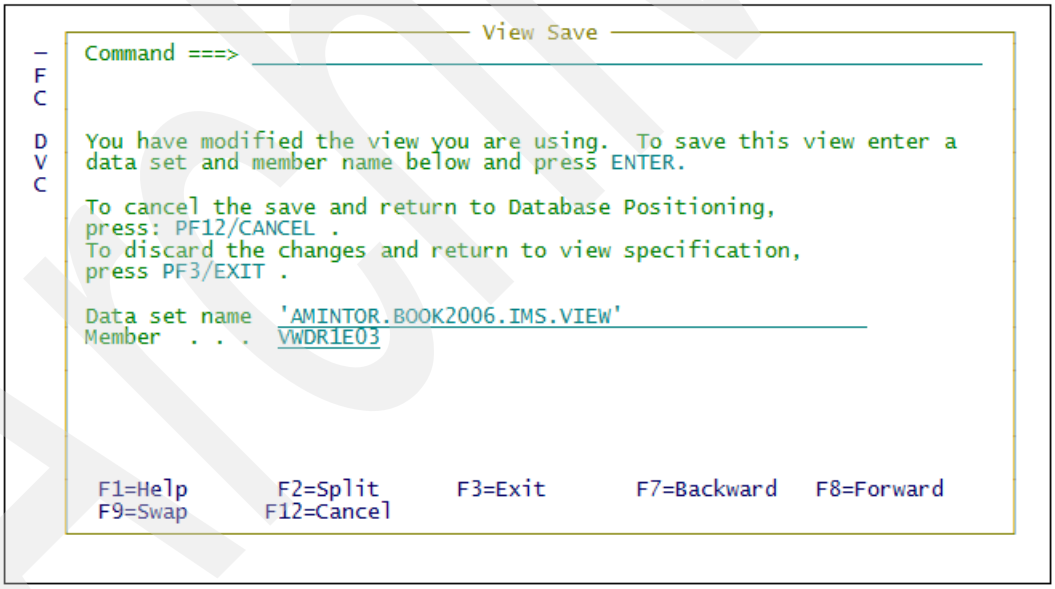


Figure 26-83 View save

. As we have changed the view by deselecting one segment, we have following options:

- Replace the view.
- Cancel the update of the view.
- Save the view with a new name.

As shown in Figure 26-84, we save the view as *VWDR1E04*.

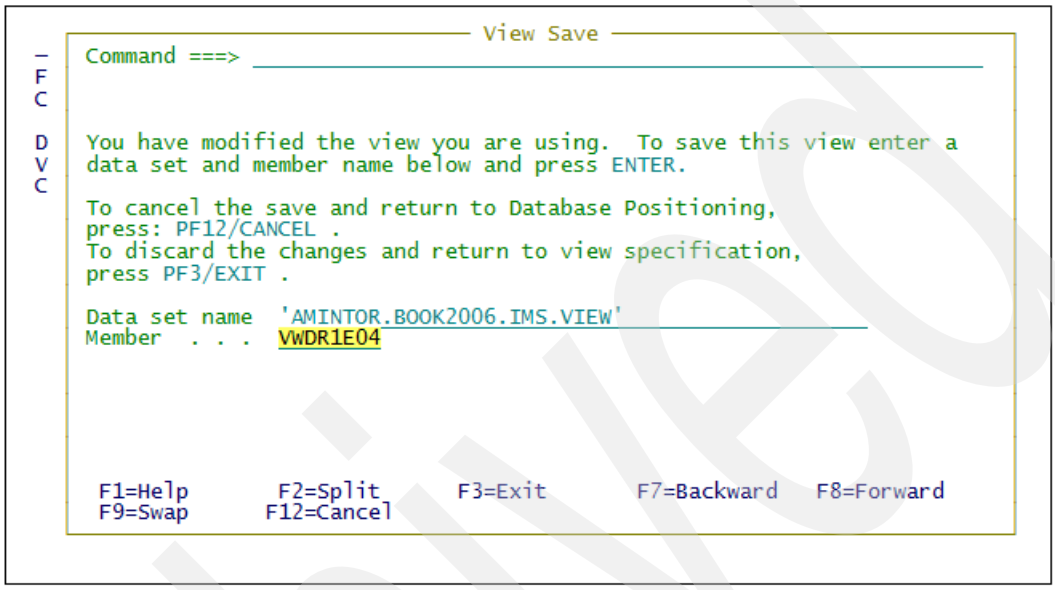


Figure 26-84 Saving as VWDR1E04

After pressing Enter, the view is saved, as shown in Figure 26-85.

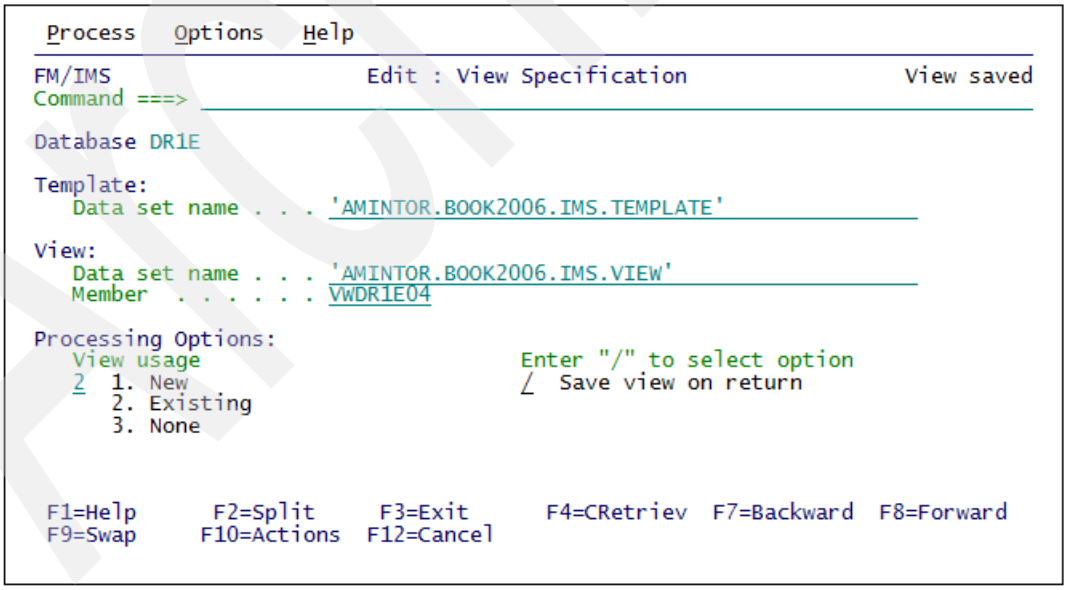


Figure 26-85 View saved

26.8.2 Reordering and suppressing fields from display

We have already shown how to create a view from a template. Here we show how to create a view to show selected fields of a segment.

After we have created the view, we edit the view. We do this by entering an **E** on the command line, as shown in Figure 26-86, and pressing Enter.

```
 Process   Options   Help
FM/IMS                          View Entry Panel
Command ===> e

CT Create view from template          E  Edit view
CM Create view from model             U  Update view from template

Template:
   Data set name . 'AMINTOR.BOOK2006.IMS.TEMPLATE'
   Member  . . . . DR1E

View:
   Data set name . 'AMINTOR.BOOK2006.IMS.VIEW'
   Member  . . . . VWDR1E04

Model view:
   Data set name .
   Member  . . . .

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap      F10=Actions  F12=Cancel
```

Figure 26-86 View Entry Panel

The Segment Selection panel is now displayed, as shown in Figure 26-87.

```
 Process   Options   Help
FM/IMS                    View : Segment Selection
Command ===>                                                Scroll CSR

View     AMINTOR.BOOK2006.IMS.VIEW(VWDR1E04)
Template AMINTOR.BOOK2006.IMS.TEMPLATE(DR1E)

Cmd  Sel  Level          Segment   Description
          1              CUSTOMER
      *    2             COMPANY
  ****  End of data  ****

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=Describe
 F7=Up        F8=Down      F9=Swap      F10=RunTemp  F12=Cancel
```

Figure 26-87 Segment Selection panel

Under the header *Sel* column, an asterisk (*) is displayed. This means that the segment is selected.

We also see information about:

- Where the view is fetched from
- Where the template is stored

As we do not want to show the COMPANY segment, we enter an **S*** on the Cmd line, as shown in Figure 26-88, to deselect the COMPANY segment, and press Enter.

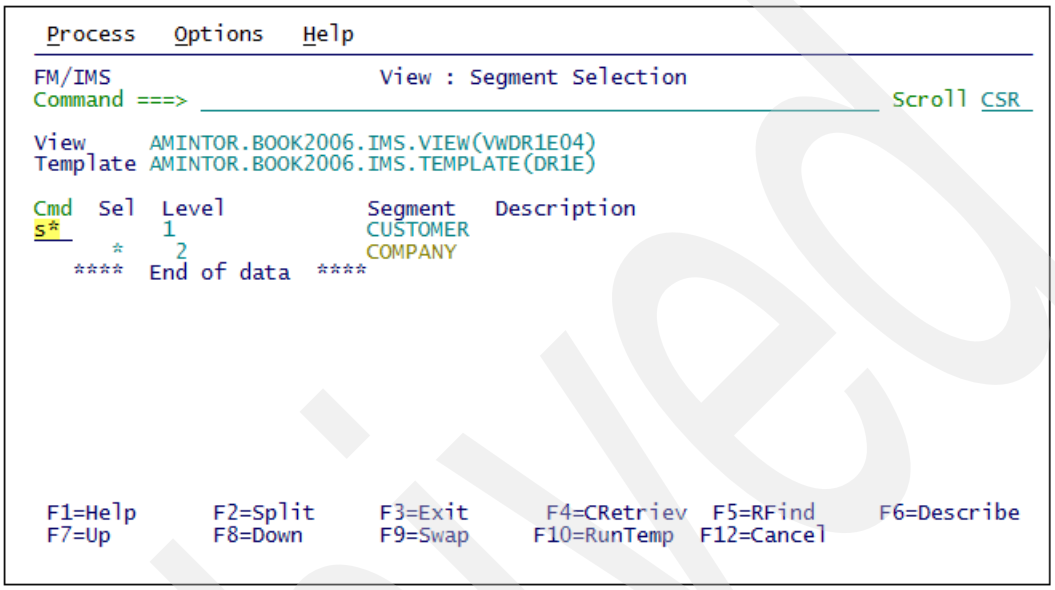


Figure 26-88 Deselect Company segment

We now see, as shown in Figure 26-89, that the COMPANY segment has been deselected.

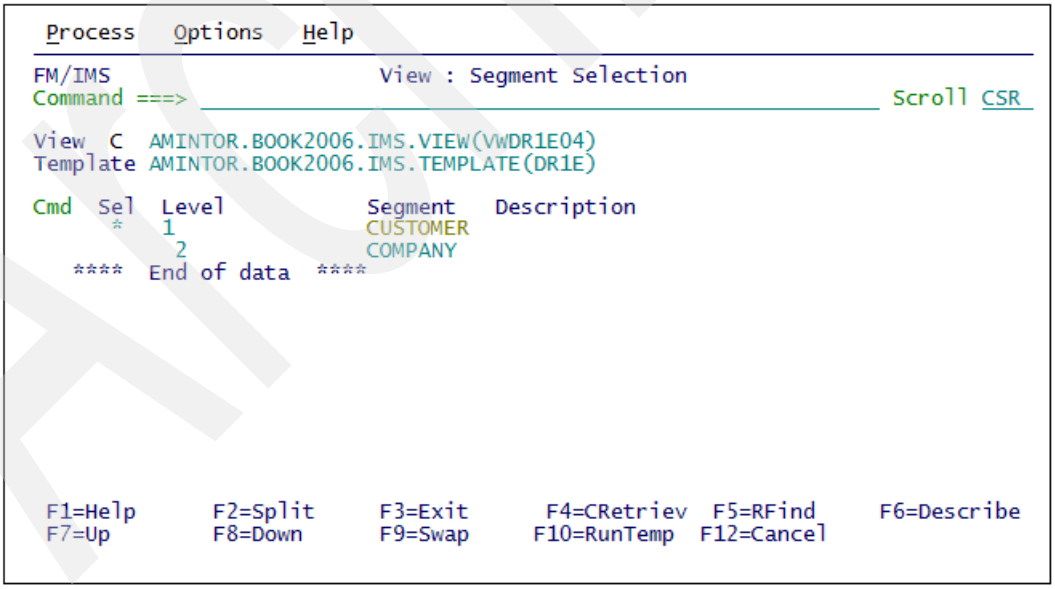


Figure 26-89 Company segment deselected

We now can edit the segment CUSTOMER. This allows us to have selection criteria for the segment and also manipulate the layout of the segment.

To edit the segment, we enter an **E** on the Cmd line to the left of segment COMPANY, as shown in Figure 26-90, and press Enter.

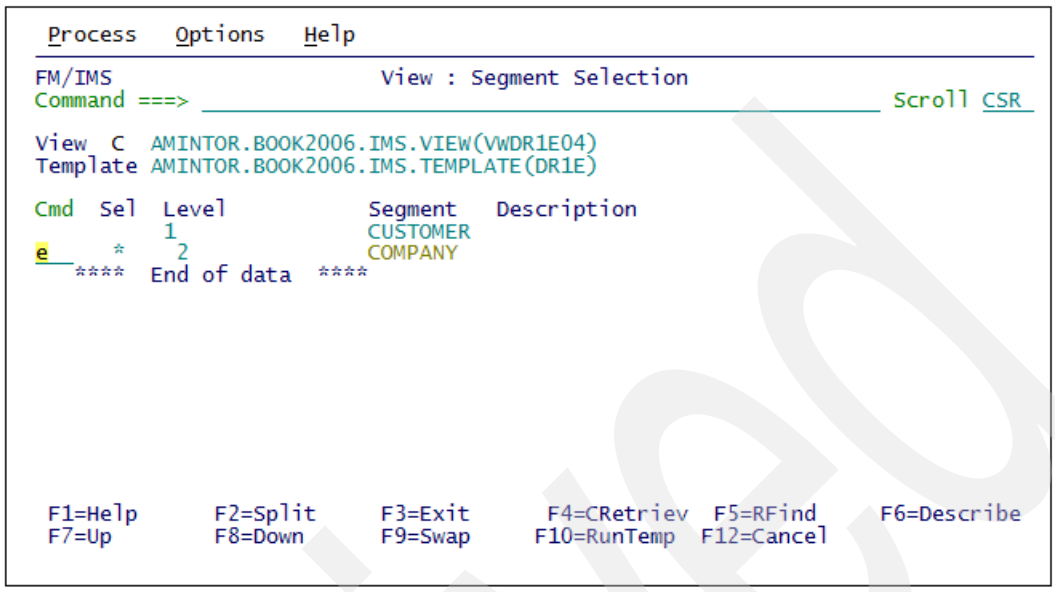


Figure 26-90 Edit criteria for COMPANY segment

The fields we want to display in this view are:

- COMPANY-CUST
- NO-SHARES-2
- NO-SHARES-3
- CUST-LVL

We also display the records that have a CUST-LEVEL = A.

We select those fields by entering an S on the Cmd line, as shown in Figure 26-91.

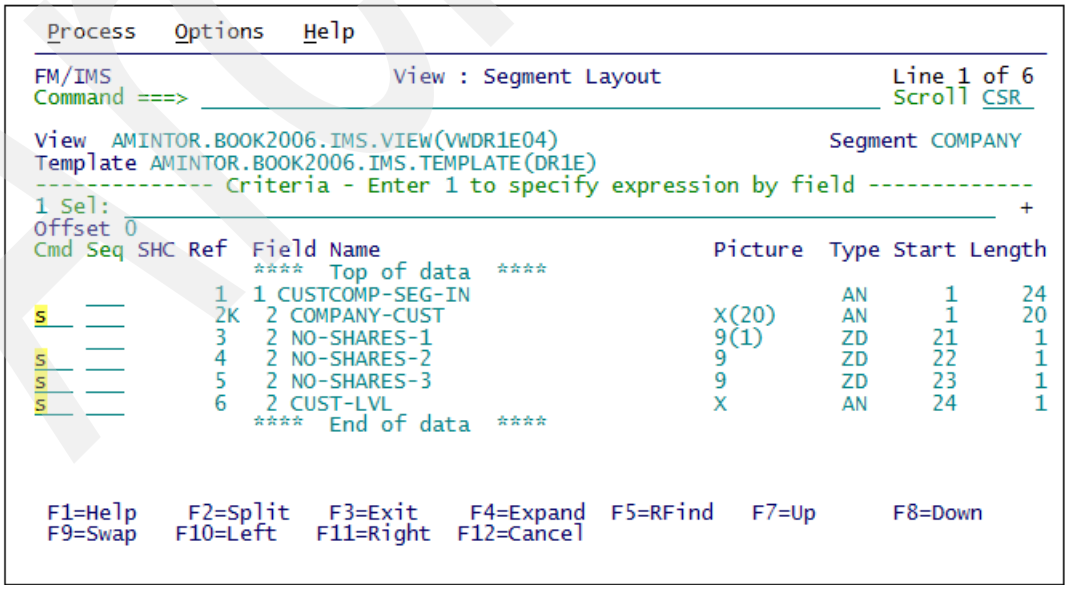


Figure 26-91 Selecting fields to display

After pressing Enter, the Segment Layout panel is redisplayed, as shown in Figure 26-92. Under the header SHC, we see an **S** for those fields that have been selected for display.

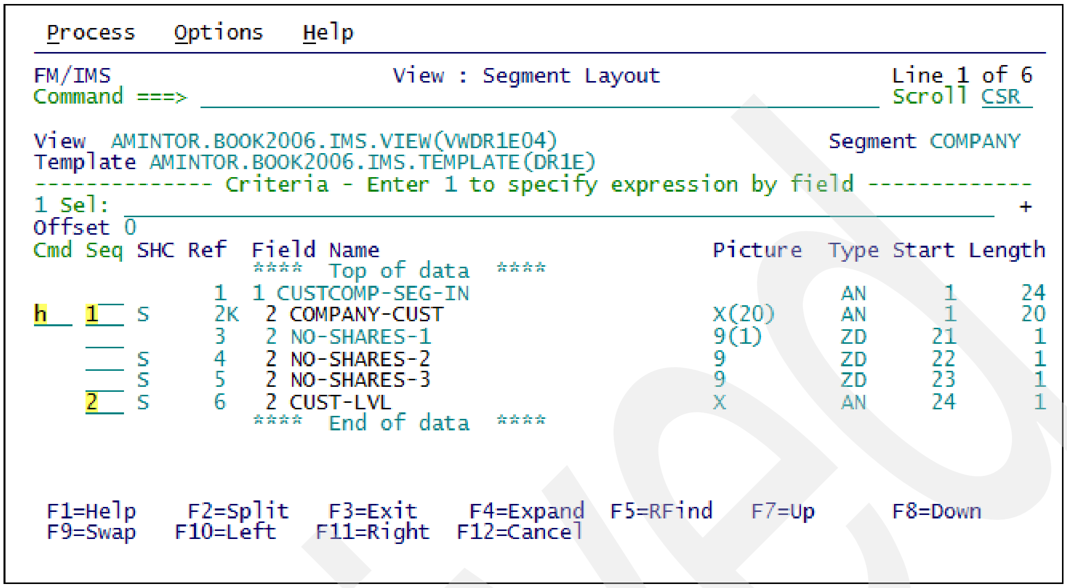


Figure 26-92 Sequence number and H indicator

We now reorder the display of the fields. This is done by defining the sequence numbers, as shown in Figure 26-92.

The order the fields are displayed in is as follows:

1. COMPANY-CUST (We choose Seq nr 1.)
2. CUST-LVL (We choose Seq nr 2.)
3. NO-SHARES-2
4. NO-SHARES-3

We also see an **H** in the Cmd field. This means that the field is not scrolled when we display the data and use the scroll key.

After pressing Enter, the Segment Layout panel is redisplayed, as shown in Figure 26-93.

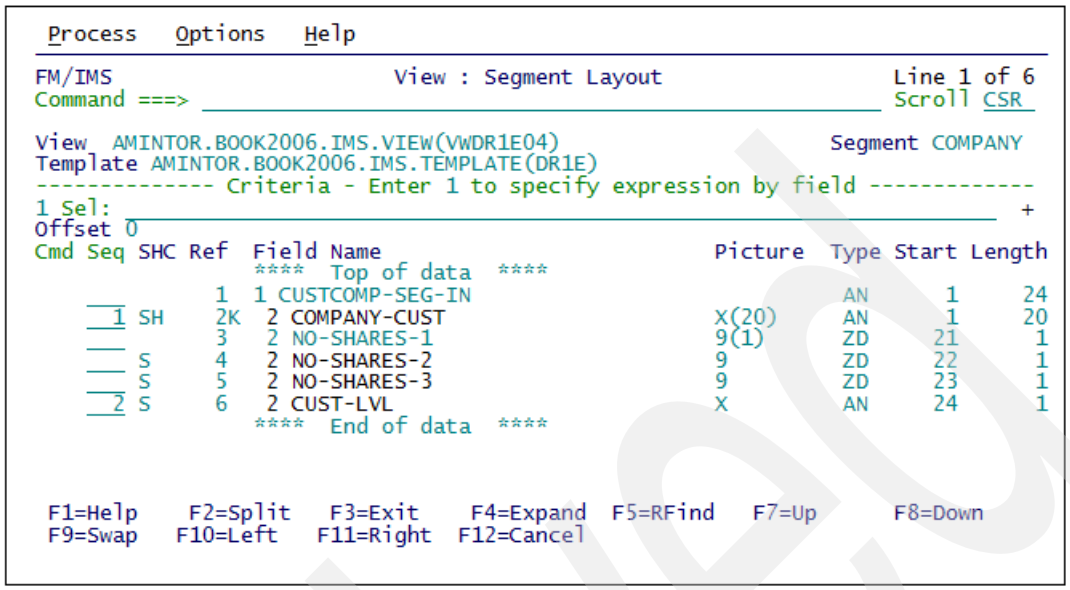


Figure 26-93 Segment Layout panel

We now do the last thing for this view, which is to specify which segments we want to display. We do this by typing a **1** on the command line, as shown in Figure 26-94.

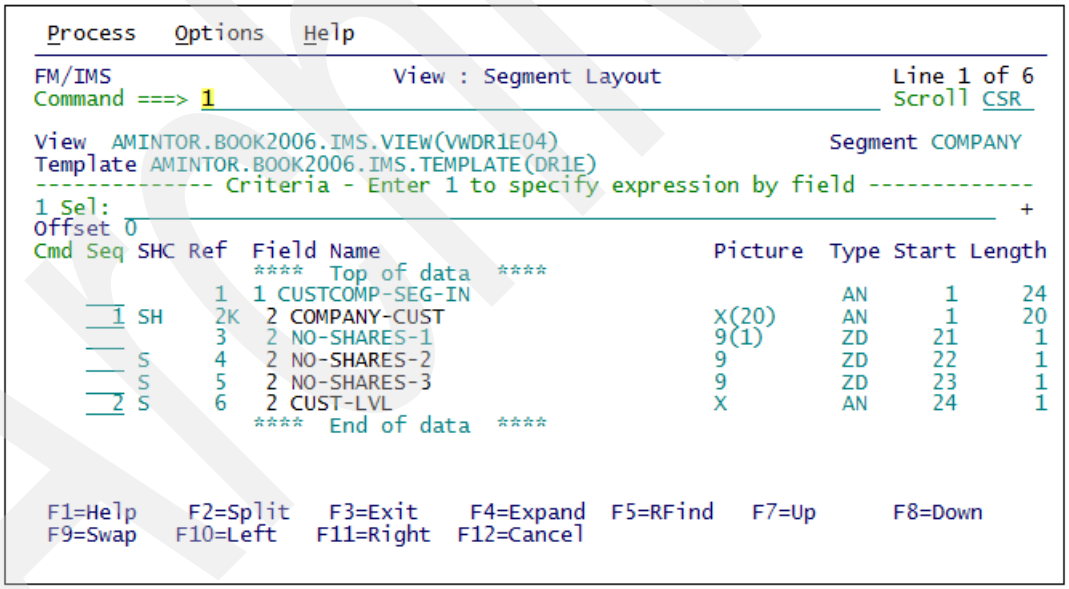


Figure 26-94 Segment Layout panel

After pressing Enter, the Field Selections Criteria Panel is displayed. The only segments we want to display are the ones with an A in the CUST-LVL field. We enter an equals sign (=) in the Op field and an **A** in the Value field, as shown in Figure 26-95.

```
 Process   Options   Help
FM/IMS                Field Selection Criteria                  Line 1 of 6
Command ===>                                                    Scroll CSR

Cmd Con ( Field Name                          Op  Value                    )
    <-> - <---+----1----+----2----+----3----> <-> <---+----1----+----2----+-- -
*** **** Top of data  ****
         CUSTCOMP-SEG-IN
___      _ COMPANY-CUST                       ___ ____________________________ _
___  AND _ NO-SHARES-1                        ___ ____________________________ _
___  AND _ NO-SHARES-2                        ___ ____________________________ _
___  AND _ NO-SHARES-3                        ___ ____________________________ _
___  AND _ CUST-LVL                           =   A                          _
*** **** End of data  ****


 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=RunTemp
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 26-95 Field Selection Criteria panel

After pressing Enter, the Segment Layout panel is displayed, as shown in Figure 26-96. We see that the line for selection criteria has been changed to Ä6 = 'A'. We exit by pressing F3.

```
 Process   Options   Help
FM/IMS                    View : Segment Layout                 Line 1 of 6
Command ===>                                                    Scroll CSR

View  AMINTOR.BOOK2006.IMS.VIEW(VWDR1E04)                  Segment COMPANY
Template AMINTOR.BOOK2006.IMS.TEMPLATE(DR1E)
------------- Criteria - Enter 1 to specify expression by field -------------
1 Sel: #6 = 'A'                                                             +
Offset 0
Cmd Seq SHC Ref  Field Name                          Picture  Type Start Length
              ****  Top of data  ****
            1  1 CUSTCOMP-SEG-IN                                AN     1     24
___ 1 SH    2K  2 COMPANY-CUST                       X(20)     AN     1     20
___         3   2 NO-SHARES-1                        9(1)      ZD    21      1
___ S       4   2 NO-SHARES-2                        9         ZD    22      1
___ S       5   2 NO-SHARES-3                        9         ZD    23      1
___ 2 S     6   2 CUST-LVL                           X         AN    24      1
              ****  End of data  ****


 F1=Help    F2=Split   F3=Exit    F4=Expand  F5=RFind   F7=Up      F8=Down
 F9=Swap    F10=Left   F11=Right  F12=Cancel
```

Figure 26-96 Selection Layout panel

After pressing PF3, the View Segment Selection panel is displayed, as shown in Figure 26-97.

```
 Process   Options   Help
FM/IMS                    View : Segment Selection        Must read all roots
Command ===>                                                        Scroll CSR

View  C  AMINTOR.BOOK2006.IMS.VIEW(VWDR1E04)
Template AMINTOR.BOOK2006.IMS.TEMPLATE(DR1E)

Cmd  Sel  Level          Segment   Description
          1              CUSTOMER
      *    2             COMPANY
   ****  End of data  ****









 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=Describe
 F7=Up        F8=Down      F9=Swap     F10=RunTemp  F12=Cancel
```

Figure 26-97 View Segment Selection panel

After pressing F3, the View Entry Panel is displayed, as shown in Figure 26-98.

```
 Process   Options   Help
FM/IMS                          View Entry Panel                  View saved
Command ===>

CT Create view from template          E  Edit view
CM Create view from model             U  Update view from template

Template:
   Data set name . 'AMINTOR.BOOK2006.IMS.TEMPLATE'
   Member  . . . . DR1E

View:
   Data set name . 'AMINTOR.BOOK2006.IMS.VIEW'
   Member  . . . . VWDR1E04

Model view:
   Data set name .
   Member  . . . .



 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 26-98 View Specification panel

We now go to edit the database, as shown in Figure 26-99.

```
 Process   Options   Help
FM/IMS                    Edit : Database Positioning      Chkpt FM000001 taken
Command ===> _____________________________________________________ Scroll CSR

Database DR1E                          Key sequence                 Format TABL
View     AMINTOR.BOOK2006.IMS.VIEW(VWDR1E04 Key
Cmd SXE Level     Segment   Description     len  Key value
___ S   1         CUSTOMER                  25  AmiRoot1_____________________
s__  XE  2        COMPANY                   20  ....................
   ****  End of data  ****

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=Key >=    F7=Backward
 F8=Forward   F9=Swap      F10=Actions  F12=Cancel
```

Figure 26-99 Database Positioning panel

After selecting the segment, we see the display format we have defined for the VIEW5, as shown in Figure 26-100.

```
 Process   Options   Help
FM/IMS                        Edit : IMS Database DR1E
Command ===> _____________________________________________________ Scroll CSR
          CHKPID FM000001 Autosave ON  SHOW SUP ON  Scope DB  Format TABL
                 COMPANY-CUST          CUST-LVL NO-SHARES-2 NO-SHARES-3
Cmd Level Segment #2                   #6              #4           #5
                 AN 1:20               AN 24:1      ZD 22:1      ZD 23:1
                 <---+----1----+----> -                <>           <>
                 ****   Top of window   ****
___   2   COMPANY  Ami01 company        A                2            2
___   2   COMPANY  MARIANNA CINZIA      A                3            2
___   2   COMPANY  NEW COMPANY          A                5            3
___   2   COMPANY  IBM                  A                7            7
___   2   COMPANY  CINZIA MARIANNA      A                7            8
___   2   COMPANY  Glass_and_Luget_plc  A                4            5
___   2   COMPANY  Headworth_Electrical A                4            4
___   2   COMPANY  ShareSelect          A                3            5
___   2   COMPANY  SportSelect          A                3            2
___   2   COMPANY  Veck_transports      A                3            3
___   2   COMPANY  Glass_and_Luget_plc  A                3            3
___   2   COMPANY  Headworth_Electrical A                2            3
 F1=Help     F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up       F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-100 Using VWDR1E04

26.9 More functions

In this section we discuss several additional functions that you can use.

26.9.1 Finding/changing data

We use the FIND primary command to locate a string within selected segments in the database. We can use the CHANGE command to find a string with multiple segments and replace it with another string. The FIND command is available in both the Browse and Edit Database panels. The CHANGE command is only available in the Edit Database panel. To search data for a specific string, we use the FIND primary command (which can be abbreviated to **F** or **/**.

Find a string anywhere in the data when not using a view

To find the string *CINZIA*, we type in the command **F CINZIA**, as shown in Figure 26-101.

```
 Process    Options    Help
FM/IMS                          Edit : IMS Database DR1E
Command ===>                                                  Scroll CSR
             CHKPID FM000001 Autosave ON  SHOW SUP ON  Scope DB  Format TABL
                    COMPANY-CUST         CUST-LVL NO-SHARES-2 NO-SHARES-3
Cmd Level Segment   #2                   #6                #4          #5
                    AN 1:20              AN 24:1      ZD 22:1     ZD 23:1
                    <---+----1----+----> -                 <>          <>
                    ****   Top of window   ****
___   2   COMPANY   Ami01 company         A                 2           2
___   2   COMPANY   MARIANNA CINZIA       A                 3           2
___   2   COMPANY   NEW COMPANY           A                 5           3
___   2   COMPANY   IBM                   A                 7           7
___   2   COMPANY   CINZIA MARIANNA       A                 7           8
___   2   COMPANY   Glass_and_Luget_plc   A                 4           5
___   2   COMPANY   Headworth_Electrical  A                 4           4
___   2   COMPANY   ShareSelect           A                 3           5
___   2   COMPANY   SportSelect           A                 3           2
___   2   COMPANY   Veck_transports       A                 3           3
___   2   COMPANY   Glass_and_Luget_plc   A                 3           3
___   2   COMPANY   Headworth_Electrical  A                 2           3
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 26-101 FIND command

After pressing Enter, FM/IMS highlights the first occurrence of the string. If we want to find the next occurrence, we press F5 (repeat find), as shown in Figure 26-102.

```
 Process    Options    Help
FM/IMS                          Edit : IMS Database DR1E      Chars cinzia found
Command ===> f cinzia                                         Scroll CSR
             CHKPID FM000001 Autosave ON  SHOW SUP ON  Scope DB  Format TABL
                    COMPANY-CUST         CUST-LVL NO-SHARES-2 NO-SHARES-3
Cmd Level Segment   #2                   #6                #4          #5
                    AN 1:20              AN 24:1      ZD 22:1     ZD 23:1
                    <---+----1----+----> -                 <>          <>
                    ****   Top of window   ****
___   2   COMPANY   Ami01 company         A                 2           2
___   2   COMPANY   MARIANNA CINZIA       A                 3           2
___   2   COMPANY   NEW COMPANY           A                 5           3
___   2   COMPANY   IBM                   A                 7           7
___   2   COMPANY   CINZIA MARIANNA       A                 7           8
___   2   COMPANY   Glass_and_Luget_plc   A                 4           5
___   2   COMPANY   Headworth_Electrical  A                 4           4
___   2   COMPANY   ShareSelect           A                 3           5
___   2   COMPANY   SportSelect           A                 3           2
___   2   COMPANY   Veck_transports       A                 3           3
___   2   COMPANY   Glass_and_Luget_plc   A                 3           3
___   2   COMPANY   Headworth_Electrical  A                 2           3
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 26-102 Result after find

Find all occurrences of a string when not using a view

To find all occurrences of data, we add the parameter ALL to the FIND command, as shown in Figure 26-103.

```
 Process   Options   Help
FM/IMS                        Edit : IMS Database DR1E
Command ===> f new all                                        Scroll CSR
   CHKPID FM000001 Autosave ON              Scope DB  Col 1     Format CHAR
Cmd Level Segment    ----+----1----+----2----+
                     ****   Top of window   ****
___ 1     CUSTOMER   AmiRoot1
___  2    COMPANY    AmiSeg1              272C
___  2    COMPANY    Ami01 company        222A
___  2    COMPANY    MARIANNA CINZIA      132A
___  2    COMPANY    NEW COMPANY          353A
___  2    COMPANY    NE2 COMPANY          212c
___  2    COMPANY    NE3 COMPANY          272c
___ 1     CUSTOMER   AAB
___  2    COMPANY                         122
___  2    COMPANY    AAB                  319B
___  2    COMPANY    IBM                  777A
___  2    COMPANY    08                   2953
___ 1     CUSTOMER   ABBA CANNING COMP
___  2    COMPANY                         X02
___  2    COMPANY     001                 8881
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-103 Find ALL

As shown in Figure 26-104, the occurrences are highlighted, and we also get information about how many were found.

```
 Process   Options   Help
FM/IMS                        Edit : IMS Database DR1E
Command ===>                                                  Scroll CSR
Search arg new found 6 times
Cmd Level Segment    ----+----1----+----2----+
___  2    COMPANY    NEW COMPANY          353A
___  2    COMPANY    NE2 COMPANY          212c
___  2    COMPANY    NE3 COMPANY          272c
___ 1     CUSTOMER   AAB
___  2    COMPANY                         122
___  2    COMPANY    AAB                  319B
___  2    COMPANY    IBM                  777A
___  2    COMPANY    08                   2953
___ 1     CUSTOMER   ABBA CANNING COMP
___  2    COMPANY                         X02
___  2    COMPANY     001                 8881
___  2    COMPANY    KAVIAR               3A3B
___  2    COMPANY    NEW COMPANY          123B
___  2    COMPANY    NEW2COMPANY          153C
___ 1     CUSTOMER   ALBANE
___  2    COMPANY                         765
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-104 Result after find

Find a string in a specific position in the data when not using a view

We use the command F OM 15 20 ALL, then FMM/IMS search for the string between position 15 and 20, as shown in Figure 26-105.

```
 Process   Options   Help
FM/IMS                          Edit : IMS Database DR1E
Command ===> f om 15 20 all                                    Scroll CSR
   CHKPID FM000001 Autosave ON                Scope DB  Col 1  Format CHAR
Cmd Level Segment    ----+----1----+----2----+
                     ****   Top of window   ****
___ 1     CUSTOMER   AmiRoot1
___  2    COMPANY    AmiSeg1              272C
___  2    COMPANY    Ami01 company        222A
___  2    COMPANY    MARIANNA CINZIA      132A
___  2    COMPANY    NEW COMPANY          353A
___  2    COMPANY    NE2 COMPANY          212c
___  2    COMPANY    NE3 COMPANY          272c
___ 1     CUSTOMER   AAB
___  2    COMPANY                         122
___  2    COMPANY    AAB                  319B
___  2    COMPANY    IBM                  777A
___  2    COMPANY    08                   2953
___ 1     CUSTOMER   ABBA CANNING COMP
___  2    COMPANY                         X02
___  2    COMPANY     001                 8881
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-105 Find in a specific segment

Tip: You can now specify the parameters in any order you prefer, and you obtain the same results. This means that you can type the command (on the command line in the Browse, Edit, or Object List utility) using either of the following syntax examples:

```
FIND "om" 15 20 ALL (ISPF- like)
FIND ALL "om" 15 20
```

We see the result in Figure 26-106.

```
 Process   Options   Help
FM/IMS                          Edit : IMS Database DR1E
Command ===>                                                   Scroll CSR
Search arg om found 2 times
Cmd Level Segment    ----+----1----+----2----+
                     ****   Top of window   ****
___ 1     CUSTOMER   AmiRoot1
___  2    COMPANY    AmiSeg1              272C
___  2    COMPANY    Ami01 company        222A
___  2    COMPANY    MARIANNA CINZIA      132A
___  2    COMPANY    NEW COMPANY          353A
___  2    COMPANY    NE2 COMPANY          212c
___  2    COMPANY    NE3 COMPANY          272c
___ 1     CUSTOMER   AAB
___  2    COMPANY                         122
___  2    COMPANY    AAB                  319B
___  2    COMPANY    IBM                  777A
___  2    COMPANY    08                   2953
___ 1     CUSTOMER   ABBA CANNING COMP
___  2    COMPANY                         X02
___  2    COMPANY     001                 8881
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-106 Search result

Find a string in a specific segment

When we want to search in a specific segment, we use the FIND command. Here we only want to search in the COMPANY segment. The result is shown in Figure 26-107.

```
 Process   Options   Help
FM/IMS                        Edit : IMS Database DR1E
Command ===> f seg(company) an 15 20 all                          Scroll CSR
Search arg an found 224 times
Cmd Level Segment    ----+----1----+----2----+
                     ****   Top of window   ****
___ 1     CUSTOMER   AmiRoot1
___  2    COMPANY    AmiSeg1              272C
___  2    COMPANY    Ami01 company        222A
___  2    COMPANY    MARIANNA CINZIA      132A
___  2    COMPANY    NEW COMPANY          353A
___  2    COMPANY    NE2 COMPANY          212c
___  2    COMPANY    NE3 COMPANY          272c
___ 1     CUSTOMER   AAB
___  2    COMPANY                         122
___  2    COMPANY    AAB                  319B
___  2    COMPANY    IBM                  777A
___  2    COMPANY    08                   2953
___ 1     CUSTOMER   ABBA CANNING COMP
___  2    COMPANY                         X02
___  2    COMPANY     001                 8881
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-107 Find in a specific segment

Finding strings when a view is used

As shown in Figure 26-108, we can use the field reference number when we do a find.

The field reference number for the NO-SHARES-1 is #3.

The field reference number for the NO-SHARES-3 is #5.

```
 Process   Options   Help
FM/IMS                        Edit : IMS Database DR1E
Command ===> f 2 #3 all                                           Scroll CSR
Search arg 2 found 9 times
                     COMPANY-CUST         NO-SHARES-1 NO-SHARES-2 NO-SHARES-3
Cmd Level Segment    #2                         #3          #4          #5
                     AN 1:20                 ZD 21:1     ZD 22:1     ZD 23:1
                     <---+----1----+---->         <>          <>          <>
___ 1     CUSTOMER   Key=AmiRoot1
___  2    COMPANY    AmiSeg1                     2           7           2
___  2    COMPANY    Ami01 company               2           2           2
___  2    COMPANY    MARIANNA CINZIA             1           3           2
___  2    COMPANY    NEW COMPANY                 3           5           3
___  2    COMPANY    NE2 COMPANY                 2           1           2
___  2    COMPANY    NE3 COMPANY                 2           7           2
___ 1     CUSTOMER   Key=AAB
___  2    COMPANY                                1           2           2
___  2    COMPANY    AAB                         3           1           9
___  2    COMPANY    IBM                         7           7           7
___  2    COMPANY    08                          2           9           5
___ 1     CUSTOMER   Key=ABBA CANNING COMP
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-108 Find when data is displayed with a view

We can also perform a FIND in multiple fields, as shown in Figure 26-109.

```
 Process   Options   Help
FM/IMS                          Edit : IMS Database DR1E
Command ===> f 7 #3,#5 all                                    Scroll CSR
Search arg 7 found 4 times
                    COMPANY-CUST          NO-SHARES-1 NO-SHARES-2 NO-SHARES-3
Cmd Level Segment   #2                            #3          #4          #5
                    AN 1:20                    ZD 21:1     ZD 22:1     ZD 23:1
                    <---+----1----+---->           <>          <>          <>
___ 1     CUSTOMER  Key=AmiRoot1
___  2    COMPANY   AmiSeg1                        2           7           2
___  2    COMPANY   Ami01 company                  2           2           2
___  2    COMPANY   MARIANNA CINZIA               1           3           2
___  2    COMPANY   NEW COMPANY                   3           5           3
___  2    COMPANY   NE2 COMPANY                   2           1           2
___  2    COMPANY   NE3 COMPANY                   2           7           2
___ 1     CUSTOMER  Key=AAB
___  2    COMPANY                                 1           2           2
___  2    COMPANY   AAB                           3           1           9
___  2    COMPANY   IBM                           7           7           7
___  2    COMPANY   08                            2           9           5
___ 1     CUSTOMER  Key=ABBA CANNING COMP
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 26-109 Find in multiple fields

Command syntax for the FIND command

The syntax for the FIND command is shown in Figure 26-110.

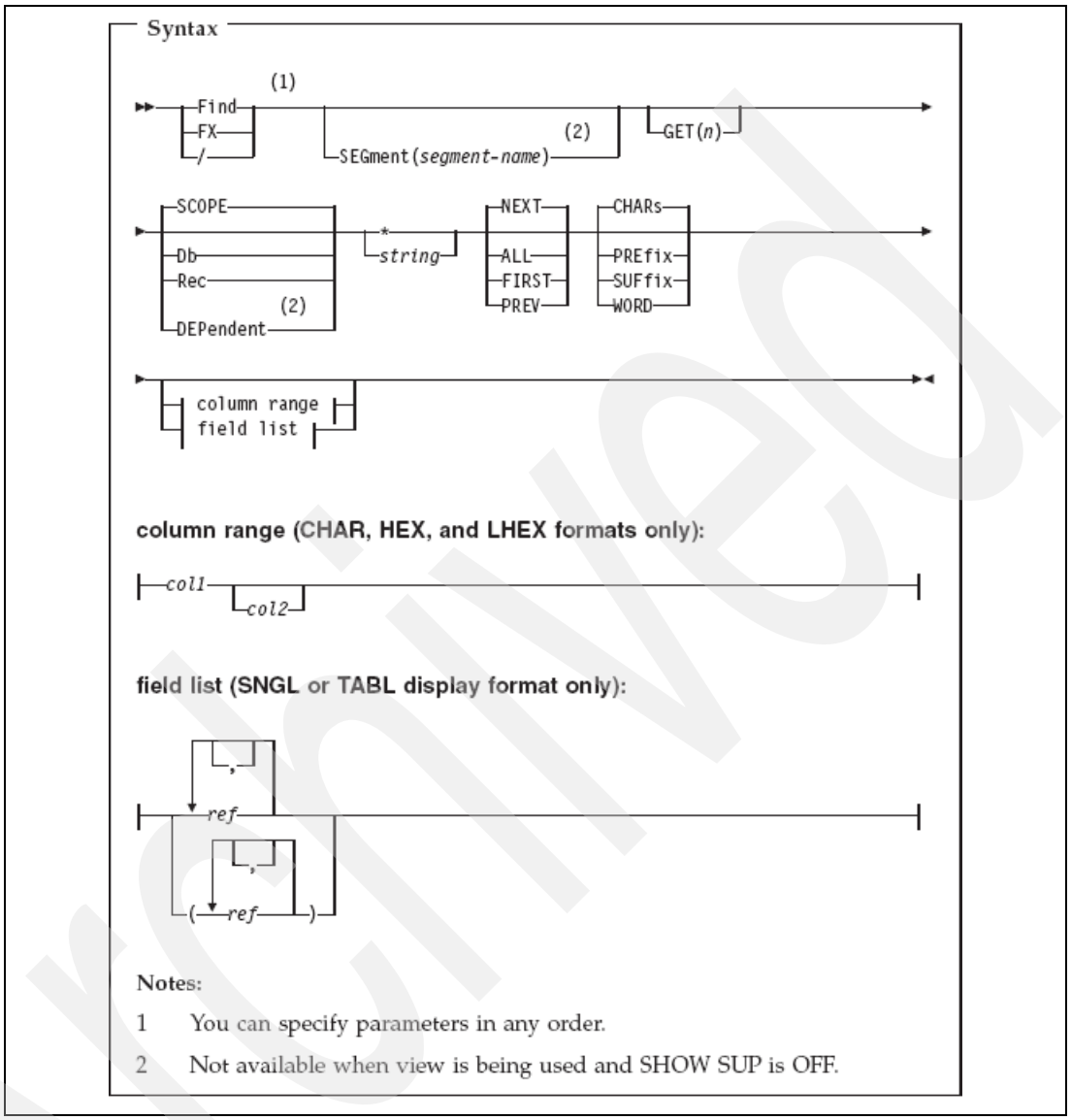


Figure 26-110 Syntax of FIND command

26.9.2 Changing data

We can use the CHANGE command to find a string of multiple segments and replace it with another string.

Command syntax for the change command

The change command syntax is shown in Figure 26-111.

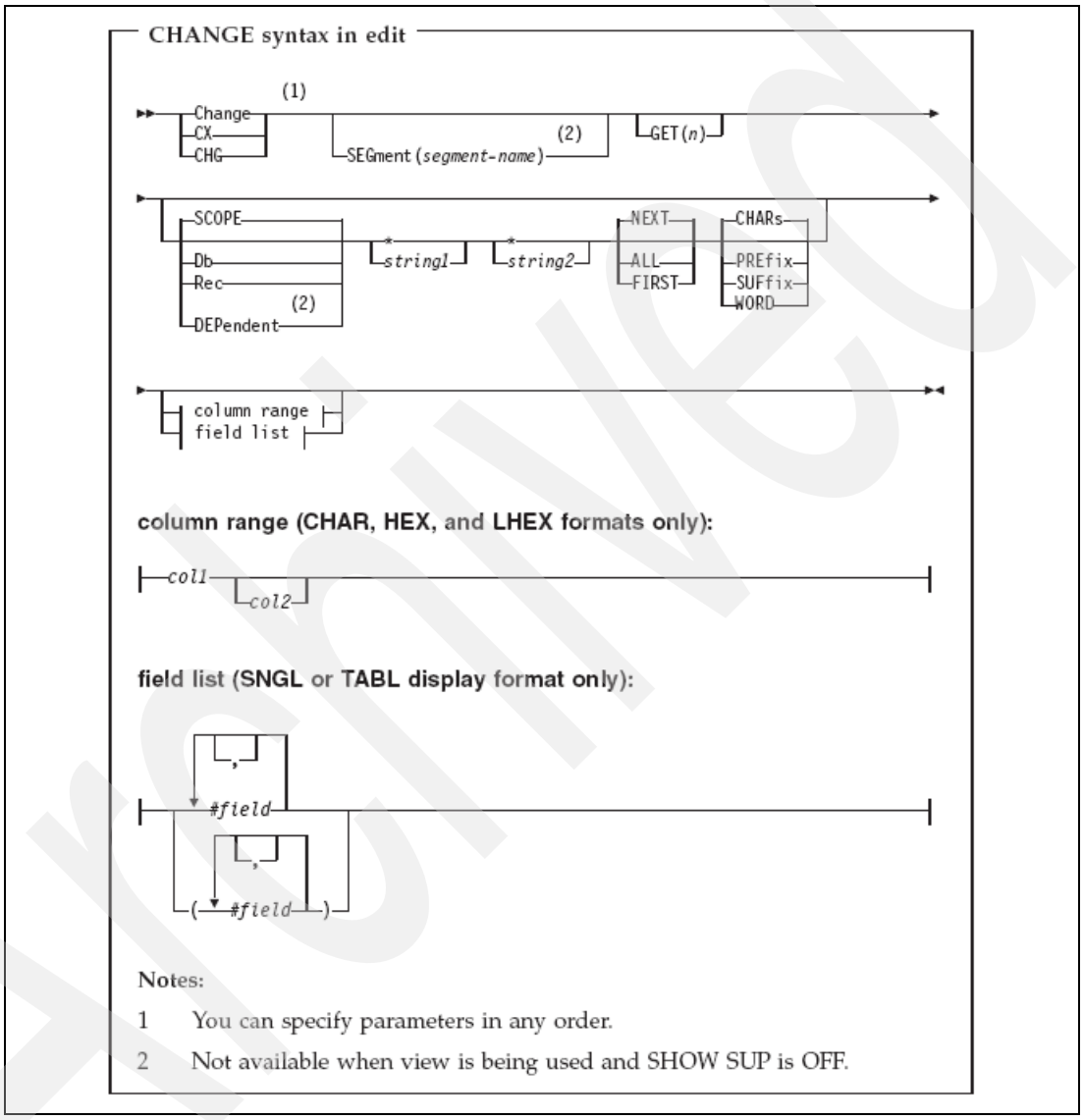


Figure 26-111 Change command

Change data when not using a view

As shown in Figure 26-112, we want to replace string 21 with 33 for all occurrences within the range of columns 20 to 25. We type the command on the command line and press Enter.

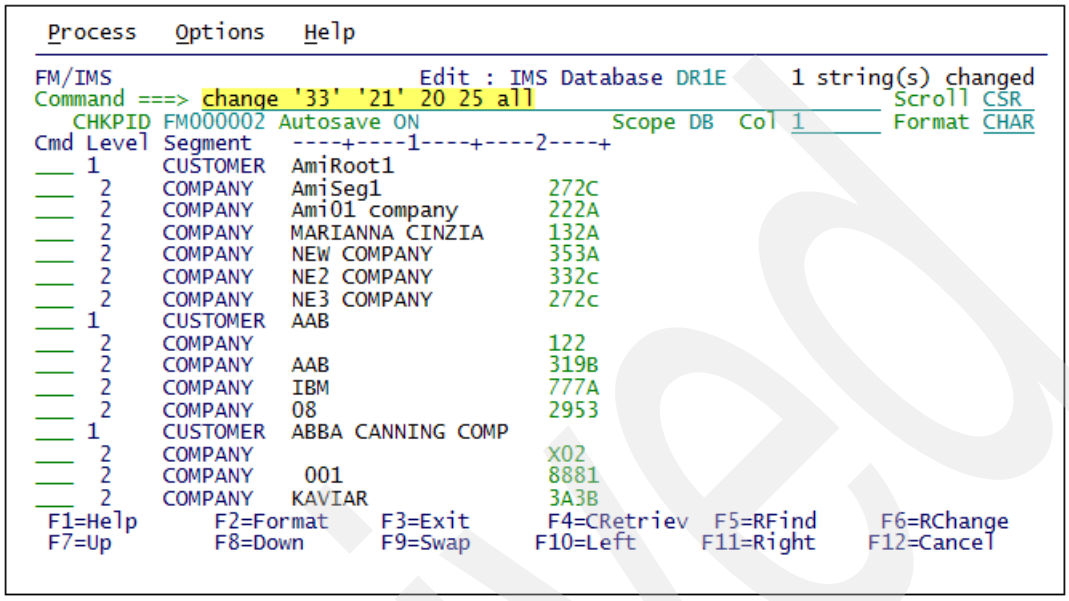


Figure 26-112 Change command

As shown in Figure 26-113, the data has been changed. We also get information about how many strings have been replaced.

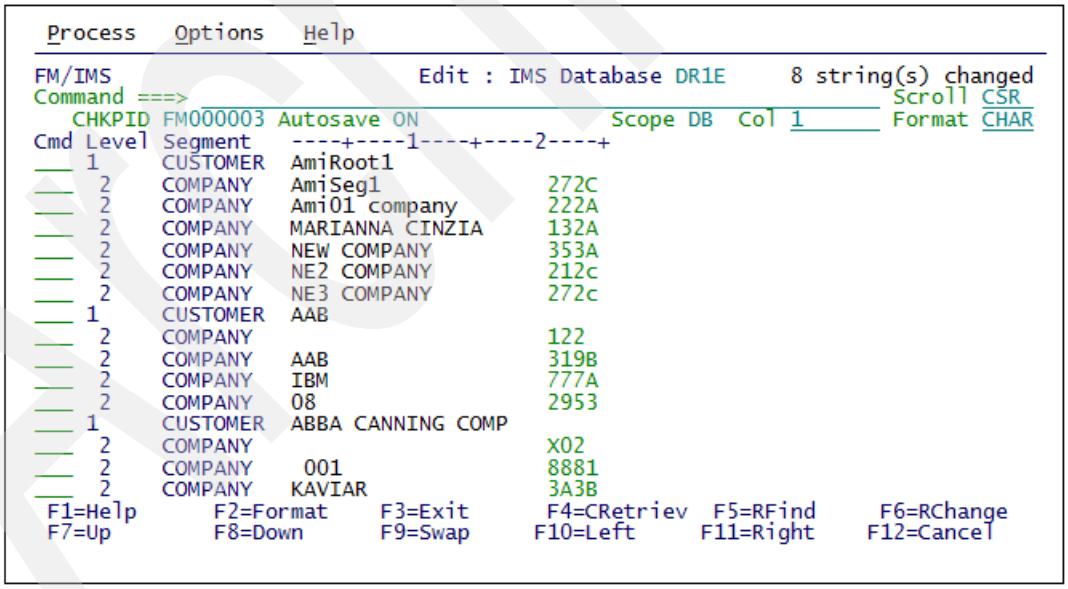


Figure 26-113 After change

Change data when using a view

When using a view, we can use the field reference number as a parameter in the change command. As shown in Figure 26-114, we want to change NO-SHARES-2 /field reference #4 and NO-SHARES-5 / field reference #5.

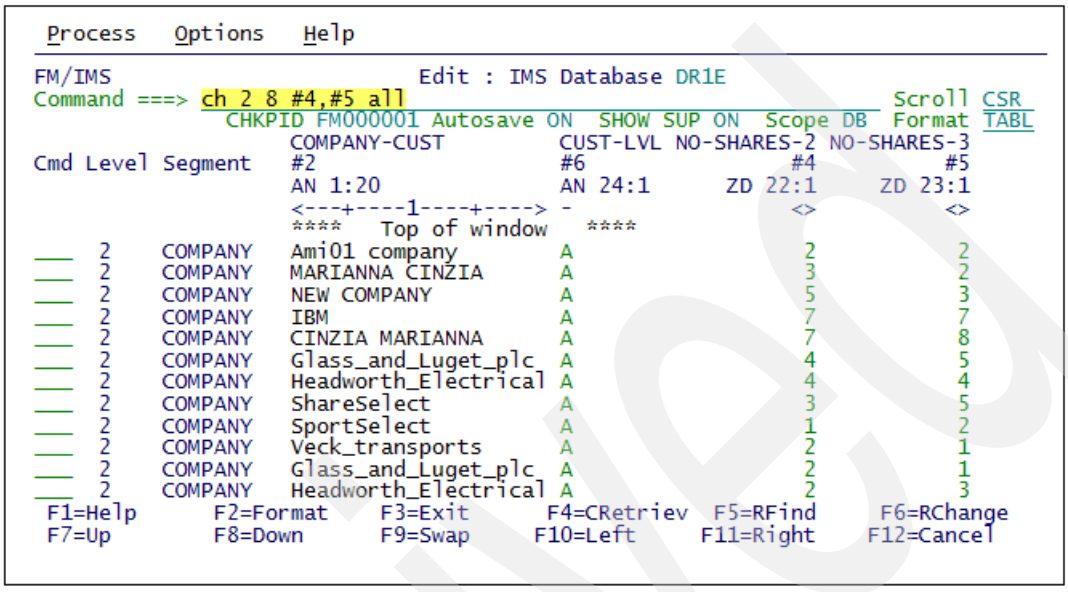


Figure 26-114 Change using field reference number

As shown in Figure 26-115, the change has been made for fields #4 and #5.

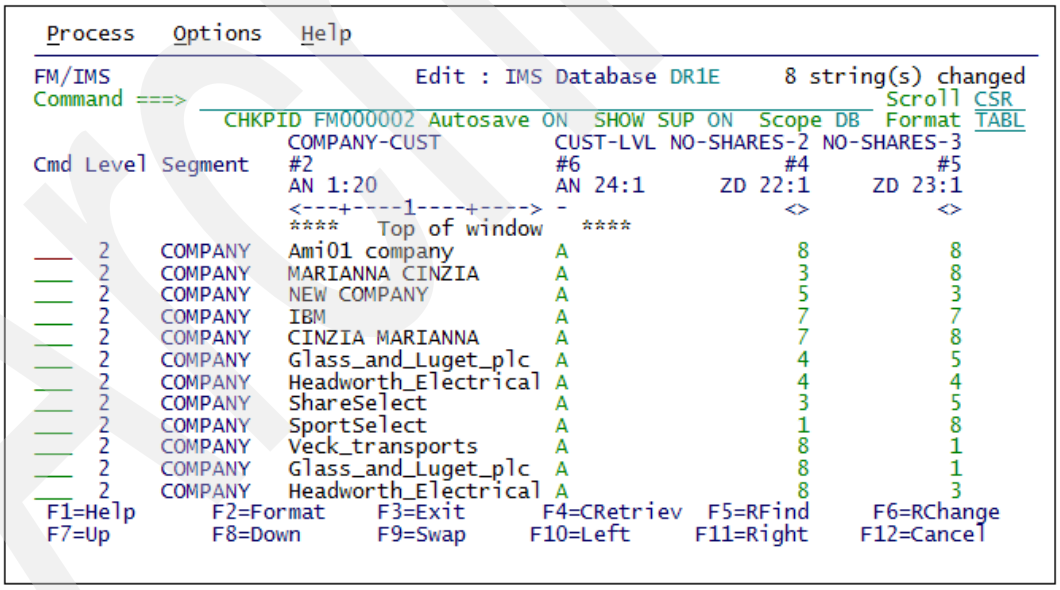


Figure 26-115 After change

Change extended command

When we want to change long strings, the command line does not fit the whole change command. We then type **CX** (change extended) on the command line. The Extended Command Entry Panel is displayed, as shown in Figure 26-116.

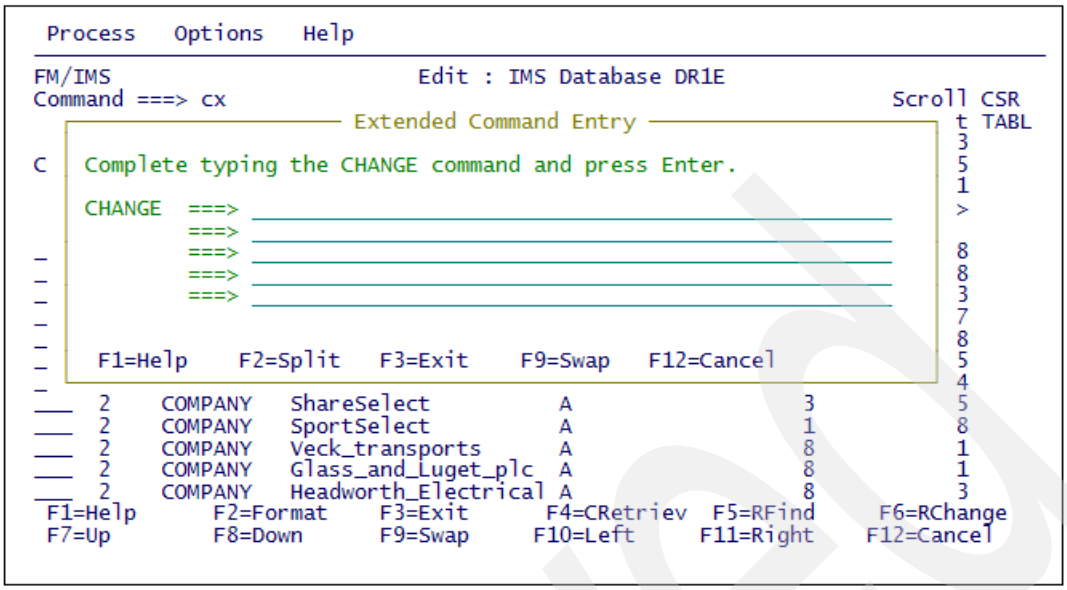


Figure 26-116 Extended Command Entry panel

26.9.3 The Find Error (FE) command

The **FE** command finds data that is not corrected to its COBOL or PL/I declaration, as shown in Figure 26-117.

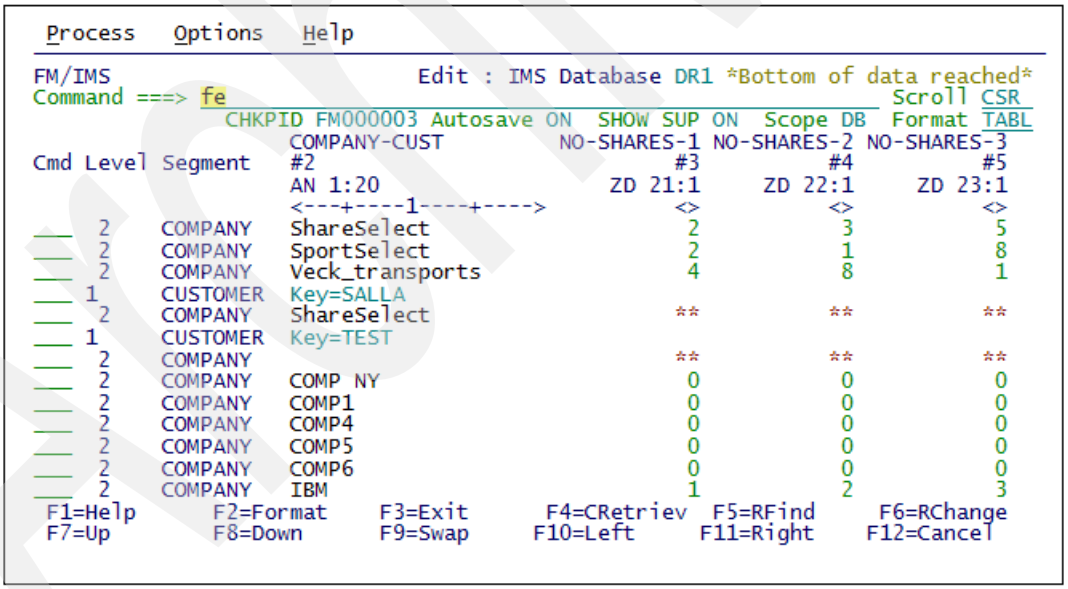


Figure 26-117 FE command

As shown above, fields in error are highlighted and the cursor is placed on the first one.

As shown, if the fields are highlighted, they are all in error, but the report displays in the upper right corner that one error is found. The default in the command is NEXT, so it stops counting after the first FE has been found.

26.9.4 HIERARCH command

The HIERARCH primary command displays a graphical representation of the structure of the database being edited or browsed. The Database Hierarchy panel is shown in Figure 26-118.

You can type this command on all panels within the Browse, Edit, Utilities (except Audit Trail) and Template/View/Criteria Set dialogs, except the initial Entry panel for each function.

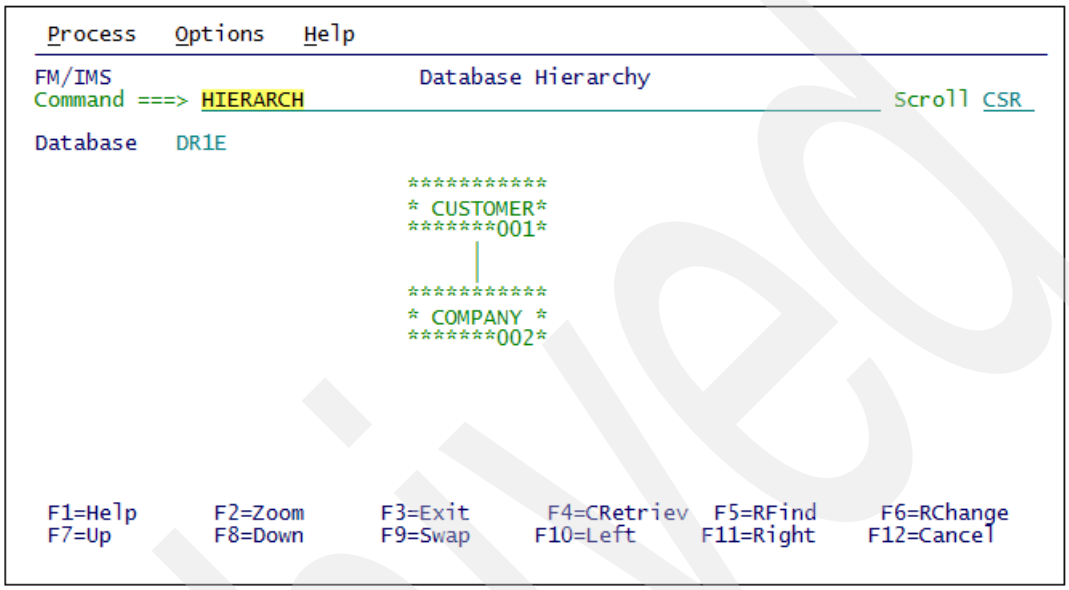


Figure 26-118 HIERARCH command

26.9.5 DBD command

We can get information about the database that is currently accessed, as shown in Figure 26-119.

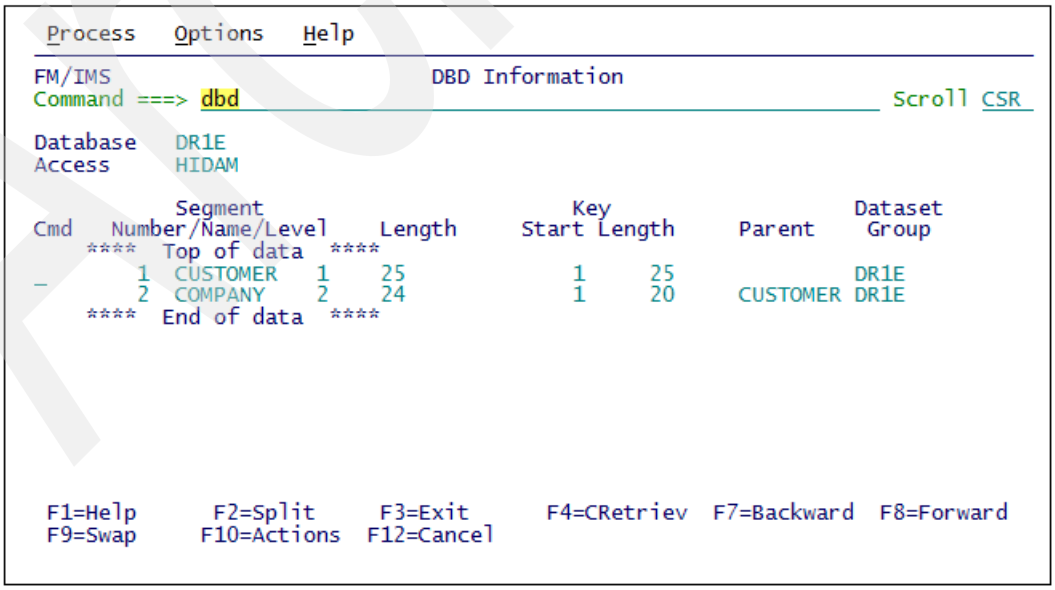


Figure 26-119 DBD primary command

As shown in Figure 26-120, the segment information is displayed on the DBD information.

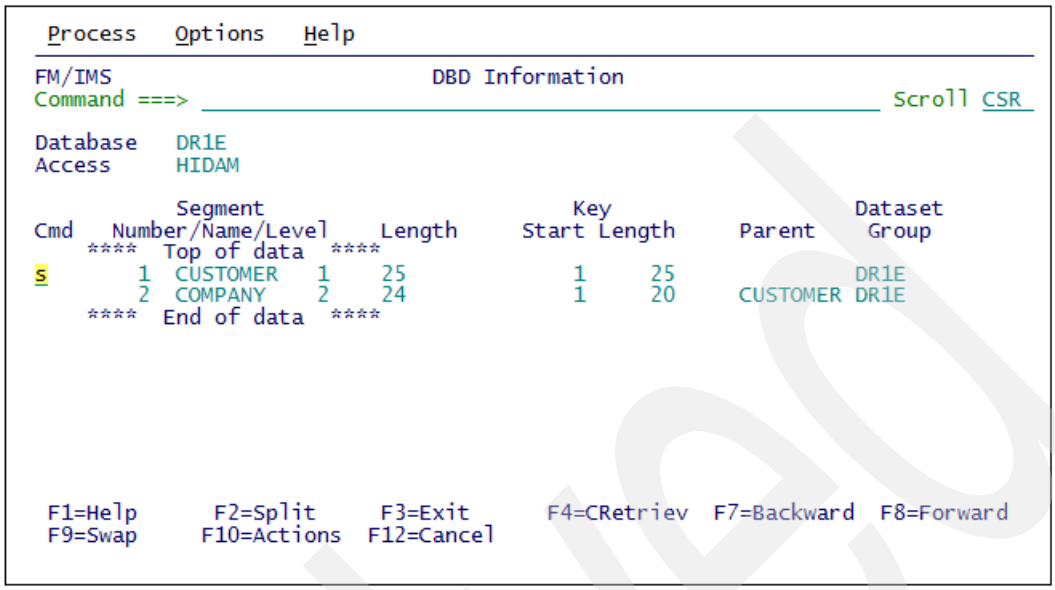


Figure 26-120 DBD information

By entering an **S** on the Cmd line to the left of the segment name, we are shown more information about the selected segment, as shown in Figure 26-121.

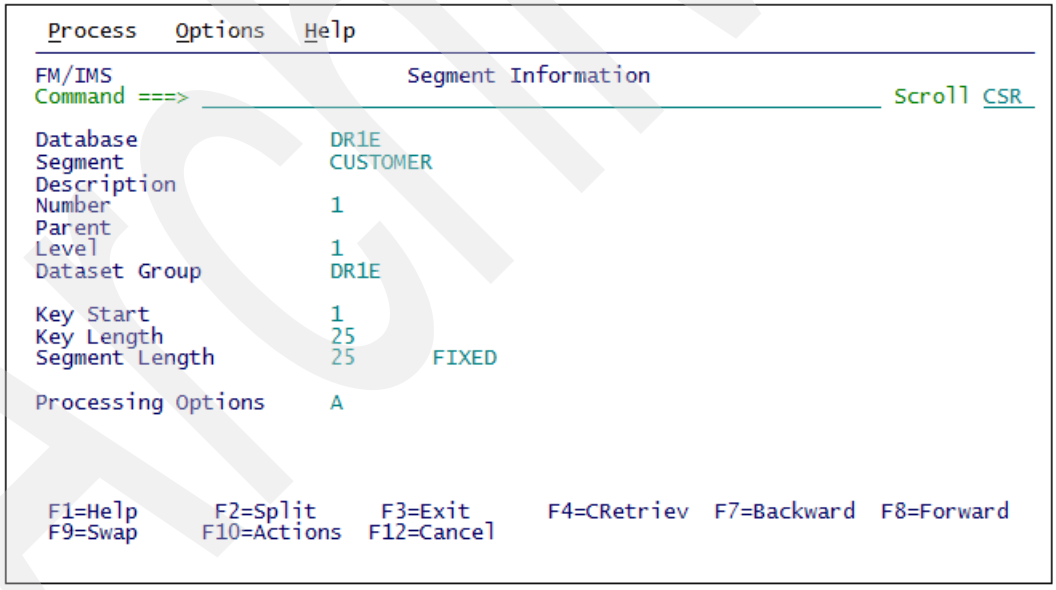


Figure 26-121 Segment information

26.9.6 SEGSTATS command

The SEGSTATS primary command lists statistics for the records included in the current edit or browse session, as shown in Figure 26-122.

```
 Process   Options   Help
FM/IMS                          Edit : IMS Database DR1E
Command ===> segstats                                          Scroll CSR
               CHKPID FM000001 Autosave ON  SHOW SUP ON  Scope DB  Format TABL
                      COMPANY-NAME
Cmd Level Segment     #2
                      AN 1:25
                      <---+----1----+----2---->
                      ****   Top of window   ****
___ 1     CUSTOMER    AmiRoot1
___  2    COMPANY     Key=AmiSeg1
___  2    COMPANY     Key=Ami01 company
___  2    COMPANY     Key=MARIANNA CINZIA
___  2    COMPANY     Key=NEW COMPANY
___  2    COMPANY     Key=NE2 COMPANY
___  2    COMPANY     Key=NE3 COMPANY
___ 1     CUSTOMER    AAB
___  2    COMPANY     Key=
___  2    COMPANY     Key=AAB
___  2    COMPANY     Key=IBM
___  2    COMPANY     Key=08
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 26-122 Segstats Command

Figure 26-123 shows the results of the SEGSTATS command.

```
 Process   Options   Help
FM/IMS                          Edit : IMS Database DR1E
Command ===> segstats                                          Scroll CSR
                          File Manager Messages
 Command ===> ____________________________________________________ Scroll CSR

 IBM File Manager for z/OS IMS Component
 Segment statistics

 Total segments                                                 :  799
 Total data bytes                                               :  19297

 Segment CUSTOMER                                               :  121
 Layout  COMPANY-SEG-IN                                         :  121

 Segment COMPANY                                                :  678
 Layout  CUSTCOMP-SEG-IN                                        :  678

  F1=Help       F2=Split      F3=Exit       F4=CRetriev   F7=Backward
  F8=Forward    F9=Swap       F10=Actions   F12=Cancel
```

Figure 26-123 Segstats panel messages

27

File Manager/CICS

In this chapter we describe some of the capabilities of File Manager and show you how to access CICS resources under the FM/CICS transactions. We illustrate how to Edit, Browse, Print, and alter the status of VSAM Files, Temporary Storage (TS) and Transient Data (TD) queues using FM/CICS. Finally, we demonstrate that all the File Manager utilities (whether Base or IMS) are available without ISPF support.

This chapter covers the following aspects of File Manager/CICS:

- FM/CICS overview
- Architecture overview
- Starting FM/CICS and setting the processing options
- Editing and browsing CICS resources
- Scenario for VSAM sharing and related concepts

27.1 Overview of FM/CICS

IBM File Manager for z/OS introduced a new component to run the FM functions and utilities from the CICS environment. FM/CICS has been introduced with Version 7.1 to provide powerful features that let you access and manage CICS resources through the familiar user-friendly ISPF look-alike interface.

The CICS resources that are supported are:

- VSAM Files (KSDS, ESDS, RRDS, URRDS) and CICS (LOAD) Table
- Temporary storage (TS) queues
- Transient data (TD) queues

File Manager for CICS has incorporated much of the FM/Base and FM/IMS functions into the CICS environment. You can invoke the functions to use all the existing File Manager capabilities such as template/copybook usage, edit, browse, and print in the CICS environment, as well as providing services to inquire about and modify the status and various attributes of the supported CICS resources. Depending on your authorization level, you can invoke the File Manager base application where you can use edit, browse, and most of the common utilities with CICS or File Manager supported data sets. For example, you can copy from any CICS resource or File Manager supported data set to any other CICS resource or File Manager supported data set.

27.2 Architecture overview

IBM File Manager for z/OS introduced a new component to run the FM functions and utilities from the CICS environment as shown in Figure 27-1. The graphic and panels maintained similar design (ISPF model) for editing, browsing, printing, and altering the status of CICS resources, but the 3270 screens that are displayed are not Common CICS Basic Mapping Support (BMS) maps created using special Assembler Language macros.

)

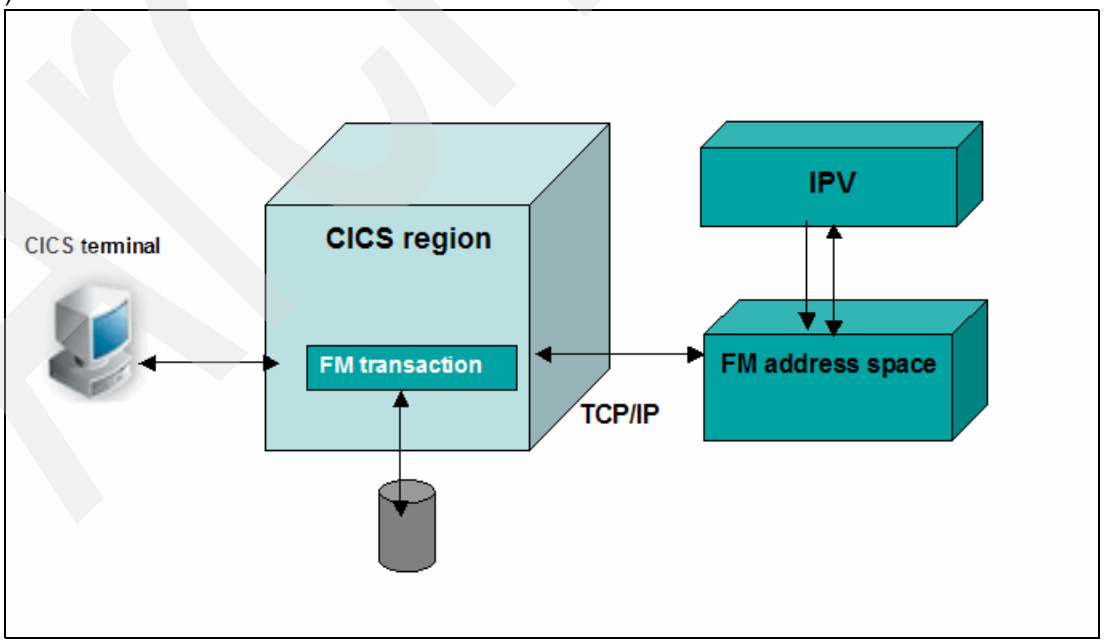


Figure 27-1 FM/CICS Architectural overview

FM/CICS provides access to CICS Resources through the familiar user-friendly ISPF “look-alike” interface, and displays panels on CICS with a similar capability to ISPF panels. When you activate a local CICS request (for example: to retrieve CICS attributes for a file) under any FM/CICS screen or modify the panel to edit any data, the FM/CICS transaction sends the altered 3270 data stream to the FM address space (FMas).

The FMas 3270 data stream sends the request to the Interactive Panel Viewer (IPV).

IPV translates the data and returns to the FMas, which processes the translated data stream and issues a request to the running FM/CICS transaction (FMT) to INQUIRE about CICS resources (that is, to retrieve the CICS attributes for the file). The FMT receives the request and issues an INQUIRE to obtain the file attributes into the buffer. The buffer is then sent to the FMas. The FMas receives all the buffers from the FMT, processes the records, and then sends the BROWSE panel to IPV for translation into a 3270 data stream. Finally, FMas sends the 3270 data stream from the BROWSE panel to FMT to complete the process and display the BROWSE panel.

This solution allows the usage of FM/Base for customers that are not licensed under TSO. It means that you do not require the TSO userid and your authority is related to the userid that is used to connect to the CICS environment (CSSN Logon procedure).

Figure 27-1 shows that FM/CICS Component could Process “Local”, as well as “Remote” CICS resources.

27.3 Starting FM/CICS and setting the processing options

To use FM/CICS, you have to activate the CICS session where FM/CICS has been already installed. The target system required to install and use File Manager is z/OS V01.04.00 or later. The Operational Requisites required for FM/CICS in order to operate all or some of the CICS TS functions have been described in Table 27-1.

Table 27-1 Mandatory CICS Requisites for File Manager CICS

Program Number	Product Name and Minimum VRM Service Level
5655-HAL	Communication Server IP Service V01.04.000 or later
5688-198	Language Environment for z/OS
5697-E	CICS Transaction Server for z/OS V02.02.00 or later

File Manager installs in the z/OS (Z038) SRE and starts the CESN transaction as shown in Figure 27-2.

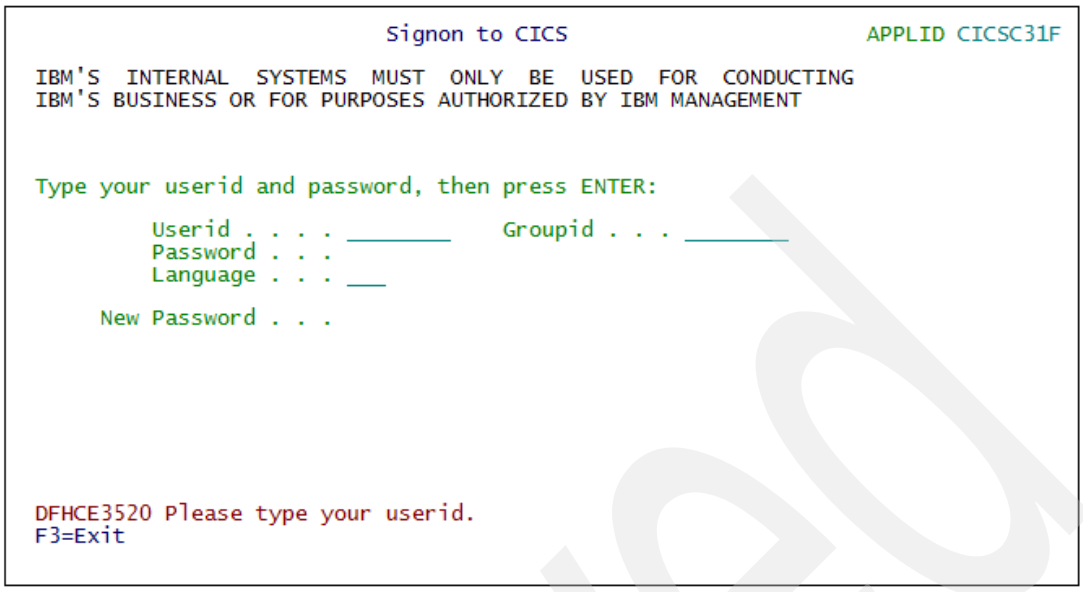


Figure 27-2 Sign on to CICS where FM/CICS component has been configured

According your CICS userid authorization level, you are allowed to process CICS commands while you are using CICS Enqueue Resources under FM/CICS.

Note: CICS command security has prevented your userid from issuing INQUIRE commands on the CICS region. Your userid must have the authority to issue INQUIRE commands to run FM/CICS.

Type your userid and password and press ENTER to display Figure 27-3.

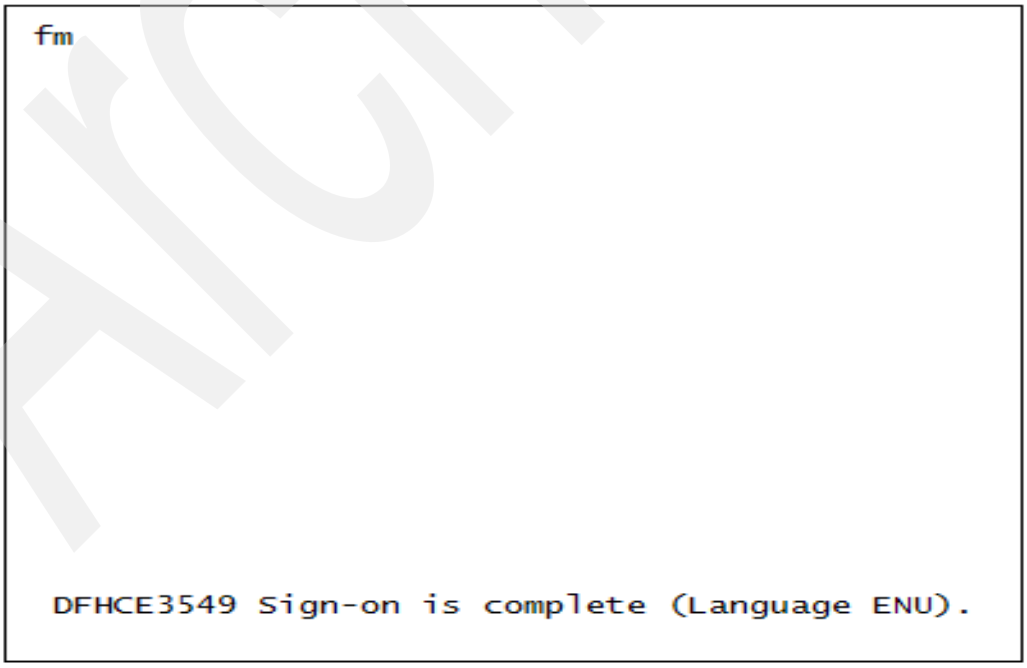


Figure 27-3 Sign on to the CICS has been Completed; type “fm” to activate FM/CICS

Fill the transaction name for File Manager using the default name, FM, on the native CICS map, to start the FM/CICS component. We use FM as default transaction name to start File Manager/CICS. You might have changed the name at your site during installation due to local standards or other reasons.

You can type the following command: **`FM userid`** (*userid* is optional).

Each FM/CICS user is required to provide an MVS userid and (on the next panel) the password. If you omit the userid, then:

- If you have signed on using the CESN transaction and the product has been installed with either *DEFAULT=SIGNON or *PASSWORD=REMEMBER, then the userid defaults to the signed-on userid.
- If you have not signed on, then you are prompted for a userid before the logon panel is presented. Once a valid userid has been specified to the FM transaction, FM/CICS displays the File Manager CICS Logon panel.

Note: The File Manager CICS Logon panel is displayed for every user the first time the FM transaction is entered. The logon panel can be bypassed on subsequent invocations of the FM transaction if the *PASSWORD=REMEMBER install option has been specified, and you have already signed on to CICS, and the userid has been omitted when invoking the FM transaction.

After pressing Enter, the CICS session displays the Primary Options Menu as shown in Figure 27-4.

```
 Process   Options   Help
------------------------------------------------------------------------------
FMN3ST00                    Primary Option Menu

0  Settings      Set processing options              User ID . : AMINTOR
1  Browse        Browse data                         CICS User : AMINTOR
2  Edit          Edit data                           Version . : 7.1.0
3  Utilities     Perform utility functions           Date. . . : 2006/11/19
4  Templates     Template and copybook utilities     Time. . . : 16:53
FM FM            File Manager z/OS
FI FM/IMS        File Manager for IMS z/OS
X  Exit          Terminate FM/CICS

Processing Options:
 CICS Resource
 3  1. File
    2. Temporary Storage
    3. Transient Data


Command ===> vercics
F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
F12=Cancel
```

Figure 27-4 File Manager CICS has been activated. Primary Option Menu

To display the level of connected CICS systems, enter VERCICS on the command line of any panel. The current level for each connected CICS systems is displayed in Figure 27-5.

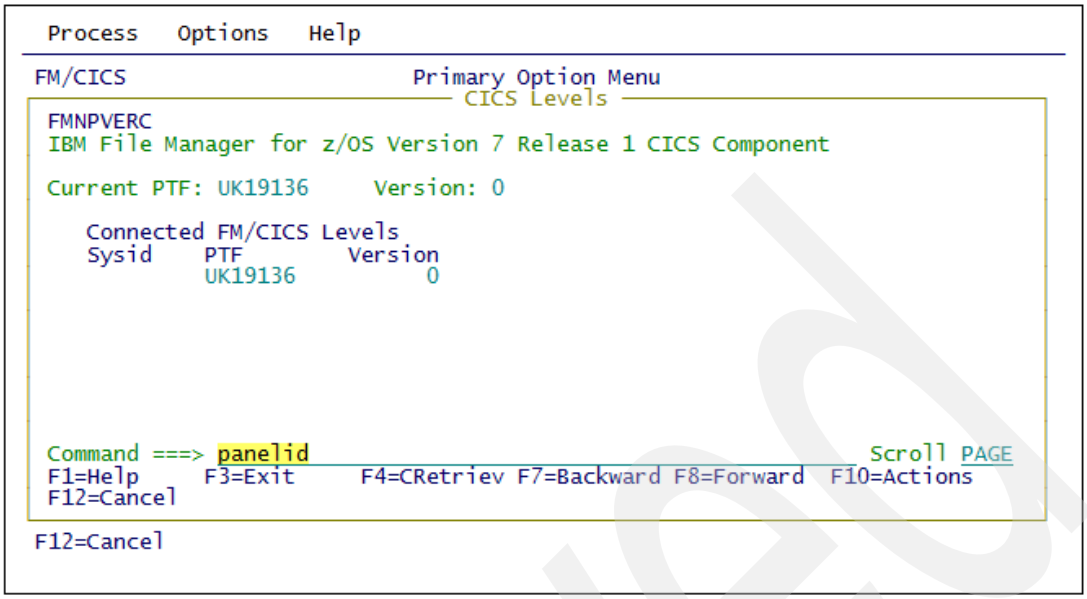


Figure 27-5 FM/CICS Level, Version, and Release information

In Figure 27-5, there is only one connected CICS regions at Version Level =0. When there are more connected SYSIDs, and if the levels or versions are different, these values are highlighted in red.

Note: To avoid unforeseen errors, ensure that all connected systems are running the same level of File Manager.

Set the processing options

The following sections describe how you can set the parameters and change your screen layout and function key values according your setting. Many of the processing operations performed by File Manager utilize default values that can be set from within the FM/CICS application.

Default options

By adjusting these values, you can customize FM/CICS so that its behavior is best suited to your requirements. Your settings are stored in your Interactive Panel Viewer (IPV) profile (see the description at the beginning of this chapter), and are invoked when you log in, regardless of which workstation you use. You can update these default values by accessing the relevant processing option panel.

To access a processing option panel, use any one of the following methods, as shown in Figure 27-6.

```
 Process   Options   Help
------------------------------------------------------------------------------
FM/CICS                        Primary Option Menu

0  Settings      Set processing options                 User ID . : AMINTOR
1  Browse        Browse data                            CICS User : AMINTOR
2  Edit          Edit data                              Version . : 7.1.0
3  Utilities     Perform utility functions              Date. . . : 2006/11/19
4  Templates     Template and copybook utilities        Time. . . : 20:33
FM FM            File Manager z/OS
FI FM/IMS        File Manager for IMS z/OS
X  Exit          Terminate FM/CICS

Processing Options:
 CICS Resource
 3  1. File
    2. Temporary Storage
    3. Transient Data


Command ===> 0
F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
F12=Cancel
```

Figure 27-6 Method to access the processing options (1,2)

- First method:

  Type **0** (Settings) on the command line and press Enter. The Set Processing Options Menu panel is displayed as shown in Figure 27-7.

```
 Process   Options   Help
------------------------------------------------------------------------------
FM/CICS    | 6 1. Print settings                    |
           |   2. System settings                   |
0  Settin  |   3. Compiler language selection       |   User ID . : AMINTOR
1  Browse  |   4. COBOL compiler specifications     |   CICS User : AMINTOR
2  Edit    |   5. PL/I compiler specifications      |   Version . : 7.1.0
3  Utilit  |   6. Edit/Browse options               |   Date. . . : 2006/11/19
4  Templa  |   7. VSAM Edit sharing options         |   Time. . . : 20:39
FM FM      |   8. IPV settings                      |
FI FM/IMS  ------------------------------------------
X  Exit          Terminate FM/CICS

Processing Options:
 CICS Resource
 3  1. File
    2. Temporary Storage
    3. Transient Data


Command ===>
F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
F12=Cancel
```

Figure 27-7 Method to access the processing options (2,2)

- Second method:

  From any FM/CICS panel, use the Options pull-down menu to select the required processing options type. Type **6** to display the Edit/Browse options as shown in Figure 27-8.

```
  Process    Options    Help
 ------------------------------------------------------------------------------
 File Manager                 Edit/Browse Options
                                                                    More:    +
 Related command if applicable, shown in ()
 Miscellaneous:           Enter "/" to select options           Initial Display
 /  Prefix on    (PRE)             Prefix on right (PRE RIGHT)  1  1. Previous
 /  Recognize and interpret ISPF packed data                          2. Table
    Show RBA and Length when browsing VSAM          (RBALEN)          3. Single
    CAPS initially ON - translate changed data to uppercase (CAPS)    4. Character
 Expose (do not group) records of types: (SHOW)                       5. Hex
    Not selected      Suppressed  /  Length error                     6. LHex
 See shadow lines (deselect to hide) for groups of: (SHADOW)
 /  Not selected  /  Suppressed  /  Excluded
 Record formatting options for SNGL display or print:
 _  Field reference number  (REF)     Field type and length values (TYPE)
    Picture clause          (PIC)     Start location               (SLOC)
    Structure               (STR)     Left justify numeric fields  (JUST)
 /  Redefined fields        (RDF)  (also affects TABL format)

 Auxiliary Data Set Allocation Defaults
 Command ===> ____________________________________________________________
 F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
 F12=Cancel
```

Figure 27-8 Edit/Browse Options panel

You can use the Set Processing Options panel to control processing of the current File Manager session. To get detailed help for a specific processing option, place the cursor on the appropriate input field on the panel and enter the HELP command (F1).

When you have set the processing options you require, you can:

- Use the EXIT command (F3) to save the changes you have made.
- Use the CANCEL command (F12) to discard the changes you have made.
- Use the RESET command to set all options back to the File Manager installation defaults.

If an input field is left blank, the default as defined for your File Manager installation is used.

CUAATTR command to change color, intensity, or highlighting attributes

Use the CUAATTR primary command (from any FM/CICS panel) to adjust panel colors, intensity, and highlighting, as shown in Figure 27-9.

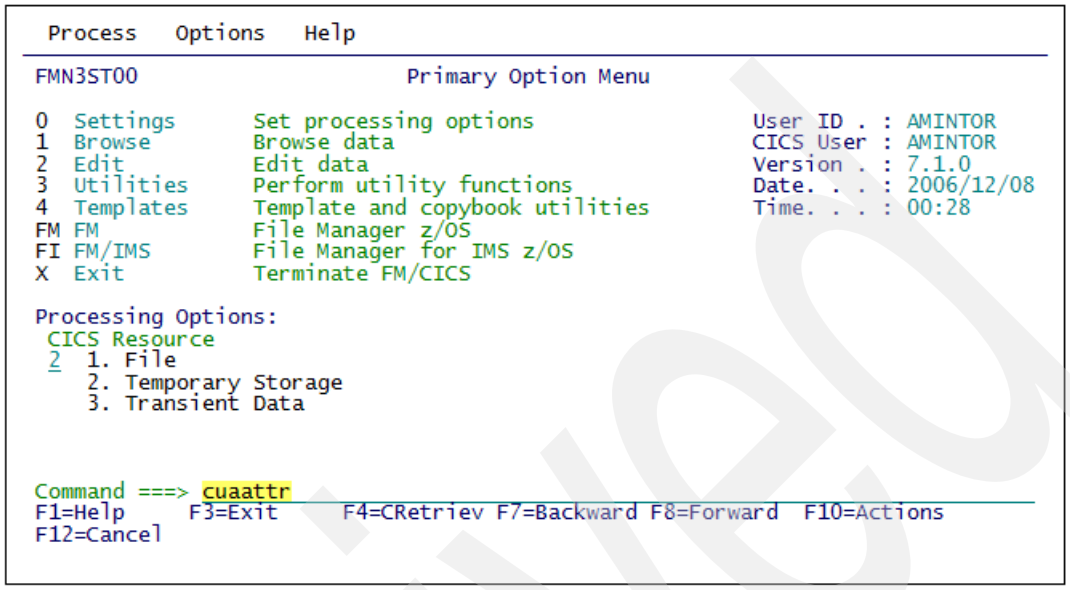


Figure 27-9 How to adjust panel colors, intensity and highlighting

Press Enter and FM/CICS displays the Change CUA® Attributes panel, as shown in Figure 27-10.

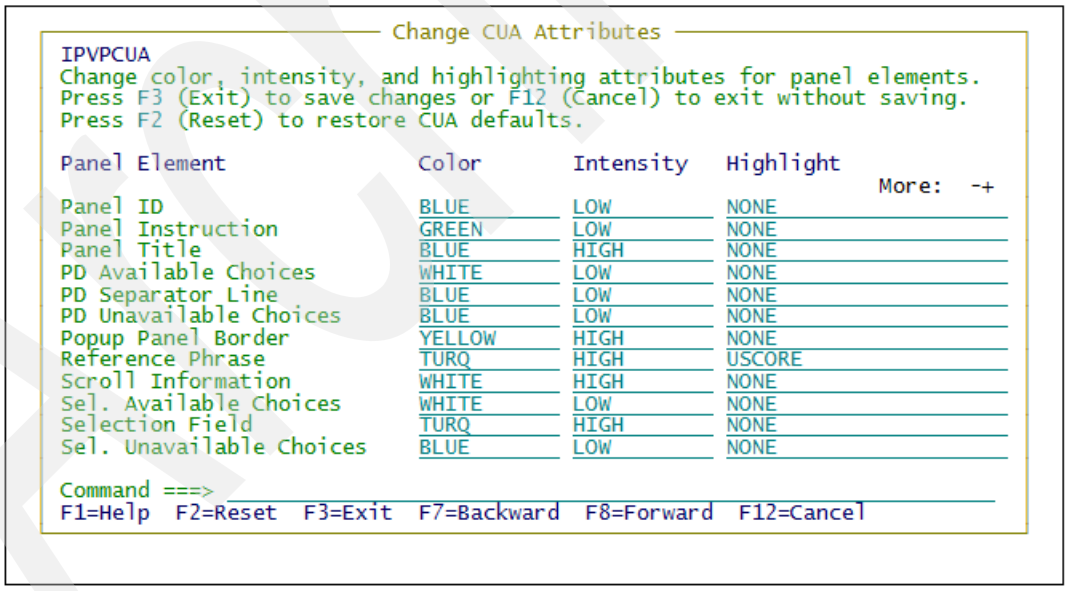


Figure 27-10 CUA attributes Panel

You can change the existing attributes by overtyping with the values that you prefer.

Tip: You can navigate through the panels by typing an equal sign (=) followed by the option number for the required processing type, on the command line of any FM/CICS panel. For example, to display the Compiler Language Selection panel, enter **=0.3**.

System options

You set any of the processing options on the Set Processing Options panel (within FM/CICS).

In this session we show how to invoke the *VSAM Edit sharing options* panel, starting from the Primary Option Menu.

Type **=0.7** from the Primary Option Menu (FMN3ST00) as shown in Figure 27-11.

```
 Process   Options   Help
FMN3ST00                    Primary Option Menu

0  Settings      Set processing options              User ID . : AMINTOR
1  Browse        Browse data                         CICS User : AMINTOR
2  Edit          Edit data                           Version . : 7.1.0
3  Utilities     Perform utility functions           Date. . . : 2006/11/21
4  Templates     Template and copybook utilities     Time. . . : 23:21
FM FM            File Manager z/OS
FI FM/IMS        File Manager for IMS z/OS
X  Exit          Terminate FM/CICS

Processing Options:
 CICS Resource
 1  1. File
    2. Temporary Storage
    3. Transient Data


Command ===> =0.7
F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
F12=Cancel
```

Figure 27-11 How to invoke the "VSAM Edit sharing options" panel

Press Enter to display the "VSAM Edit Sharing Options" panel. The following panel presents options which apply only when editing VSAM files that are potentially being shared by other users. The value and the parameters that we can set are shown in Figure 27-12.

```
 Process   Options   Help
File Manager            VSAM Edit Sharing Options

These options apply only when editing VSAM files that are being shared.

 Autosave frequency             0

Options for the CHANGE ALL command
 SAVE before starting           on   Enter ON or OFF.
 Change all autosave frequency
 Auto retry for change all      off  Enter ON or OFF.




Command ===>
F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
F12=Cancel
```

Figure 27-12 VSAM Edit sharing panel. An example of setting for the CHANGE ALL command

In this case, since you are closing the session without modifying the data, you are notified when CUSTOM3 has been updated from another application.

This means that you are notified during the confirm stage if another user updated the same data before you complete your “Changed records”.

As seen before, this panel above shows the options that are designed to provide improved usability for the files that have been allocated with DISP=SHR in order to allow access to the datasets in shared mode.

In this session we explain the meaning related to the setting of these options:

- Auto SAVE frequency:
  - For shared files, we now have the ability to automatically issue a SAVE command after a given number of changes. Set this value to 0 if you do not want an auto-save command to be issued. When operating upon a shared file, this determines the frequency of periodic SAVE of the file.If 0 is specified, File Manager does not automatically issue a SAVE command.
  - If a positive value is specified, File Manager automatically issues a SAVE command for the file being operated upon after the given number of updates have occurred. An update is counted during each File Manager operation. For example:
    - Overtyping data in one record is counted as one update.
    - If the same record is again overtyped, it is counted as another update.
- Save before starting:
  - When operating upon a shared file, during a CHANGE command with the ALL operand, this determines if a SAVE is issued before commencing the CHANGE. Setting this ON ensures that any updates made to the file so far are saved.
  - During the CHANGE ALL process, when using AUTORTRY (and a CHGASAVE value), the change process might require a value), the change process might have to refresh records as it goes (due to concurrent updates by other users). That refresh might discard other updates that had been made in this edit session to records on the file. Leaving the setting at ON, to issue a SAVE before the CHANGE commences, is therefore recommended as it ensures the state of the records.
- Change ALL autosave:
  - This applies specifically to the operation of the Change command when used with the ALL parameter. Similar to the AUTOSAVE option above, it provides the ability to automatically save records after a given number of changes. If used, the changes are counted at the record level, that is, if a record is changed (regardless of how many times the change command changed that record), this is counted as one change. When the number of records changed reaches the number set in this option, a SAVE command is issued and the change command continues. These are the values that you can specify:
    - If 0 is specified, File Manager does not automatically issue a SAVE command.
    - If a positive value is specified, File Manager automatically issues a SAVE command for the file being operated upon after the given number of updates have occurred. An update is counted during each File Manager change operation upon a record element. For example, assuming a record contains the data 'FREDFRED' and one enters a change command of C FRED JOHN ALL, then the change to the record would be counted as one change.

- Change ALL autoretry:
  - This also applies specifically to the operation of the Change command when used with the ALL parameter. There is a relationship between this option and the *Change all autosave frequency* parameter. Depending on the value (ON/OFF) you set, FM/CICS performs and takes different actions when a SAVE command is issued during operation of the Change process. It occurs when another user has updated a record while the change was running.
    - If this option is set *on*, then the change command automatically refreshes (that is, it rereads) the record that this occurred on and attempts to reapply the change.
    - If this option is *off*, then each time that File Manager detects that the record was updated before the change command saved the record, then you are given a pop-up window asking for your decision as to what to do about the situation. In this case, you have the choice of updating the record anyway, refreshing the record, and reapplying the update or aborting the change command.

27.4 Editing/browsing CICS resources

Before we show some scenarios in detail, we want to describe how we specify a CICS resource to File Manager. FM/CICS provides specific functions to process your CICS resources. These are referred to as CICS ONLY functions. When entering the CICS ONLY resource name, the following rules apply:

- Resource name:
  - The resource name has:
    - A 1–8 character file name
    - A 1–16 character temporary storage queue
    - A 1–4 character transient data queue
  - The resource name is used in conjunction with SYSID and POOLNAME (TS queues only) to produce a list of matching resources. You can enter a fully-qualified or generic resource name. In a generic resource, use the percent sign (%) to represent a single character, and an asterisk (*) to represent any number of characters.
- SYSID:
  - The default value is blank. This field contains the ID of the CICS systems where the resource resides. A blank value defaults to the local system. You can enter a fully-qualified or generic SYSID. In a generic SYSID, use the percent sign (%) to represent a single character, and an asterisk (*) to represent any number of characters.
- TS Pool name (TS data sharing only):
  - The default value is blank. This field is used to specify the pool name of a shared temporary storage pool and temporary storage processing is shipped to a temporary storage server that manages the pool. You can enter a fully-qualified or generic pool name. In a generic pool name, use the percent sign (%) to represent a single character, and an asterisk (*) to represent any number of characters. If you specify a SYSID value and a TS Pool name, then a list of queues that match either is produced from which you can select the appropriate queue for processing.

Note: FM/CICS can process CICS resources that are owned by a region other than the region where FM/CICS is running if:

- An active connection between the current region where FM/CICS is running and the remote region that owns the resource is active.
- The FM/CICS program, FMN3CICS, is available on the remote region.

To process a remote CICS resource, you must specify the SYSID that owns the resource on the FM/CICS panels (generics are allowed). A remote definition on the local CICS region for the remote resource is not required by FM/CICS. In addition, specifying a blank SYSID value for a remote CICS resource that is owned by another region causes a "not found" condition.

If you enter a pattern value on the Resource name o SYSID, FM/CICS processes the panel and displays a list of the data sets that match the pattern. When the member selection panel is displayed, select a member by typing an **S** in the Sel field next to the member or with the SELECT primary command, and then pressing Enter. Because you are selecting a copybook or template, you select only one name.

If you have to reset values entered in the prefix area that have not yet been processed, use the RESET primary command.

To re-read the directory and display the current member list, use the REFRESH primary command.

Tip: When you are selecting a member, you might find it useful to sort the list of displayed members. To do this, place the cursor on the column header of the column you want to sort and press Enter.

When you sort members, the sequence (ascending or descending) for a given column is predetermined and consistent with ISPF.

Displaying more information when receive an error Message

If an error occurs when FM/CICS attempts to process a panel, a short text message displays in the upper right corner of the screen. While this message is displayed, pressing F1 displays the expanded text of the error message at the bottom of the screen.

27.4.1 Viewing of CICS resources queues

The CICS resources supported from FM/CICS are files, temporary storage queues, and transient data queues. If you have the authority, you can also modify the status of the CICS resources. FM/CICS panels allow you to select options and to specify parameters, commands, and program function (PF) keys to simplify requests for common functions. FM/CICS panels provide full-screen format for information display and editing.

The main functions provided by FM/CICS are:

- The ability to edit or browse CICS resources
- The ability to list CICS resources and change their status and attributes
- The ability to run File Manager for z/OS Base Component (FM/Base) under FM/CICS, hence using it without requiring ISPF; this includes the ability to use FM Base tools and utilities against FM/CICS resources when running under FM/CICS
- The ability to run File Manager IMS Component (FM/IMS) under FM/CICS, using it without requiring ISPF

The CICS TD control facility provides a generalized queuing facility. Data can be queued (stored) for subsequent internal or external processing. Selected data, specified in the application program, can be routed to or from predefined symbolic TD queues: either intrapartition or extrapartition.

Transient data queues are *intrapartition* when are associated with a facility allocated to the CICS region, and *extrapartition* (usually sequential files) if the data is directed to a destination that is external to the CICS region. TD queues must be defined and installed before first reference by an application program.

Regarding the TD queues, FM/CICS provides a panel to manage both extrapartition as well intrapartition. When FM/CICS reads an intrapartition queue (READQ TD), the entire queue is read into memory by the Browse Transient Data Entry Panel.

When you exit, you are given the option of restoring (or not) all the records processed back to the queue. These records are appended to the existing queue if you select to rewrite the record.

This section is provided to help the new user to easily use and navigate within FM/CICS primary commands, panels, fields, resources, and CICS enqueues.

Browsing TD queues using FM/CICS

You can browse the following types of transient data queues:

- Intrapartition:
  - FM/CICS reads the entire queue into memory. You can then browse the queue and when you exit you are given the option of restoring all the records read back to the queue. These records are appended to the existing queue if you select to rewrite the records.
- Extrapartition:
  - You can only browse extrapartition queues defined for input.
- Indirect:
  - If the associated queue is the same as the foregoing, then browsing is supported as stated for those queues.

Starting from the *Primary Options Menu* panel, type **1** on the command line as shown in Figure 27-13.

```
 Process   Options   Help
------------------------------------------------------------------------------
FM/CICS                        Primary Option Menu

0  Settings      Set processing options                 User ID . : AMINTOR
1  Browse        Browse data                            CICS User : AMINTOR
2  Edit          Edit data                              Version . : 7.1.0
3  Utilities     Perform utility functions              Date. . . : 2006/12/06
4  Templates     Template and copybook utilities        Time. . . : 11:32
FM FM            File Manager z/OS
FI FM/IMS        File Manager for IMS z/OS
X  Exit          Terminate FM/CICS

Processing Options:
 CIC
 1      IBM* File Manager for z/OS Version 7 Release 1 CICS Component
        Licensed Materials - Property of IBM
        5655-R47
        (C) Copyright IBM Corporation 1986, 2006 - All Rights
        Reserved.              * Trademark of International Business
        Machines
Comm
F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
F12=Cancel
```

Figure 27-13 Select the options '1' using the command line

Press Enter to start Transient Data Browsing as shown in Figure 27-14.

```
 Process   Options   Help
------------------------------------------------------------------------------
FM/CICS              Browse Transient Data Entry Panel

Input Transient Data Queue:
   Queue name . . . . *___
   Sysid  . . . . . . *___
   Start position . . _____          Record number
   Record limit . . . memory__       Record sampling  _
Copybook or Template:
   Data set name  . . ____________________________________________
   Member . . . . . . ________       Blank or pattern for member list
Processing Options:
 Copybook/template       Enter "/" to select option
 3  1. Above               Edit template    Type (1,2,S)
    2. Previous            Include only selected records
    3. None
    4. Create dynamic


Command ===> ____________________________________________________________
F1=Help     F3=Exit     F4=Expand   F7=Backward F8=Forward  F10=Left
F11=Right    F12=Cancel
```

Figure 27-14 How to browse a TD queue

After pressing Enter, the browse entry panel is displayed. You use this panel to enter the name of a resource to be browsed. You also use this panel to specify the name of the COBOL copybook or template that describes a logical view of the data set.

Selection and formatting criteria

The Record limit field is used to restrict the number of records retrieved from a data set (from the start point or top) resulting in an edit or browse of a portion of the data set. As seen in the foregoing panel, we type the keyword `MEMORY` as the Record limit to restrict the number of records retrieved to as many as can fit in the available virtual storage. Press Enter to get the panel in Figure 27-15.

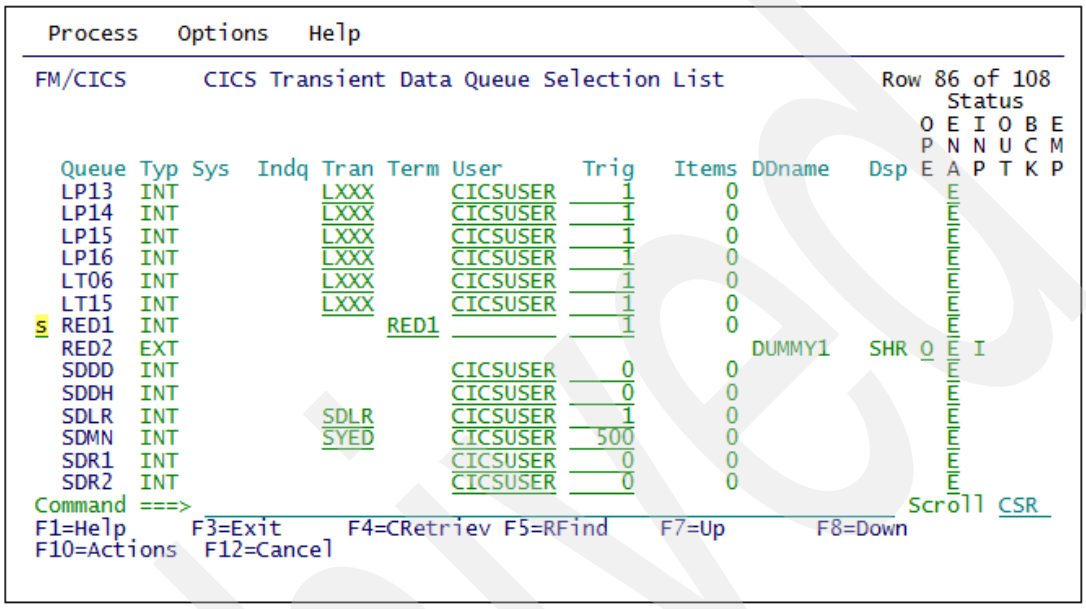


Figure 27-15 Select the RED1 Transient Data from the queue list

Next we describe the fields (column descriptions) of this panel.

Prefix area (first column):

This is where you enter the prefix commands S, A, B, E, I, V, and X:

S Selects a single CICS queue and returns to the name for processing by the current function. (Note: Subsequent selections are ignored).

A Displays and alters the CICS information associated with a given CICS queue.

D Deletes the items for an intrapartition queue.

I Performs an INFO command against the CICS queue if a z/OS data set is associated with it. This command displays the data set details, and, for VSAM entries, also statistics and allocation information.

X Performs an EXTENTS command against the CICS queue if a z/OS data set is associated with it. This command displays the statistics and extent information for a data set.

Queue:

This is the 1-4 character CICS queue id.

Typ:

This is one of three following values:

- IND indirect queue
- EXT extrapartition queue
- INT intrapartition queue

Sys:

This is the System ID of the CICS system on which the file has been defined.

INDQ (indirect):

This field contains the 1-4 character queue name that an indirect queue points to. This column is blank for other types of queues.

Tran (intrapartition):

This field contains the 1-4 character transaction to be executed when CICS initiates a task automatically to process the queue. This column only applies to intrapartition queues and is protected for other types of queues. You can overtype the transaction ID with valid CICS transaction characters.

Term (intrapartition):

This field contains the 1-4 character terminal or session associated with the queue. If the facility is a terminal then you can overtype this value with valid terminal ID. This column only applies to intrapartition queues with a terminal facility.

User (intrapartition):

This contains the 1-8 character user identifier associated with the queue. If the facility is a not a terminal, then you can overtype this value with valid user ID. This column only applies to intrapartition queues without a terminal facility.

Trig (intrapartition):

This field contains the number of items the queue must reach before automatic transaction initiation (ATI) occurs. You can overtype this value with a valid number.

Items (intrapartition):

This field contains the number of items currently on this queue.

DDname (extrapartition):

This field contains the DD name for the extrapartition queue.

DSP (extrapartition):

This field contains the disposition of the extrapartition data set.

Changing the STATUS on the resource list

On the CICS resource list or corresponding attributes panel, you can change any value that is underlined by overtyping the change into the field and pressing Enter. For example, to close a VSAM file from the CICS file selection list, type **C** in the open status column and press Enter.

OPE (extrapartition):

You can change the open status by overtyping the character with an **O** to open or **C** to close.

ENA:

You can change the enable status by overtyping the displayed character with an **E** to enable or **D** to disable. This field only applies to extrapartition and intrapartition queues.

INP (extrapartition):

The extrapartition data set is opened for input.

OUT (extrapartition):

The extrapartition data set is opened for output.

BCK (extrapartition):

The queue is defined for input and is read backward.

EMP (extrapartition):

CICS has detected an empty E or full F status for the queue.

The CICS Transient Data Queue Selection List panel shows all the queues that match the queue name and sysid values specified on the previous panel. Standard scrolling applies and you can sort all of the columns with exception of the status columns. You can also perform the following tasks:

- Open, close and modify various queue attributes.
- Display catalog information for extrapartition data sets.
- Invoke various File Manager functions against a queue.

Select (by typing an **S** in the Prefix line) the RED1 (intrapartition) queue. Press Enter to browse the TD RED1 panel, as shown in Figure 27-16.

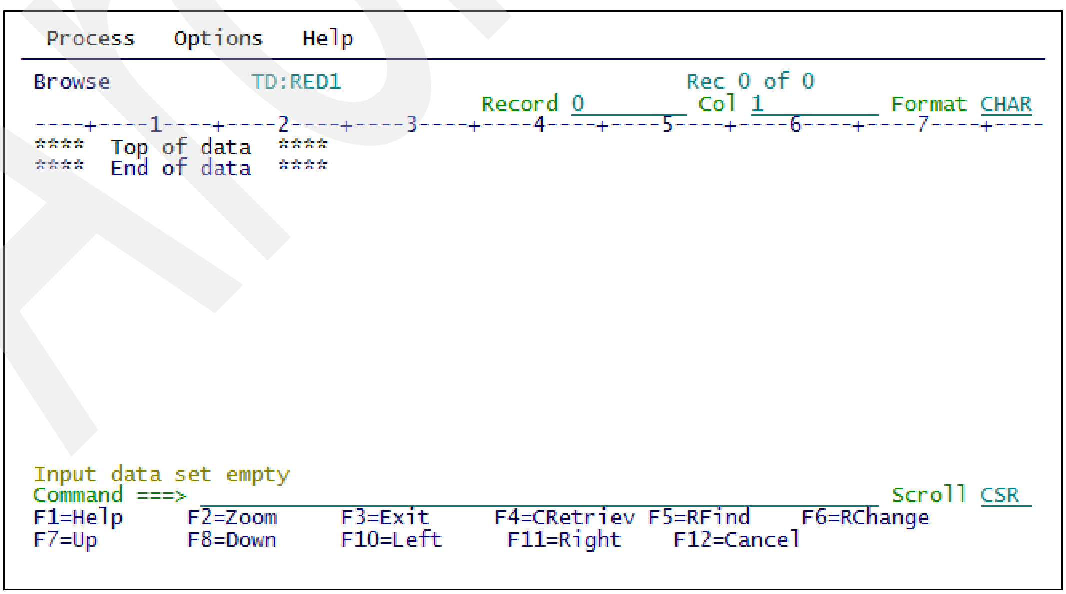


Figure 27-16 The RED1 TD is empty

We note that the RED1 INT queue is empty. To fill up the RED1 dataset, we invoke the RED0 Transaction (in CEDF mode, in order to clarify and follow all the steps) as shown in Figure 27-17.

```
TRANSACTION: RED0 PROGRAM: S737201  TASK: 0004143 APPLID: CICSC31F DISPLAY:  00
 STATUS:  PROGRAM INITIATION

     EIBTIME      = 53548
     EIBDATE      = 0106341
     EIBTRNID     = 'RED0'
     EIBTASKN     = 4143
     EIBTRMID     = '0024'

     EIBCPOSN     = 4
     EIBCALEN     = 0
     EIBAID       = X'7D'                                    AT X'001400EA'
     EIBFN        = X'0000'                                  AT X'001400EB'
     EIBRCODE     = X'000000000000'                          AT X'001400ED'
     EIBDS        = '........'
+    EIBREQID     = '........'


ENTER:  CONTINUE
PF1 : UNDEFINED          PF2 : SWITCH HEX/CHAR     PF3 : END EDF SESSION
PF4 : SUPPRESS DISPLAYS  PF5 : WORKING STORAGE     PF6 : USER DISPLAY
PF7 : SCROLL BACK        PF8 : SCROLL FORWARD      PF9 : STOP CONDITIONS
PF10: PREVIOUS DISPLAY   PF11: EIB DISPLAY         PF12: UNDEFINED
```

Figure 27-17 The panel displays the RED0 program (CEDF)

The RED0 Tran uses the **S737201** program to write the TS queues data, as shown in Figure 27-18.

```
TRANSACTION: RED0 PROGRAM: S737201  TASK: 0004143 APPLID: CICSC31F DISPLAY:  00
 STATUS:  COMMAND EXECUTION COMPLETE
 EXEC CICS WRITEQ TS
  QNAME ('REDBOOKTSQUEUE  ')
  FROM ('TS Queue Data item 1 created by AMINTOR  on system CICSC31F.    '...)
  LENGTH (80)
  AUXILIARY
  NOHANDLE




 OFFSET:X'0003A4'    LINE:00006400       EIBFN=X'0A02'
 RESPONSE: NORMAL                        EIBRESP=0

ENTER:  CONTINUE
PF1 : UNDEFINED          PF2 : SWITCH HEX/CHAR     PF3 : END EDF SESSION
PF4 : SUPPRESS DISPLAYS  PF5 : WORKING STORAGE     PF6 : USER DISPLAY
PF7 : SCROLL BACK        PF8 : SCROLL FORWARD      PF9 : STOP CONDITIONS
PF10: PREVIOUS DISPLAY   PF11: EIB DISPLAY         PF12: ABEND USER TASK
```

Figure 27-18 The RED0 Tran is writing the REDBOOKTSQUEUE queue

The **S737201** program continues to write TS queues (REDBOOKTSQUEUE) and a TD queues (RED1) until it reaches 1000 records, as shown in Figure 27-19.

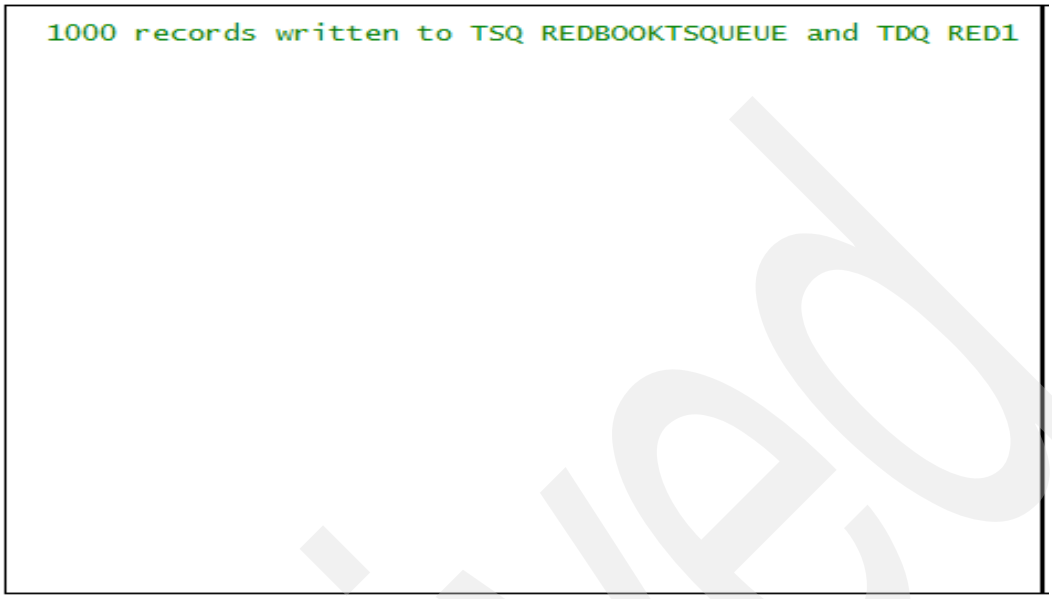


Figure 27-19 The RED0 Tran written 1000 records

We come back in the Browse Transient Data Entry panel to select the RED1 TD again, as shown in Figure 27-20.

':

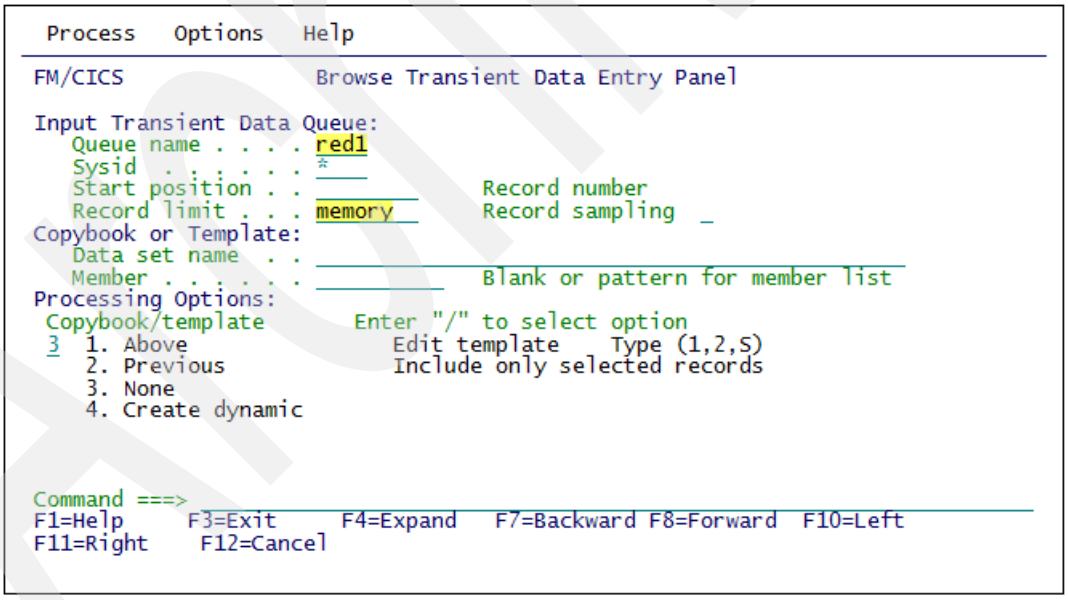


Figure 27-20 Browsing a Transient Data Queue

After pressing Enter, the CICS Transient Data Queue Selection List panel is displayed, as shown in Figure 27-21.

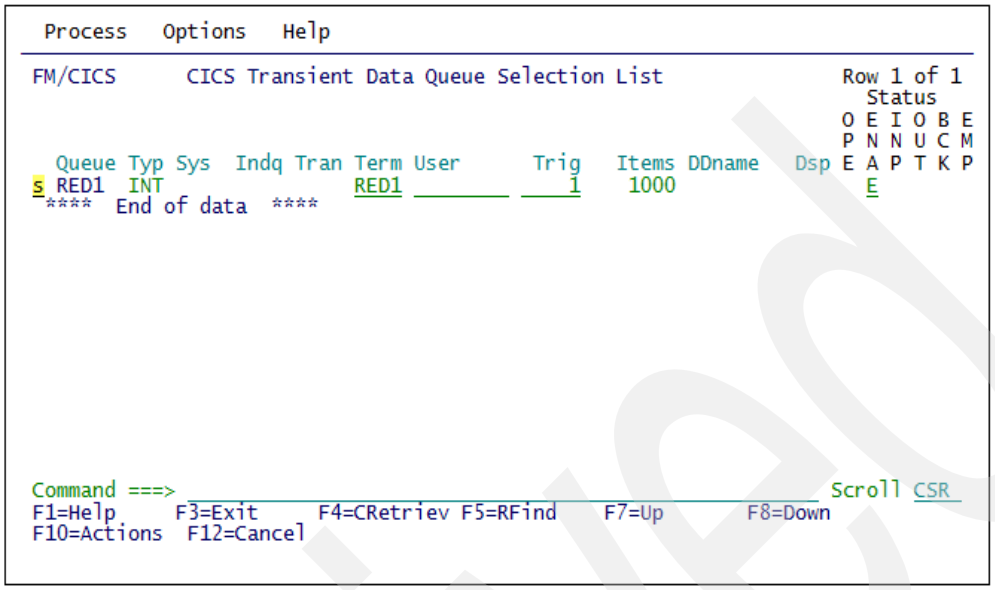


Figure 27-21 TD selection List panel

The RED1 queue that we have just selected is an INT (intrapartition queue) type. Press Enter to display the 1000 items, as shown in Figure 27-22.

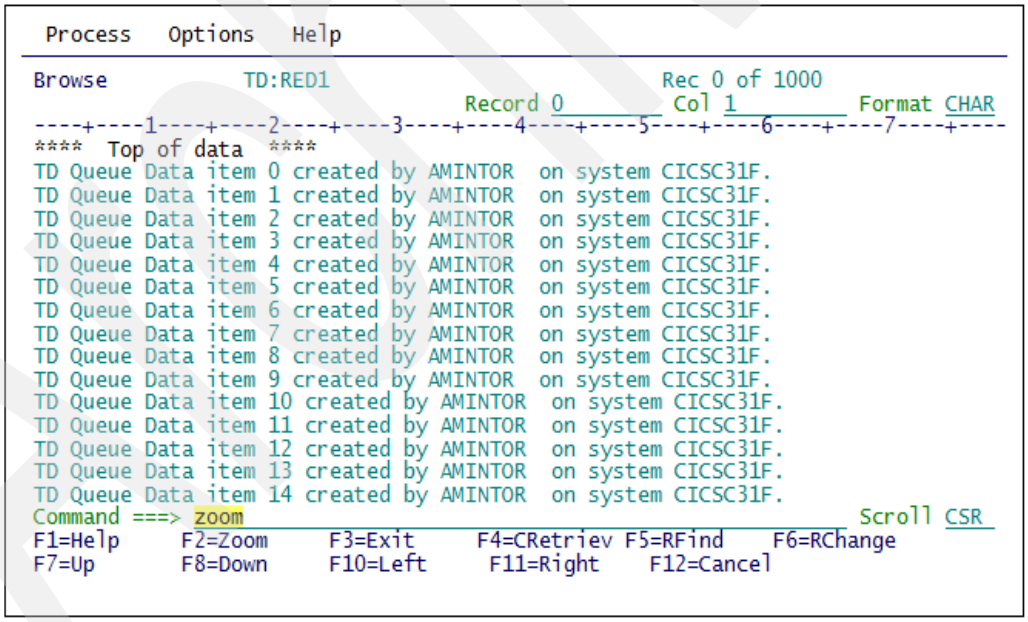


Figure 27-22 The RED1 dataset items

You can invoke the RED1 data set browsing with a specified copybook or template. In this scenario we ZOOM the second item, and we choose to display the data using the LHEX format (from CHAR, HEX, and LHEX), as shown in Figure 27-23.

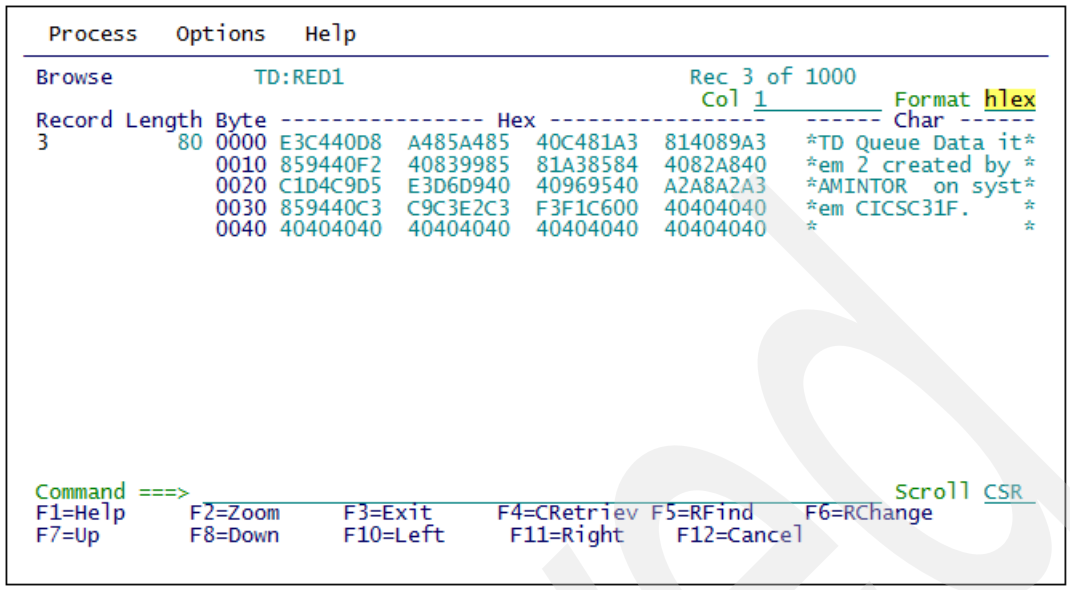


Figure 27-23 To display data in HLEX format

LHEX displays data in hexadecimal format in multiple-record display mode, and in both hexadecimal and character (system dump) format in single-record mode.

Since we are browsing TD intrapartition, Fm/CICS reads (READQ TD) all the entire queue into memory by the Browse Transient Data Entry Panel.

When you exit, you are given the option of restoring (or not) all the records read back to the queue, as shown in Figure 27-24.

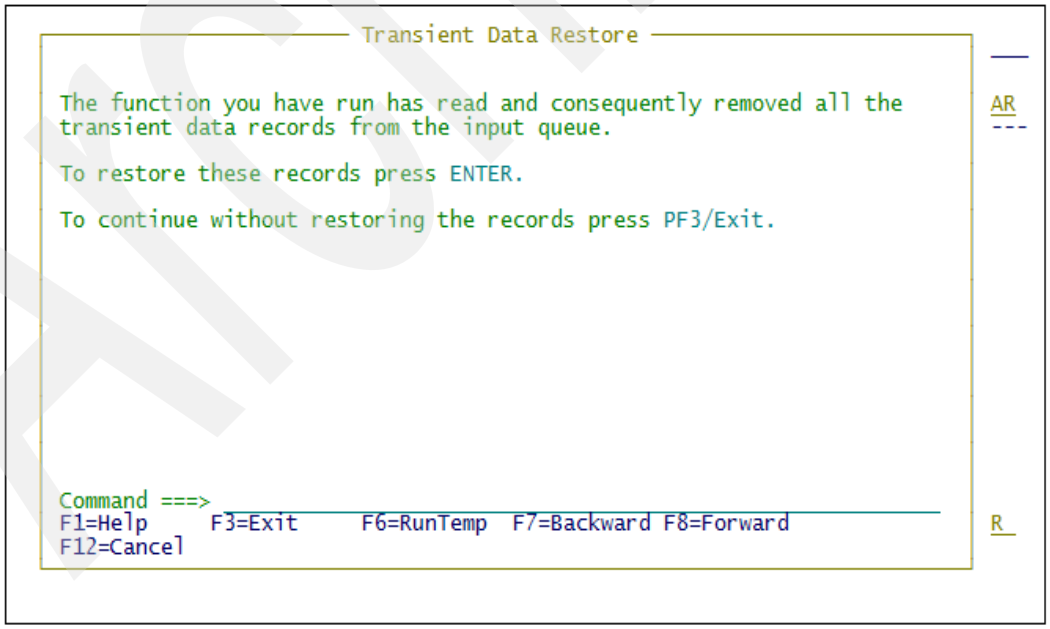


Figure 27-24 ENTER to restore, PF3 to delete all the TD records

We press Enter, since we decided to restore these records.

Viewing TS queues using a template

Starting from the Primary Options Menu panel, type **3.4** on the command line as shown in Figure 27-25.

```
 Process   Options   Help
------------------------------------------------------------------------------
FMN3ST00                      Primary Option Menu

0  Settings      Set processing options                 User ID . : AMINTOR
1  Browse        Browse data                            CICS User : AMINTOR
2  Edit          Edit data                              Version . : 7.1.0
3  Utilities     Perform utility functions              Date. . . : 2006/12/07
4  Templates     Template and copybook utilities        Time. . . : 14:59
FM FM            File Manager z/OS
FI FM/IMS        File Manager for IMS z/OS
X  Exit          Terminate FM/CICS

Processing Options:
 CICS Resource
 2  1. File
    2. Temporary Storage
    3. Transient Data


Command ===> 3.4
F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
F12=Cancel
```

Figure 27-25 Type 3.4 on the command line to display the CICS resources

Press Enter to display the *List CICS Resources Entry Panel*, as shown in Figure 27-26.

```
 Process   Options   Help
------------------------------------------------------------------------------
FM/CICS               List CICS Resources Entry Panel

CICS File, Transient Data, Temporary Storage Queue, or Enqueue Name:
   Resource name  . . REDxxxxTS*
   Sysid  . . . . . . *___
   TS pool name . . . ________

Processing Options:
 CICS Resource
 2  1. File
    2. Temporary Storage
    3. Transient Data
    4. Enqueue




Command ===>
F1=Help     F3=Exit     F4=Expand   F7=Backward F8=Forward  F10=Left
F11=Right    F12=Cancel
```

Figure 27-26 Wildcard example

The List CICS resources function provides you with the ability to list the resources matching the entered name, sysid, and TS pool. Subsequently you can modify the listed resources or perform various functions such as edit, browse and print against any listed file, TS queue, and TD queue.

You can use a wildcard when specifying a CICS resource name on a FM/CICS entry panel. When you enter a wildcard on a FM/CICS entry panel, only the function from the previously displayed FM/CICS entry panel (edit, browse, or print) can be selected for the listed resource. We are filling in the *Resource name* field with a generic character string value such as “RED%%%%TS*” and we use the % sign to represent a single character, and an asterisk (*) to represent any number of characters. Press Enter to list the Temporary Storage resources, as shown in Figure 27-27.

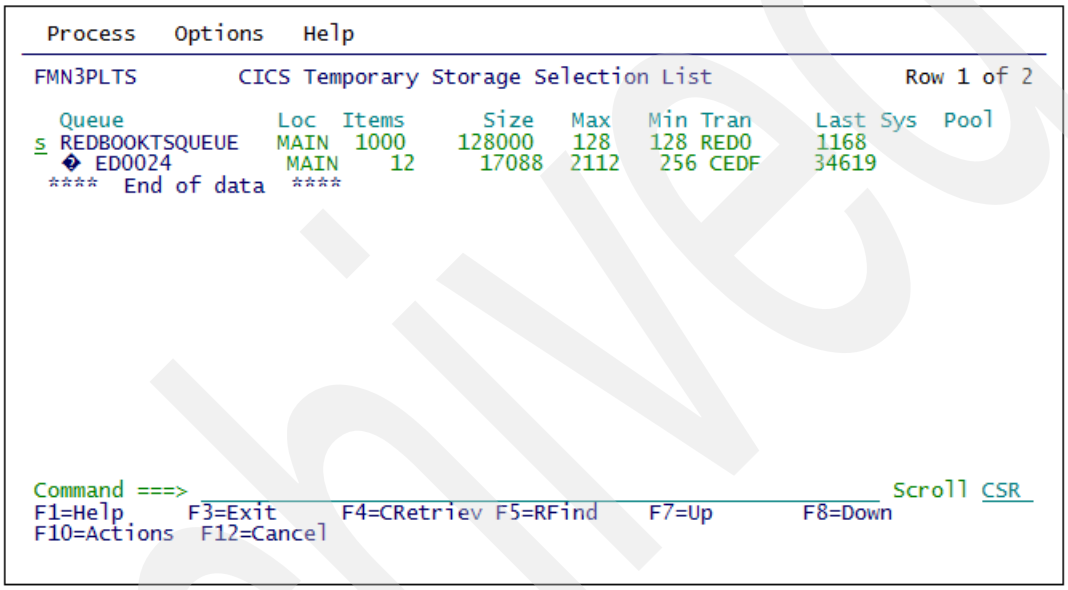


Figure 27-27 CICS TS Selection List panel

In this panel, notice the Tran column value. It specifies the transaction that created the queue (RED0). We select the REDBOOKTSQUEUE that we generated during the previous sample and press Enter. to display the *Edit Temporary Storage Entry* panel, as shown in Figure 27-28.

```
 Process   Options   Help
------------------------------------------------------------------------------
FM/CICS              Browse Temporary Storage Entry P        No name selected

Input Temporary Storage Queue:
   Queue name . . . . RED%%%%TS*
   Sysid  . . . . . . *
   Pool name  . . . . ________
   Start position . . ________      Record number
   Record limit . . . ________      Record sampling  _
Copybook or Template:
   Data set name  . . 'AMINTOR.BOOK2006.COPYBOOK'
   Member . . . . . . CopyTS01     Blank or pattern for member list
Processing Options:
 Copybook/template       Enter "/" to select option
 1  1. Above                Edit template    Type (1,2,S)
    2. Previous             Include only selected records
    3. None
    4. Create dynamic


Command ===> ____________________________________________________________
F1=Help     F3=Exit     F4=Expand   F7=Backward F8=Forward  F10=Left
F11=Right    F12=Cancel
```

Figure 27-28 Browse TS queue using a Copybook

Press Enter to display the formatted TS Edit panel, as shown in Figure 27-29.

```
 Process   Options   Help
------------------------------------------------------------------------------
Edit            TS:REDBOOKTSQUEUE                Rec 0 of 1000
                                                          Format TABL
       REC-TS TS-TYPE ITEM-NAME    ITEM-PROG NAME            TYPE-PHONE NO-PH
       #2     #3      #4                 #5 #6              #7         #8  +
       AN 1:2 AN 3:6  AN 9:10         BI 19:4 AN 23:16       AN 39:10   AN 49
       <>     <---+>  <---+----> <---+----1> <---+----1----+> <---+----> <---+
000000 ****  Top of data  ****
000001 TS      Queue   Data EXT   1089486979 created by AMINT R  on sysB em CI
000002 TS      Queue   Data item  1089552515 created by AMINT R  on sysB em CI
000003 TS      Queue   Data INT   1089618051 created by AMINT R  on sysB em CI
000004 TD      Queue   Data INT   1089683587 created by AMINT R  on syst em CI
000005 TD      Queue   Data INT   1089749123 created by AMINT R  on syst em CI
000006 TD      Queue   Data EXT   1089814659 created by AMINT R  on syst em CI
000007 TS      Queue   Data EXT   1089880195 created by AMINT R  on syst em CI
000008 TS      Queue   Data ININ  1089945731 created by AMINT R  on syst em CI
000009 TS      Queue   Data INT   1090011267 created by AMINT R  on syst em CI
000010 TD      Queue   Data EXT   1090076803 created by AMINT R  on syst em CI
000011 TS      Queue   Data item  1089597504 created by AMINT OR on sysA tem C
000012 TS      Queue   Data item  1089597760 created by AMINT OR on sysA tem C
Command ===> c 'data EXT' 'EXT' #4 all                          Scroll CSR
F1=Help     F2=Zoom     F3=Exit     F4=CRetriev F5=RFind    F6=RChange
F7=Up       F8=Down     F10=Left    F11=Right   F12=Cancel
```

Figure 27-29 Change data syntax

You can now perform the same functions (Change, Delete, Insert) as usual, using the same FM/Base syntax. In this case: type the command `C 'DATA EXT' 'EXT' #4 ALL'` on the command line to change the values of the "ITEM-NAME" column, as shown in Figure 27-30.

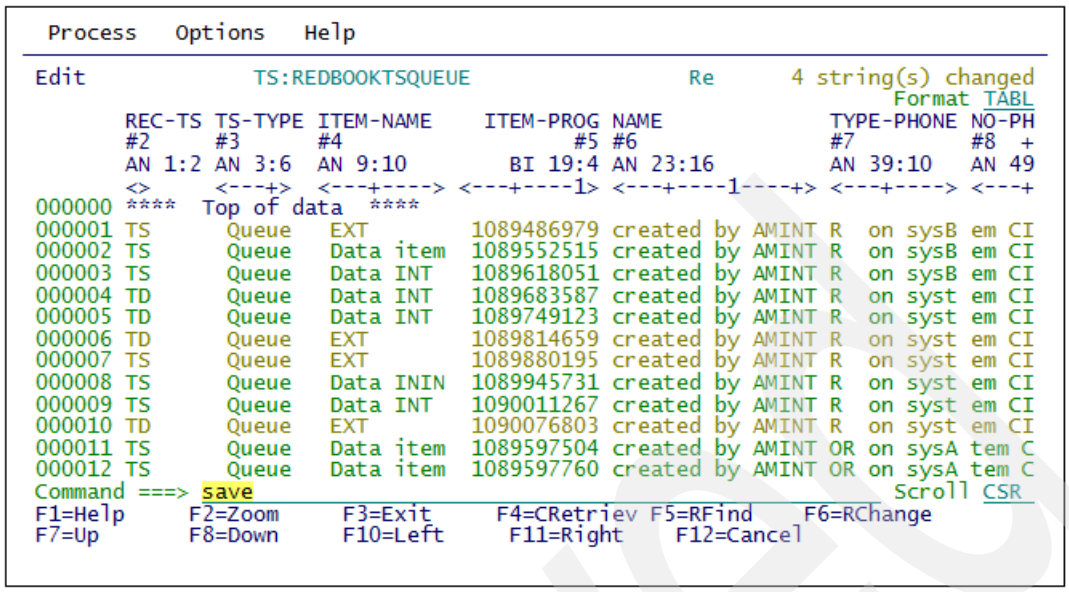


Figure 27-30 Type “SAVE” to confirm the update

You can confirm the update by typing **SAVE** on the command line. Press Enter to complete the function.

27.5 VSAM sharing and related concepts scenario

Unless DFSMS™ Record Level Sharing (RLS) is implemented, the file would have to be closed in all CICS regions before it could be processed through File Manager/Base for z/OS with integrity. If the file was opened in a CICS region with a DISP of SHARE and it was defined with SHAREOPTIONS 2, 3, or 4, it is possible that you can read from or write to the file through base FM while it remains open to CICS. But READ/WRITE integrity is not guaranteed and VSAM corruption could occur. Therefore, FM/CICS resolves this issue by allowing CICS resources to be updated with integrity while it remains open in the CICS region because all read/write requests to the file are performed through CICS.

When you open a file with READ/WRITE, FM/CICS automatically creates a buffer to store all the initial values of the record occurrences that you are editing. The FM/CICS buffer freezes the values of the records that you modify, insert, and delete during this unit of work.

For instance, we explain in the next sessions, a sample scenario to illustrate the most important concepts just described above. Starting from FM/CICS, we invoke FM/Base to perform several types of definitions and provide a high-level description of each function that we use during the following tasks:

- Create and define a VSAM file with FM/CICS
- Define and install the CSD descriptions
- Open, close, and modify file attributes with FM/CICS
- Create a job to update in batch the VSAM file defined before
- Initialize the VSAM Sharing Options and explain the “Integrity Warning” aspects
- Concurrent VSAM Update: Two test cases

27.5.1 How to create/define a VSAM file by File Manager /CICS

Here we show how to invoke the File Manager/Base utilities via FM/CICS in order to create and initialize a VSAM KSDS data set called CUSTFIL4. This file is defined and allocated with SHAREOPTIONS =3,4 to accomplish our test. Following are the steps necessary to perform this function:

1. Log in FM/CICS and start the menu navigation from the Primary Options menu (FMN3ST00). Select option **FM** (File Manager z/OS) from the main FM menu, as shown in Figure 27-31.

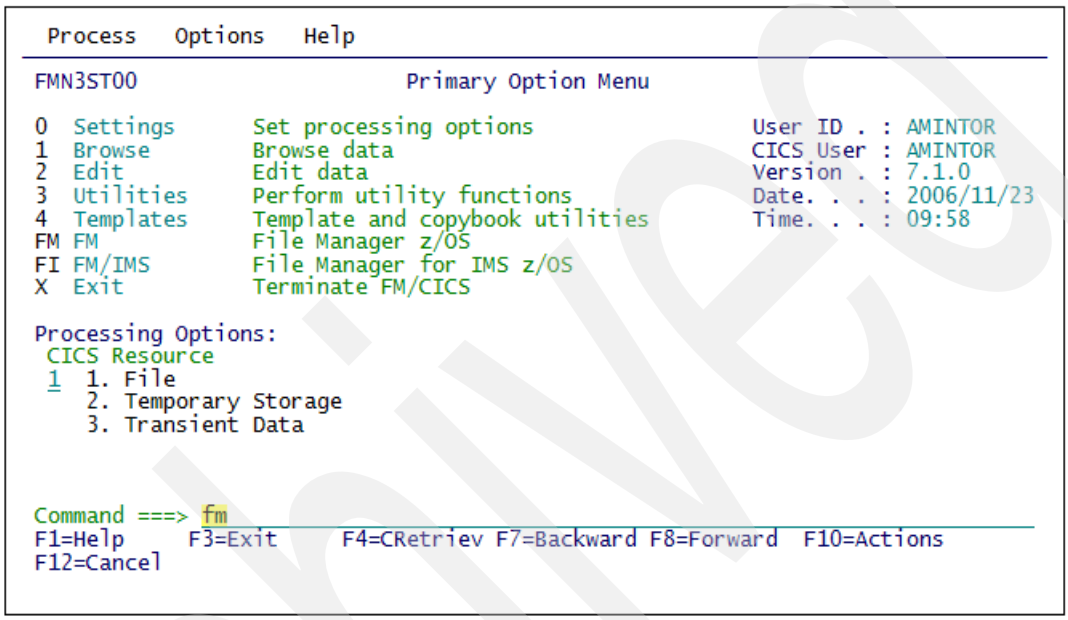


Figure 27-31 Start File Manager by entering FM on the command line

2. Press Enter to invoke the FM/Base *Primary Option Menu* (`FMNPST00 ger`) as shown in Figure 27-32.

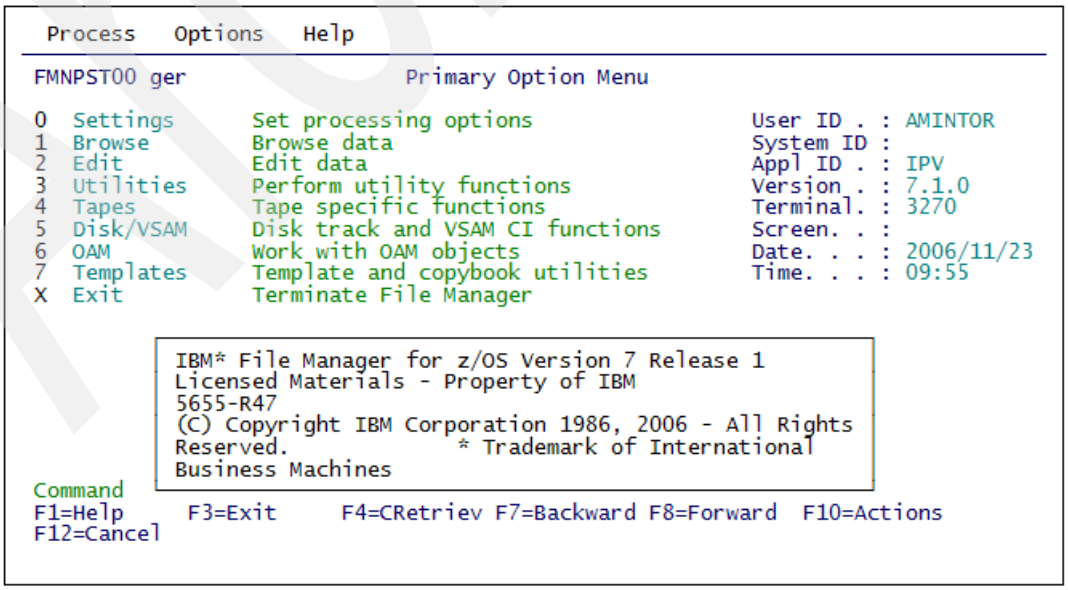


Figure 27-32 Primary Option Menu (type 3.4 on the command line to invoke the Catalog Service)

3. Fill in **3.4** on the command line to perform the VSAM definition using an existing dataset as model and press Enter to get to the Catalog Services panel (FMNPSCS) as shown in Figure 27-33.

```
 Process   Options   Help
------------------------------------------------------------------------------
FMNPSCS  ger                    Catalog Services
                                                                More:   +
  blank List catalog entries                 A Alter catalog entry
    DEF Define catalog entry               DEL Delete catalog entry
      I Display entry information            P Print catalog entries

Data Set:
   Data set name . 'AMINTOR.BOOK2006.CUS**'
   Catalog ID  . .

Processing Options:
   Entry Type                                              Sort field
   3  1. Any           6. Cluster      11. Page space    1  1. Name
      2. Non-VSAM      7. Data         12. Path             2. Creation date
      3. VSAM          8. GDG          13. User catalog     3. Free space
      4. AIX           9. Index                             4. Allocated space
      5. Alias        10. OAM          Enter "/" to select option
                                       _  Batch execution      with list
                                       _  Include Additional Qualifiers

Command ===>
F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
F12=Cancel
```

Figure 27-33 Catalog Service panel

4. File Manager displays the VSAM Data Set List already defined in your environment. Select a file that has attributes resembling those of the file that you want to create. Type **list** in the command area next to the corresponding VSAM dataset that has the attributes you want. Or you can position the dataset that you wish to select as the first occurrence in the list, select the Process pull-down menu, and select **5** List, as shown in Figure 27-34.

```
 Process   Options   Help
------------------------------------------------------------------------------
File Manager                  Data Set List                    Line 1 of 18
Catalog ID ''                                                  Types ALLVSAM
         --- Data Set Name --- sorted by NAME      ---   Entry  Prim  M Created
Command  1...5...10...15...20...25...30...35...40....   type  volume V YYYY.DDD
list     AMINTOR.BOOK2006.CUSTFILE                      KSDS            2006.308
           AMINTOR.BOOK2006.CUSTFILE.DATA                DATA  STF620    2006.308
           AMINTOR.BOOK2006.CUSTFILE.INDEX               INDEX STF620    2006.308
         AMINTOR.BOOK2006.CUSTFILE.SEQ                   KSDS            2006.308
           AMINTOR.BOOK2006.CUSTFILE.SEQ.DATA            DATA  STFS63    2006.308
           AMINTOR.BOOK2006.CUSTFILE.SEQ.INDEX           INDEX STFS63    2006.308
         AMINTOR.BOOK2006.CUSTFILE.SEQ.CLUSTER           AIX             2006.311
           AMINTOR.BOOK2006.CUSTFILE.SEQ.FRO9B2SC        DATA  STF636    2006.311
           AMINTOR.BOOK2006.CUSTFILE.SEQ.FY5UM2SC        INDEX STF636    2006.311
         AMINTOR.BOOK2006.CUSTFIL2                       KSDS            2006.308
           AMINTOR.BOOK2006.CUSTFIL2.DATA                DATA  STF505    2006.308
           AMINTOR.BOOK2006.CUSTFIL2.INDEX               INDEX STF505    2006.308
         AMINTOR.BOOK2006.CUSTFIL3                       KSDS            2006.317
           AMINTOR.BOOK2006.CUSTFIL3.DATA                DATA  STF64D    2006.317
           AMINTOR.BOOK2006.CUSTFIL3.INDEX               INDEX STF64D    2006.317
Command ===>                                                          Scroll CSR
F1=Help     F3=Exit     F4=CRetriev F5=RFind    F6=Process  F7=Up
F8=Down     F10=Left    F11=Right   F12=Cancel
```

Figure 27-34 Entering List in the Command area (Prefix line)

5. Press Enter to display the VSAM Entry Detail panel with the information for the current file as shown in Figure 27-35.

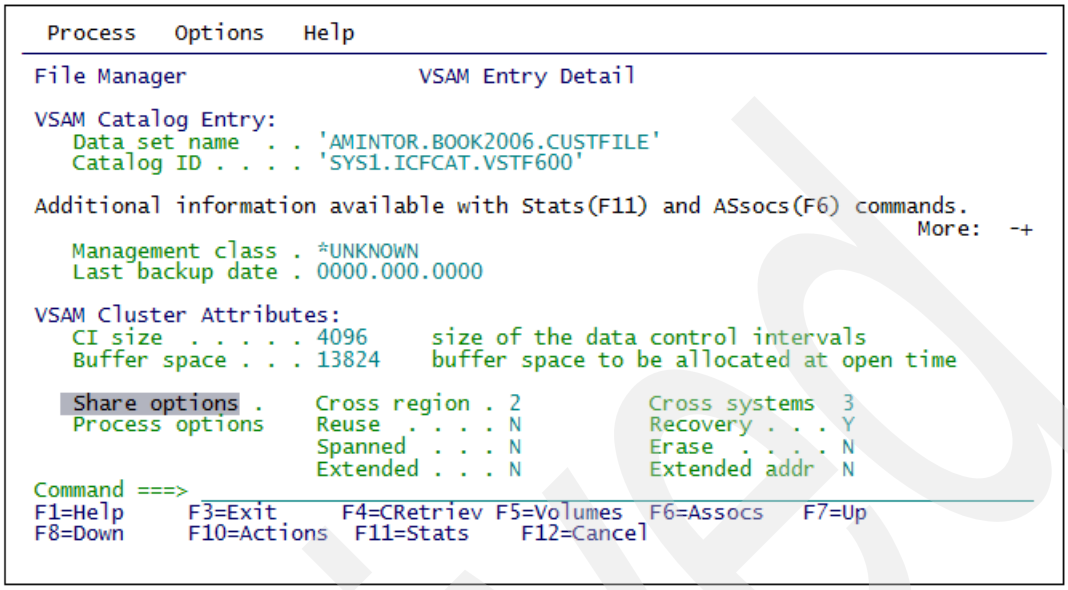


Figure 27-35 VSAM Entry Detail panel (CUSTFILE Share options)

6. When you finish checking the definitions inside your selected file, press PF3 to return to the Data Set List panel and select **Define** in the Process pull-down as option. Or, you can type `DEFINE` in the command line to obtain the same result as shown in Figure 27-36.

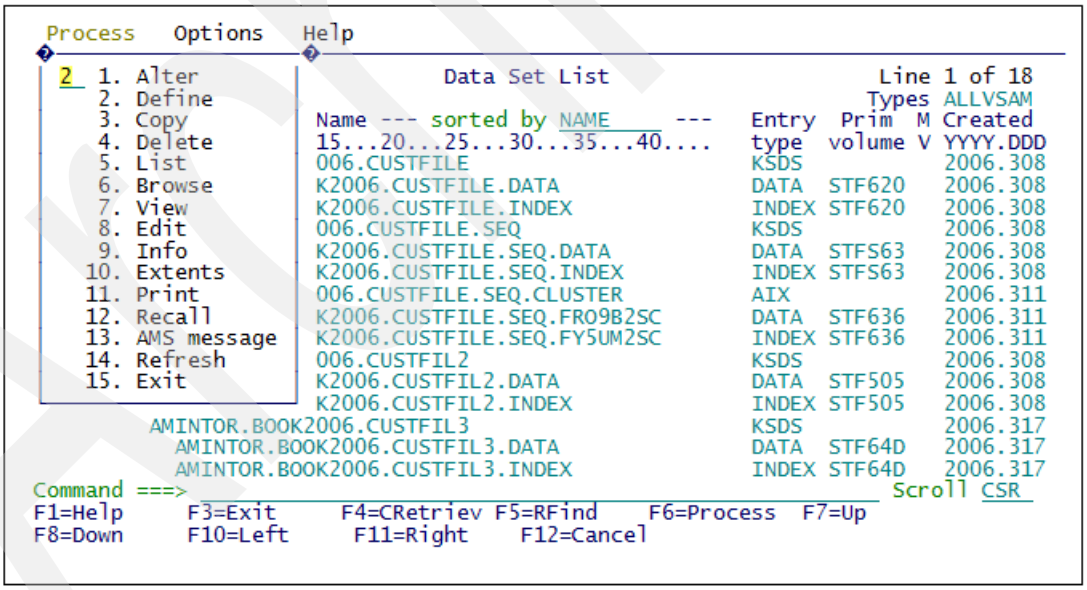


Figure 27-36 Selecting DEFINE from the Process pull down menu

7. Change the dataset name, VSAM cluster attributes, and VSAM data allocation to the new file by typing over the existing information as shown in Figure 27-37.

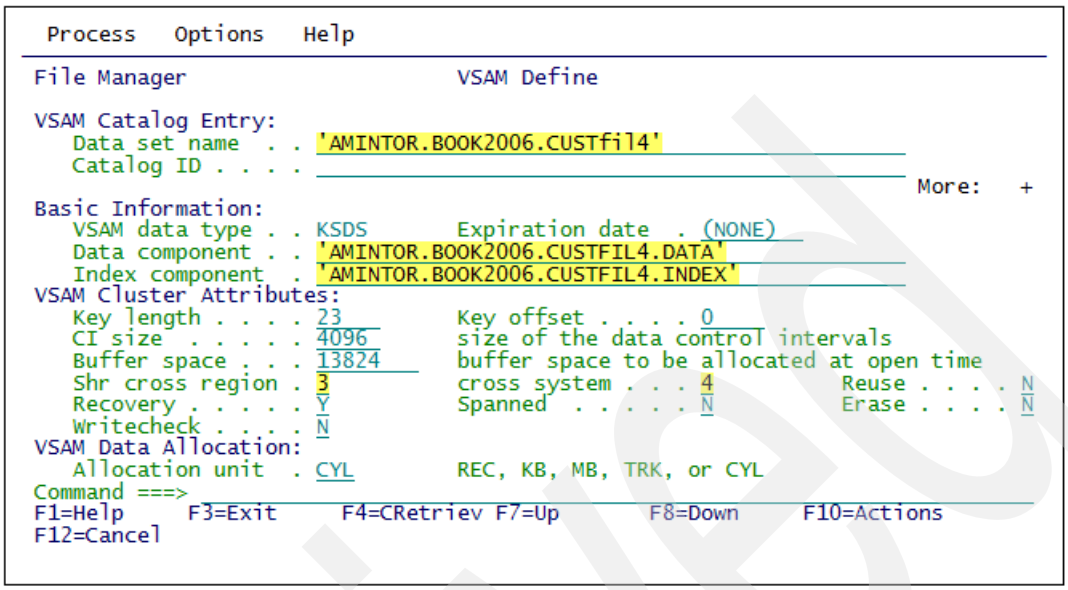


Figure 27-37 VSAM Define panel

27.5.2 Creating and defining CICS resources (CSD definitions)

When you have finished the VSAM define and the allocation stage is completed, and before the commencement of the usage of this new CUSTFIL4 under FM/CICS, you have to submit the following JCL, as shown in Example 27-1, in order to update the CICS resources in batch way. Use this job to define the CSD DESCRIPTION definitions in your CICS to guarantee your work. In most places you require the batch job to be run by the CICS administrator.

8. To continue the scenario and complete the process definition of the VSAM file, AMINTOR.BOOK2206.CUSTFIL4, to CICS, you have to modify the CICS library dataset names in the following JCL to conform to the local naming conventions.

Example 27-1 Job to define CICS resources

```
SDSF EDIT    AMINTORB (JOB04818) JCLEDIT                  Columns 00001 00072
Command ===>                                                 Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (A),
000002 //            AMINTOR,MSGCLASS=A,
000003 //            NOTIFY=AMINTOR,CLASS=A,
000004 //            MSGLEVEL=(1,1)
000005 //COPYCSD  EXEC PGM=DFHCSDUP,REGION=1M
000006 //STEPLIB  DD DISP=SHR,DSN=IBM.TS310.CICS.SDFHLOAD
000007 //DFHCSD   DD DISP=SHR,DSN=CICSVS.C31F.DFHCSD
000008 //SYSUT1   DD UNIT=SYSDA,SPACE=(1024,(100,100))
000009 //SYSPRINT DD SYSOUT=*
000010 //SYSIN    DD *
000011 DEFINE FILE(CUST004) GROUP(REDBOOK)
000012 DESCRIPTION(KSDS DATA SET FOR PD TOOLS REDBOOK)
000013        DSNAME(AMINTOR.BOOK2006.CUSTFIL4) RLSACCESS(NO) LSRPOOLID(1)
000014        READINTEG(UNCOMMITTED) DSNSHARING(ALLREQS) STRINGS(5)
000015        RECORDSIZE(136) KEYLENGTH(23) STATUS(ENABLED) OPENTIME(FIRSTREF)
000016        DISPOSITION(SHARE) DATABUFFERS(6) INDEXBUFFERS(6) TABLE(NO)
000017        MAXNUMRECS(NOLIMIT) UPDATEMODEL(LOCKING) LOAD(NO)
```

```
000018        RECORDFORMAT(F) ADD(YES) BROWSE(YES) DELETE(YES) READ(YES)
000019        UPDATE(YES) JOURNAL(NO) JNLREAD(NONE) JNLSYNCREAD(NO)
000020        JNLUPDATE(NO) JNLADD(NONE) JNLSYNCWRITE(YES) RECOVERY(NONE)
000021        FWDRECOVLOG(NO) BACKUPTYPE(STATIC)
000022 /*
****** ***************************** Bottom of Data **************************
```

9. After the run, we receive the following SYSOUT as shown in Example 27-2.

Example 27-2 Log out with description of CICS resource CUST004

```
******************************** TOP OF DATA ********************************
                  J E S 2  J O B  L O G  --  S Y S T E M  F 6      --  N O D E

18.37.48 JOB04818 ---- WEDNESDAY, 22 NOV 2006 ----
18.37.48 JOB04818  IRR010I  USERID AMINTOR  IS ASSIGNED TO THIS JOB.
18.37.48 JOB04818  ICH70001I AMINTOR  LAST ACCESS AT 17:38:16 ON WEDNESDAY, NOVE
18.37.48 JOB04818  $HASP373 AMINTORB STARTED - INIT 5    - CLASS A - SYS F6
18.37.48 JOB04818  SMF000I  AMINTORB    COPYCSD     DFHCSDUP    0000
18.37.48 JOB04818  $HASP395 AMINTORB ENDED
------ JES2 JOB STATISTICS ------
  22 NOV 2006 JOB EXECUTION DATE
           22 CARDS READ
           76 SYSOUT PRINT RECORDS
            0 SYSOUT PUNCH RECORDS
            3 SYSOUT SPOOL KBYTES
         0.00 MINUTES EXECUTION TIME
        1 //AMINTORB JOB (A),
          //            AMINTOR,MSGCLASS=A,
          //            NOTIFY=AMINTOR,CLASS=A,
          //            MSGLEVEL=(1,1)
        2 //COPYCSD  EXEC PGM=DFHCSDUP,REGION=1M
        3 //STEPLIB  DD DISP=SHR,DSN=IBM.TS310.CICS.SDFHLOAD
        4 //DFHCSD   DD DISP=SHR,DSN=CICSVS.C31F.DFHCSD
        5 //SYSUT1   DD UNIT=SYSDA,SPACE=(1024,(100,100))
        6 //SYSPRINT DD SYSOUT=*
        7 //SYSIN    DD *
ICH70001I AMINTOR  LAST ACCESS AT 17:38:16 ON WEDNESDAY, NOVEMBER 22, 2006
IEF236I ALLOC. FOR AMINTORB COPYCSD
IGD103I SMS ALLOCATED TO DDNAME STEPLIB
IGD103I SMS ALLOCATED TO DDNAME DFHCSD
IGD101I SMS ALLOCATED TO DDNAME (SYSUT1  )
        DSN (SYS06326.T183748.RA000.AMINTORB.R0114656    )
        STORCLAS (SCTEMP) MGMTCLAS (        ) DATACLAS (        )
        VOL SER NOS= TMP001
IEF237I JES2 ALLOCATED TO SYSPRINT
IEF237I JES2 ALLOCATED TO SYSIN
IEF142I AMINTORB COPYCSD - STEP WAS EXECUTED - COND CODE 0000
IGD104I IBM.TS310.CICS.SDFHLOAD                      RETAINED,  DDNAME=STEPLIB
IGD104I CICSVS.C31F.DFHCSD                           RETAINED,  DDNAME=DFHCSD
IGD105I SYS06326.T183748.RA000.AMINTORB.R0114656     DELETED,   DDNAME=SYSUT1
IEF285I   AMINTOR.AMINTORB.JOB04818.D0000102.?         SYSOUT
IEF285I   AMINTOR.AMINTORB.JOB04818.D0000101.?         SYSIN
IEF373I STEP/COPYCSD /START 2006326.1837
IEF374I STEP/COPYCSD /STOP  2006326.1837 CPU    0MIN 00.03SEC SRB    0MIN 00.00S
IEF375I  JOB/AMINTORB/START 2006326.1837
IEF376I  JOB/AMINTORB/STOP  2006326.1837 CPU    0MIN 00.03SEC SRB    0MIN 00.00S
```

```
DEFINE FILE(CUST004) GROUP(REDBOOK)
DESCRIPTION(KSDS DATA SET FOR PD TOOLS REDBOOK)
       DSNAME(AMINTOR.BOOK2006.CUSTFIL4) RLSACCESS(NO) LSRPOOLID(1)
       READINTEG(UNCOMMITTED) DSNSHARING(ALLREQS) STRINGS(5)
       RECORDSIZE(136) KEYLENGTH(23) STATUS(ENABLED) OPENTIME(FIRSTREF)
       DISPOSITION(SHARE) DATABUFFERS(6) INDEXBUFFERS(6) TABLE(NO)
       MAXNUMRECS(NOLIMIT) UPDATEMODEL(LOCKING) LOAD(NO)
       RECORDFORMAT(F) ADD(YES) BROWSE(YES) DELETE(YES) READ(YES)
       UPDATE(YES) JOURNAL(NO) JNLREAD(NONE) JNLSYNCREAD(NO)
       JNLUPDATE(NO) JNLADD(NONE) JNLSYNCWRITE(YES) RECOVERY(NONE)
       FWDRECOVLOG(NO) BACKUPTYPE(STATIC)
DFH5120 I PRIMARY CSD OPENED;  DDNAME: DFHCSD
DFH5159 I FILE CUST004  DEFINED IN GROUP REDBOOK
DFH5101 I DEFINE COMMAND EXECUTED SUCCESSFULLY.
DFH5123 I PRIMARY CSD CLOSED;  DDNAME: DFHCSD



DFH5107 I COMMANDS EXECUTED SUCCESSFULLY: 1     COMMANDS GIVING WARNING(S): 0
DFH5108 I COMMANDS NOT EXECUTED AFTER ERROR(S): 0
DFH5109 I END OF DFHCSDUP UTILITY JOB. HIGHEST RETURN CODE WAS: 0
******************************** BOTTOM OF DATA ********************************
```

What is happening in these jobs

We run this job to create and define CICS resources that are used to perform the scenario under FM/CICST. The JCL defines a new FILE(CUST004) under the GROUP(REDBOOK). The CUST004 is the logical definition for the dataset AMINTOR.BOOK2006.CUSTFIL4. The job uses the CICS module DFHCSDUPT to OPEN Define in Group Redbook and close as shown in the SYSOUT above. After this stage you use the CEDA transaction to complete the install of the Group Redbook as shown in Figure 27-38,

```
DI G(REDBOOK)
ENTER COMMANDS
 NAME      TYPE          GROUP                                    DATE   TIME
 ANS0001   FILE          REDBOOK                                  06.326 07.19.30
 CUST004   FILE          REDBOOK                                  06.326 18.37.48
 REDFILE   FILE          REDBOOK                                  06.323 23.01.53
 PROGA     PROGRAM       REDBOOK                                  06.299 10.10.49
 S737201   PROGRAM       REDBOOK                                  06.298 09.48.46
 S737202   PROGRAM       REDBOOK                                  06.312 00.01.24
 RED0      TRANSACTION   REDBOOK                                  06.298 09.48.35
 RED1      TRANSACTION   REDBOOK                                  06.299 10.10.36
 RED2      TRANSACTION   REDBOOK                                  06.312 00.01.13
 RED1      TDQUEUE       REDBOOK                                  06.298 08.23.57
 RED2      TDQUEUE       REDBOOK                                  06.323 23.06.44
 TSMOD1    TSMODEL       REDBOOK                                  06.298 08.56.41




                                                  SYSID=C31F APPLID=CICSC31F
 RESULTS: 1 TO 12 OF 12                     TIME:  18.46.21  DATE: 06.326
PF 1 HELP       3 END 4 TOP 5 BOT 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Figure 27-38 CICS CEDA transaction

10. We have just completed all the CICS definitions and are able now to Open, Browse, and Edit the VSAM CUST004 using FM/CICS.

Note: Verify the update access in the Security System with your CICS Administrator before running the job.

27.5.3 Opening, closing, and modifying file attributes with FM/CICS

FM/CICS allows you to create, edit, browse, or change data in a CICS resource. The Interactive Panel Viewer (IPV) interface is provided to assist users and perform the following tasks:

- Open, close, and modify various file attributes.
- Display catalog information associated with a data set name.
- Invoke various File Manager functions against a file.

Since you have just completed the file VSAM definition under CICS, you can display the catalog information associated with the file CUST004, as shown the next steps:

11. Log in FM/CICS and start the menu navigation from the Primary Options menu (FMN3ST00), then type **2.1** on the command line, as shown in Figure 27-39.

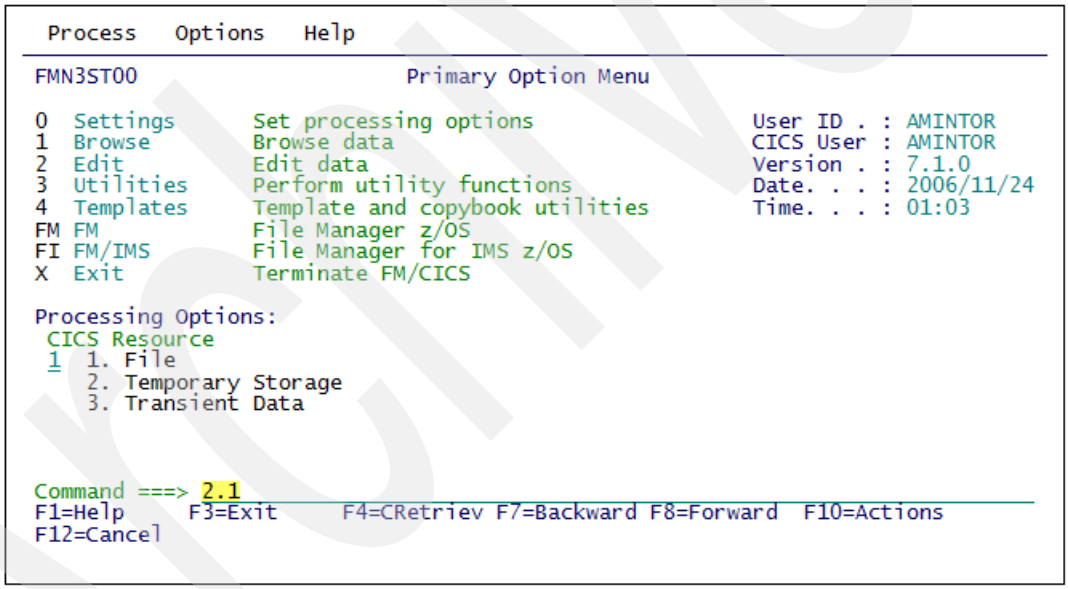


Figure 27-39 FM Primary Option Menu panel (type 2.1 on the command line)

12. Press Enter to select the CICS File Selection List panel, as shown in Figure 27-40.

```
 Process   Options   Help
FMN3PLFI                  CICS File Selection List              Row 1 of 20
                                                                   Status
                                                              O E R U A B D
                                                              P N E P D R E
  File     Data set name                            Type Sys  E A A D D O L
  CSQKCDF                                           VSAM      C E R U A B D
_ DFHCMACD                                          VSAM      C E R _ _ _ _
_ DFHDBFK                                           VSAM      C E R U A B D
_ LGCUST   ADTOOLS.MNA.DATA.CUSTOMER.ESDS           VSAM      C E R _ _ B _
_ LGENDOW  ADTOOLS.MNA.DATA.ENDOW.ESDS              VSAM      C E R _ _ B _
_ LGHOUSE  ADTOOLS.MNA.DATA.HOUSE.ESDS              VSAM      C E R _ _ B _
_ LGMOTOR  ADTOOLS.MNA.DATA.MOTOR.ESDS              VSAM      C E R _ _ B _
_ LGPOLICY ADTOOLS.MNA.DATA.POLICY.ESDS             VSAM      C E R _ _ B _
_ ANS0001  AMINTOR.BOOK2006.CUSTFIL3                KSDS      O E R U A B D
_ REDFILE  AMINTOR.BOOK2006.CUSTFIL3                KSDS      O E R U _ B _
_ CUST004  AMINTOR.BOOK2006.CUSTFIL4                VSAM      C U R U A B D
_ COMPFILE CHABERT.C31F.TRADER.COMPFILE             VSAM      C E R U A B D
_ CUSTFILE CHABERT.C31F.TRADER.CUSTFILE             KSDS      O E R U A B D
_ ERCFILE  CHABERT.FBI.TS31.FILE                    VSAM      C E R _ _ B _
Command ===> sort file                                    Scroll CSR
F1=Help     F3=Exit     F4=CRetriev F5=RFind    F7=Up       F8=Down
F10=Actions  F12=Cancel
```

Figure 27-40 CICS File Selection List panel (how to re-order the columns displayed)

13. The panel above displays all the files defined in CICS. Using this panel, you can perform many activities within the CICS resources. You are able to perform commands such as FIND, Locate, REFRESH, RESet, SELECT, and SORT. The SORT command can be executed to re-order the columns displayed on the selection list, except the line command field and Status columns. Type `Sort file` on the command line and press ENTER to sort the list by the first column as shown in Figure 27-41.

```
 Process   Options   Help
FM/CICS                   CICS File Selection List              Row 1 of 20
                                                                   Status
                                                              O E R U A B D
                                                              P N E P D R E
  File     Data set name                            Type Sys  E A A D D O L
  ANS0001  AMINTOR.BOOK2006.CUSTFIL3                KSDS      O E R U A B D
_ COMPFILE CHABERT.C31F.TRADER.COMPFILE             VSAM      C E R U A B D
_ CSQKCDF                                           VSAM      C E R U A B D
  CUSTFILE CHABERT.C31F.TRADER.CUSTFILE             KSDS      O E R U A B D
s CUST004  AMINTOR.BOOK2006.CUSTFIL4                VSAM      C U R U A B D
_ DFHCMACD                                          VSAM      C E R
_ DFHCSD   CICSVS.C31F.DFHCSD                       VSAM      C U R U A B D
_ DFHDBFK                                           VSAM      C E R U A B D
_ DFHDPFMB CICSVS.TS310.CICS.DFHDPFMB               VSAM      C E R U A B D
_ DFHDPFMP CICSVS.TS310.CICS.DFHDPFMP               PATH      O E R U A B D
_ DFHLRQ   CICSVS.C31F.DFHLRQ                       KSDS      O E R U A B D
_ ERCFILE  CHABERT.FBI.TS31.FILE                    VSAM      C E R _ _ B _
_ EZACACHE EZACACHE                                 VSAM      C U R U A B D
_ EZACONFG CICSVS.C31F.TCPIP.CONFIG                 KSDS      O D R _ _ B _
Command ===>                                              Scroll CSR
F1=Help     F3=Exit     F4=CRetriev F5=RFind    F7=Up       F8=Down
F10=Actions  F12=Cancel
```

Figure 27-41 Select a CLOSED file from the list

We have sorted the list by File name. The panel shows the dataset resources status. The CUST004 file is in *CLOSE* status. If we try to edit it, by typing an **S** in the corresponding CUST004 file in the Prefix area, when the file is in *CLOSE* status, we receive the message `Resource not enabled`, as shown in Figure 27-42.

```
 Process    Options    Help
------------------------------------------------------------------------------
FMN3PFIE                  Edit CICS File Entry Panel      Resource not enabled

Input CICS VSAM File:
   File name  . . . . CUST004
   Sysid  . . . . . . *
   Start position . .
   Record limit . . .          Record sampling
   Inplace edit . . .          Prevent inserts and deletes
   Lock resource  . .          Name
Copybook or Template:
   Data set name  . . 'AMINTOR.BOOK2006.TEMPLATE'
   Member . . . . . . TEMPCOST    Blank or pattern for member list
Processing Options:
 Copybook/template   Start position type   Enter "/" to select option
 1  1. Above            1. Key               Edit template    Type (1,2,S)
    2. Previous         2. RBA               Include only selected records
    3. None             3. Record number     Create audit trail
    4. Create dynamic

Command ===>
F1=Help     F3=Exit     F4=Expand   F7=Backward F8=Forward  F10=Left
F11=Right    F12=Cancel
```

Figure 27-42 The panel shows the message, Resource not enabled

14. Type * in the corresponding Sysid field and press ENTER to receive the panel *CICS File Selection List* again, as shown in Figure 27-43.

```
 Process    Options    Help
------------------------------------------------------------------------------
FMN3PLFI                  CICS File Selection List                 Row 1 of 1
                                                                     Status
                                                                O E R U A B D
                                                                P N E P D R E
  File     Data set name                              Type Sys  E A A D D O L
s CUST004  AMINTOR.BOOK2006.CUSTFIL4                  KSDS      o E R U A B D
 ****  End of data  ****




Command ===>                                                    Scroll CSR
F1=Help     F3=Exit     F4=CRetriev F5=RFind    F7=Up       F8=Down
F10=Actions  F12=Cancel
```

Figure 27-43 CICS File Selection List panel (how to change STATUS)

15. Type **o** in the OPE column to open and permit at the FM/CICS *Edit CICS File Entry* panel (FMN3PFIE) to modify the CUST004 dataset data, as shown in Figure 27-44.

```
 Process    Options    Help
------------------------------------------------------------------------------
FMNPVET          FI:CUST004 DS:AMINTOR.BOOK2006    Rec 0
                                                     Type KSDS     Format TABL
       REC-A  DEPT   EMP-NO EMP-COD NAME              CONTACT     TEL
       #2     #3        #4 #5      #6                #7          #8
       AN 1:2 AN 3:1 ZD 4:2 AN 6:1  AN 7:17           AN 24:10    AN 34:12
       <>     -      <-> -        <---+----1----+-> <---+----> <---+----1->
000000 ****  Top of data  ****
000001 b0     1           0 B      Grans, Marianna   cell Phonx 0233-777-958
000002 b0     1          16 B      Graham, Cinzia    cell Phonx 0112-555-673
000003 b0     1          27 B      Grahan, Cinzia    cell Phonx 0112-555-673
000004 b1     2           9 B      Liszt, Mackenzie  cell Phonx 0481-555-474
000005 b2     1          30 C      Kleiner, Giulio   cell phone 054-11-5555-
000006 b3     1          15 C      Graham, Holly     cell phone 0135-555-233
000007 b3     1          15 C      Graham, Marianna  work phone 135-555-2338
000008 b4     1          15 B      Graham, Holly     work phone 112-555-6736
000009 b4     1          20 D      Parris,April Lynn work phone 54-11-5555-4
000010 b4     2           0 B      Dewitt, Howdy     work phone 642-555-3852
000011 b4     3           0 C      Dewitt, cinzia    work phone 650-555-0547
000012 b4     3           0 C      Dewitt, Cinzia    work phone 650-555-0547
Command ===>                                                   Scroll CSR
F1=Help      F2=Zoom      F3=Exit      F4=CRetriev F5=RFind     F6=RChange
F7=Up        F8=Down      F10=Left      F11=Right    F12=Cancel
```

Figure 27-44 Edit the CUST04 using a template

16. You are now able to add, delete, and update the CUST004 occurrences under a CICS session using a template.

Note: The previous panel displays the CUST004 data using a Template (TEMPCOST) that has been defined by FM/Base before while under ISPF. FM/CICS is able to connect these external CICS resources, and to use the Templates definitions, through the IPV interface.

You could receive a short text message "*Update required*" displayed in the upper right corner of the *Edit CICS File Entry* (FMN3PFIE) Panel. This error occurs while you attempts to process an Edit function on a VSAM that does not have the "UPD" Status properly set as "U".

27.5.4 Create a batch process to verify the VSAM concurrent access

We continue the scenario and we use and invoke FM/Base, starting from FM/CICS, to define the batch process (JCL) that updates the CUST004 occurrences.

In this example we describe how to generate JCL to perform a global change and replace the new values in CUST004 in batch. The steps necessary to perform this function is described by the following steps.

17. Log in FM/CICS and start the menu navigation from the Primary Options menu (FMN3ST00). Type **FM.3.6** on the command line, as shown in Figure 27-45.

```
 Process   Options   Help
-------------------------------------------------------------------------------
FMN3ST00                      Primary Option Menu

0  Settings      Set processing options                 User ID . : AMINTOR
1  Browse        Browse data                            CICS User : AMINTOR
2  Edit          Edit data                              Version . : 7.1.0
3  Utilities     Perform utility functions              Date. . . : 2006/11/24
4  Templates     Template and copybook utilities        Time. . . : 13:35
FM FM            File Manager z/OS
FI FM/IMS        File Manager for IMS z/OS
X  Exit          Terminate FM/CICS

Processing Options:
 CICS Resource
 1  1. File
    2. Temporary Storage
    3. Transient Data


Command ===> fm.3.6
F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
F12=Cancel
```

Figure 27-45 Type 3.6 to invoke the Find/Change Utility panel

18. Press Enter to get to the *Find/Change Utility* (FMNPFCH) panel as shown in Figure 27-46.

```
 Process   Options   Help
-------------------------------------------------------------------------------
File Manager                Find/Change Utility

Input Partitioned, Sequential or VSAM Data Set:
   Data set name  . 'amintor.book2006.custfil4'
   Member . . . . . *        (Blank - selection, pattern - process list)
   Volume serial  .          (If not cataloged)
   Record count . . ALL      (Number of records to be searched)
- Additional options
Listing data set  . SRCHFOR.LIST

Enter "/" to select option                      Listing Option  ISPF Packing
_ JCL Source format          Immediate change   1 1. Long       1 1. Asis
  Use REXX proc *          / Batch execution      2. Summary      2. Pack
  REXX no update             Directory integrity                  3. Unpack
  Advanced member selection  Stats off                            4. None
                           _ CAPS initially on                    5. Skip

Process List:
Sel Name     Prompt       Alias-of     Size Created  Changed          ID
Command ===> c 'CELL.Phone' 'cell phonx'                          Scroll CSR
F1=Help     F3=Exit     F4=CRetriev F5=Refresh  F7=Up       F8=Down
F12=Cancel
```

Figure 27-46 Create a JCL to execute a CHANGE command in batch

19. Type the command **`c 'CELL.Phone' 'cell Phonx'`** as shown above in the previous figure. Set the Batch execution to define the JCL that defines the statement to CHANGE in batch mode the CUST004 Dataset. Press Enter to generate the JCL statements that we use subsequently, as shown in Figure 27-47.

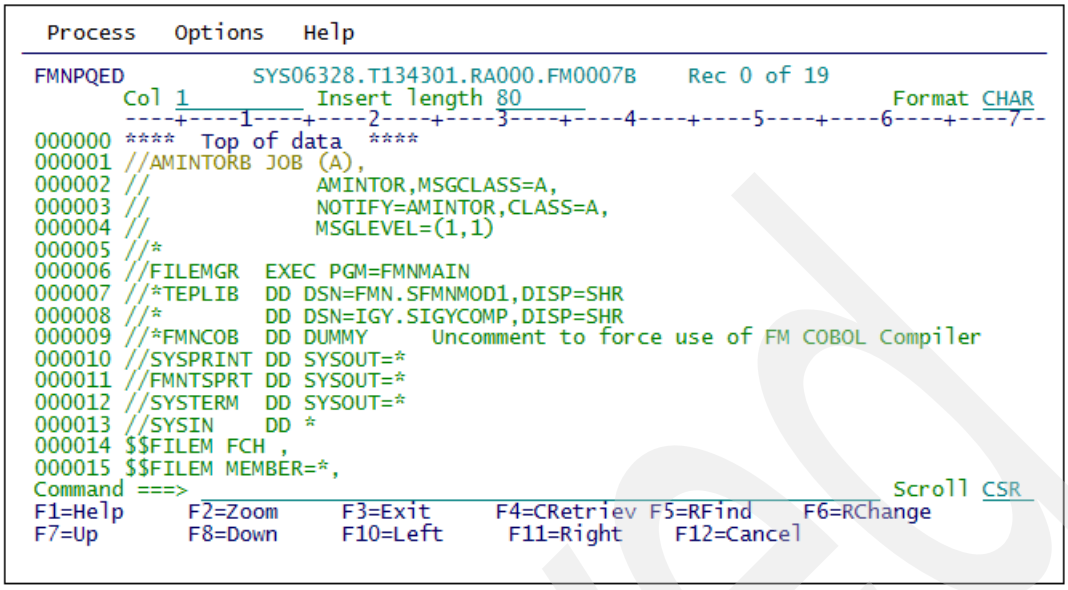


Figure 27-47 The JCL has been created

We have completed the automatic generation stage and created the JCL using the File Manager/CICS panels to test and verify the “concurrent update aspects” under FM/CICS. The JCL that we have just created has been saved into a PDS library AMINTOR.BOOK2006.JCL. The name of this member is $FMCHAN1 and the details are shown in Example 27-3.

Example 27-3 Batch process to modify the CUST004 dataset

```
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (A),
000002 //            AMINTOR,MSGCLASS=A,
000003 //            NOTIFY=AMINTOR,CLASS=A,
000004 //            MSGLEVEL=(1,1)
000005 //*
000006 //FILEMGR  EXEC PGM=FMNMAIN
000007 //*TEPLIB  DD DSN=FMN.SFMNMOD1,DISP=SHR
000008 //*        DD DSN=IGY.SIGYCOMP,DISP=SHR
000009 //*FMNCOB  DD DUMMY     UNCOMMENT TO FORCE USE OF FM COBOL COMPILER
000010 //SYSPRINT DD SYSOUT=*
000011 //FMNTSPRT DD SYSOUT=*
000012 //SYSTERM  DD SYSOUT=*
000013 //SYSIN    DD *
000014 $$FILEM FCH ,
000015 $$FILEM MEMBER=*,
000016 $$FILEM PACK=ASIS,
000017 $$FILEM DSNIN=AMINTOR.BOOK2006.CUSTFIL4
000018 C 'CELL.Phone' 'cell Phonx'
000019 /+
************************** Bottom of Data **********************************
```

These are the $FILEM statements. We assume that you are still editing (under FM/CICS) the same VSAM file (AMINTOR.BOOK2006.CUSTFIL4). Afterwards, we open a new PCOMM session and we run (submit) the $FMCHAN1 job from SDSF under TSO.

27.5.5 Initializing the VSAM sharing options; explaining the Integrity Warning

In this scenario, you are editing and updating record occurrences, using the previous FM/CICS (FMNPVET) panel under the CUST004 VSAM file.

While you are editing a VSAM file with Share Options 3,4 (Cross region=3, and Cross systems=4), another user or Application could access and update the CUST004 Dataset (online or in batch mode) in the same time.

Accordingly, we initialize the *VSAM Edit sharing options* panel twice, using different values every time. We analyze the results and the FM/CICS behavior in both cases within the following sessions:

- Test Case 1: Save before starting using the CHANGE ALL command
- Test Case 2: Auto Retry for CHANGE ALL

We note that the *VSAM Processing Options* configuration provides security, and we improve the usability for those files that are in such a shared environment.

FM/Base is able to identify these events during the OPEN stage, and we display a `Warning` message through a panel, every time you edit a shared VSAM file allocated to another task, as shown in Figure 27-48.

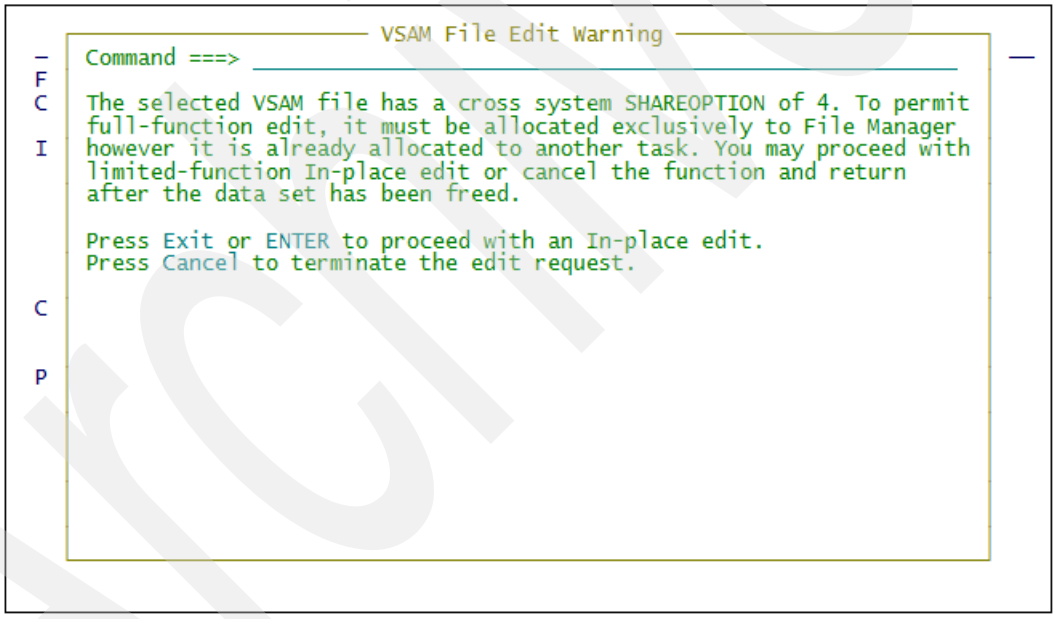


Figure 27-48 FM/Base shows the Edit Warning Panel - the VSAM could be processed in-place edit

Instead, by using FM/CICS, you receive a different panel when you are attempting to confirm (using the EXIT or SAVE command) the updating activities done on this Dataset.

In fact, when you process an Edit function on a shared VSAM file that has been already allocated and modified by another task, FM/CICS is able to identify if the same VSAM occurrences has been modified (using the buffer described before), during the CLOSE stage, In this case, FM/CICS provide you the panel, *Record Updated by other user*, as shown in Figure 27-49.

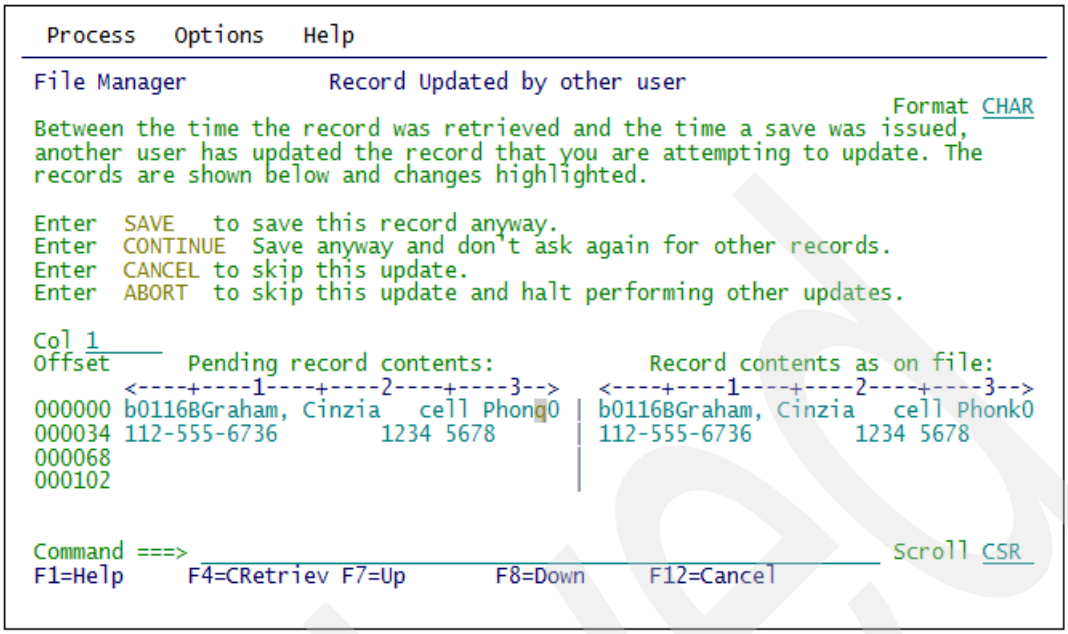


Figure 27-49 FM/CICS displays the Record Update by other user panel

When you receive the panel above, you can choose among the following four possibilities:

- SAVE:
  - Save this record.
- CONTINUE:
  - Save and do not ask again for other records.
- CANCEL:
  - Skip this update.
- ABORT:
  - Skip this update and halt performing other updates.

Depending on your FM/CICS ‘Settings Processing Options” you receive different behavior and information while you are editing the VSAM files. But, you are always notified if you change the occurrences, when others applications modify the same records (CUST004) that you are updating.

We have already explained the meaning of these “Processing Options” attributes, one by one, inside the Set the processing options paragraph. We verify, within the next *Settings processing Options* Panel, the details of the first configuration that we use as Test Case (1). After, we analyze and compare the results, using the second configuration as Test Case (2).

TestCase1: Save before starting using the CHANGE ALL command

Here we show how to set the VSAM Processing Options in order to provide the security and improve the usability for those files that are in such a shared environment. We display the actual *VSAM Edit sharing options* to justify how the settings and the values of these parameters could influence the FM/CICS behaviors.

We proceed as follows:

1. To start the TestCase1 and verify your VSAM Edit sharing options panel, type the command **=0.7** from the Primary Option Menu (FMN3ST00) as shown in Figure 27-50.

```
 Process   Options   Help
FMN3ST00                    Primary Option Menu

0  Settings      Set processing options              User ID . : AMINTOR
1  Browse        Browse data                         CICS User : AMINTOR
2  Edit          Edit data                           Version . : 7.1.0
3  Utilities     Perform utility functions           Date. . . : 2006/11/21
4  Templates     Template and copybook utilities     Time. . . : 23:21
FM FM            File Manager z/OS
FI FM/IMS        File Manager for IMS z/OS
X  Exit          Terminate FM/CICS

Processing Options:
 CICS Resource
 1  1. File
    2. Temporary Storage
    3. Transient Data


Command ===> =0.7
F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
F12=Cancel
```

Figure 27-50 Type 0.7 to invoke the VSAM Edit Sharing Options panel

2. Press Enter to display the current "*VSAM edit Sharing Options*" configuration within the FM/ CICS panel, as shown in Figure 27-51.

```
 Process   Options   Help
File Manager           VSAM Edit Sharing Options

These options apply only when editing VSAM files that are being shared.

 Autosave frequency            0

Options for the CHANGE ALL command
 SAVE before starting          on   Enter ON or OFF.
 Change all autosave frequency 1
 Auto retry for change all     off  Enter ON or OFF.




Command ===> =2
F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
F12=Cancel
```

Figure 27-51 The panel shows the actual VSAM sharing options

3. After pressing Enter, FM/CICS displays the panel above with the actual *VSAM Edit sharing options* values. In order to clarify the next steps of this TestCase1, we explain the meaning and the settings of the following parameters:

- Autosave = 0. There is no “automatic save”. The only way to SAVE the changes done during the edit session are:
  - Type SAVE on the command line.
  - Press PF3 and confirm the updating.
- Save before Starting = ON. This option is related to the CHANGE ALL command and takes place when you are editing a shared VSAM File. If set to “ON”, this parameter ensures that during the CHANGE ALL process, all the updates done are saved, even if another user makes a different change on the same records. It means (and we verify this aspect during this TestCase) that your “session updates” can replace the eventual updates done before (but within your task) from another application.
- Change All Autosave frequency = 1. This option is related, and it take places only, when the CHANGE ALL command is executed. If you initialize with the value 1, FM/CICS does not ask you to type **SAVE** on the command line to confirm the changes, every time you press Enter to perform a CHANGE ALL command. File Manager automatically issues a SAVE command for the file.

4. Check that the Processing Option for COPYBOOK/TEMPLATE has been set to 1 (Check it from the FM/CICS Main Panel), type “=2” on the command line and press ENTER. The Edit Entry session panel is displayed, as shown in Figure 27-52.

```
 Process   Options   Help
FM/CICS                  Edit CICS File Entry Panel

Input CICS VSAM File:
   File name  . . . . cust004
   Sysid  . . . . . .
   Start position . .
   Record limit . . .          Record sampling
   Inplace edit . . .          Prevent inserts and deletes
   Lock resource  . .          Name
Copybook or Template:
   Data set name  . . 'amintor.book2006.template'
   Member . . . . . . tempcost    Blank or pattern for member list
Processing Options:
 Copybook/template   Start position type   Enter "/" to select option
 1  1. Above            1. Key               Edit template    Type (1,2,S)
    2. Previous         2. RBA               Include only selected records
    3. None             3. Record number     Create audit trail
    4. Create dynamic

Command ===>
F1=Help     F3=Exit     F4=Expand   F7=Backward F8=Forward  F10=Left
F11=Right    F12=Cancel
```

Figure 27-52 Edit CICS File Entry Panel

5. On the *Edit CICS File Entry Panel* we specify:
   - For the Input CICS VSAM File:
     - The Dataset name CUST004 already described inside the CSD
   - For the Copybook o Template:
     - The dataset where the Template is stored; in this example, AMINTOR.BOOK2006.TEMPLATE
   - Member: name of the Template
     - In this example, TEMPCOST

Select **1** for Processing Options and press Enter to display the actual file values, as shown in Figure 27-53.

```
 Process    Options    Help
FMNPVET          FI:CUST004 DS:AMINTOR.BOOK2006     Rec 0
                                                      Type KSDS      Format TABL
       REC-A  DEPT    EMP-NO EMP-COD NAME              CONTACT     TEL
       #2     #3        #4 #5       #6                #7          #8
       AN 1:2 AN 3:1 ZD 4:2 AN 6:1  AN 7:17           AN 24:10    AN 34:12
       <>     -       <-> -        <---+----1----+-> <---+----> <---+----1->
000000 ****  Top of data  ****
000001 b0      1          0 B       Grans, Marianna   cell Phonx wk0233-777-9
000002 b0      1         16 B       Graham, Cinzia    cell Phonx wk0112-555-6
000003 b0      1         27 B       Grahan, Cinzia    cell Phonx wx0112-555-6
000004 b1      2          9 B       Liszt, Mackenzie  cell Phonx wx0481-555-4
000005 b2      1         30 C       Kleiner, Rick     CELL.Phonx e054-11-5555
000006 b3      1         15 C       Graham, Holly     CELL.Phonx e0135-555-23
000007 b3      1         15 C       Graham, Marianna  work phone 135-555-2338
000008 b4      1         15 B       Graham, Holly     work phone 112-555-6736
000009 b4      1         20 D       Parris,April Lynn work phone 54-11-5555-4
000010 b4      2          0 B       Dewitt, Howdy     work phone 642-555-3852
000011 b4      3          0 C       Dewitt, cinzia    work phone 650-555-0547
000012 b4      3          0 C       Dewitt, Cinzia    work phone 650-555-0547
Command ===>                                                    Scroll CSR
F1=Help      F2=Zoom      F3=Exit      F4=CRetriev F5=RFind     F6=RChange
F7=Up        F8=Down      F10=Left      F11=Right     F12=Cancel
```

Figure 27-53 The panel shows the lines before the update

6. We now concentrate our attention on the fifth and sixth line displayed above. While we are continuing to edit the file, we open a new PCOMM session under ISPF interface and execute the $FMCHAN1 Job from FM/Base. as shown in Figure 27-54.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       AMINTOR.BOOK2006.JCL($FMCHAN1) - 01.07          Columns 00001 00072
Command ===> sub                                            Scroll ===> CSR
****** ***************************** Top of Data ********************************
000001 //AMINTORB JOB (A),
000002 //            AMINTOR,MSGCLASS=A,
000003 //            NOTIFY=AMINTOR,CLASS=A,
000004 //            MSGLEVEL=(1,1)
000005 //*
000006 //FILEMGR  EXEC PGM=FMNMAIN
000007 //*TEPLIB  DD DSN=FMN.SFMNMOD1,DISP=SHR
000008 //*         DD DSN=IGY.SIGYCOMP,DISP=SHR
000009 //*FMNCOB  DD DUMMY     UNCOMMENT TO FORCE USE OF FM COBOL COMPILER
000010 //SYSPRINT DD SYSOUT=*
000011 //FMNTSPRT DD SYSOUT=*
000012 //SYSTERM  DD SYSOUT=*
000013 //SYSIN    DD *
000014 $$FILEM FCH ,
000015 $$FILEM MEMBER=*,
000016 $$FILEM PACK=ASIS,
000017 $$FILEM DSNIN=AMINTOR.BOOK2006.CUSTFIL4
000018 C 'CELL.Phonx' 'cell PhonT'
000019 /+
```

Figure 27-54 Submit the job to modify the CUST04 file in batch

7. The previous panel shows the CHANGE function (`C 'CELL.Phonx ' 'cell PhonK'`) that we execute in batch mode. We are using this solution to execute a concurrent update access against the same VSAM (CUST004) instances. Press Enter to verify the job's results, as shown in Example 27-4.

Example 27-4 AMINTORB SYSOUT

```
------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY AMINTORB JOB01020  DSID   102 LINE 1       COLS 02- 81
 COMMAND INPUT ===>                                            SCROLL ===> CSR
IBM File Manager for z/OS


 DEFAULT SET FUNCTION VALUES:

PRINTOUT=SYSPRINT  HEADERPG=YES  ASCII=BOTH     RECLIMIT=(1,*)
PRINTLEN=132       PAGESKIP=NO   PAD=OFF        EOD=/*
PAGESIZE=60        DATAHDR=YES   DBCSPRT=OFF    LANGUAGE=ENGLISH
PRTTRANS=ON        DUMP=UPDOWN   TAPELBL=SL     CYLHD=ABSOLUTE
SMFNO=000          PRTDISP=MOD   USEIOX=DISABLE IOX=
TEMPHLQ=
ABENDCC=NONE       WIDEPRT=YES

IBM File Manager for z/OS
$$FILEM FCH ,
$$FILEM MEMBER=*,
$$FILEM PACK=ASIS,
$$FILEM DSNIN=AMINTOR.BOOK2006.CUSTFIL4
C 'CELL.Phonx' 'cell PhonT'
Find/Change Listing DSN: AMINTOR.BOOK2006.CUSTFIL4

                           ----- STRING(S) FOUND/CHANGED -----
Record Number
        5  b2130CKleiner, Rick    cell PhonTe054-11-5555-6298     1234 5678
        6  b3115CGraham, Holly    cell PhonTe0135-555-2338        9012 3456

  -- Find/Change summary section --
  Records found: 2 Records processed: 18
  ISPF packing option: ASIS
  Search cols: 1:136
  Longest record: 136
  Edit options in effect: CAPS:OFF

                Count  Value
  Found   :         2  >CELL.PHONX<
  Changed :         2  >cell PhonK<
  Listing lines may be truncated due to listing output line width.
------------------------------------------------------------------------------
******************************** BOTTOM OF DATA ********************************
```

8. You can verify in this SYSOUT that job has just changed the fifth and sixth records in the CUST004 file (they are the corresponding rows that we have highlighted above). Now, we can perform the CHANGE ALL command using the FM/CICS Panel, as shown in Figure 27-55.

```
 Process    Options    Help
FMNPVET           FI:CUST004 DS:AMINTOR.BOOK2006     Rec 0
                                                       Type KSDS      Format TABL
       REC-A  DEPT   EMP-NO EMP-COD NAME              CONTACT     TEL
       #2     #3         #4 #5      #6                #7          #8
       AN 1:2 AN 3:1 ZD 4:2 AN 6:1  AN 7:17           AN 24:10    AN 34:12
       <>     -        <-> -       <---+----1----+-> <---+----> <---+----1->
000000 ****  Top of data  ****
000001 b0     1           0 B       Grans, Marianna   cell Phonx wk0233-777-9
000002 b0     1          16 B       Graham, Cinzia    cell Phonx wk0112-555-6
000003 b0     1          27 B       Grahan, Cinzia    cell Phonx wx0112-555-6
000004 b1     2           9 B       Liszt, Mackenzie  cell Phonx wx0481-555-4
000005 b2     1          30 C       Kleiner, Rick     CELL.Phonx e054-11-5555
000006 b3     1          15 C       Graham, Holly     CELL.Phonx e0135-555-23
000007 b3     1          15 C       Graham, Marianna  work phone 135-555-2338
000008 b4     1          15 B       Graham, Holly     work phone 112-555-6736
000009 b4     1          20 D       Parris,April Lynn work phone 54-11-5555-4
000010 b4     2           0 B       Dewitt, Howdy     work phone 642-555-3852
000011 b4     3           0 C       Dewitt, cinzia    work phone 650-555-0547
000012 b4     3           0 C       Dewitt, Cinzia    work phone 650-555-0547
Command ===> CHANGE 'Phonx' 'PhonK' #7 ALL                          Scroll CSR
F1=Help     F2=Zoom     F3=Exit     F4=CRetriev F5=RFind    F6=RChange
F7=Up       F8=Down     F10=Left     F11=Right    F12=Cancel
```

Figure 27-55 This CHANGE command modifies the first six lines

9. The **CHANGE 'Phonx' 'PhonK' #7 ALL** Command modifies all the CONTACT columns occurrences that have the value "Phonx". Press Enter to verify the results of this CHANGE ALL command, as shown in Figure 27-56.

Note: The occurrences 5 and 6 have been already modified from the Job. The actual value is: `cell PhonT`. We do not see the new value because we have not refreshed the panel.

```
 Process    Options    Help
File Manager           Record Updated by other user
                                                          Format CHAR
Between the time the record was retrieved and the time a save was issued,
another user has updated the record that you are attempting to update. The
records are shown below and changes highlighted.

Enter  SAVE     to save this record anyway.
Enter  CONTINUE  Save anyway and don't ask again for other records.
Enter  CANCEL to skip this update.
Enter  ABORT  to skip this update and halt performing other updates.

Col 1
Offset      Pending record contents:            Record contents as on file:
       <----+----1----+----2----+----3-->   <----+----1----+----2----+----3-->
000000 b2130CKleiner, Rick     CELL.PhonKe | b2130CKleiner, Rick     cell PhonTe
000034 054-11-5555-6298     1234 5678      | 054-11-5555-6298     1234 5678
000068                                     |
000102                                     |


Command ===>                                                        Scroll CSR
F1=Help     F4=CRetriev F7=Up       F8=Down     F12=Cancel
```

Figure 27-56 This panel is displayed when another user has updated the records that you are attempting to update

10. Because the job ($FMCHAN1) had updated the same record number 5 (Kleiner Rick), FM/CICS displays the “Record Update by other User” panel, with the possibility to choose the best option for these pending records. If you choose the “SAVE” option, you then overwrite (lose) the updates done from the other application ($FMCHAN1). Type SAVE and press Enter. FM/CICS displays again the same panel for the Record number 6, as shown in Figure 27-57.

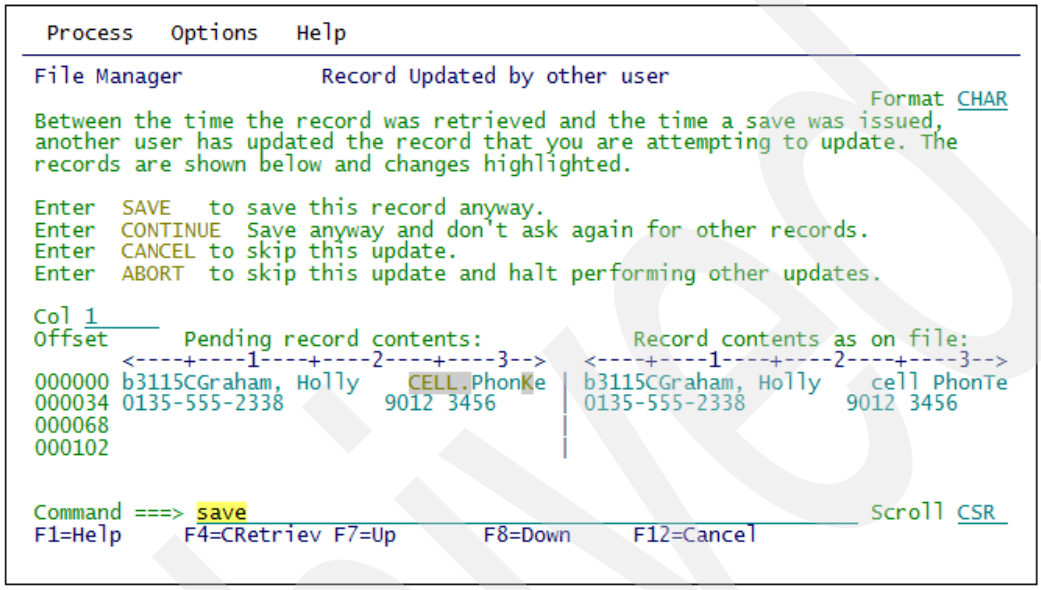


Figure 27-57 The SAVE command overwrites the changed done from $FMCHAN1 via batch

11. Type SAVE to overwrite the changes from the job and confirm your update. Press Enter to complete the CHANGE ALL command and receive the confirm, as shown in Figure 27-58.

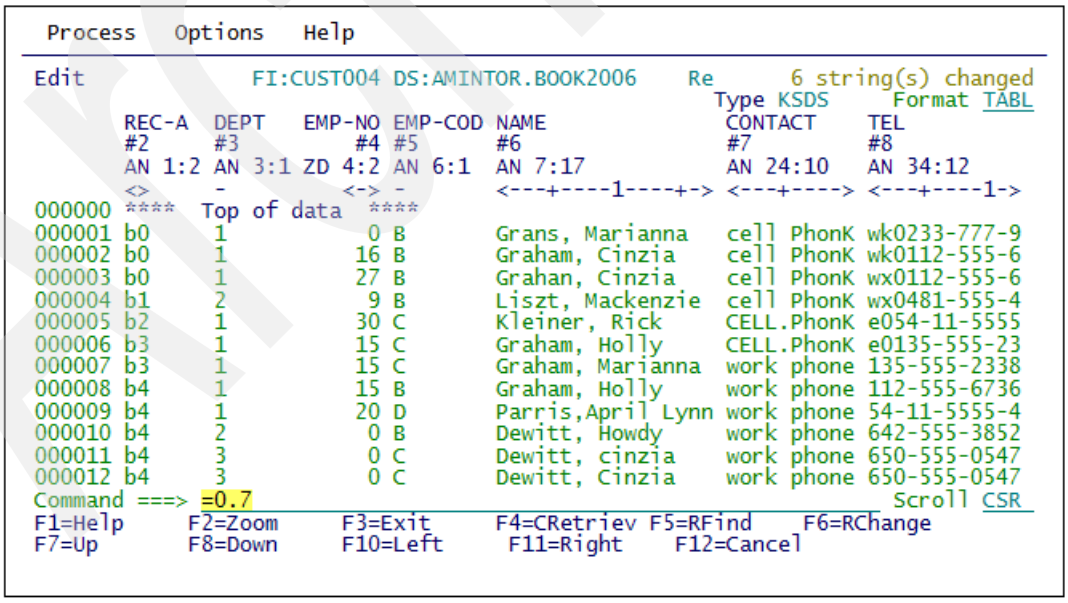


Figure 27-58 How to invoke the “VSAM Edit sharing options panel

12. The panel notified the end of TestCase1. To start the TestCase2 and verify the different FM/CICS behavior you only have to modify the “VSAM Edit sharing options” panel. Type the command **=0.7** to display this panel.

TestCase2: Auto Retry for CHANGE ALL command

Here we show how to set the VSAM Processing Options in order to provide the security and improve the usability for files in a shared environment. We skip many steps because they have been already described in the previous paragraph.

TestCase2 differs from the previous one because the *Auto Retry for Change* value is set to 'ON', as shown in Figure 27-59.

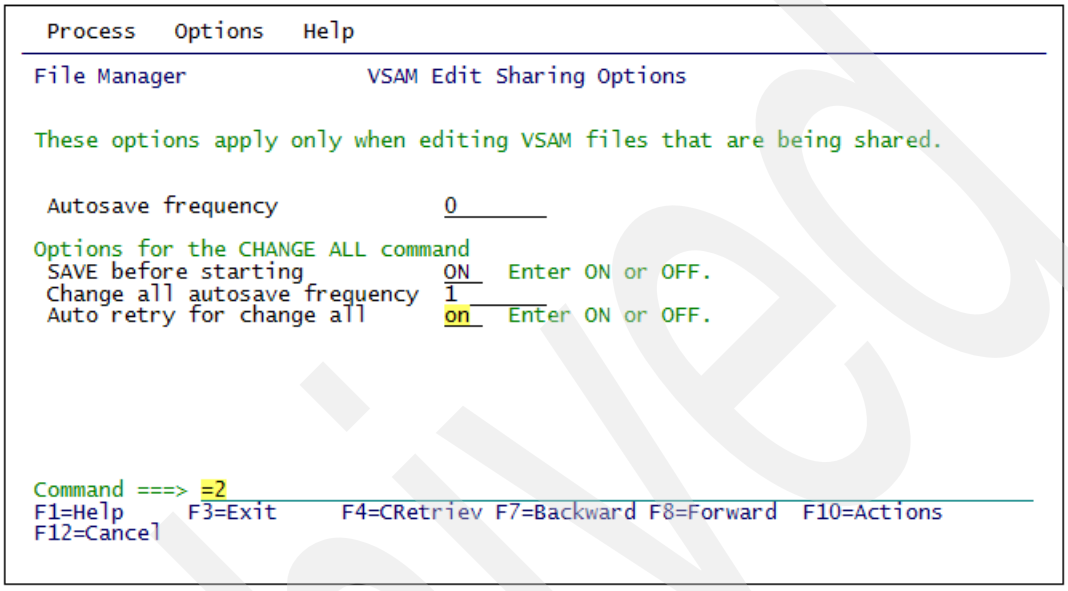


Figure 27-59 Set the Auto Retry for Change option to 'ON'

Note: In the previous paragraph we have already discussed the features of the *VSAM Edit sharing options*. We explained how the settings and the values of these parameters could influence the behavior of FM/CICS. Step 3 described the details of these options.

To complete TestCase2, we have to reset the column `CELL.Phonx`, instead using `CELL.PhonK` for records 5 and 6 CONTACT with the original values as:

- `CELL.Phonx` instead of `CELL.PhonK` for records 5 and 6
- `cell Phonx` instead of `cell PhonK` for records 1, 2, 3, and 4

We have to reset these values in order to permit another correct run for the $FMCHAN1 job. After completing this activity, we can repeat steps 4 to 9, already described in TestCase1.

When we have completed that, as shown in Figure 27-60, we can retype the command, `CHANGE 'phonx' 'PhinK' #7 ALL`

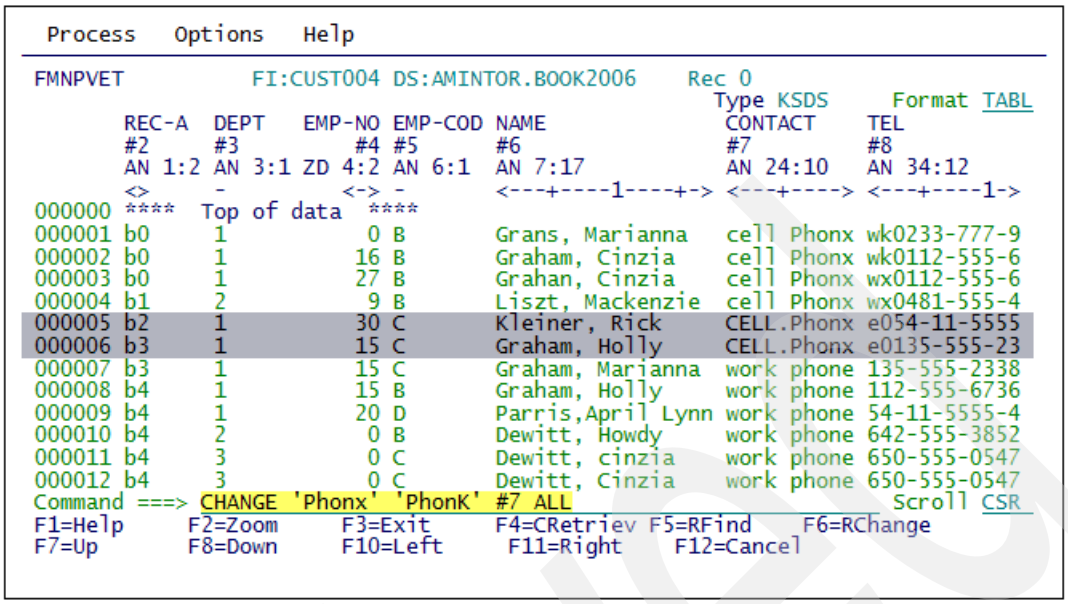


Figure 27-60 Perform the Change command together the run of $FMCHAN1

FM/CICS, with the value ‘ON’ that we have just set for the *Auto Retry for Change* option, performs the change immediately, as shown in Figure 27-61.

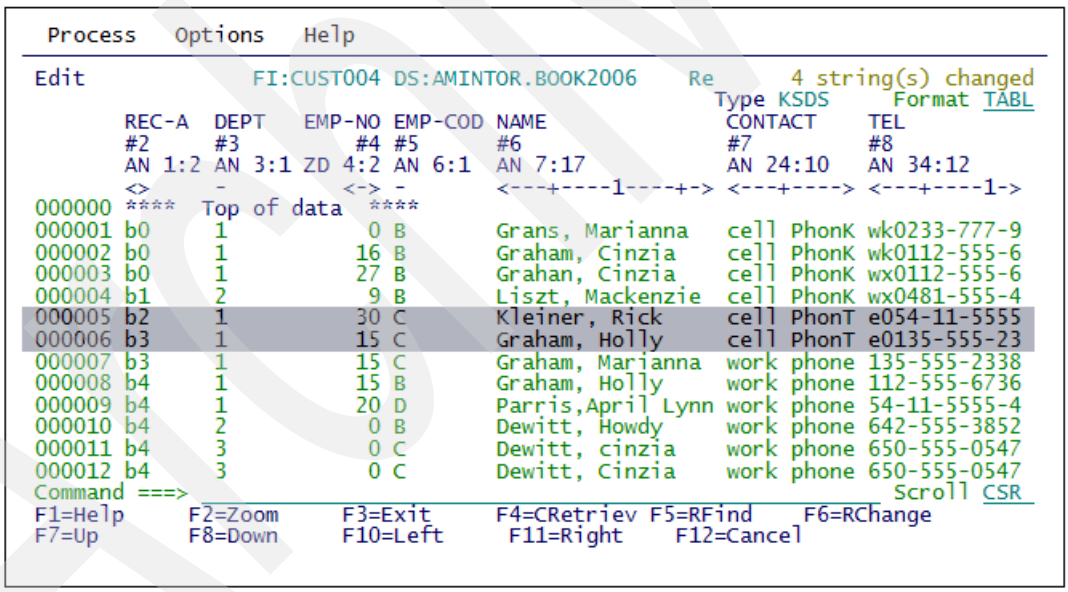


Figure 27-61 The lines 5 and 6 has been changed immediately

Figure 27-61 closes TestCase2.

Note: In TestCase2, we note the following two interesting results:

- When you execute the CHANGE ALL command, FM/CICS does not display the Record *Update from other User* panel, but performs the command immediately.
- The updates done from the other (batch) application have been correctly kept.

27.5.6 Summary of scenario

In this topic we described the various components that make up the FM/CICS components within the CICS environment on our system and how they are set up. We reviewed the CSD definitions and installation processing performed by CEDA.

We detailed a process whereby FM/CICS was used to identify the VSAM Sharing Options between online and batch applications. We described the “Integrity Warning” messages and the File Manager’s ability to determine whether any update, on the VSAM file occurrences, which you were editing, has been done using another application.

FM/CICS uses an internal buffer to compare and verify if the data has been changed from external events. The “original values” are compared with the “actual CUST004 values,” according to the VSAM Sharing Options, every time you perform CHANGE ALL, SAVE, CONTINUE, or press PF3 within your FM/CICS session.

We present this scenario to illustrate how these FM/CICS functions are designed to provide improved usability for those files that are in such a shared environment.

Now, with the objects that we have already created during TestCase 1 and 2, you can modify the parameters as you prefer, to verify these functions and enhance your understanding of FM/CICS.

28

Test data and File Manager

When an application is tested, the data is required. There could be several types of data required for testing, for example, sequential and VSAM data sets, data from DB2 and IMS databases, mostly existing on DASD. The data could contain dummy values generated according to templates, or it could be extracted from production data. Moreover, some relations between data from different sources might exist.

In this chapter we describe some capabilities of File Manager for z/OS and its DB2 and IMS components.

In this chapter some panels are intentionally omitted. Please refer to the corresponding product documentation.

28.1 Creating MVS data using File Manager and copybooks

Using File Manager online or in a batch mode, you can create data from scratch, using the File Manager data generator, which allows you to create data based on a record layout.

28.1.1 Data creation online using a PDS as model

To define a PDS that we use as a model, we start from the Main panel of File Manager as shown in Figure 28-1.

```
 Process   Options   Help
-------------------------------------------------------------------------------
FMNSTASK ger                 Primary Option Menu
Command ===> 3.4

0  Settings      Set processing options                 User ID . : AMINTOR
1  Browse        Browse data                            System ID : STLABF6
2  Edit          Edit data                              Appl ID . : FMN
3  Utilities     Perform utility functions              Version . : 7.1.0
4  Tapes         Tape specific functions                Terminal. : 3278A
5  Disk/VSAM     Disk track and VSAM CI functions       Screen. . : 2
6  OAM           Work with OAM objects                  Date. . . : 2006/11/16
7  Templates     Template and copybook utilities        Time. . . : 10:31
8  HFS           Access Hierarchical File System
X  Exit          Terminate File Manager
```

Figure 28-1 FM/Base (FMNSTASK) Primary Option Menu

Type **3.4** on the Command Line and press Enter to navigate directly on the *Catalog Services* panel. The following command is shown in Figure 28-2.

```
 Process   Options   Help
-------------------------------------------------------------------------------
FMNPSCS ager                  Catalog Services
Command ===>
                                                                More:      +
  blank List catalog entries                A Alter catalog entry
    DEF Define catalog entry              DEL Delete catalog entry
      I Display entry information           P Print catalog entries

Data Set:
   Data set name . 'amintor.book2006.**'
   Catalog ID  . .

Processing Options:
   Entry Type                                              Sort field
   2  1. Any            6. Cluster      11. Page space    1  1. Name
      2. Non-VSAM       7. Data         12. Path             2. Creation date
      3. VSAM           8. GDG          13. User catalog     3. Free space
      4. AIX            9. Index                             4. Allocated space
      5. Alias         10. OAM          Enter "/" to select option
                                        _  Batch execution     with list
                                        _  Include Additional Qualifiers
                                        _  YY/MM/DD date format (def. YYYY.DDD)
```

Figure 28-2 Catalog Services panel to filter non-VSAM entry

You can select your dataset's name and filter your selection using the Processing Options with an Entry Type of **2** to search your existing "non-VSAM" dataset. Press Enter to get the panel shown in Figure 28-3.

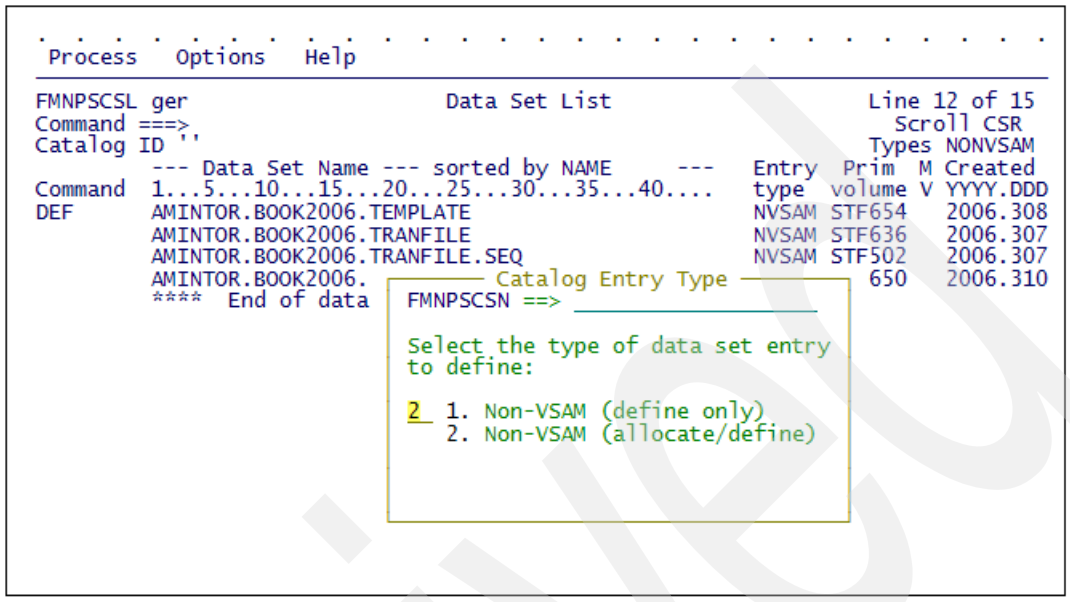


Figure 28-3 Define a PDS Dataset from the Data Set List

To generate your PDS dataset, type **def** in the Prefix Area to the corresponding dataset identifier position that you want use as an image. After pressing Enter, you receive a pop-up panel. Choose **2** to Allocate and Define the new PDS file and press Enter to show the panel in Figure 28-4.

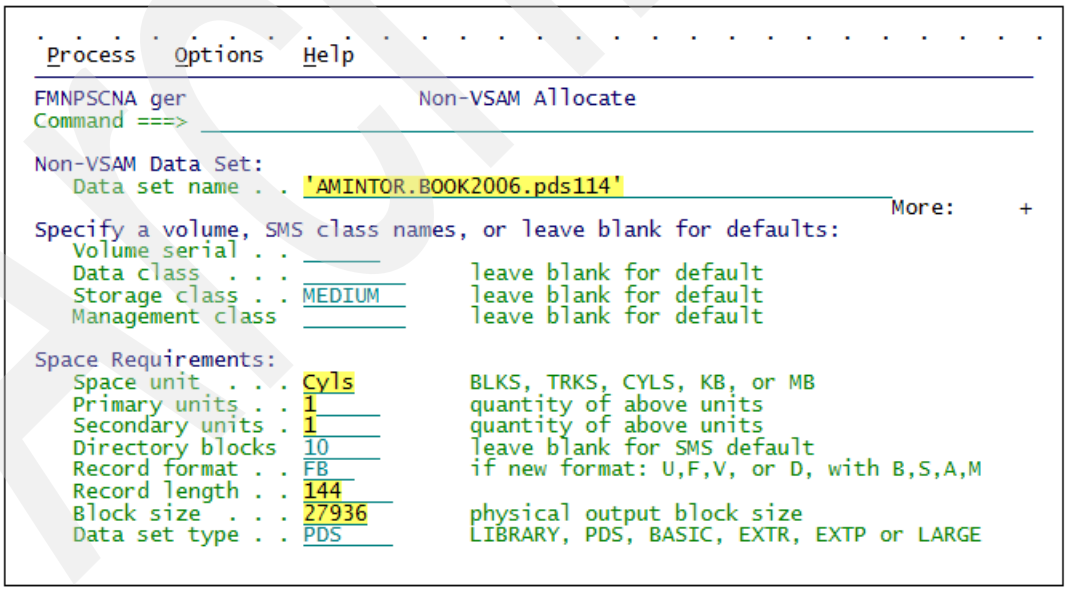


Figure 28-4 Next panel

Change the data according your definitions as shown. Press Enter to create the PDS and receive the confirmation as shown in Figure 28-5.

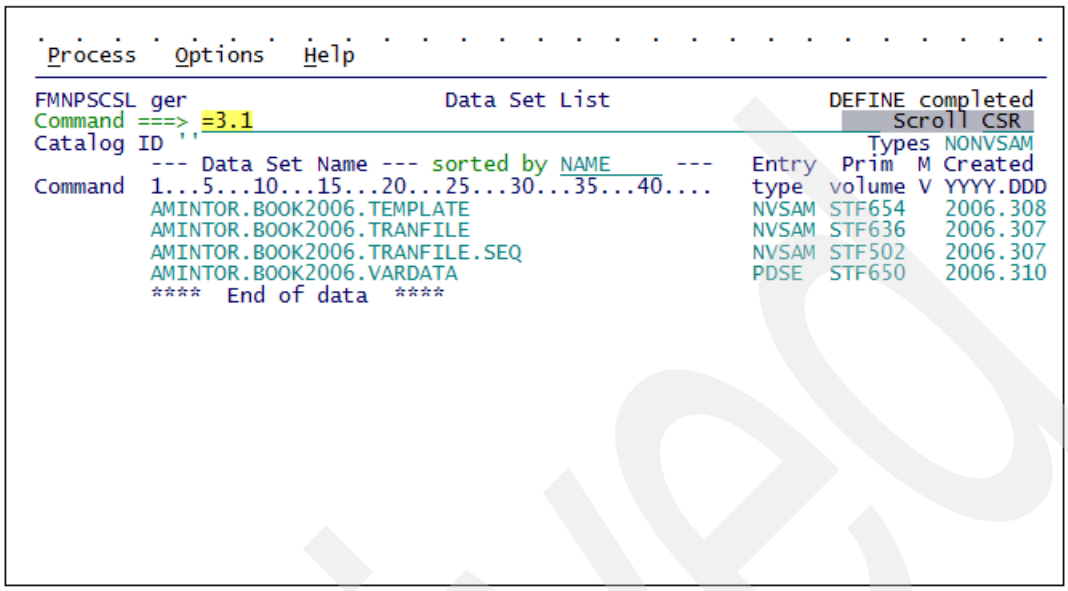


Figure 28-5 The panel confirms the new PDS creation

Type **=3.1** as shown above in the command line to navigate in the *Data Create Utility* panel to generate the data using a PDS as model. Press Enter to start the new dataset definition as shown in Figure 28-6.

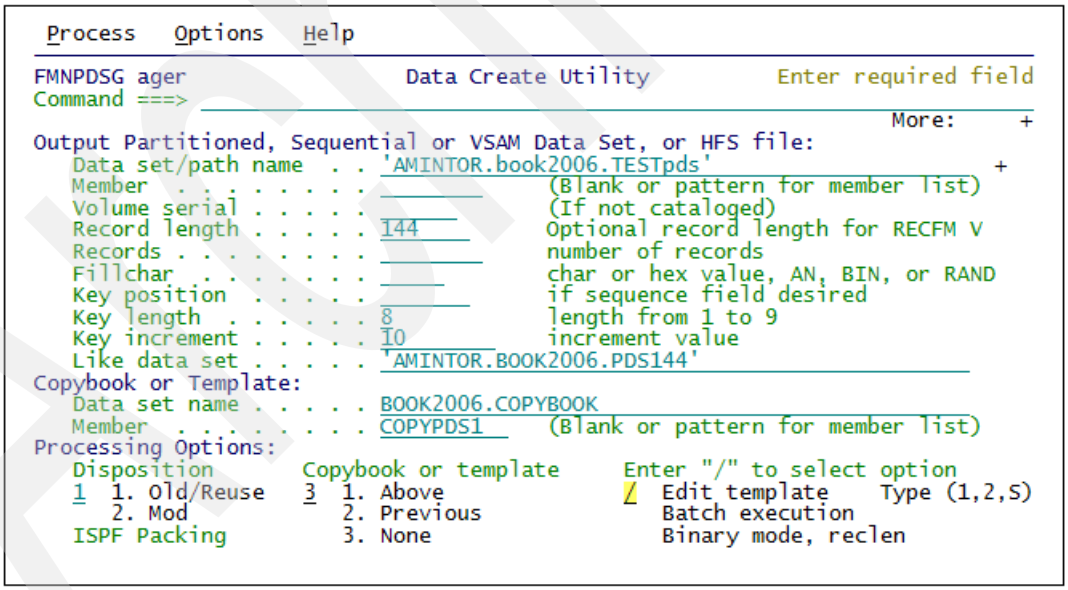


Figure 28-6 Specifying target data set and template for data generation

Fill in the required and optional fields (for example, the name of the target data set, the number of records, and the name of copybook or template), as shown above.

If the target data set does not exist, File Manager guides you through its creation. For a non-VSAM data set, you can specify a model, as shown in Figure 28-7.

```
 Process   Options   Help
FMNPALLN AMINTOR.BOOK2006.TESTPDS
Command ===>

New Data Set Organization:
   Select option          Instructions
   6 1. KSDS              The above data set does not exist.
     2. ESDS              To define or allocate a new data set select a data
     3. RRDS              set organization and press ENTER or press PF3/EXIT
     4. VRRDS             or PF12/CANCEL to return without allocation.
     5. LDS
     6. Non VSAM
                          For a new data set, enter a data set name
                          below to copy existing allocation attributes.


Existing Data Set:
   Like data set . . . . . 'AMINTOR.BOOK2006.PDS144'
   Volume serial . . . . . ______
```

Figure 28-7 Creating a new dataset using a model

File Manager can use the model for allocation attributes, as shown in Figure 28-8.

```
 Process   Options   Help
FMNPQALL AMINTOR.BOOK2006.TESTPDS
Command ===>

Specify a model data set, volume, SMS class names, or leave blank for defaults:
   Like data set . . 'AMINTOR.BOOK2006.PDS144'
   Volume serial . . ______
   Data class  . . . ________     leave blank for default
   Storage class . . MEDIUM__     leave blank for default
   Management class  ________     leave blank for default

Space Requirements:
   Space unit  . . . CYLS____     BLKS, TRKS, CYLS, KB, or MB
   Primary units . . 1_______      quantity of above units
   Secondary units . 1_______      quantity of above units
   Directory blocks  10______     leave blank for SMS default
   Record format . . FB______     if new format: U,F,V, or D, with B,S,A,M
   Record length . . 144_____
   Block size  . . . 27936___     physical output block size
   Data set type . . PDS_____     LIBRARY, PDS, BASIC, EXTR, EXTP or LARGE
   Release unused  . NO______     enter YES to free unused space, else NO
   Expiration date . ________     yyyy.ddd, blank for default
```

Figure 28-8 Allocation attributes for target data set

Since the record layout has more than one 01-Level inside the copybook COPYPDS1, FM gives you a panel to select the structures as shown in Figure 28-9.

```
 Process   Options   Help
FMNPRDS1 ger                 Record Type Selection                 Line 1 of 2
Command ===> ____________________________________________________  Scroll CSR

Processing Option: _  Template for segmented data
Cmd SIE Field Name                                           Prompt   Offset Length
        ****  Top of data  ****
e__  S  REC-T-A                                                           0    144
     S  REC-T-B                                                           0    120
        ****  End of data  ****
```

Figure 28-9   Segment Data selection Panel

File Manager presents you with an opportunity to select the segment data, and the template fields to be included in the target data set, as shown in Figure 28-10. The fields with default values are marked by the character **C** in the SHC column.

```
 Process   Options   Help
FMNPRDS2 ger                  Field Selection/Edit                Line 1 of 13
Command ===> ____________________________________________________  Scroll CSR

----------- Criteria - Enter 1 or 2 to specify expression by field -----------
1 Id : #2 = 'b'                                                             +
2 Sel:                                                                      +
Offset     0
Cmd Seq SHC Ref Field Name                              Picture  Type Start Length
                ****  Top of data  ****
              1  1 REC-T-A                                       AN      1    144
    ___ S     2  2 REC-A                                X        AN      1      1
    ___ S     3  2 DEPT                                 X(02)    AN      2      2
    ___ S     4  2 EMP-NO                               9(4)     BI      4      2
    ___ S     5  2 EMP-COD                              9(4)     BI      6      2
    ___ S     6  2 OF-NO                                X(3)     AN      8      3
    __1 S     7  2 NAME                                 X(17)    AN     11     17
    ___ S     8  2 JOB                                  X(11)    AN     28     11
    ___ S     9  2 ADDR                                 X(20)    AN     39     20
    ___      10  2 CITY                                 X(14)    AN     59     14
    ___      11  2 FILLER                               X(12)    AN     73     12
    ___ S C  12  2 CITY                                 X(4)     AN     85      4
    ___      13  2 FILLER                               X(56)    AN     89     56
```

Figure 28-10   Field Selection panel

You can edit the field definitions to specify default values for data generation. When done, File Manager displays before the panel above, to confirm the changes to your template TMPFILE4 generated from the COPYPDS1 copybook. Press Enter and PF3 to save the template. After pressing PF3, you receive the confirmation message `Template saved`, as shown in Figure 28-11.

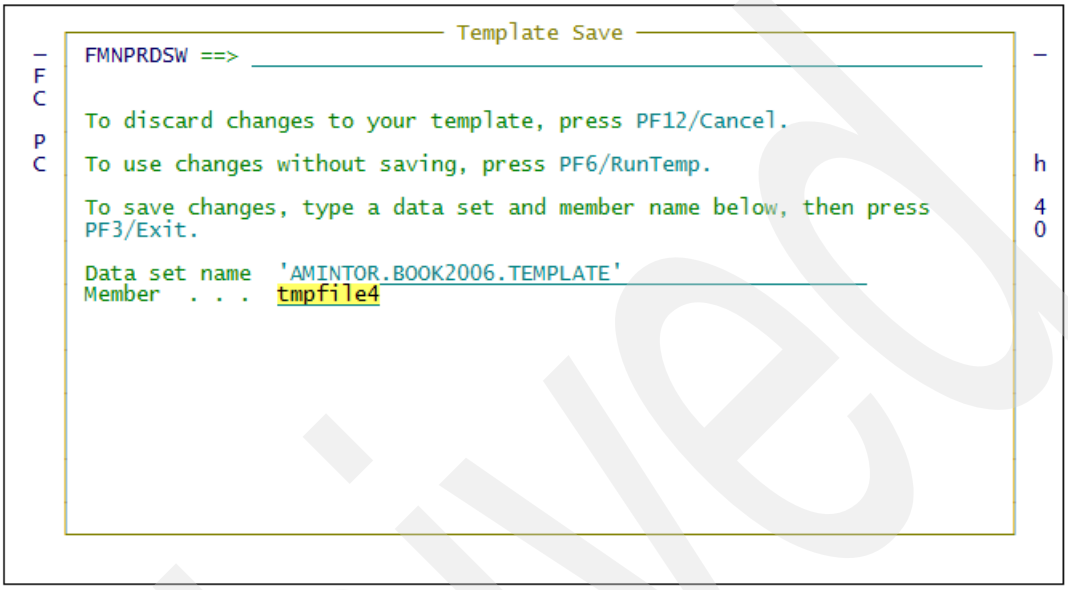


Figure 28-11 Confirm the Template Save panel when you change the copybook definitions

Confirmation is shown in Figure 28-12.

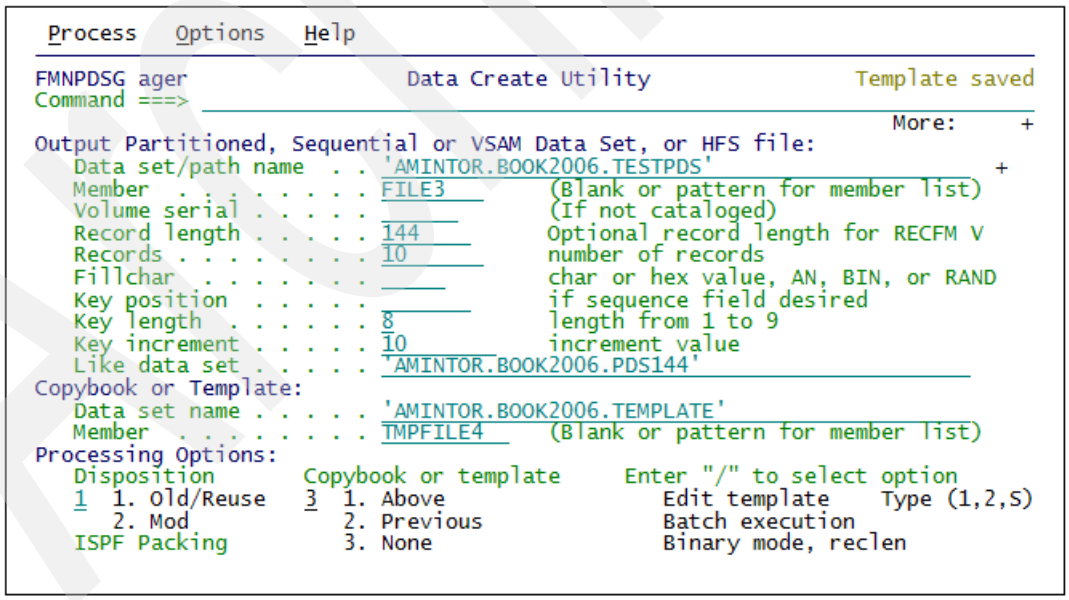


Figure 28-12 Confirmation message with the new template in the copybook Template field

When you press Enter or PF3 on this panel, you receive the creation result as shown in Figure 28-13.

```
 Process   Options   Help
FMNPDSG ager                Data Create Utility          10 record(s) written
Command ===> _____________________________________________________________
                                                                More:     +
Output Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . 'AMINTOR.BOOK2006.TESTPDS'                    +
   Member  . . . . . . . . FILE3___     (Blank or pattern for member list)
   Volume serial . . . . . ________     (If not cataloged)
   Record length . . . . . 144_____     Optional record length for RECFM V
   Records . . . . . . . . 10______     number of records
   Fillchar  . . . . . . . _____        char or hex value, AN, BIN, or RAND
   Key position  . . . . . _____        if sequence field desired
   Key length  . . . . . . 8           length from 1 to 9
   Key increment . . . . . 10______     increment value
   Like data set . . . . . 'AMINTOR.BOOK2006.PDS144'
Copybook or Template:
   Data set name . . . . . 'AMINTOR.BOOK2006.TEMPLATE'
   Member  . . . . . . . . TMPFILE4     (Blank or pattern for member list)
Processing Options:
   Disposition       Copybook or template     Enter "/" to select option
   1  1. Old/Reuse   3  1. Above                Edit template    Type (1,2,S)
      2. Mod            2. Previous             Batch execution
   ISPF Packing         3. None                 Binary mode, reclen
```

Figure 28-13 Data creation online results

The results are presented in Figure 28-14.

```
 Process   Options   Help
Edit            AMINTOR.BOOK2006.TESTPDS(FILE3  +            Rec 0 of 10
Command ===> _____________________________________________________ Scroll CSR
                                                                  Format TABL
       REC-A  DEPT    EMP-NO EMP-COD OF-NO  NAME              JOB          ADDR
       #2     #3          #4      #5 #6     #7                #8           #9  +
       AN 1:1 AN 2:2 BI 4:2  BI 6:2 AN 8:3 AN 11:17           AN 28:11     AN 39
       -      <>     <---+> <---+> <->    <---+----1----+-> <---+----1> <---+
000000 ****  Top of data  ****
000001 0       1       16448   16448
000002 0       1       16458   16448
000003 0       1       16468   16448
000004 0       1       16478   16448
000005 0       1       16488   16448
000006 0       1       16498   16448
000007 0       1       16508   16448
000008 0       1       16518   16448
000009 0       1       16528   16448
000010 0       1       16538   16448
000011 ****  End of data  ****
```

Figure 28-14 Generated data

You have to update the generated records with pertinent values, unless the generated values are acceptable.

28.1.2 Data creation in a batch mode

A sample procedure to generate data in a batch mode is presented in Example. This job creates five records in the output data set according to the field definitions in the specified File Manager template (TCOUT parameter). The results of the batch generation job can be seen together the SDSF output, as shown in Example 28-1.

Example 28-1 Generating data in batch mode

```
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (A),
000002 //             AMINTOR,MSGCLASS=A,
000003 //             NOTIFY=AMINTOR,CLASS=A,
000004 //             MSGLEVEL=(1,1)
000005 //*
000006 //FILEMGR  EXEC PGM=FMNMAIN
000007 //*TEPLIB  DD DSN=FMN.SFMNMOD1,DISP=SHR
000008 //*        DD DSN=IGY.SIGYCOMP,DISP=SHR
000009 //*FMNCOB  DD DUMMY     Uncomment to force use of FM COBOL Compiler
000010 //SYSPRINT DD SYSOUT=*
000011 //FMNTSPRT DD SYSOUT=*
000012 //SYSTERM  DD SYSOUT=*
000013 //SYSIN    DD *
000014 $$FILEM DSG DSNOUT=AMINTOR.BOOK2006.TESTPDS(FILE10),
000015 $$FILEM LIKE='AMINTOR.BOOK2006.PDS144',
000016 $$FILEM DISP=OLD,
000017 $$FILEM TCOUT=AMINTOR.BOOK2006.TEMPLATE(TMPFILE4),
000018 $$FILEM PACK=NONE,
000019 $$FILEM NLRECS=10
****** ***************************** Bottom of Data ****************************

 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY AMINTORB JOB04495  DSID   102 LINE 5       COLS 02- 81
 COMMAND INPUT ===>                                            SCROLL ===> CSR

PRINTOUT=SYSPRINT  HEADERPG=YES  ASCII=BOTH     RECLIMIT=(1,*)
PRINTLEN=132       PAGESKIP=NO   PAD=OFF        EOD=/*
PAGESIZE=60        DATAHDR=YES   DBCSPRT=OFF    LANGUAGE=ENGLISH
PRTTRANS=ON        DUMP=UPDOWN   TAPELBL=SL     CYLHD=ABSOLUTE
SMFNO=000          PRTDISP=MOD   USEIOX=DISABLE IOX=
TEMPHLQ=
ABENDCC=NONE       WIDEPRT=YES

IBM File Manager for z/OS
$$FILEM DSG DSNOUT=AMINTOR.BOOK2006.TESTPDS(FILE10),
$$FILEM LIKE='AMINTOR.BOOK2006.PDS144',
$$FILEM DISP=OLD,
$$FILEM TCOUT=AMINTOR.BOOK2006.TEMPLATE(TMPFILE4),
$$FILEM PACK=NONE,
$$FILEM NLRECS=10
IBM File Manager for z/OS
Template contains multiple record types marked for use: only first type used
10 record(s) written
******************************** BOTTOM OF DATA ********************************
```

28.1.3 Field attributes for template-based data creation

The opportunity to define default values to be used in generated data is available to the user when File Manager presents the Field Attributes panel, shown in Figure 28-15.

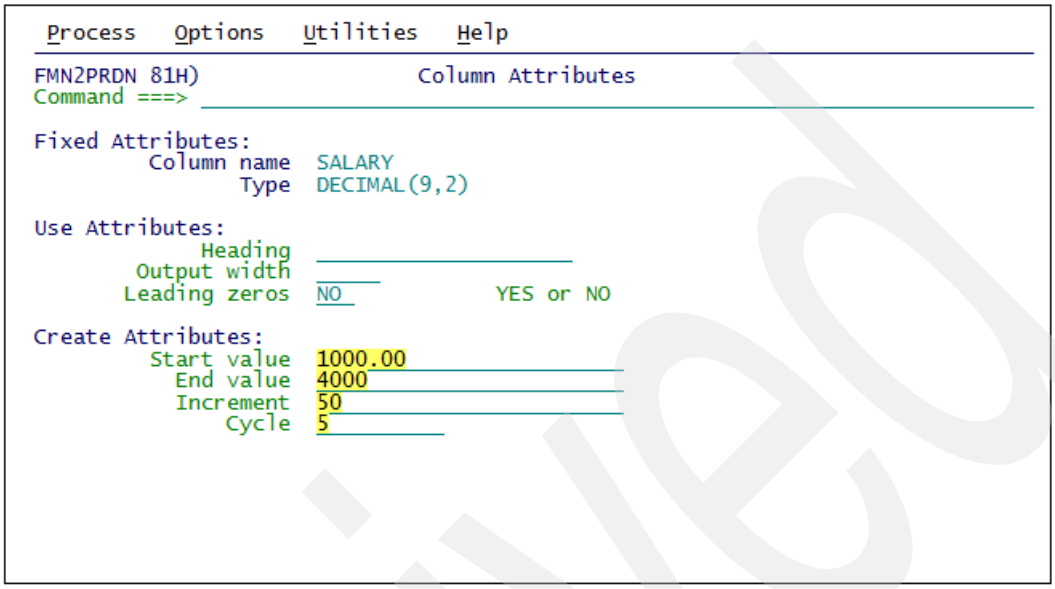


Figure 28-15 Attributes for numeric fields

Numeric values

For numeric fields, the creation attributes are the start and end values, the increment, and the cycle.

Table 28-1 contains samples illustrating how the values of these attributes affect the resulting values.

Table 28-1 Attributes for numeric fields

Start	End	Increment	Cycle	Generated values
1,000		10		1000, 1010, 1020, 1030, and so on
1,000	1,020	5		1000, 1005, 1010, 1015, 1020, 1020, 1020, and so on
1,000		10	3	1000, 1000, 1000, 1010, 1010, and so on

Alphanumeric values

For alphanumeric fields, there is a different set of creation attributes (for example, the filler, the action, the start character, the pattern, and the pattern repetition indicator), as shown in Figure 28-16.

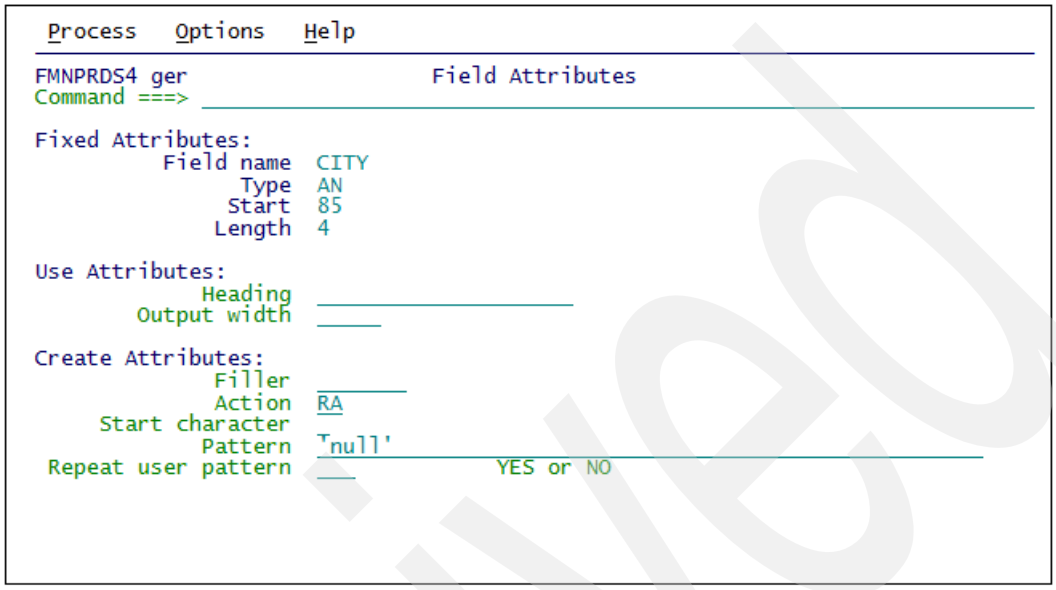


Figure 28-16 Attributes for alphanumeric fields

Table 28-2 contains samples illustrating how the values of these attributes affect the resulting values.

Table 28-2 Attributes for alphanumeric fields

Filler	Action	Pattern	Generated values
	FX	'01'	'01' in all records
?			'??????' in all records, filling the field
	FX	AN	'ABCDEF' for all records
	RP	AN	Rippling effect ABCDEF BCDEFG CDEFGH DEFGHI EFGHIJ

The list of available values for the action field is as follows:

FX The contents of the field are to remain fixed.

RA The contents of the field are to be generated from characters selected at random from the specified pattern. A different set of characters is selected for each record.

RO The contents of the field are to be rolled. The pattern you specify is shifted left one byte for each record until the first non-blank character of the pattern is in the first byte of the field. Then the pattern is shifted right one byte for each output record until it returns to its original position in the field. RO can only be used with a user-supplied pattern, not with an IBM-supplied pattern. The pattern must start with a blank, or the result is the same as FX. The roll only occurs within the length of the pattern.

RP The contents of the field are to be rippled. The pattern you specify is shifted left one byte for each record and the truncated character is appended to the right-hand end of the pattern.

SL The contents of the field are to be shifted left for each record. The pattern you specify is shifted left one character and filled to the right with a space. When all characters in the pattern have been shifted out, the original pattern is restored and the process is repeated.

SR The contents of the field are to be shifted right for each record. The pattern you specify is shifted right one character and filled to the left with a space. When all characters in the pattern have been shifted out, the original pattern is restored and the process is repeated.

TL The contents of the field are to be truncated on the left for each record. The pattern you specify is left truncated (the leftmost character replaced with a space) one character at a time until all characters in the pattern have been truncated. Then the original pattern is restored and the process is repeated.

TR The contents of the field are to be truncated on the right for each record. The pattern you specify is right truncated (the rightmost character replaced with a space) one character at a time until all characters in the pattern have been truncated. Then the original pattern is restored and the process is repeated.

WV The contents of the field are to be waved. The pattern you specify is shifted left one byte for each record until the first non-blank character of the pattern is in the first byte of the field. Then the original pattern is restored and the process repeated. WV can only be used with a user-supplied pattern, not with an IBM-supplied pattern. The pattern must start with a blank, or the result is the same as FX. The roll only occurs within the length of the pattern.

The list of IBM-supplied patterns is as follows:

AL Alphabetic: Characters A–Z, inclusive
AN Alphanumeric: Characters A–Z, 0–9 inclusive
CO Collating sequence: All printable characters in the range X'40'–X'F9'

28.2 Copying MVS data using File Manager

When copying data using File Manager, you can:

- Select records and fields using Boolean expressions.
- Map fields from the input data set to the output data set.
- Reformat and generate data while copying.

28.2.1 Data copy online

Select option **3** (Copy data) from the File Manager Utility Functions panel and then fill in the fields about the source data set and the template or copybook to be used, as shown in Figure 28-17.

```
 Process   Options   Help
------------------------------------------------------------------------------
File Manager                      Copy Utility
Command ===> ____________________________________________________________

From Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . book2006.psb144                                  +
   Member  . . . . . . . . fileaaa4       (Blank or pattern for member list)
   Volume serial . . . . . ______
   Start key . . . . . . . ________________________ key or slot
   Skip count  . . . . . . ________     number of records to be skipped
   Copy count  . . . . . . ALL          number of records to be copied
From Copybook or Template:
   Data set name . . . . . BOOK2006.TEMPLATE
   Member  . . . . . . . . tmpfile4     (Blank or pattern for member)
Processing Options:
   Copybook/template    Enter "/" to select option
   1  1. Above          Batch execution          Advanced member selection
      2. None           Use proc ________        Skip member name list
      3. Create dynamic /  Ignore length mismatch  REXX member selection: P
   _  Edit template     JCL Source format        Directory integrity
   _  Type (1,2,S)      Export mode              Report PDS record counts
                                                 _ Binary mode, reclen
```

Figure 28-17 Specifying source data set and template for copying

Then do the same thing for the target data set, as shown in Figure 28-18.

```
 Process   Options   Help
Copy from AMINTOR.BOOK2006.PDS144(FILEAAA4)
Command ===> _____________________________________________________________
                                                                More:     +
To Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . BOOK2006.DATA.SEQ                            +
   Member name (or mask) . ________
   Volume serial . . . . . ______

To Copybook/Template From:  AMINTOR.BOOK2006.TEMPLATE(TMPFILE4)
   Data set name . . . . . BOOK2006.TEMPLATE
   Member  . . . . . . . . TEMPDATA     (Blank or pattern for member)

Processing Options:
  Copybook/template usage   Disposition          Enter "/" to select option
  1  1. Above               1  1. Old or Reuse      Replace members
     2. None                   2. Mod               /  Edit template mapping
     3. Create dynamic                              /  Edit template source
  ISPF Packing                                      _  Binary mode, reclen
  1  1. Asis                                        _  Stats Off
     2. Pack
     3. Unpack
     4. None
```

Figure 28-18 Specifying target data set and template for copying

After pressing Enter, FM displays the Record Type Mapping if the copybook structure consists of more than one 01-Level. Since in this example we have a 01- Level only, FM shows the next *Field Mapping* panel.

Edit the templates if there is a requirement to specify selection criteria. Edit the mapping if there are differences between source and target formats. Type **e** or **s** in the Prefix line to create your *Selected Fields* list and press Enter. The panel in Figure 28-19 is displayed.

```
 Process   Options   Help
FMNPRDS6 ger                    Field Mapping                  Line 1 of 28
Command ===> _____________________________________________________ Scroll CSR

To          AMINTOR.BOOK2006.TEMPLATE(TEMPDATA)
From        AMINTOR.BOOK2006.TEMPLATE(TMPFILE4)

Cmd Lv To Field Name        Type  Len   Lv From Field Name             Type  Len
        ****  Top of data  ****
      1 REC-DATA             AN   133 |  1 REC-T-A                     AN   144
      2 REC-TYPE             AN     1 |  2 REC-A                       AN     1
      2 DEPART-1             AN     2 |  2 DEPT                        AN     2
e     2 PGR-BIN              BI     2 |
      2 COD-EMP              AN     3 |  2 EMP-COD                     BI     2
      2 NOMINATIVO           AN    26 |  2 NAME                        AN    17
      3 NAME                 AN    16 |  2 FILLER                      AN    12
      3 SURNAME              AN    10 |  2 FILLER                      AN    12
e     2 NOMIN-R              AN    26 |
e     2 CAPABIL              AN     5 |
s     2 CAPABIL-R            PD     5 |
      2 EMP-NO               AN     3 |  2 EMP-NO                      BI     2
s     2 BILL                 PD     5 |
s     2 STARTDATE            AN    10 |
```

Figure 28-19 Field Mapping panel

You can specify a relationship between source and target fields, as shown in Figure 28-20 (SEX and FILLER). Press PF3 to receive again the *From Field Mapping* (FMNPRD7) panel according to the next *Selected fields* order, defined before inside the *Field Mapping* (FMNPRDS7) panel.

```
 Process   Options   Help
FMNPRDS7 ger                  From Field Mapping
Command ===> _____________________________________________ Scroll CSR
To   template/copybook : AMINTOR.BOOK2006.TEMPLATE(TEMPDATA)
From template/copybook : AMINTOR.BOOK2006.TEMPLATE(TMPFILE4)
To field . . . . . : #20 SEX
From field . . . . . :
Corresponding(Y/N) . . : N              (Auto map for group items).

Sel   Ref Lvl From Field                                  Typ Start Length
        1  1  REC-T-A                                     AN      1    144
        2  2  REC-A                                       AN      1      1
        3  2  DEPT                                        AN      2      2
        4  2  EMP-NO                                      BI      4      2
        5  2  EMP-COD                                     BI      6      2
        6  2  OF-NO                                       AN      8      3
        7  2  NAME                                        AN     11     17
        8  2  JOB                                         AN     28     11
        9  2  ADDR                                        AN     39     20
       10  2  CITY                                        AN     59     14
s      11  2  FILLER                                      AN     73     12
       12  2  CITY                                        AN     85      4
       13  2  FILLER                                      AN     89     56
```

Figure 28-20 Source and target field mapping

You can change the correspondence between fields and edit new fields to specify default values for data creation. Creation attributes for a target field can be specified, as shown in Figure 28-21.

```
 Process   Options   Utilities   Help
FMN2PRDN 81H)                Column Attributes
Command ===> ____________________________________________________

Fixed Attributes:
        Column name  SALARY
              Type  DECIMAL(9,2)

Use Attributes:
           Heading  ____________________
      Output width  ____
     Leading zeros  NO          YES or NO

Create Attributes:
       Start value  1000.00_____________
         End value  4000.00_____________
         Increment  50________________
             Cycle  5_______
```

Figure 28-21 Creation attributes specified for target field

Press PF3 as often as you complete a relation between source and target fields. FM displays the confirmation message when all the correspondence has been finished, as shown in Figure 28-22.

```
 Process   Options   Help
FMNPDSCI ger                    Copy Utility
Command ===> ____________________________________________________________
51 record(s) copied: 51 truncated: 0 fields truncated
From Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name  . . BOOK2006.PDS144                                +
   Member  . . . . . . . . FILEAAA4       (Blank or pattern for member list)
   Volume serial . . . . . ______
   Start key . . . . . . . ____________________________ key or slot
   Skip count  . . . . . . 0_______       number of records to be skipped
   Copy count  . . . . . . ALL_____       number of records to be copied
From Copybook or Template:
   Data set name . . . . . BOOK2006.TEMPLATE
   Member  . . . . . . . . TMPFILE4__     (Blank or pattern for member)
Processing Options:
   Copybook/template    Enter "/" to select option
   1  1. Above             Batch execution          Advanced member selection
      2. None              Use proc *_______        Skip member name list
      3. Create dynamic /  Ignore length mismatch   REXX member selection: P
   _  Edit template        JCL Source format        Directory integrity
   _  Type (1,2,S)         Export mode              Report PDS record counts
                                                  _ Binary mode, reclen
```

Figure 28-22 File Manager confirmation message

28.3 Creating DB2 test data using File Manager

You can create DB2 data using the File Manager DB2 Component, which is referred to as File Manager/DB2.

28.3.1 DB2 data creation online

There is a specific option (8) for data creation on the File Manager/DB2 Utility Functions panel shown in Figure 28-23.

```
 Process   Options   Utilities   Help
FMN2STUT 81H)                Utility Functions
Command ===> 8

1  Print          Print DB2 table or view
2  Objects        Create and drop DB2 objects
3  Copy           Copy data within DB2
4  Object List    Display and process DB2 object lists
5  Privileges     Manage DB2 privileges
6  Import         Import sequential or VSAM data into DB2
7  Export         Export DB2 data to sequential or VSAM data set
8  Create         Create DB2 test data
9  Utilities      DB2 utility job generation
10 Audit trail    Print audit trail report
11 Print browse   Browse FM/DB2 print data set
```

Figure 28-23 Utility Functions panel of File Manager/DB2

The target DB2 object must exist prior to data generation. There is no option to create it presented at this point. On the next panel, shown in Figure 28-24, specify the target DB2 object.

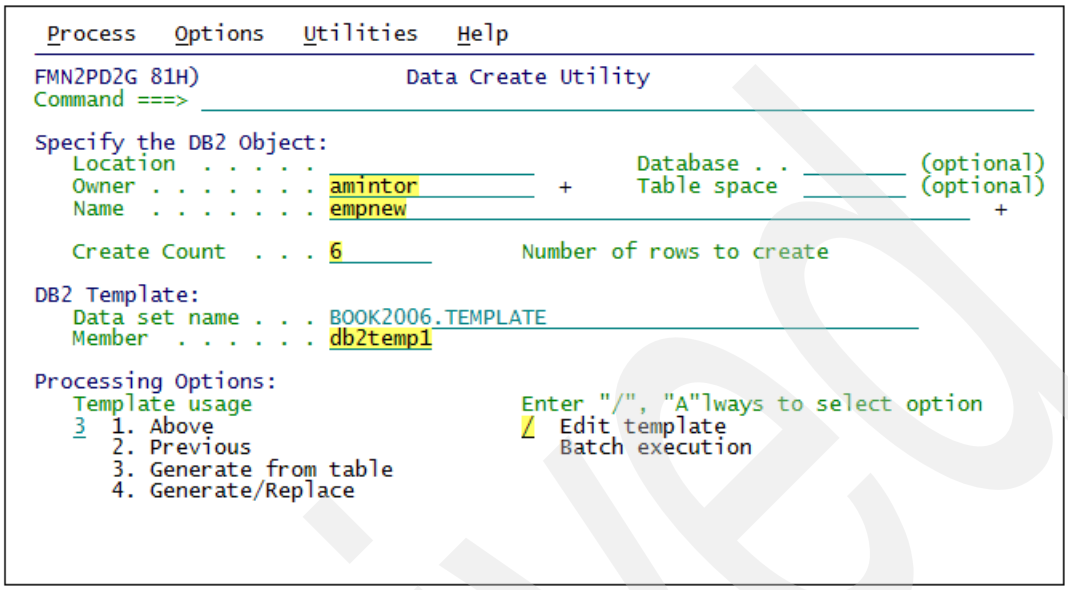


Figure 28-24 Data Create Utility panel of File Manager/DB2

Template usage

A template data set is never required to copy a DB2 object. You can select from four options:

1. To copy an object using a previously created and stored template, specify **1** in the template usage field. A template data set is mandatory for this option.
2. To use the last template associated with the DB2 object, specify **2** in the template usage field.
3. Option **3** is the default value. The template is automatically built during the copy function. A template data set name is not required for this option, and is ignored if specified.
4. Option **4** allows you to generate a new template for the DB2 object, and save this template in the data set specified in the template section of the panel. A template data set is mandatory for this option. If the template already exists, its contents are over-written.

You can edit the corresponding template, as shown in Figure 28-25.

```
 Process   Options   Utilities   Help
FMN2PRD2 81H)              Column Selection/Edit                 Line 1 of 14
Command ===>                                                     Scroll PAGE
TABLE AMINTOR.EMPNEW
------- Row Selection Criteria ---- (Use SQL/PF4 for full screen edit) -------
1 Sel:                                                                      +

Cmd Seq SHC CL# Column name          Data type(length)  Null Default  Order A/D
                ****  Top of data  ****
     ___ S     1 EMPNO               CHARACTER(6)            None
     ___ S     2 FIRSTNME            VARCHAR(12)             None
     ___ S     3 MIDINIT             CHARACTER(1)            None
     ___ S     4 LASTNAME            VARCHAR(15)             None
e    ___ S     5 WORKDEPT            CHARACTER(3)       Y    Null
     ___ S     6 PHONENO             CHARACTER(4)       Y    Null
e    ___ S     7 HIREDATE            DATE               Y    Null
     ___ S     8 JOB                 CHARACTER(8)       Y    Null
     ___ S     9 EDLEVEL             SMALLINT           Y    Null
     ___ S    10 SEX                 CHARACTER(1)       Y    Null
     ___ S    11 BIRTHDATE           DATE               Y    Null
e    ___ S    12 SALARY              DECIMAL(9,2)       Y    Null
     ___ S    13 BONUS               DECIMAL(9,2)       Y    Null
     ___ S    14 COMM                DECIMAL(9,2)       Y    Null
```

Figure 28-25 Column Selection/edit panel

You can specify creation criteria for the fields. Column attributes for a character field are shown in Figure 28-26.

```
 Process   Options   Utilities   Help
FMN2PRDC 81H)                Column Attributes
Command ===>

Fixed Attributes:
         Column name  WORKDEPT
               Type  CHARACTER(3)

Use Attributes:
            Heading  ____________________
       Output width  _____

Create Attributes:
             Filler  _______
             Action  __
    Start character
            Pattern  'A00'
 Repeat user pattern ___          YES or NO
```

Figure 28-26 Column attributes for character field

Column attributes for a numeric field are shown in Figure 28-27.

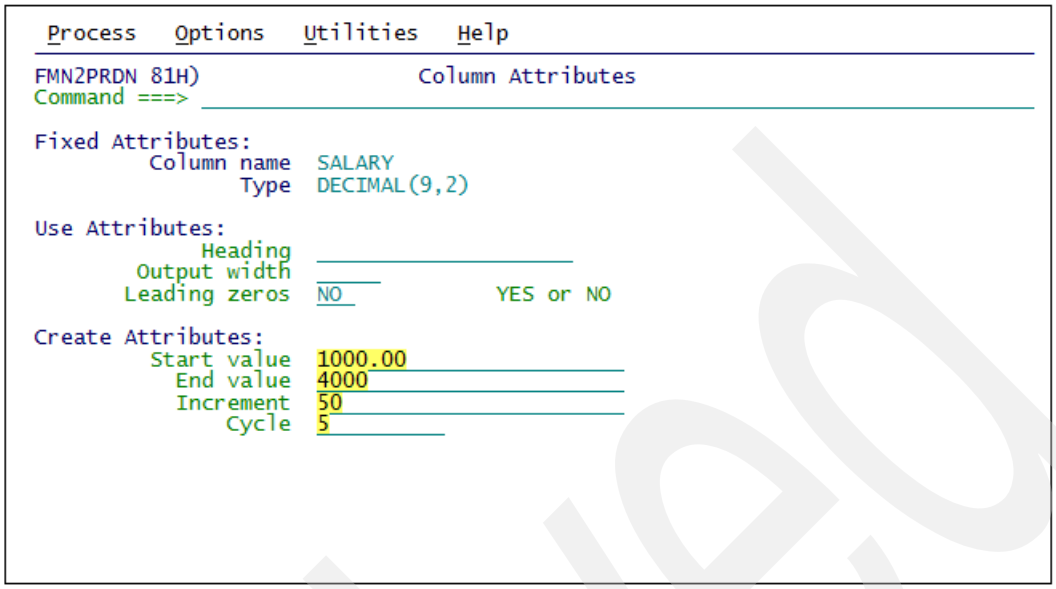


Figure 28-27 Column attributes for numeric field

Column attributes for a date field that is specific for the File Manager/DB2 are shown in Figure 28-28.

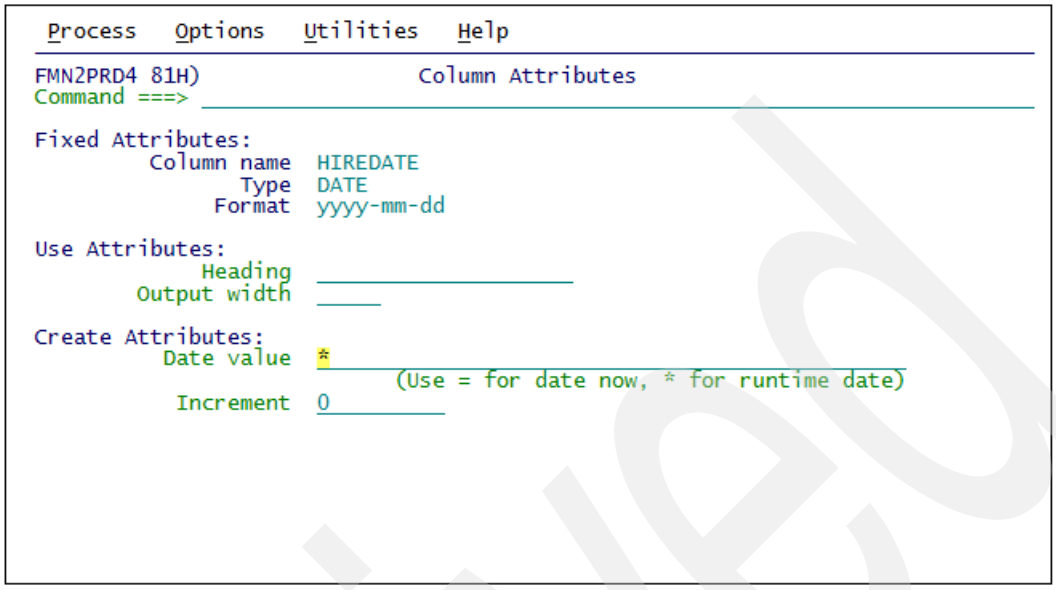


Figure 28-28 Column attributes for date field

Note: The date field is specific to the DB2 component. It does not exist in the MVS component.

The fragment of the result table is shown in Figure 28-29. Note that no default values were specified.

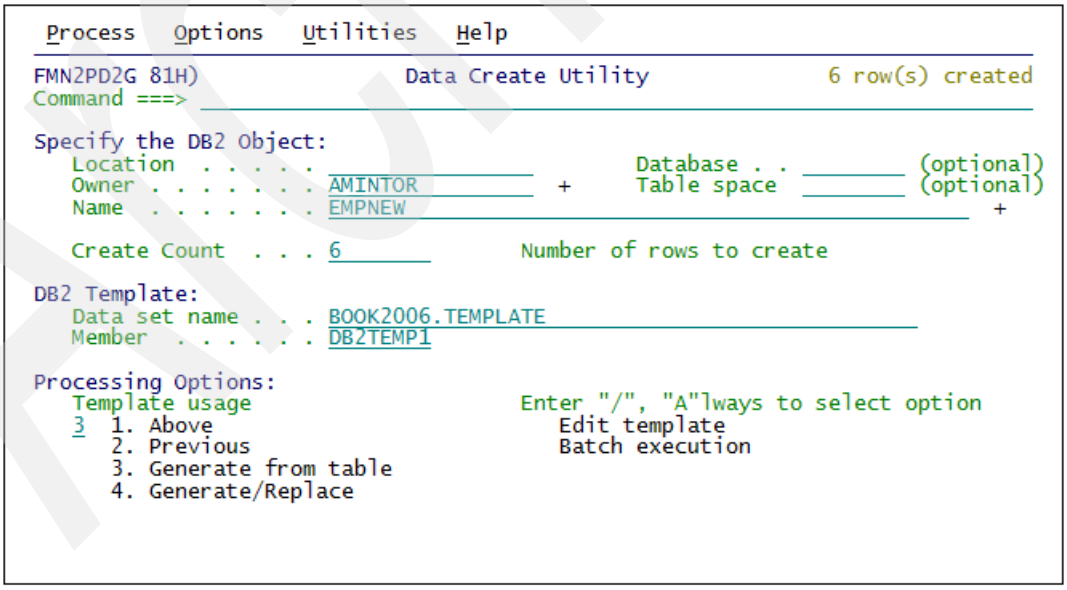


Figure 28-29 File Manager confirmation message: 6 new rows have been generated

28.3.2 DB2 data creation in a batch mode

DB2 data can also be created using File Manager/DB2 in a batch mode. The sample procedure is presented in Example 28-2. This job fills the table EMPNEW with six records according to the columns definitions from the specified template.

Example 28-2 Sample procedure to generate DB2 data

```
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (A),
000002 //             AMINTOR,MSGCLASS=A,
000003 //             NOTIFY=AMINTOR,CLASS=A,
000004 //             MSGLEVEL=(1,1)
000005 //*
000006 //FMNDB2   EXEC PGM=FMNDB2,PARM=('SSID=D81H,SQID=AMINTOR')
000007 //STEPLIB  DD DSN=ADTOOLS.FM710.SFMNMOD1,DISP=SHR
000008 //         DD DSN=SYS1.D8OH.SDSNEXIT,DISP=SHR
000009 //         DD DSN=SYS1.D81H.SDSNLOAD,DISP=SHR
000010 //SYSPRINT DD SYSOUT=*
000011 //FMNTSPRT DD SYSOUT=*
000012 //SYSTERM  DD SYSOUT=*
000013 //SYSIN    DD *
000014 $$FILEM D2G OBJOUT="AMINTOR"."EMPNEW",
000015 $$FILEM TMOUT=AMINTOR.BOOK2006.TEMPLATE(DB2TEMP1),
000016 $$FILEM ROWS=6
****** **************************** Bottom of Data ***************************
```

28.4 Copying DB2 data using File Manager

When copying DB data, you can make a one-for-one copy:

- Select rows.
- Map columns in a source DB2 table to columns in a target DB2 table.
- Reformat and generate data while copying.

28.4.1 DB2 data copy online

Select option **3** (Copy data within DB2) from the File Manager/DB2 Utility Functions panel. Fill in the fields about the source object and the template to be used, as shown in Figure 28-30.

```
 Process   Options   Utilities   Help
FMN2STUT 81H)              Utility Functions
Command ===> 3

1  Print         Print DB2 table or view
2  Objects       Create and drop DB2 objects
3  Copy          Copy data within DB2
4  Object List   Display and process DB2 object lists
5  Privileges    Manage DB2 privileges
6  Import        Import sequential or VSAM data into DB2
7  Export        Export DB2 data to sequential or VSAM data set
8  Create        Create DB2 test data
9  Utilities     DB2 utility job generation
10 Audit trail   Print audit trail report
11 Print browse  Browse FM/DB2 print data set
```

Figure 28-30 Utility Functions panel.

Type the source DB2 table, the Copy count (number of rows), and the template that you wish to copy as shown Figure 28-31.

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)                    Copy Utility
Command ===>

From DB2 Object:
   Location  . . . . .                        Database . .          (optional)
   Owner . . . . . . . AMINTOR          +     Table space           (optional)
   Name  . . . . . . . EMPNEW                                                 +

   Copy count  . . . . 6             Number of rows to copy

From Template:
   Data set name . . . BOOK2006.TEMPLATE
   Member  . . . . . . DB2TEMP1

Processing Options:
   Template usage                    Enter "/", "A"lways to select option
   2  1. Above                       /  Edit template
      2. Previous                       Copy "From" values to "To" panel
      3. Generate from table
      4. Generate/Replace
```

Figure 28-31 Source DB2 object and template

Do the same thing for the target object, as shown in Figure 28-32.

```
 Process   Options   Utilities   Help
FMN2PRD2 81H)              Column Selection/Edit                Line 1 of 14
Command ===> _____________________________________________________ Scroll PAGE
TABLE AMINTOR.EMPNEW
------- Row Selection Criteria ---- (Use SQL/PF4 for full screen edit) -------
1 Sel: _____________________________________________________________________ +

Cmd Seq SHC CL# Column name          Data type(length)  Null Default  Order A/D
              ****  Top of data  ****
___ ___ S     1 EMPNO                CHARACTER(6)            None           _
___ ___3S     2 FIRSTNME             VARCHAR(12)             None           _
___ ___4S     3 MIDINIT              CHARACTER(1)            None           _
___ ___2S     4 LASTNAME             VARCHAR(15)             None           _
e__ ___ S     5 WORKDEPT             CHARACTER(3)       Y    Null           _
___ ___ S     6 PHONENO              CHARACTER(4)       Y    Null           _
___ ___ S     7 HIREDATE             DATE               Y    Null           _
___ ___ S     8 JOB                  CHARACTER(8)       Y    Null           _
___ ___ S     9 EDLEVEL              SMALLINT           Y    Null           _
___ ___ S    10 SEX                  CHARACTER(1)       Y    Null           _
___ ___ S    11 BIRTHDATE            DATE               Y    Null           _
e__ ___ S    12 SALARY               DECIMAL(9,2)       Y    Null           _
___ ___ S    13 BONUS                DECIMAL(9,2)       Y    Null           _
___ ___ S    14 COMM                 DECIMAL(9,2)       Y    Null           _
```

Figure 28-32 Target DB2 object and template

If required, you can edit the templates to specify selection criteria and edit the template mapping if there are differences between source and target formats, as shown in Figure 28-33.

```
 Process   Options   Help
FMNPRDS6 81H)              Template Mapping                     Line 1 of 14
Command ===> _____________________________________________________ Scroll PAGE

To          AMINTOR.EMPNEW
From        AMINTOR.EMPNEW

Cmd CL# To Column name  Datatype  NI   CL# From Column name       Datatype  NI
      ****  Top of data  ****
      1 EMPNO           CH(6)     NN |  1 EMPNO                  CH(6)     NN
      2 FIRSTNME        VC(12)    NN |  2 FIRSTNME               VC(12)    NN
e__   3 MIDINIT         CH(1)     NN |  3 MIDINIT               CH(1)     NN
      4 LASTNAME        VC(15)    NN |  4 LASTNAME               VC(15)    NN
e__   5 WORKDEPT        CH(3)        |  5 WORKDEPT               CH(3)
      6 PHONENO         CH(4)        |  6 PHONENO                CH(4)
      7 HIREDATE        DATE         |  7 HIREDATE               DATE
      8 JOB             CH(8)        |  8 JOB                    CH(8)
      9 EDLEVEL         SMINT        |  9 EDLEVEL                SMINT
     10 SEX             CH(1)        | 10 SEX                    CH(1)
     11 BIRTHDATE       DATE         | 11 BIRTHDATE              DATE
     12 SALARY          DEC(9,2)     | 12 SALARY                 DEC(9,2)
     13 BONUS           DEC(9,2)     | 13 BONUS                  DEC(9,2)
     14 COMM            DEC(9,2)     | 14 COMM                   DEC(9,2)
```

Figure 28-33 Template Mapping panel of File Manager/DB2

You can also change the correspondence between fields and edit new fields to specify default values for data creation.

A sample of specific relationships between fields is shown in Figure 28-34.

```
 Process   Options   Utilities   Help
FMN2PRD7 81H)                From Column Mapping
Command ===>                                                  Scroll PAGE
To DB2 object . . : AMINTOR.EMPNEW
From DB2 object . : AMINTOR.EMPNEW
To column . . . . : #3 MIDINIT
From column . . . : #3 MIDINIT

Sel   Cl#  From Column              Data type(length)
           ****  Top of data  ****
___      D  Delete "From column"
___      1  EMPNO                   CHARACTER(6)
         2  FIRSTNME                VARCHAR(12)
         3  MIDINIT                 CHARACTER(1)
         4  LASTNAME                VARCHAR(15)
         5  WORKDEPT                CHARACTER(3)
         6  PHONENO                 CHARACTER(4)
         7  HIREDATE                DATE
         8  JOB                     CHARACTER(8)
         9  EDLEVEL                 SMALLINT
        10  SEX                     CHARACTER(1)
        11  BIRTHDATE               DATE
        12  SALARY                  DECIMAL(9,2)
```

Figure 28-34 File Manager/DB2 Column Mapping panel

The File Manager/DB2 Column Attributes panel where creation attributes can be defined is shown in Figure 28-35.

```
 Process   Options   Utilities   Help
FMN2PRDC 81H)                Column Attributes
Command ===>

Fixed Attributes:
         Column name  WORKDEPT
                Type  CHARACTER(3)

Use Attributes:
             Heading  ____________________
        Output width  _____

Create Attributes:
              Filler  _______
              Action  __
     Start character
             Pattern  'A00'
 Repeat user pattern  ___          YES or NO
```

Figure 28-35 File Manager/DB2 Column Attributes panel

File Manager/DB2 displays the confirmation message when data is copied, as shown in Figure 28-36.

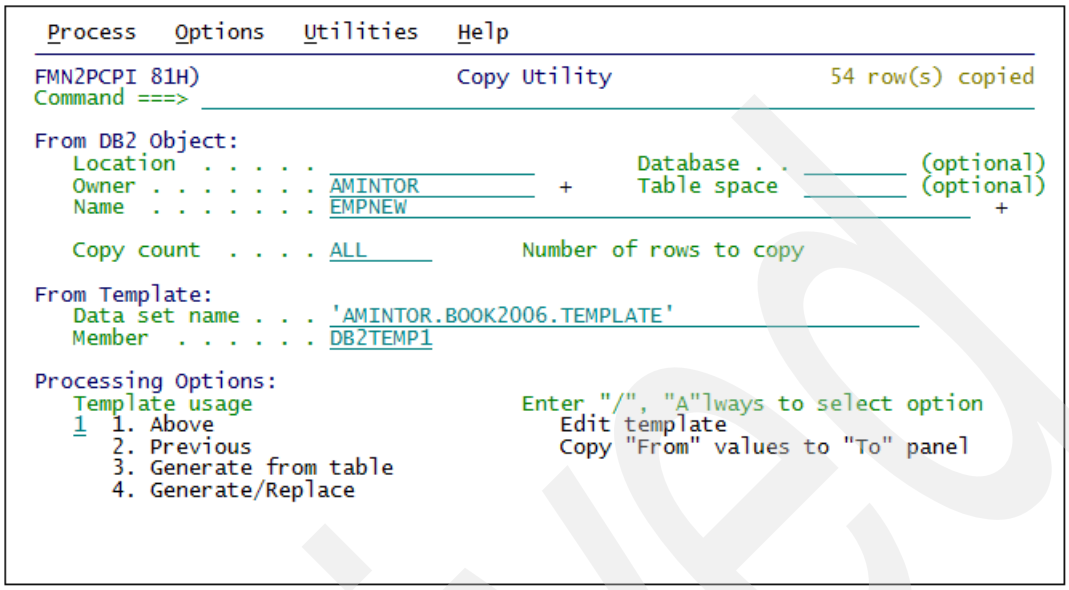


Figure 28-36 File Manager/DB2 confirmation message

This process of copying DB2 data using File Manager/DB2 online is very similar to the process of copying data online using File Manager.

28.4.2 DB2 data copy in a batch mode

DB2 data can also be copied to another DB2 object using File Manager/DB2 in a batch mode. The sample procedure is presented in Example 28-3. This job copies all records from one table to the other according to the columns definitions in the catalog with automatic mapping by field name.

Example 28-3 Sample procedure to copy DB2 data

```
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (A),
000002 //            AMINTOR,MSGCLASS=A,
000003 //            NOTIFY=AMINTOR,CLASS=A,
000004 //            MSGLEVEL=(1,1)
000005 //*
000006 //FMNDB2   EXEC PGM=FMNDB2,PARM=('SSID=D81H,SQID=AMINTOR')
000007 //STEPLIB  DD DSN=ADTOOLS.FM710.SFMNMOD1,DISP=SHR
000008 //         DD DSN=SYS1.D8OH.SDSNEXIT,DISP=SHR
000009 //         DD DSN=SYS1.D81H.SDSNLOAD,DISP=SHR
000010 //SYSPRINT DD SYSOUT=*
000011 //FMNTSPRT DD SYSOUT=*
000012 //SYSTERM  DD SYSOUT=*
000013 //SYSIN    DD *
000014 $$FILEM DBC OBJIN="AMINTOR"."EMPNEW",
000015 $$FILEM     TMIN=AMINTOR.BOOK2006.TEMPLATE(DB2TEMP1),
000016 $$FILEM     OBJOUT="AMINTOR"."EMPNEW2",
000017 $$FILEM     LOCK=NONE,
000018 $$FILEM     DUPKEY=IGNORE,
000019 $$FILEM     DUPMAX=ALL,
000020 $$FILEM     REFI=FAIL,
000021 $$FILEM     AUDIT=YES,
```

```
000022 $$FILEM     ROWS=ALL
****** ***************************** Bottom of Data ***************************
```

28.5 Exporting and importing DB2 data using File Manager

When exporting and importing DB2 data, you can:

- Select rows from a table.
- Export to a data set in:
  - The default File Manager/DB2 format
  - A format defined by a copybook
  - A comma-delimited format
- Import from a file:
  - Generated by the export function
  - Described by a copybook

28.5.1 DB2 data export online

To export data from DB2 to a sequential or VSAM data set, select option **7** on the Utility Functions panel of File Manager/DB2, as shown in Figure 28-37.

```
 Process   Options   Utilities   Help
FMN2STUT 81H)                Utility Functions
Command ===> 7

1  Print          Print DB2 table or view
2  Objects        Create and drop DB2 objects
3  Copy           Copy data within DB2
4  Object List    Display and process DB2 object lists
5  Privileges     Manage DB2 privileges
6  Import         Import sequential or VSAM data into DB2
7  Export         Export DB2 data to sequential or VSAM data set
8  Create         Create DB2 test data
9  Utilities      DB2 utility job generation
10 Audit trail    Print audit trail report
11 Print browse   Browse FM/DB2 print data set
```

Figure 28-37 Option 7: Export on File Manager/DB2 Utility Functions panel

Fill in the fields about the source object and the template to be used, as shown in Figure 28-38.

```
 Process   Options   Utilities   Help
FMN2PEXI 81H)                  Export Utility
Command ===>

From DB2 Object:
   Location  . . . . .                          Database . .          (optional)
   Owner . . . . . . . AMINTOR          +       Table space           (optional)
   Name  . . . . . . . empnew                                                 +

   Export Count  . . . ALL          Number of rows to export

From Template:
   Data set name . . . BOOK2006.TEMPLATE
   Member  . . . . . . db2temp1

Processing Options:
   Template usage                        Enter "/", "A"lways to select option
   3  1. Above                              Edit options
      2. Previous                           Edit template
      3. Generate from table
      4. Generate/Replace
```

Figure 28-38 Source DB2 object and template on File Manager/DB2 Export Utility panel

Do the same thing for the target object, as shown in Figure 28-39.

```
 Process   Options   Utilities   Help
FMN2PEXO om AMINTOR.EMPNEW
Command ===>

To Partitioned, Sequential or VSAM Data Set:
   Data set name . . . book2006.data
   Member  . . . . . . empnew
   Volume  . . . . . .

To Copybook or Template:
   Data set name . . . book2006.template
   Member  . . . . . . copypds1

Processing Options:
   Template usage                        Enter "/", "A"lways to select option
   3  1. Above                              Edit options
      2. Previous                           Edit template mapping
      3. Generate from input
      4. Generate and save
      5. None. (CSV output)
```

Figure 28-39 Target MVS object and template definition for data export

If required, you can edit the templates to specify selection criteria and edit the template mapping if there are differences between source and target formats, as shown in Figure 28-40.

```
 Process    Options    Help
FMNPRDS6 81H)                  Template Mapping                  Line 1 of 14
Command ===> _____________________________________________________ Scroll PAGE

To          AMINTOR.BOOK2006.DATA.EXP
From        AMINTOR.EMPNEW

Cmd Lv To Field Name       Type  Len   CL# From Column name       Datatype  NI
      ****  Top of data  ****
      2 EMPNO               AN     2 |   1 EMPNO                 CH(6)     NN
      2 PHONENO             AN     4 |   6 PHONENO               CH(4)
      2 EDLEVEL             BI     2 |   9 EDLEVEL               SMINT
      2 FIRSTNME            AN    12 |   2 FIRSTNME              VC(12)    NN
      2 MIDINIT             AN     1 |   3 MIDINIT               CH(1)     NN
      2 EDLEVE1             BI     2 |   9 EDLEVEL               SMINT
      2 LASTNAME            AN    17 |   4 LASTNAME              VC(15)    NN
      2 WORKDEPT            AN     3 |   5 WORKDEPT              CH(3)
e     2 ADDR2               AN    10 |
e     2 ADDR3               AN    10 |
      2 JOB                 AN    14 |   8 JOB                   CH(8)
      2 SEX                 AN     1 |  10 SEX                   CH(1)
      2 POSTCODE            AN     4 |  14 COMM                  DEC(9,2)
      2 FILLER              AN    62 |  14 COMM                  DEC(9,2)
```

Figure 28-40 Template mapping when exporting DB2 data

You can change the correspondence between fields and edit new fields to specify default values for data creation.

A specific relationship between fields is defined using the File Manager/DB2 Column mapping panel, as shown in Figure 28-41.

```
 Process    Options    Utilities    Help
FMN2PRU7 81H)                 From Column Mapping
Command ===>                                                  Scroll PAGE
To template/copybook  : AMINTOR.BOOK2006.DATA.EXP
From DB2 object . . . : AMINTOR.EMPNEW
To field  . . . . . . : #8 LASTNAME
From column . . . . . : #4 LASTNAME

Sel   Cl#  From Column            Data type(length)
           ****  Top of data  ****
        D  Delete "From column"
___     1  EMPNO                 CHARACTER(6)
___     2  FIRSTNME              VARCHAR(12)
        3  MIDINIT               CHARACTER(1)
s__     4  LASTNAME              VARCHAR(15)
        5  WORKDEPT              CHARACTER(3)
        6  PHONENO               CHARACTER(4)
        7  HIREDATE              DATE
        8  JOB                   CHARACTER(8)
        9  EDLEVEL               SMALLINT
       10  SEX                   CHARACTER(1)
       11  BIRTHDATE             DATE
       12  SALARY                DECIMAL(9,2)
```

Figure 28-41 Column Mapping panel

Creation attributes for columns can be specified on the Column Attributes panel, as shown in Figure 28-42.

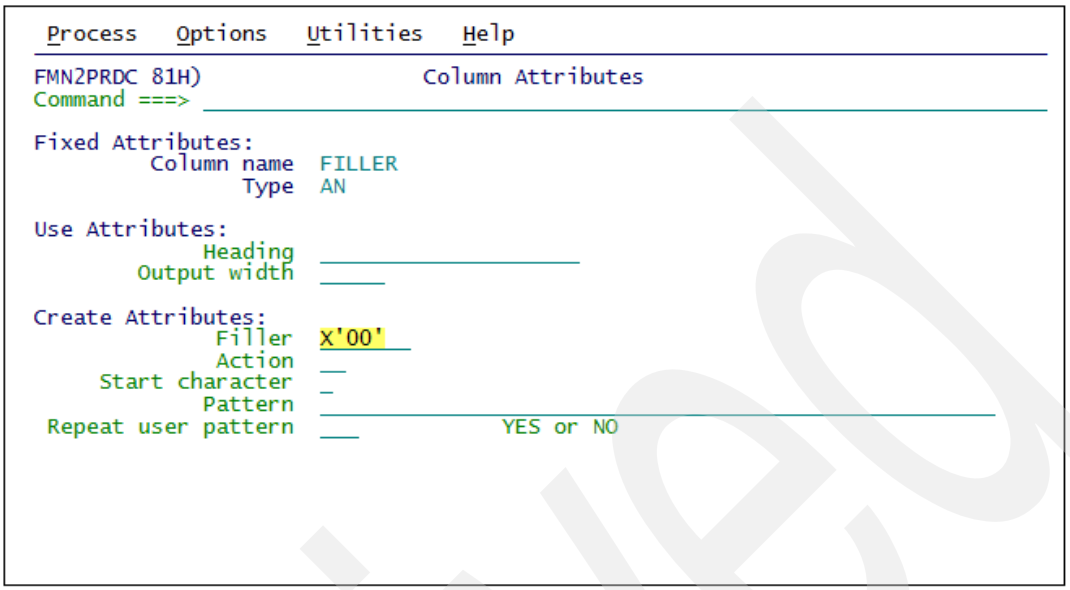


Figure 28-42 Data creation attributes for an alphanumeric column

When data is exported, File Manager displays the panel to confirm the template definition and shows the confirmation message, as shown in Figure 28-43.

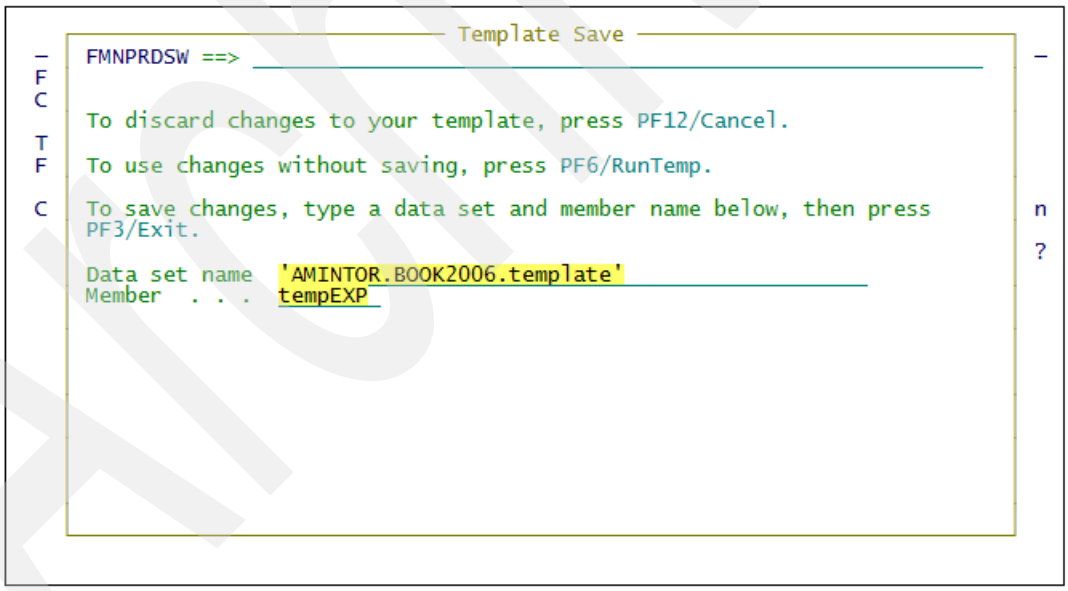


Figure 28-43 FM displays the panel to confirm the new template

The data exported from DB2 to a sequential has been 54 rows, as can be seen in the message at the top of Figure 28-44.

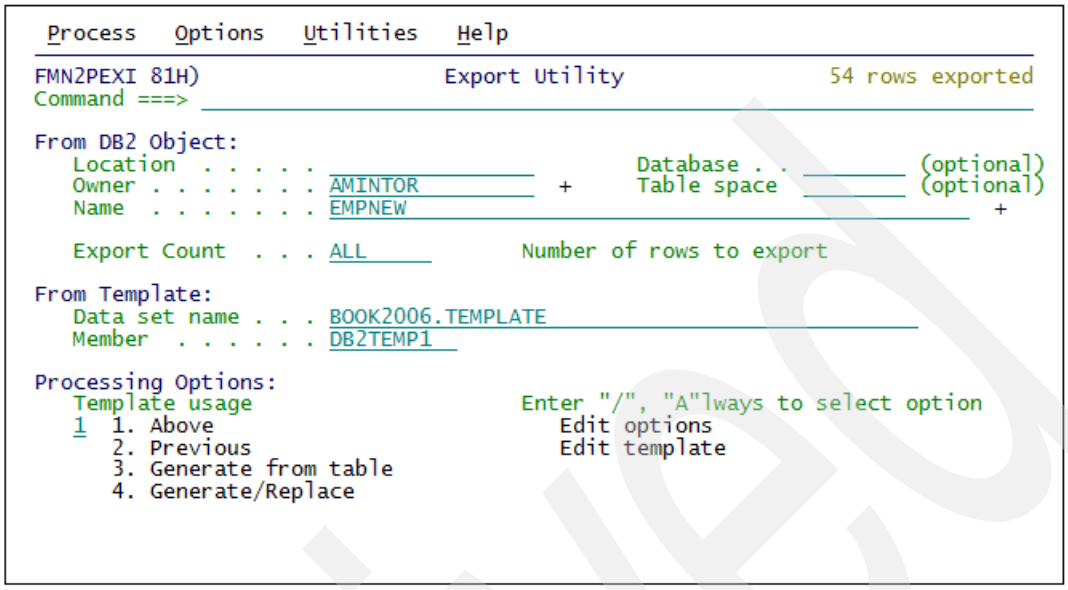


Figure 28-44 DB2 data is exported successfully

The fragment of the source DB2 table is shown in Figure 28-45.

```
 Process    Options    Help
------------------------------------------------------------------------------
Browse           AMINTOR.BOOK2006.DATA.EXP                      Rec 0 of 378
Command ===> ____________________________________________________ Scroll CSR
                                                 Record 0        Format TABL
EMPNO  EDLEVEL FIRSTNME     MIDINIT EDLEVE1 LASTNAME       EDLEVE2 WORKDEPT
#2          #3 #4           #5           #6 #7                 #8 #9
AN 1:6  BI 7:2 AN 9:12      AN 21:1 BI 22:2 AN 24:15       BI 39:2 AN 41:3
<---+>  <---+> <---+----1-> -        <---+> <---+----1----> <---+> <->
****  Top of data  ****
000010       9 CHRISTINE    I             4 HAAS                 0 A00
000020       7 MICHAEL      L             8 THOMPSON             0 B01
000030       5 SALLY        A             4 KWAN                 0 C01
000050       4 JOHN         B             5 GEYER                0 E01
000060       6 IRVING       F             5 STERN                0 D11
000070       3 EVA          D             7 PULASKI              0 D21
000090       6 EILEEN       W             9 HENDERSON            0 E11
000100       8 THEODORE     Q             7 SPENSER              0 E21
000110       8 VINCENZO     G             8 LUCCHESI             0 A00
000120       4 SEAN                       9 O'CONNELL            0 A00
000130       7 DOLORES      M             8 QUINTANA             0 C01
000140       7 HEATHER      A             8 NICHOLLS             0 C01
000150       5 BRUCE                      7 ADAMSON              0 D11
000160       9 ELIZABETH    R             6 PIANKA               0 D11
```

Figure 28-45 Fragment of source DB2 table

The fragments of the resulting sequential data set are shown in Figure 28-46.

```
 Process    Options    Help
------------------------------------------------------------------------------
Browse           AMINTOR.BOOK2006.DATA.EXP                      Rec 0 of 378
Command ===> ____________________________________________________ Scroll CSR
                                                 Record 0        Format TABL
PHONENO POSTCODE PROGRES RTARTDATE     CONT JOB          CONT       BONUS SEX
    #10 #11          #12 #13            #14 #15           #16         #17 #18
BI 44:2 AN 46:4  BI 50:2 AN 52:10   BI 62:2 AN 64:8  BI 72:2     BI 74:4 AN 78
 <---+> <-->     <---+> <---+----> <---+> <---+--> <---+> <---+----1> -
****  Top of data  ****
      0 3978           0 1965-01-01       0 PRES           0     1179648 F
      0 3476           0 1973-10-10       0 MANAGER        0     1179648 M
      0 4738           0 1975-04-05       0 MANAGER        0     1310720 F
      0 6789           0 1949-08-17       0 MANAGER        0     1048576 M
      0 6423           0 1972-08-14       0 MANAGER        0     1048576 M
      0 7831           0 1980-09-30       0 MANAGER        0     1048576 F
      0 5498           0 1970-08-15       0 MANAGER        0     1048576 F
      0 0972           0 1980-06-19       0 MANAGER        0      917504 M
      0 3490           0 1958-05-16       0 SALESREP       0     1245184 M
      0 2167           0 1963-12-05       0 CLERK          0      917504 M
      0 4578           0 1971-07-28       0 ANALYST        0     1048576 F
      0 1793           0 1976-12-15       0 ANALYST        0     1179648 F
      0 4510           0 1972-02-12       0 DESIGNER       0     1048576 M
      0 3782           0 1977-10-11       0 DESIGNER       0     1114112 F
```

Figure 28-46 Fragment of exported DB2 data, part 1

Press PF11 to scroll right. The field COMM with default values X'00" is added, as shown in Figure 28-47.

```
 Process   Options   Help
Browse          AMINTOR.BOOK2006.DATA.EXP                Rec 0 of 378
Command ===> ____________________________________________ Scroll CSR
                                          Record 0       Format TABL
SEX        COMM ENDATA
#18         #19 #20
AN 78:1 BI 79:2 AN 81:10
-       <---+> <---+---->
****  Top of data  ****
F             0 1933-08-14
M             0 1948-02-02
F             0 1941-05-11
M             0 1925-09-15
M             0 1945-07-07
F             0 1953-05-26
F             0 1941-05-15
M             0 1956-12-18
M             0 1929-11-05
M             0 1942-10-18
F             0 1925-09-15
F             0 1946-01-19
M             0 1947-05-17
F             0 1955-04-12
```

Figure 28-47 Fragment of exported DB2 data, part 2

28.5.2 DB2 data export in batch mode

The corresponding DB2 data export procedure using File Manager/DB2 is presented in Example 28-4. This job exports data from table EMP to an output PDS member according to an MVS template.

You can run the export utility in foreground, or as a batch job. To run the export utility in batch mode, we have to set the execution options. The latter option (batch) is recommended when exporting large volumes of data.

- Online execution:
  - Select option **1** to run this export in foreground.
- Batch execution:
  - Select option **2** to run this export as a batch job.
  - FM/DB2 generates JCL for the export job, and display an edit session so that you can review the JCL prior to submitting the job.
- Batch, using DB2 utility:
  - If the subsystem to which you are connected is running DB2 V7 or higher, the panel shows an additional selection of the DB2 UNLOAD utility. Select option **3** to run this export using the DB2 UNLOAD utility.
  - FM/DB2 generates JCL for the UNLOAD job, and displays an edit session so that you can review the JCL prior to submitting the job.

Select with **/** the *Option Edit* field to run your Export Utility in Batch mode. Figure 28-48 shows the setting attributes within the Source (From DB2 Object) and Target (To Dataset) panel.

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)                  Export Utility
Command ===> ____________________________________________________________

From DB2 Object:
   Location  . . . . . ____________          Database . . ________ (optional)
   Owner . . . . . . . AMINTOR      +        Table space  ________ (optional)
   Name  . . . . . . . EMPNEW                                      +

   Export Count  . . . ALL       Number of rows to export

From Template:
   Data set name . . . 'AMINTOR.BOOK2006.TEMPLATE'
   Member  . . . . . . DB2TEMP1

Processing Options:
   Template usage                    Enter "/", "A"lways to select option
   3  1. Above                       /  Edit options
      2. Previous                       Edit template
      3. Generate from table
      4. Generate/Replace
```

Figure 28-48 Set Edit Options to execute the Export Utility via batch

Select the Edit Options to choose the *Execution Options* as shown in Figure 28-49.

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)            Export Options - (1 of 3)
Command ===> ____________________________________________________________

Execution Options:
   Select option
   2  1. Online
      2. Batch
      3. Batch, using DB2 utility

Export Data Set:
   Enter "/" to select option
   /  Batch data set creation

More Options:
   Enter "/" to select option
   /  Export data set
   /  Data type format
   /  Edit UNLOAD options
```

Figure 28-49 Execution Options in batch

After pressing PF3, FM/DB2 shows the next panel. Do the same thing for the target object, and type / on the *Edit Options field*, as shown in Figure 28-50. Select the *Output Dataset*.

```
 Process   Options   Utilities   Help
Export from AMINTOR.EMPNEW
Command ===> ____________________________________________________________

To Partitioned, Sequential or VSAM Data Set:
   Data set name . . . BOOK2006.DATA.EXP_____________________________
   Member  . . . . . . ________
   Volume  . . . . . . ______

To Copybook or Template:
   Data set name . . . 'AMINTOR.BOOK2006.TEMPLATE'___________________
   Member  . . . . . . TEMPEXP2__

Processing Options:
   Template usage                      Enter "/", "A"lways to select option
   3  1. Above                         /  Edit options
      2. Previous                         Edit template mapping
      3. Generate from input
      4. Generate and save
      5. None. (CSV output)
```

Figure 28-50 Output database selection

Set the field *Edit Options* to add and complete your settings as shown in Figure 28-51.

```
 Process   Options   Utilities   Help
FM/DB2 (D81H)            Export Options - (2 of 3)
Command ===> ____________________________________________________________

Export Data Set:
   Data format                         Disposition
   1  1. FM/DB2 (SQLDA) format         1  1. Old or Reuse
      2. DB2 unload format                2. Mod
      3. DSNTIAUL format
      4. User defined
      5. Delimited Variables (CSV)     Separator character

Null Indicators:
   Usage                 Placement          Type             Indicator
   1  1. Separate        1  1. Before      2  1. One byte      Char
      2. None               2. After           2. Two byte   1_____ Integer
      3. CSV, use NI char   3. User defined

More Options:                                        Execution:  Allocation:
   Enter "/" to select option                        BATCH       BATCH
   /  Data type format
```

Figure 28-51 Export Options panel

When you have completed the settings as shown before, press PF3 to generate the following sample JCL shown in Example 28-4.

Example 28-4 Sample procedure to export DB2 data

```
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (A),
000002 //             AMINTOR,MSGCLASS=A,
000003 //             NOTIFY=AMINTOR,CLASS=A,
000004 //             MSGLEVEL=(1,1)
000005 //*
000006 //FMNDB2   EXEC PGM=FMNDB2,PARM=('SSID=D81H,SQID=AMINTOR')
000007 //STEPLIB  DD DSN=ADTOOLS.FM710.SFMNMOD1,DISP=SHR
000008 //         DD DSN=SYS1.D8OH.SDSNEXIT,DISP=SHR
000009 //         DD DSN=SYS1.D81H.SDSNLOAD,DISP=SHR
000010 //SYSPRINT DD SYSOUT=*
000011 //FMNTSPRT DD SYSOUT=*
000012 //SYSTERM  DD SYSOUT=*
000013 //FMNOUT   DD DISP=SHR,
000014 //            DSN=AMINTOR.BOOK2006.DATA.EXP
000015 //SYSIN    DD *
000016 $$FILEM DBX OBJIN="AMINTOR"."EMPNEW",
000017 $$FILEM OUTPUT=FMNOUT,
000018 $$FILEM TCOUT=AMINTOR.BOOK2006.TEMPLATE(TEMPEXP2),
000019 $$FILEM NULLIND=_,
000020 $$FILEM CSV=NO,
000021 $$FILEM ROWS=ALL
***** **************************** Bottom of Data ****************************
```

28.5.3 DB2 data import online

To import data from a sequential or VSM data set into DB2, select option **6** on the Utility Functions panel of File Manager/DB2, as shown in Figure 28-52.

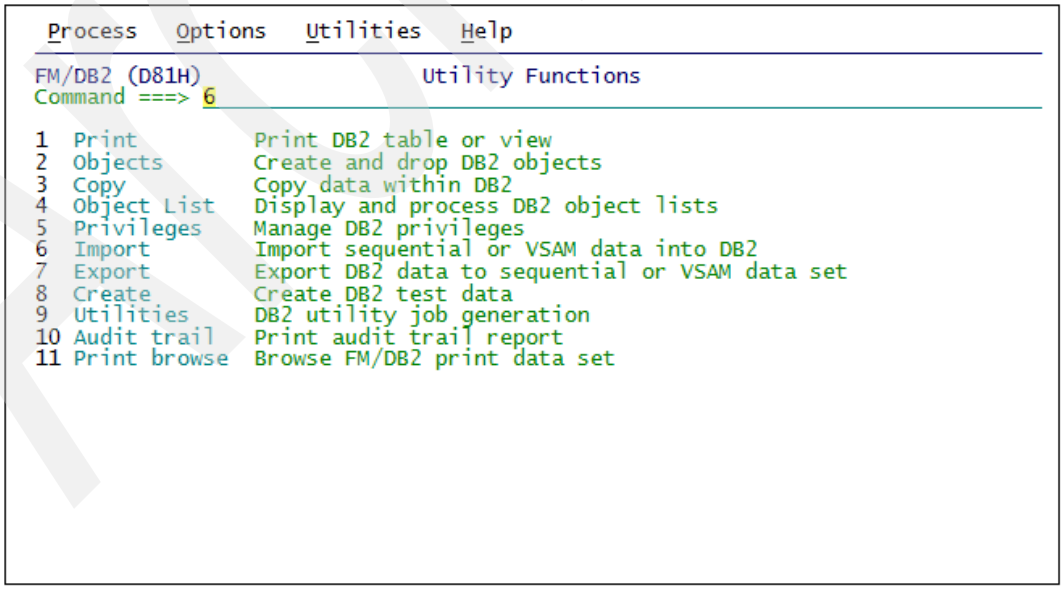
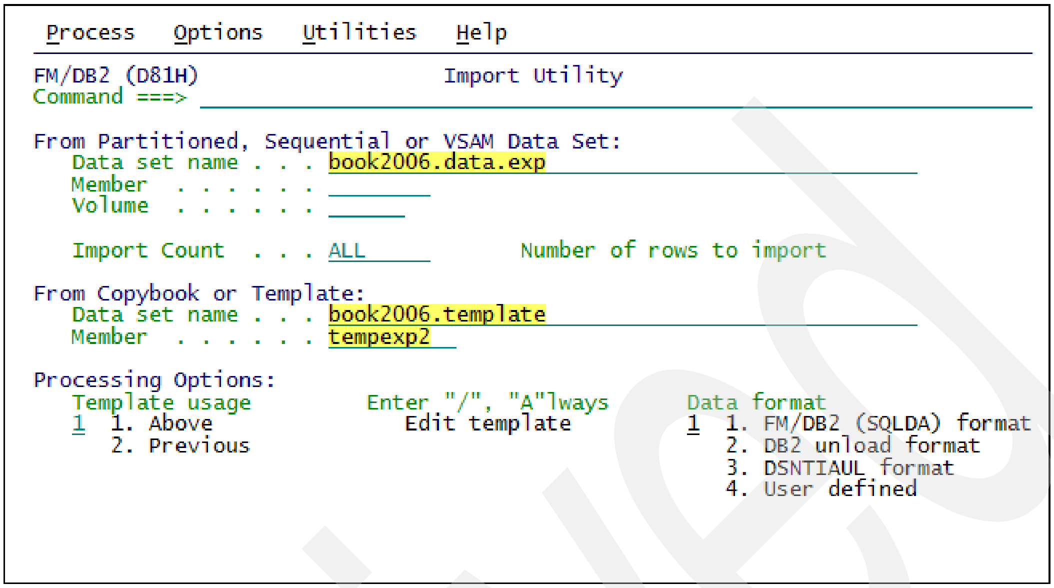


Figure 28-52 Option 6 to import data on File Manager/DB2 Utility Functions panel

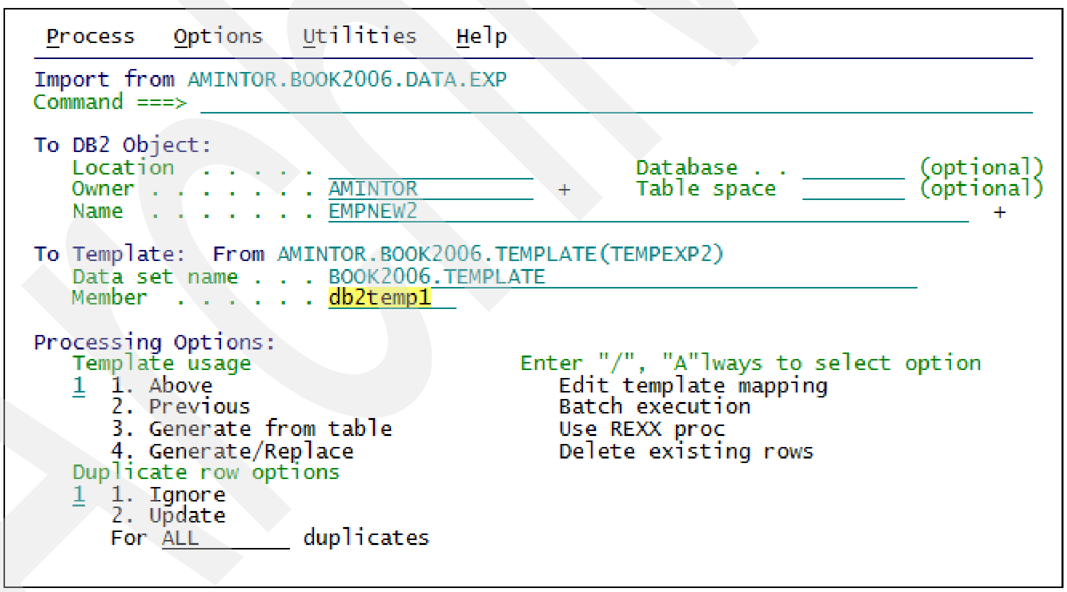
Fill in the fields about the source data set and the template to be used when importing data, as shown in Figure 28-53.



Process Options Utilities Help
FM/DB2 (D81H) Import Utility
Command ===>
From Partitioned, Sequential or VSAM Data Set:
Data set name . . . book2006.data.exp
Member . . . .
Volume . . . .
Import Count . . . ALL Number of rows to import
From Copybook or Template:
Data set name . . . book2006.template
Member tempexp2
Processing Options:
Template usage
1 1. Above
2. Previous
Enter "/", "A"lways
Edit template
Data format
1 1. FM/DB2 (SQLDA) format
2. DB2 unload format
3. DSNTIAUL format
4. User defined

Figure 28-53 Specifying MVS source data set and templates

Do the same thing for the target object, as shown in Figure 28-54.



Process Options Utilities Help
Import from AMINTOR.BOOK2006.DATA.EXP
Command ===>
To DB2 Object:
Location . . . . .
Owner . . . . AMINTOR +
Name . . . . EMPNEW2 +
Database . . (optional)
Table space (optional)
To Template: From AMINTOR.BOOK2006.TEMPLATE(TEMPEXP2)
Data set name . . . BOOK2006.TEMPLATE
Member db2temp1
Processing Options:
Template usage
1 1. Above
2. Previous
3. Generate from table
4. Generate/Replace
Duplicate row options
1 1. Ignore
2. Update
For ALL duplicates
Enter "/", "A"lways to select option
Edit template mapping
Batch execution
Use REXX proc
Delete existing rows

Figure 28-54 Specifying target DB2 object for importing data

You can edit, if required, the templates to specify selection criteria and edit the template mapping if there are differences between source and target formats. See Figure 28-55.

```
 Process   Options   Help
FM/DB2 (D81H)                 Template Mapping                 Line 1 of 14
Command ===> _____________________________________________________ Scroll PAGE

To          AMINTOR.EMPNEW2
From        AMINTOR.BOOK2006.DATA.EXP

Cmd CL# To Column name  Datatype  NI   Lv From Field Name            Type  Len
        ****  Top of data  ****
      1 EMPNO           CH(6)     NN |  2 REC-TYPE                   AN      2
      2 FIRSTNME        VC(12)    NN |  2 NAME                       AN     20
      3 MIDINIT         CH(1)     NN |  2 REC-ID                     AN      2
      4 LASTNAME        VC(15)    NN |
      5 WORKDEPT        CH(3)        |
      6 PHONENO         CH(4)        |
      7 HIREDATE        DATE         |
      8 JOB             CH(8)        |
      9 EDLEVEL         SMINT        |
     10 SEX             CH(1)        |
     11 BIRTHDATE       DATE         |
     12 SALARY          DEC(9,2)     |  2 SALARY                     PD      4
     13 BONUS           DEC(9,2)     |
     14 COMM            DEC(9,2)     |
```

Figure 28-55 Template mapping when importing data to DB2

You can also change the correspondence between fields and edit new fields to specify default values for data creation.

When data import is finished, File Manager/DB2 displays the confirmation message, as shown in Figure 28-56.

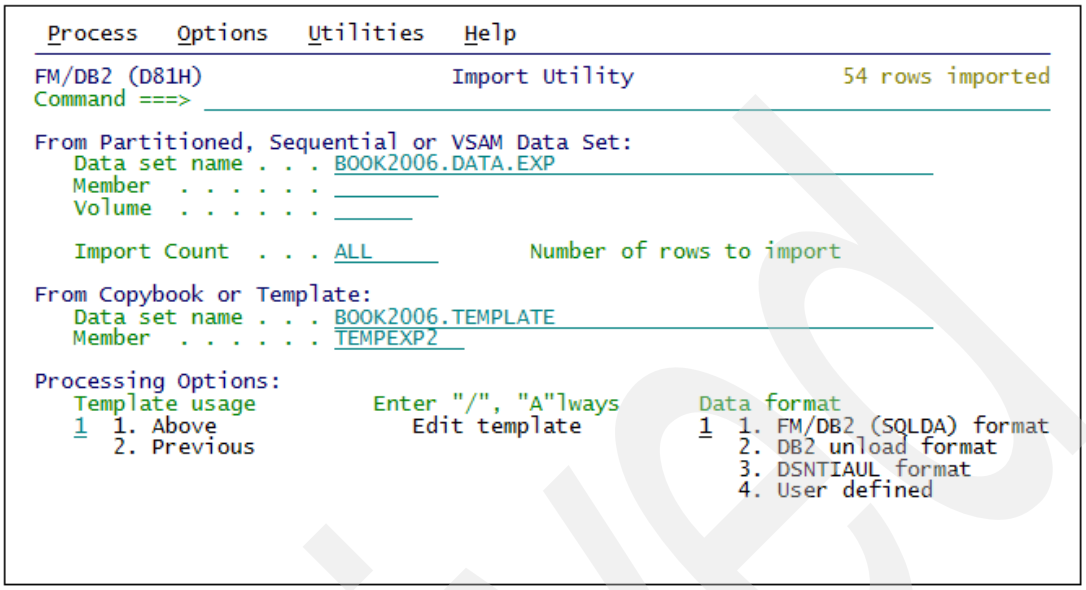


Figure 28-56 Data successfully imported to DB2

28.5.4 Data import to DB2 in a batch mode

Example 28-5 contains a sample procedure to import data into DB2 in a batch mode. This job loads data from an input PDS to table EMP2 according to an MVS template.

Example 28-5 Sample procedure to import data to DB2

```
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (A),
000002 //            AMINTOR,MSGCLASS=A,
000003 //            NOTIFY=AMINTOR,CLASS=A,
000004 //            MSGLEVEL=(1,1)
000005 //*
000006 //FMNDB2   EXEC PGM=FMNDB2,PARM=('SSID=D81H,SQID=AMINTOR')
000007 //STEPLIB  DD DSN=ADTOOLS.FM710.SFMNMOD1,DISP=SHR
000008 //         DD DSN=SYS1.D80H.SDSNEXIT,DISP=SHR
000009 //         DD DSN=SYS1.D81H.SDSNLOAD,DISP=SHR
000010 //SYSPRINT DD SYSOUT=*
000011 //FMNTSPRT DD SYSOUT=*
000012 //SYSTERM  DD SYSOUT=*
000013 //SYSIN    DD *
000014 $$FILEM DBI DSNIN=AMINTOR.BOOK2006.DATA.EXP,
000015 $$FILEM TCIN=AMINTOR.BOOK2006.TEMPLATE(TEMPEXP2),
000016 $$FILEM OBJOUT="AMINTOR"."EMPNEW2",
000017 $$FILEM TMOUT=AMINTOR.BOOK2006.TEMPLATE(DB2TEMP1),
000018 $$FILEM UPDATE=NO,
000019 $$FILEM DUPMAX=ALL,
000020 $$FILEM DELROWS=NO,
000021 $$FILEM ROWS=ALL
****** **************************** Bottom of Data ****************************
```

28.6 Extracting and loading IMS data using File Manager

The File Manager Component for IMS, which is referred to as File Manager/IMS, can be used to perform the following actions:

- Build the extract criteria.
- Run the extract in a batch mode.
- View the extract reports.
- Run the load in a batch mode.
- View the load reports.

28.6.1 IMS data extract

On the File Manager/IMS Primary Option panel, shown in Figure 28-57, type the command **3.3** to extract data from the IMS database.

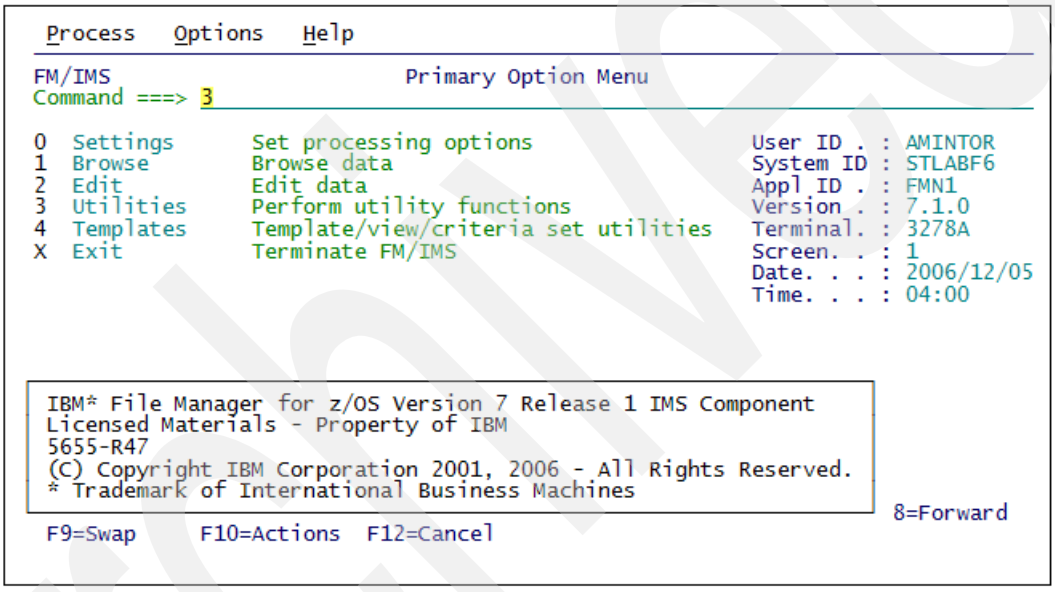


Figure 28-57 Option 3 on File Manager/IMS Primary Option panel

After pressing Enter, Figure 28-58 is displayed. Specify the source IMS database as shown in Figure 28-58.

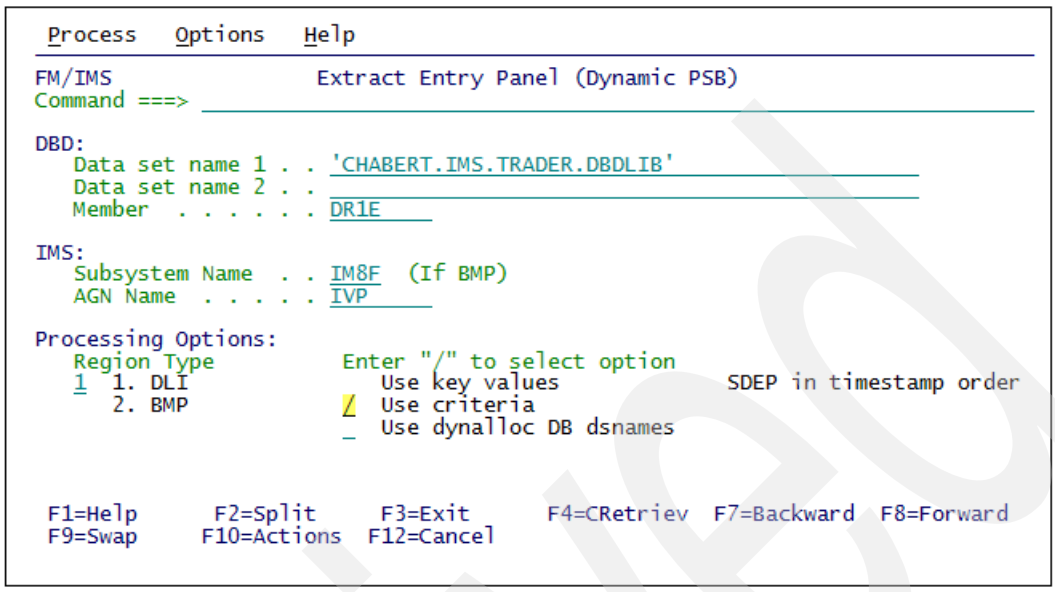


Figure 28-58 Specifying source IMS database for data extraction

Depending on the region type, DLI or BMP, supply and confirm the data sets associated with the source IMS database as shown in Figure 28-59.

Note: There is a specific option (**3**, Criteria) in the *Template/View/Criteria Set Utilities* panel (FMN1PTUT) to create and work with selection criteria. You can invoke this panel using **=4.3** from the command line.

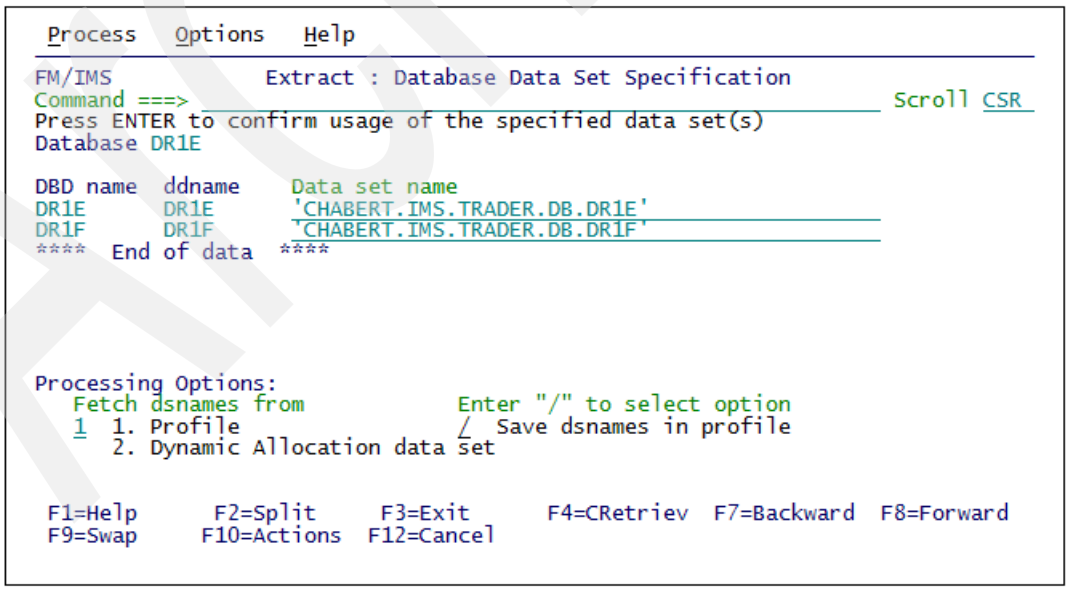


Figure 28-59 DataBase and Dataset specification

Specify the extract options as shown in Figure 28-60.

```
 Process   Options   Help
FMN1PX2             Extract : Criteria/Keys Specification
Command ===>

Database DR1E

Template:
   Data set name . . . 'AMINTOR.BOOK2006.IMS.TEMPLATE'
Criteria Set:
   Data set name . . . 'AMINTOR.BOOK2006.IMS.CRITERIA'
   Member  . . . . . . EXTDJ1E

Key Values:
   Data set name . . .
   Member  . . . . . .

Processing Options:
   Selection criteria                 Enter "/" to select option
   2  1. Use new criteria             /  Edit existing criteria
      2. Use existing criteria           Use key values
      3. Use no criteria
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 28-60 FM /IMS extract options panel

After the Criteria specification panel has been displayed, you can edit the existing criteria or add new ones, as required.

Specify the segments by editing the specifications, as shown in Figure 28-61.

```
 Process   Options   Help
FMN1PC7                 Criteria : Segment Selection
Command ===>                                                 Scroll CSR

Criteria New
Template AMINTOR.BOOK2006.IMS.TEMPLATE(DR1E)
Subset   1

Cmd  Sel  Level          Segment   Description
 e    *   1             CUSTOMER
 e    *    2            COMPANY
   ****  End of data  ****




 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F6=Describe  F7=Backward
 F8=Forward   F9=Swap     F10=Actions  F12=Cancel
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

Figure 28-61 Segment selection for criteria panel

There you have the segment description with the available layouts, as shown in Figure 28-62.

```
 Process   Options   Help
------------------------------------------------------------------------------
FMN1PC5                 Criteria : Segment Layout                Line 1 of 6
Command ===> ____________________________________________________ Scroll CSR

Criteria New                                                      Subset 1
Template AMINTOR.BOOK2006.IMS.TEMPLATE(DR1E)                      Segment COMPANY
-------------- Criteria - Enter 1 to specify expression by field -------------
1 Sel: ____________________________________________________________________ +
Offset 0
Cmd Seq SHC Ref  Field Name                          Picture  Type Start Length
                 ****  Top of data  ****
___         1  1 CUSTCOMP-SEG-IN                              AN       1     24
___        2K  2 COMPANY-CUST                        X(20)     AN       1     20
__4         3   2 NO-SHARES-1                        9(1)      ZD      21      1
__5         4   2 NO-SHARES-2                        9         ZD      22      1
__6         5   2 NO-SHARES-3                        9         ZD      23      1
__3         6   2 CUST-LVL                           X         AN      24      1
                 ****  End of data  ****


 F1=Help    F2=Split   F3=Exit    F4=Expand  F5=RFind   F7=Up      F8=Down
 F9=Swap    F10=Left   F11=Right  F12=Cancel
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

Figure 28-62 Segment layout selection for criteria

Edit the layouts to specify the selection criteria, as shown in Figure 28-63.

```
 Process   Options   Help
------------------------------------------------------------------------------
FMN1PC5                 Criteria : Segment Layout                Line 1 of 2
Command ===> ____________________________________________________ Scroll CSR

Criteria New                                                      Subset 1
Template AMINTOR.BOOK2006.IMS.TEMPLATE(DR1E)                      Segment CUSTOMER
-------------- Criteria - Enter 1 to specify expression by field -------------
1 Sel: #2 = 'A'____________________________________________________________ +
Offset 0
Cmd Seq SHC Ref  Field Name                          Picture  Type Start Length
                 ****  Top of data  ****
___ S C    1K 1 COMPANY-SEG-IN                              AN       1     25
___ S      2K  2 COMPANY-NAME                       X(025)    AN       1     25
                 ****  End of data  ****




 F1=Help    F2=Split   F3=Exit    F4=Expand  F5=RFind   F7=Up      F8=Down
 F9=Swap    F10=Left   F11=Right  F12=Cancel
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

Figure 28-63 Selection criteria added

Exit and save the criteria.

File Manager/IMS displays the confirmation message, as shown in Figure 28-64.

```
 Process   Options   Help
FMN1PX2            Extract : Criteria/Keys Specifica       Criteria set saved
Command ===>                                                          More:     +
Database DR1E

Template:
   Data set name . . . 'AMINTOR.BOOK2006.IMS.TEMPLATE'
Criteria Set:
   Data set name . . . 'AMINTOR.BOOK2006.IMS.CRITERIA'
   Member  . . . . . . EXTDR1E

Key Values:
   Data set name . . .
   Member  . . . . . .

Processing Options:
   Selection criteria                 Enter "/" to select option
   2  1. Use new criteria                Edit existing criteria
      2. Use existing criteria           Use key values
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
. . . . . .
```

Figure 28-64 Criteria set saved confirmation message

On the next panel, provide the specifications for the target sequential data set, as shown in Figure 28-65.

```
 Process   Options   Help
 MN1PX3             Extract : To Data Set Specification
Command ===>

Extract to:
   Data set name . . . 'amintor.dj1e.extract'
   Disposition . . . . new       (NEW or OLD)
   Volume serial . . .           (If not cataloged)
   Device type . . . . sysda     (Generic unit or device address)

Allocation parameters (if disposition is NEW):
   Management class  .           (Blank for default management class)
   Storage class . . .           (Blank for default storage class)
   Data class  . . . .           (Blank for default data class)
   Space units . . . . cyl       (BLK, TRK, CYL, KB or MB)
   Primary quantity  . 1         (in above units)
   Secondary quantity  1         (in above units)



 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 28-65 Target data set specifications

File Manager/IMS generates the extraction JCL. A sample is presented in Example 28-6.

Example 28-6 IMS data extract sample procedure

```
****** ***************************** Top of Data ******************************
000001 //AMINTORB JOB (*),
000002 //             AMINTOR,MSGCLASS=A,
000003 //             NOTIFY=AMINTOR,CLASS=A,
000004 //             MSGLEVEL=(1,1)
000005 //*
000006 //FILEMAN  EXEC PGM=FMN1IMSB
000007 //STEPLIB  DD DSN=FMN.SFMNMOD1,DISP=SHR
000008 //SYSPRINT DD SYSOUT=*,RECFM=FBA,LRECL=133
000009 //FMNTSPRT DD SYSOUT=*
000010 //FMN1JIN  DD DSN=FMN.SFMNSLIB,DISP=SHR
000011 //SYSTERM  DD SYSOUT=*
000012 //FMIMSIN  DD *
000013 $$FILEM IXB REGNTYPE=DLI,
000014 $$FILEM     PSBTYPE=DYNAMIC,
000015 $$FILEM     DBDDSN=CHABERT.IMS.TRADER.DBDLIB,
000016 $$FILEM     DBDMEM=DR1E,
000017 $$FILEM     CRITERIA=Y,
000018 $$FILEM     CRITDD=CRITDD,
000019 $$FILEM     CRITMEM=EXTDJ1E,
000020 $$FILEM     RESLIB1=IMSVS.IM8G.SDFSRESL,
000021 $$FILEM     MACLIB=IMSVS.I81A2AC.SDFSMAC,
000022 $$FILEM     PROCOPT=G,
000023 $$FILEM     RECON1=IMSVS.IM8F.RECON1,
000024 $$FILEM     RECON2=IMSVS.IM8F.RECON2,
000025 $$FILEM     RECON3=IMSVS.IM8F.RECON3,
000026 $$FILEM     DBRC=N,
000027 $$FILEM     IRLM=N,
000028 $$FILEM     RSR=N,
000029 $$FILEM     DFSVSAMP=IMSVS.IM8G.PROCLIB,
000030 $$FILEM     VSMPMEM=DFSVSM00,
000031 $$FILEM     KEYEXT=N
000032 /*
000033 //REPORT   DD SYSOUT=*
000034 //UNLOAD   DD DSN=AMINTOR.DJ1E.EXTRACT,
000035 //         DISP=(,CATLG,DELETE),
000036 //         UNIT=SYSDA,
000037 //         SPACE=(CYL,(1,1,0),RLSE),
000038 //         DCB=(RECFM=VB,LRECL=00110,BLKSIZE=0)
000039 //* ------------------------------------------------------------------
000040 //* -- IMS DLI DATASETS REQUIRED FOR THE EXTRACT                   --
000041 //* ------------------------------------------------------------------
000042 //DR1E DD DSN=CHABERT.IMS.TRADER.DB.DR1E,DISP=SHR
000043 //DR1F DD DSN=CHABERT.IMS.TRADER.DB.DR1F,DISP=SHR
000044 //CRITDD   DD DSN=AMINTOR.BOOK2006.IMS.CRITERIA,DISP=SHR
****** ***************************** Bottom of Data ***************************
/
```

The generated data set can be used by File Manager/IMS for upload.

Figure 28-66 presents a fragment of the extracted data.

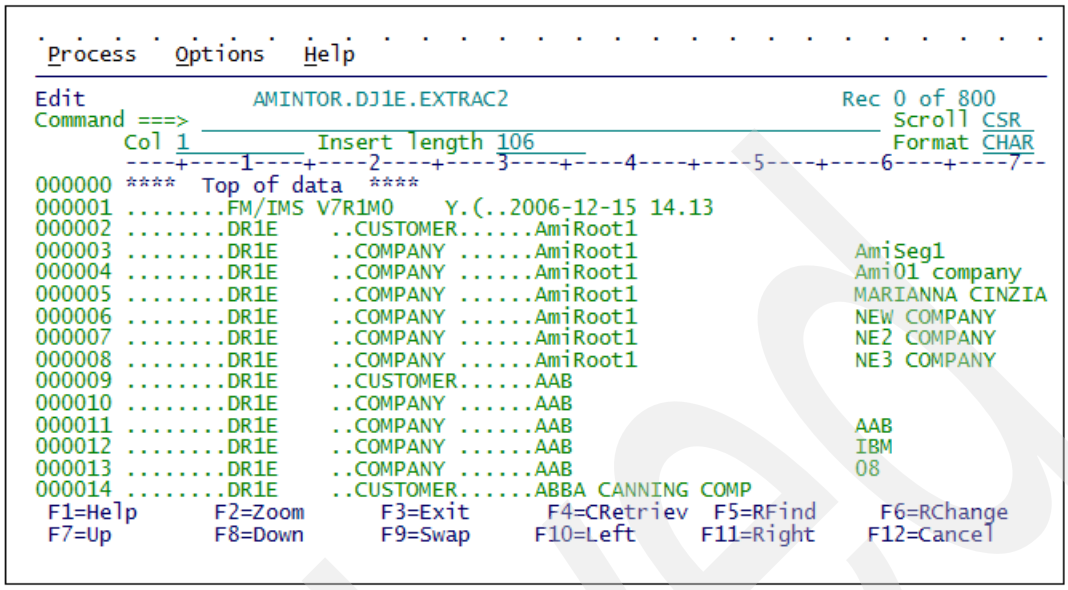


Figure 28-66 Data extracted from IMS database

In Example 28-7 and Example 28-8 we present a partial IMS data export report.

Example 28-7 IMS database extract report (part 1)

```
********************************* TOP OF DATA ***********************************
IBM File Manager for z/OS IMS Component
                                        Database Extract Summary

Primary database  DR1E
Extract
   From           CHABERT.IMS.TRADER.DB.DR1E
   To             AMINTOR.DJ1E.EXTRAC2
DBD
   Data set name 1  CHABERT.IMS.TRADER.DBDLIB
   Data set name 2
   Member         DR1E
Region Type       DLI
Processing Options:
   Use Key values N
   Use Criteria   Y
        Data set  AMINTOR.BOOK2006.IMS.CRITERIA(EXTDR1E)
Databases Extracted:
              DBD Name  ddname    Data set name
              --------  --------  -------------------------------------------
              DR1E      DR1E      CHABERT.IMS.TRADER.DB.DR1E
              DR1F      DR1F      CHABERT.IMS.TRADER.DB.DR1F
IBM File Manager for z/OS IMS Component
                                        Database Extract Statistics Report


                                  -Segm Length- ----- Extract Statistics ----
              Database Segment Level    Min   Max      Read   Rejected  Extract
              ------------------------------------------------------------------
              DR1E     CUSTOMER  01      25    25       121          0        1
                       COMPANY   02      24    24       678          0        6
   Subtotal   DR1E                                     799          0        7

   Grand total                                         799          0        7

                                       *** End of IMS Extract Report ***
```

Example 28-8 IMS Database extract report (part 2)

```
IBM File Manager for z/OS IMS Component
                                          Database Extract Statistics Report

-Segm Length- ----- Extract Statistics ------  Logically
              Database Segment Level    Min   Max      Read   Rejected  Extracted  Related DB
              ------------------------------------------------------------------------------
              DJ1E     SHIRE     01      53    53        11          0         11
                       SHIRENP   02      11    11        11          0         11
                       LINKSUB   02      18    18        87          0         87  DJ2E
   Subtotal   DJ1E                                     109          0        109

              DJ2E     SUBURB    01      62    62        57          0         57
                       LINKSTR   02      28    28      3603          0       3603  DJ3E
                       NSTREET   02      30    30        29          0         29
                       INSHIRE   02      20    20        73          0         73  DJ1E
                       LETRBOX   02      34    34        30          0         30
   Subtotal   DJ2E                                    3792          0       3792

              DJ3E     STREET    01      32    32       215          0        213
                       INSUB     02      18    18       332          0        332  DJ2E
   Subtotal   DJ3E                                     547          0        547

   Grand total                                        4448          0       4448

                                        *** End of IMS Extract Report ***
```

28.6.2 Loading data extracted from IMS by File Manager

To load data into the IMS database, select option **4** on the File Manager/IMS main panel, as shown in Figure 28-67.

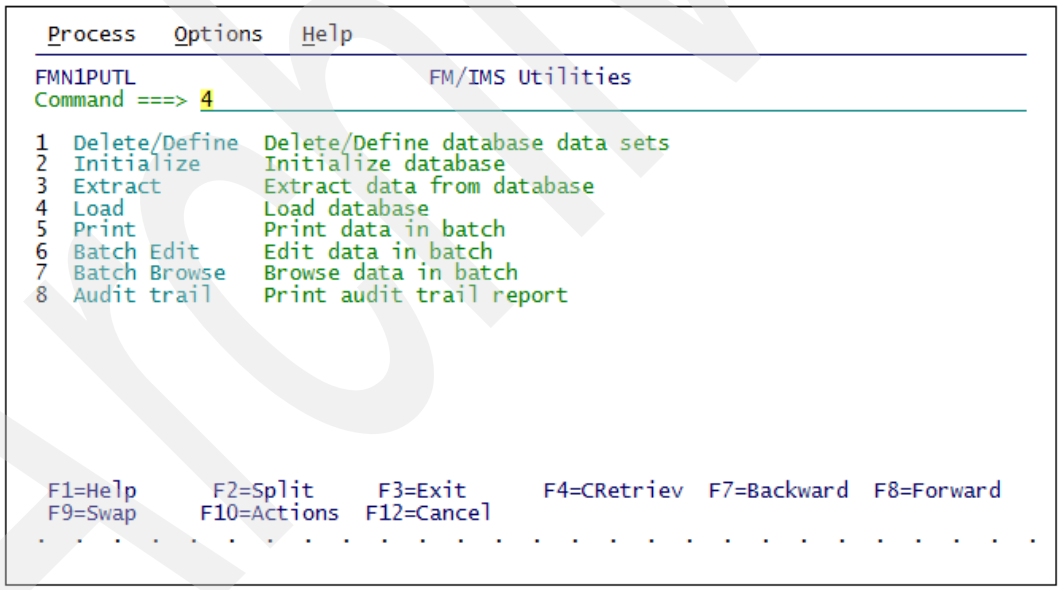


Figure 28-67 Option 4 on File Manager/IMS Primary Option panel selected

When the next panel appears, you specify the target IMS database and processing options, as shown in Figure 28-68.

```
 Process   Options   Help
FMN1PL1                Load Entry Panel (Dynamic PSB)
Command ===>

DBD:
   Data set name 1 . . 'CHABERT.IMS.TRADER.DBDLIB'
   Data set name 2 . .
   Member  . . . . . . DR1E

IMS:
   Subsystem Name  . . IM8F  (If BMP)
   AGN Name  . . . . . IVP

Processing Options:
   Region Type     Load Options                  Enter "/" to select option
   1  1. DLI       2  1. Insert only               Use dynalloc DB dsnames
      2. BMP           2. Insert or Update
                       3. Insert-Load PSB (If DLI)


 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 28-68 File Manager/IMS Load Entry panel

If required, confirm the data sets associated with the target IMS database, as shown in Figure 28-69.

```
 Process   Options   Help
FMN1PE3            Load : Database Data Set Specification
Command ===>                                                  Scroll CSR
Press ENTER to confirm usage of the specified data set(s)
Database DR1E

DBD name  ddname    Data set name
DR1E      DR1E      'CHABERT.IMS.TRADER.DB.DR1E'
DR1F      DR1F      'CHABERT.IMS.TRADER.DB.DR1F'
****  End of data  ****




Processing Options:
   Fetch dsnames from            Enter "/" to select option
   1  1. Profile                 /  Save dsnames in profile
      2. Dynamic Allocation data set

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 28-69 IMS data set specification panel

Next specify the source data set and the IMS logging options, as shown in Figure 28-70.

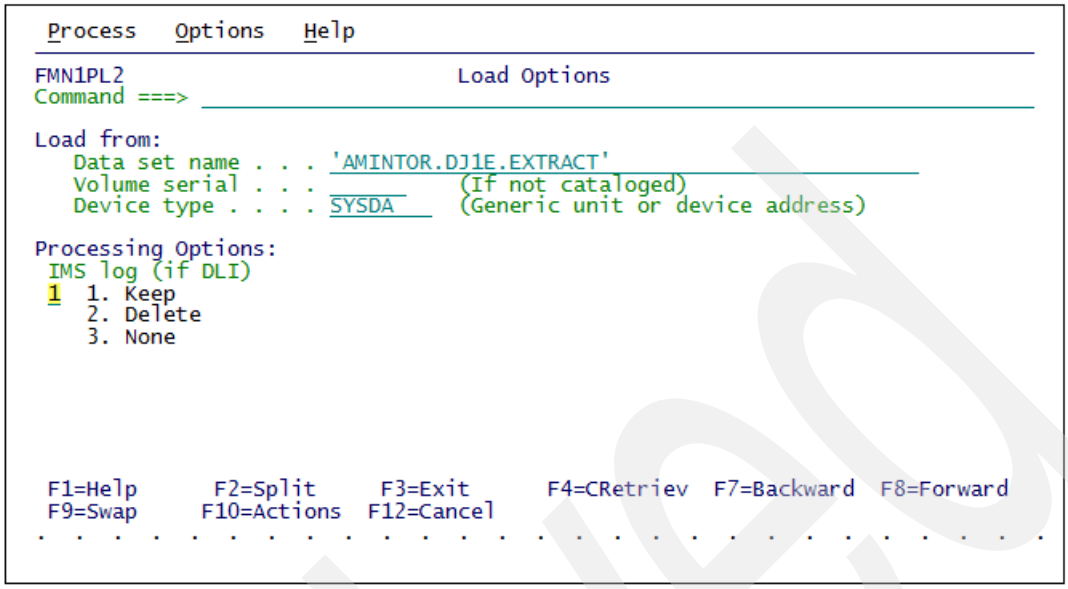


Figure 28-70 Source data set and IMS logging options for data load

File Manager/IMS generates the load JCL. The sample procedure is shown in Example 28-9.

Example 28-9 Sample procedure to load data into IMS database

```
****** ***************************** Top of Data ***************************
000001 //AMINTORB JOB (A),
000002 //            AMINTOR,MSGCLASS=A,
000003 //            NOTIFY=AMINTOR,CLASS=A,
000004 //            MSGLEVEL=(1,1)
000005 //*
000006 //* ---------------------------------------------------------------
000007 //*
000008 //*       SORT LOAD FILE.
000009 //*
000010 //* ---------------------------------------------------------------
000011 //*
000012 //TESTRC    IF (RC LE 4) THEN
000013 //SORTSTEP EXEC PGM=SORT
000014 //SYSOUT   DD SYSOUT=*
000015 //SORTIN   DD DSN=AMINTOR.DJ1E.EXTRACT,
000016 //         DISP=SHR
000017 //SORTOUT  DD DSN=&&FMN1XTSD,
000018 //       LIKE=AMINTOR.DJ1E.EXTRACT,
000019 //       UNIT=SYSALLDA,
000020 //       DISP=(NEW,PASS)
000021 //SORTWK01 DD UNIT=SYSALLDA,
000022 //       SPACE=(CYL,(10,10)),
000023 //       DISP=(NEW,DELETE,DELETE)
000024 //SORTWK02 DD UNIT=SYSALLDA,
000025 //       SPACE=(CYL,(10,10)),
000026 //       DISP=(NEW,DELETE,DELETE)
000027 //SORTWK03 DD UNIT=SYSALLDA,
000028 //       SPACE=(CYL,(10,10)),
000029 //       DISP=(NEW,DELETE,DELETE)
000030 //SYSIN    DD *
000031  RECORD TYPE=V
```

```
000032  SORT FIELDS=(05,2,BI,A,07,2,BI,A,09,4,BI,A)
000033 /*
000034 //ENDRC     ENDIF
000035 //*
000036 //* ------------------------------------------------------------
000037 //*
000038 //*       IMS LOAD
000039 //*
000040 //* ------------------------------------------------------------
000041 //*
000042 //TESTRC    IF (RC LE 4) THEN
000043 //FILEMAN  EXEC PGM=FMN1IMSB
000044 //STEPLIB  DD DSN=FMN.SFMNMOD1,DISP=SHR
000045 //SYSPRINT DD SYSOUT=*,RECFM=FBA,LRECL=133
000046 //FMNTSPRT DD SYSOUT=*
000047 //FMN1JIN  DD DSN=FMN.SFMNSLIB,DISP=SHR
000048 //SYSTERM  DD SYSOUT=*
000049 //FMIMSIN  DD *
000050 $$FILEM ILB REGNTYPE=DLI,
000051 $$FILEM     PSBTYPE=DYNAMIC,
000052 $$FILEM     LOADTYPE=2,
000053 $$FILEM     LDDSN=AMINTOR.DJ1E.EXTRACT,
000054 $$FILEM     DBDDSN=CHABERT.IMS.TRADER.DBDLIB,
000055 $$FILEM     DBDMEM=DR1E,
000056 $$FILEM     RESLIB1=IMSVS.IM8G.SDFSRESL,
000057 $$FILEM     MACLIB=IMSVS.I81A2AC.SDFSMAC,
000058 $$FILEM     RECON1=IMSVS.IM8F.RECON1,
000059 $$FILEM     RECON2=IMSVS.IM8F.RECON2,
000060 $$FILEM     RECON3=IMSVS.IM8F.RECON3,
000061 $$FILEM     DBRC=N,
000062 $$FILEM     IMSBKO=N,
000063 $$FILEM     IRLM=N,
000064 $$FILEM     RSR=N,
000065 $$FILEM     DFSVSAMP=IMSVS.IM8G.PROCLIB,
000066 $$FILEM     VSMPMEM=DFSVSM00,
000067 $$FILEM     CHKPFREQ=100
000068 /*
000069 //LOAD     DD DSN=&&FMN1XTSD,DISP=SHR
000070 //REPORT   DD SYSOUT=*
000071 //IEFRDER  DD DSN=AMINTOR.IMSLOG.D061205.T072535.DR1E,
000072 //       UNIT=SYSALLDA,
000073 //       SPACE=(TRK,(20,20),RLSE),
000074 //       DCB=(RECFM=VB,LRECL=6140,BLKSIZE=6144),
000075 //       DISP=(NEW,CATLG,CATLG)
000076 //* ------------------------------------------------------------
000077 //* -- IMS DLI DATASETS REQUIRED FOR THE LOAD                  --
000078 //* ------------------------------------------------------------
000079 //DR1E DD DSN=CHABERT.IMS.TRADER.DB.DR1E,DISP=OLD
000080 //DR1F DD DSN=CHABERT.IMS.TRADER.DB.DR1F,DISP=OLD
000081 //ENDRC     ENDIF
000082 //*
****** *************************** Bottom of Data ****************************
```

28.7 Editing and browsing data using IPT and File Manager

When an application abnormal end (abend) occurs, you have to investigate Logs, SYSOUT, and data to analyze and evaluate the information received. During this activity you edit, browse, print, and use the dataset utility within ISPF.

The following topics should help to increase your productivity, facilitate the usability of File Manager, and improve the navigation of functions and panels between FM and ISPF. To enhance several ISPF functions and integrate the FM's characteristics, we use the IBM ISPF Productivity Tool (IPT).

28.7.1 IPT overview

IPT (IBM ISPF Productivity Tool) is an enhancement to TSO/ISPF that improves its productivity and performance. IPT integrates seamlessly into the most frequently used ISPF functions, including BROWSE, EDIT, VIEW, and DSLIST. IPT extends the functions provided by ISPF for standard data sets to other objects, including VSAM files, Librarian/Panvalet libraries, UNIX System Services (USS) files, PC files, DB2 tables, as well as other object classes. Along with this support, IPT provides a powerful object browser called OLIST that has the look, feel, and functionality of DSLIST.

All IPT functions are totally integrated. IPT can perform almost any activity within ISPF, or internally invoke the function that can perform the task.

When you start using the enhanced features of IPT to perform standard BROWSE, EDIT, VIEW functions, as well as most of the ISPF capabilities, you can see that almost every ISPF capability has been enhanced and new capabilities added.

28.7.2 IPT and Object Linking and Embedding (OLE)

IPT relates objects to applications similarly to the way a PC performs Object Linking and Embedding (OLE).

OLE is the process by which a specified object (such as a VSAM file) is internally registered to a method (for example, File Manager) that performs a task the user requests. Usually, it is sufficient to click the object in order to take the appropriate action. It is up to the system to resolve the class of object, the desired action, and what program or product should perform the requested function.

28.7.3 Sample scenarios

During the generation of new procedures, or when the modification of objects requires you to create a new file, or when a testing effort requires a clean copy, you can model it based on an existing file that has common attributes.

Normally we execute these processes in batch mode and cross between the SDSF Output (SYSOUT) and the Edit or Browser Panel to modify or verify the data.

We use the IPT "point-and-shoot" capabilities to simplify the path that we normally follow to browses/edit objects and allow more cursor activated functions. Like ISPF's point-and-shoot, you can use the cursor to click on fields, press the Enter key, and have IPT take the appropriate action. If you use an emulation program, such as IBM Personal Communications, you can use the mouse to both position the cursor on a field and send the Enter key.

The following scenario illustrates how to navigate between the panel and demonstrates how you can improve your productivity.

Many ISPF panels contain data set names. *IPT* automatically treats these as input parameters to the BROWSE, EDIT, and VIEW commands. Additionally, the IPT TSO shell lets you use a slash as a symbolic placeholder for a “point-and-shoot” parameter within the parameter buffer of a TSO command.

Tip: ISPF Productivity Tool extends the point-and-shoot capabilities of ISPF to allow more cursor activated functions. Like ISPF’s point-and-shoot, you can use the cursor to click on fields, press the Enter key, and have IPT take the appropriate action. If you use an emulation program, such as IBM Personal Communications, you can use the mouse to both position the cursor on a field and send the Enter key. With a workstation and mouse, the IPT extended point-and-shoot capability makes ISPF behave more like a PC functionality.

For the scenario below, take the following steps:

1. BROWSE any JCLs using a spool viewer as SDSF. You might want to inspect a data set referred to in the SYSOUT as shown in Figure 28-71.

```
  Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 ISFPCU41 UT DISPLAY AMINTORB JOB03982  DSID     3 LINE 12       COLS 02- 81
 COMMAND INPUT ===> bf                                        SCROLL ===> CSR
        5 //SYSTERM  DD SYSOUT=*
        6 //SYSIN    DD *
ICH70001I AMINTOR  LAST ACCESS AT 21:41:31 ON SUNDAY, NOVEMBER 12, 2006
IEF236I ALLOC. FOR AMINTORB FILEMGR
IEF237I JES2 ALLOCATED TO SYSPRINT
IEF237I JES2 ALLOCATED TO FMNTSPRT
IEF237I JES2 ALLOCATED TO SYSTERM
IEF237I JES2 ALLOCATED TO SYSIN
IGD103I SMS ALLOCATED TO DDNAME SYS00001
IGD103I SMS ALLOCATED TO DDNAME SYS00002
IGD101I SMS ALLOCATED TO DDNAME (FMNSRTP )
        DSN (SYS06316.T215100.RA000.AMINTORB.R0108743    )
        STORCLAS (SCTEMP) MGMTCLAS (        ) DATACLAS (        )
        VOL SER NOS= STF668
IGD105I SYS06316.T215100.RA000.AMINTORB.R0108743     DELETED,   DDNAME=FMNSRTP
IGD104I AMINTOR.BOOK2006.OUTFILE                     RETAINED,  DDNAME=SYS00001
IGD104I AMINTOR.BOOK2006.OUTFIL2                     RETAINED,  DDNAME=SYS00002
IEF142I AMINTORB FILEMGR - STEP WAS EXECUTED - COND CODE 0000
IEF285I   AMINTOR.AMINTORB.JOB03982.D0000102.?         SYSOUT
IEF285I   AMINTOR.AMINTORB.JOB03982.D0000103.?         SYSOUT
```

Figure 28-71 Point-and-Shoot capability using IPT

2. Type `BF` (Browse File: Processor Command) on the command line, place the cursor (point-and-shoot) on the OUTFIL2 Sequential Dataset, and press Enter. IPT invokes File Manager to display the panel shown in Figure 28-72.

```
 Process   Options   Help
FMNPQBD          AMINTOR.BOOK2006.OUTFIL2                     Rec 0 of 3222
Command ===> ______________________________________________________ Scroll PAGE
                                  Record 0          Col 1          Format CHAR
----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----
****  Top of data  ****
12345ALBANE John                  .Veck_Transport      BUY_SELL       23691
12345ALBIS  Cristian              .Headworth_ElectricalBUY_SELL       01501
12345DENNY  Mark Jack             .IBM                 BUY_SELL       02501
12345BUNDLE George                .Casey_Import_Export BUY_SELL       00231
12345ALBANE Marianna Rebecca      .ShareSelect         BUY_SELL       00061
12345BELLA  Cinzia Margherita     .Glass_and_Luget_plc BUY_SELL       00782
12345ALBANE                       .Veck_Transport      BUY_SELL       23692
12345ALBANE                       .Headworth_ElectricalBUY_SELL       01502
12345ALBANE                       .IBM                 BUY_SELL       02502
12345ALBANE                       .Casey_Import_Export BUY_SELL       00232
12345ALBANE                       .ShareSelect         BUY_SELL       00062
12345ALBANE                       .Veck_Transport      BUY_SELL       23691
12345ALBANE                       .Glass_and_Luget_plc BUY_SELL       00781
12345ALBANE                       .Veck_Transport      BUY_SELL       23691
12345ALBANE                       .Headworth_ElectricalBUY_SELL       01501
12345ALBANE                       .IBM                 BUY_SELL       02501
12345ALBANE                       .Casey_Import_Export BUY_SELL       00231
```

Figure 28-72 File Manager Browse Panel (FMNPQBD)

3. The File Manager browse panel has been invoked from IPT using the point-and-shoot capability. IPT invokes (or processes) File Manager using different ways as shown in Figure 28-73.

```
  Menu  Reflist  Refmode  Utilities  Settings  Test  Help  Exit
-------------------------------------------------------------------------------
IQIMSL04              EDIT - ENTRY PANEL
COMMAND ===> _________________________________________________________________
HOTBAR?

ISPF LIBRARY:
   Project   ===>
   Group     ===>           ===>           ===>           ===>
   Type      ===>
   Member    ===>          (Blank or pattern for selection list)
 Other data set, VSAM file, @H, or @L for 'AMINTOR.BOOK2006.COPYBOOK':
   DSN/Cat. level  ===> @H
   Volume serial   ===>        (Optional VOLSER or pattern for selection list)
   Password        ===>        (If password protected)
 Default process   ===> EF     (B=Browse, V=View, E=Edit, BF, EF, VF, or ?)
 Execute TAILOR    ===> N      (Y=Yes ,N=no, D=define commands)
 EDIT/VIEW parameters:
   Initial Macro   ===>          Confirm Cancel/Move/Replace     ===> N (Y, N)
   Profile Name    ===>          Action Bar in Edit/View         ===> Y (Y, N)
   Format  Name    ===>          Highlight coloring in Edit/View ===> Y (Y, N)
                                 Exclusive access of viewed file ===> Y (Y, N)
Preserve VB record length  => N  Mixed Mode (NLS DBCS char. set) ===> N (Y, N)
```

Figure 28-73 The panel shows the default process available (EF, BF, B, E, V...)

4. The File Manager browse Panel has been invoked from Spiffy using the Point and Shot capability. IPT invokes (or Processes) File Manager using different ways as shown in Figure 28-74.

```
 Display  Filter  View  Print  Options  Help
--------------------------------------------------------------------------------
ISFPCU41 UT DISPLAY AMINTORB JOB04526  DSID     4 LINE 127      COLS 02- 81
COMMAND INPUT ===> ef                                          SCROLL ===> CSR
       VOL SER NOS= TMP001
IGD103I SMS ALLOCATED TO DDNAME SYSLIB
IEF237I DMY  ALLOCATED TO FMNADATA
IEF237I DMY  ALLOCATED TO FMNIN
IGD101I SMS ALLOCATED TO DDNAME (FMNPRINT)
       DSN (SYS06322.T222009.RA000.AMINTORB.R0113166    )
       STORCLAS (SCTEMP) MGMTCLAS (        ) DATACLAS (        )
       VOL SER NOS= TMP001
IGD104I AMINTOR.BOOK2006.COPYBOOK                    RETAINED,  DDNAME=SYSLIB
IGD105I SYS06322.T222009.RA000.AMINTORB.R0113166     DELETED,   DDNAME=FMNPRINT
IGD105I SYS06322.T222009.RA000.AMINTORB.R0113159     DELETED,   DDNAME=FMNUT1
IGD105I SYS06322.T222009.RA000.AMINTORB.R0113160     DELETED,   DDNAME=FMNUT2
IGD105I SYS06322.T222009.RA000.AMINTORB.R0113161     DELETED,   DDNAME=FMNUT3
IGD105I SYS06322.T222009.RA000.AMINTORB.R0113162     DELETED,   DDNAME=FMNUT4
IGD105I SYS06322.T222009.RA000.AMINTORB.R0113163     DELETED,   DDNAME=FMNUT5
IGD105I SYS06322.T222009.RA000.AMINTORB.R0113164     DELETED,   DDNAME=FMNUT6
IGD105I SYS06322.T222009.RA000.AMINTORB.R0113165     DELETED,   DDNAME=FMNUT7
IGD104I AMINTOR.BOOK2006.CUSTFIL3                    RETAINED,  DDNAME=SYS00001
IGD104I AMINTOR.BOOK2006.COPYBOOK                    RETAINED,  DDNAME=SYS00002
```

Figure 28-74 EF command to edit the VSAM file

5. Type **EF** on the Command Line and put the cursor on the Dataset name that you wish edit. Press enter to display Figure 28-75.

```
 Process   Options   Help
--------------------------------------------------------------------------------
FMNPVED          AMINTOR.BOOK2006.CUSTFIL3                  Rec 0 of 401
Command ===>                                                Scroll PAGE
                                                 Type KSDS  Format CHAR
      Col 1        Insert length 136
      <===+====1====+====2====+====3====+====4====+====5====+====6====+====7==
000000 ****  Top of data  ****
000001 b0115BGraham, Cinzia   Home Phone112-555-6736        1234 5678
000002 b1209BLiszt, Mackenzie Home Phone481-555-4742        3456 7890
000003 b2130CKleiner, Rick    Cell Phone54-11-5555-6298     1234 5678
000004 b3115CGraham, Holly    Cell Phone135-555-2338        9012 3456
000005 b4115BGraham, Holly    Home Phone112-555-6736        1234 5678
000006 b4120DParris,April LynnWork Phone54-11-5555-4857     5678 9012
000007 b4200BDewitt, Howdy    Work Phone642-555-3852        3456 7890
000008 b4300CDewitt, Howdy    Cell Phone650-555-0547        1234 5678
000009 b5209CLiszt, Mackenzie Work Phone487-555-3261        1234 5678
000010 b6580BMoore, Adeline   Work Phone161-555-4024        7890 1234
000011 b7580CMoore, Adeline   Home Phone221-555-7598        5678 9012
000012 b7715CGraham, Holly    Cell Phone135-555-2338        9012 3456
000013 b8580DMoore, Adeline   Cell Phone138-555-2410        3456 7890
000014 b8815BGraham, Holly    Home Phone112-555-6736        1234 5678
000015 b8815BGraham, Holly    Home Phone112-555-6736        1235 5678
```

Figure 28-75 Invoke File Manager for the SDSF (ISPF) panel

6. You are in Edit mode on the CUSTFIL3 KSDS as shown in the foregoing panel. You can perform or execute all the FM utilities. If you have been authorized, you can modify the first occurrence line from `CINZIA` to `MARIANNA`. Use the FM/Base commands in order to update the data as shown in Figure 28-76.

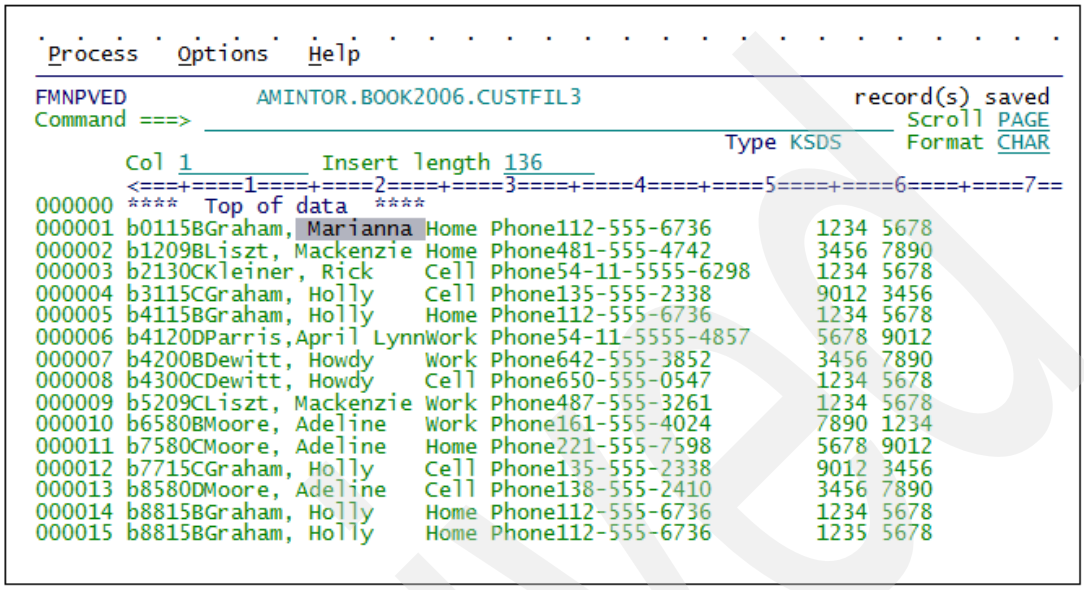


Figure 28-76 Editing VSAM data using FM/Base

7. You are now able to edit and compute the activity you want on the Dataset invoked via SPIFFY. If you predefine a PF key as BROWSE, EDIT, or VIEW, you can place the cursor on the data set name and press the PF key directly. The panel confirms the update done on the right side, at the top. The same scenario could be done with PDS members.

Once IPT has been installed, and you have run the IPT customization via the %IQIWIZRD wizard CLIST, you can invoke FM through any ISPF panels.

28.7.4 IPT %IQIWIZRD wizard

To run the IBM Customization wizard, you have to perform the following simple steps.

1. Type **%IQIWIZRD** on the Ready ISPF panel, as shown in Figure 28-77.

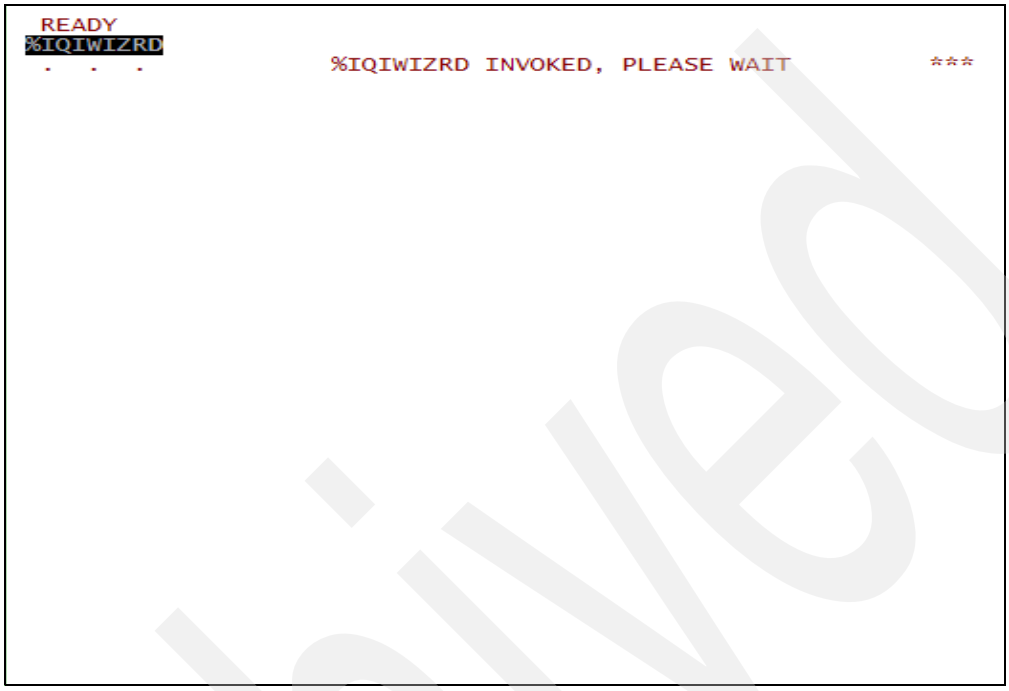


Figure 28-77 To receive the READY panel, exit from your TSO Session

2. When you press Enter, you start your automatic IPT Customization. To complete this step, you have just to know the dataset name of your -IPT- TABLE and PANEL LIBRARY (Example: AMINTOR.HIQI590.SIQITLIB and AMINTOR.HIQI590.SIQIPLIB, and the IPT Customization Wizard starts, as shown in Figure 28-78.

```
------------------------ IBMIPT - Customization Wizard ------------------------
COMMAND ===>
PREVIOUS DEFAULTS WERE UPDATED BY IBM-IPT ON 11/02/06 AT 18:21:59
Please select which tasks you want to perform:

1 Customization      - Set up IBMIPT options and defaults.

X EXIT               - Exit - do not customize, do not set up password.

      Make selection and press the Enter key or press the END key to exit.
```

Figure 28-78 This is the first ISPF panel to start the Customization stage

3. Press Enter to start the process. You receive a list of panels with the default value already set. When you receive the next panel (during the Wizard processes), you must take care to verify that the VSAM INTERFACE panel has been set as shown in Figure 28-79.

```
------------------------ IBMIPT - Customization Wizard ------------------------
COMMAND ===>
When IBMIPT detects a VSAM data set (in BROWSE, EDIT, VIEW, DSLIST, or object
list), it may use an interface to process it. You may select which kind
of VSAM support you want in your installation, if any. The table below
shows what CLIST will handle the support you have selected.

      VSAM interface ===> 2  (Select one of the following options)
Option Invoked VSAM interface                                  Interface CLIST
------ ------------------------------------------------------- ---------------
  0    None (IBMIPT will not handle VSAM files.) . . . . . .  Not applicable
  1    Installation-written VSAM interface . . . . . .  IQIVSAM
  2    IBM File Manager for z/OS . . . . . .  IQI$FMGR
  3    File-Aid VSAM browser/editor(For Version 6 or below) .  IQI$FAID
  4    File-Aid VSAM browser/editor(Version 7.0). . . . . .  IQI$XPRT
  5    File-Aid VSAM browser/editor(Version 8.0 or above) . .  IQI$FAR8
  6    Data-Xpert VSAM browser/editor . . . . . .  IQI$XPRT
  7    MacKinney ISPF VSAM Utility  . . . . . .  IQI$MCKN
  8    IBM Ditto/ESA . . . . . .  IQI$DIT
  9    Serena STARTOOL . . . . . .  IQI$STOL
  10   VSAM RECORD BROWSER (A SAMPLE WORKING INTERFACE) . . .  IQIVBRO

NOTES:- Options 2-9 require a third-party product. If such a product uses
        LIBDEFS, include the LIBDEF statements in the interface CLIST.
```

Figure 28-79 Verify if the VSAM Interface has been set as 2 (File Manager)

4. When the IPT Customization stage is finished, you receive the following confirmation panel, as shown in Figure 28-80.

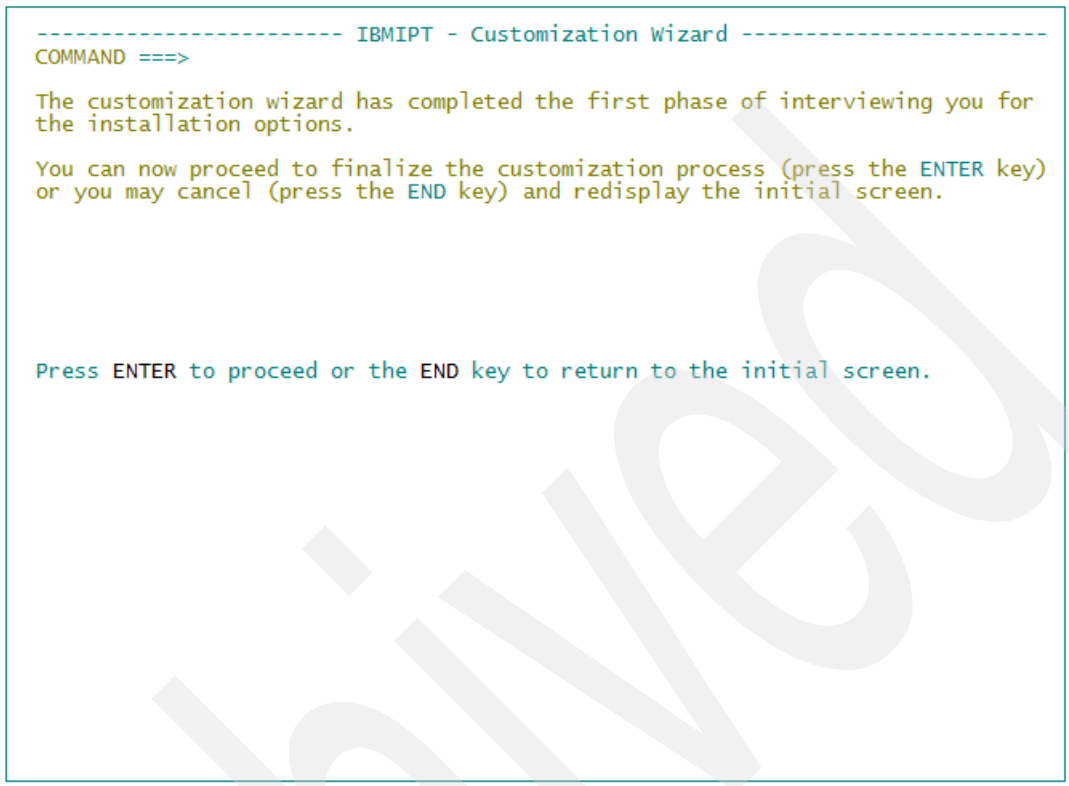


Figure 28-80 Customization Wizard has been completed.

5. To complete the installation process, you have to modify the FM library dataset names in the IQI$FMGR CLIST to conform to the local naming convention. This CLIST has been selected for VSAM support during the customization (when you specified SELECT the Invoked VSAM Interface).

6. Press Enter, and now you are able to invoke File Manager from any ISPF panel. You are able to navigate through your IPT panels now, to perform the new functions from the Dataset List panel shown in Figure 28-81.

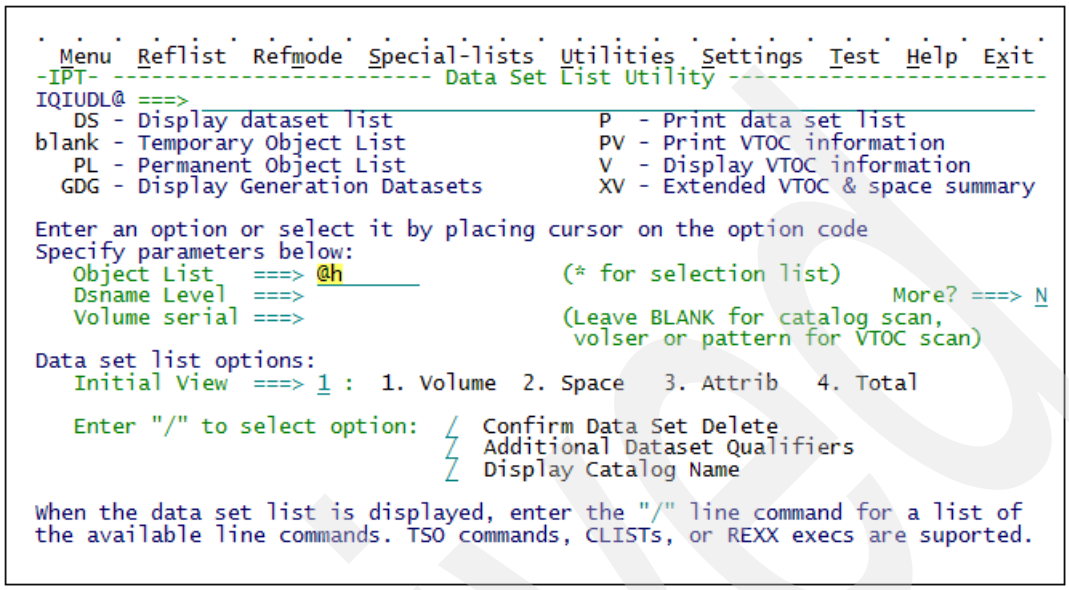


Figure 28-81 Data set List Utility panel (ISPF 3.4)

7. Type **@h** to list the most recently accessed dataset (History) and press Enter to receive the panel shown in Figure 28-82.

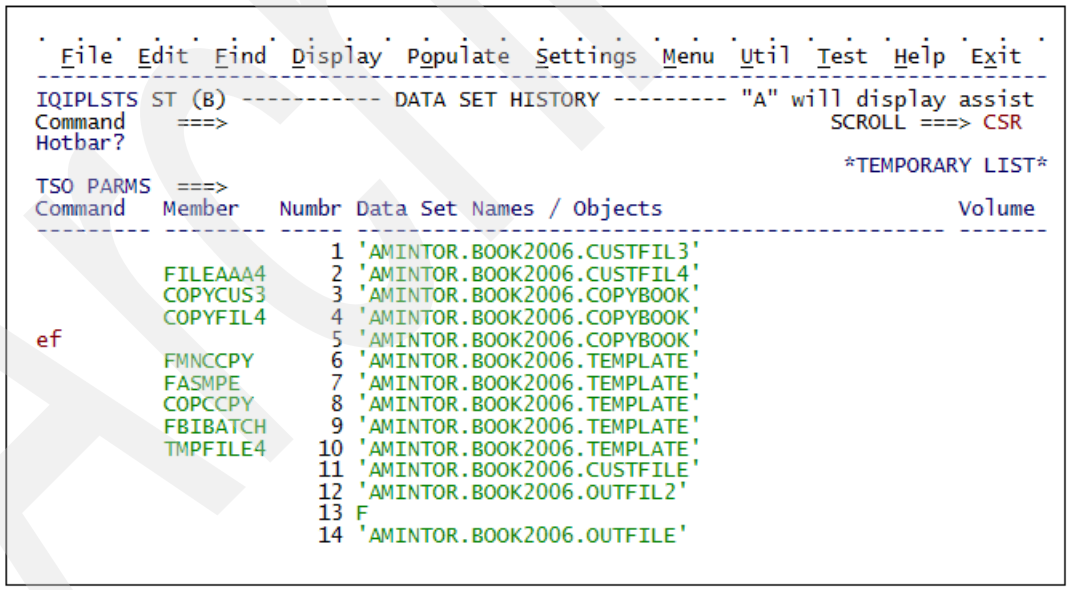


Figure 28-82 Data set History displayed using the @H command

8. Select a Partitioned dataset by using the command **EF** in the prefix Area. After pressing Enter, IPT shows the panel in Figure 28-83.

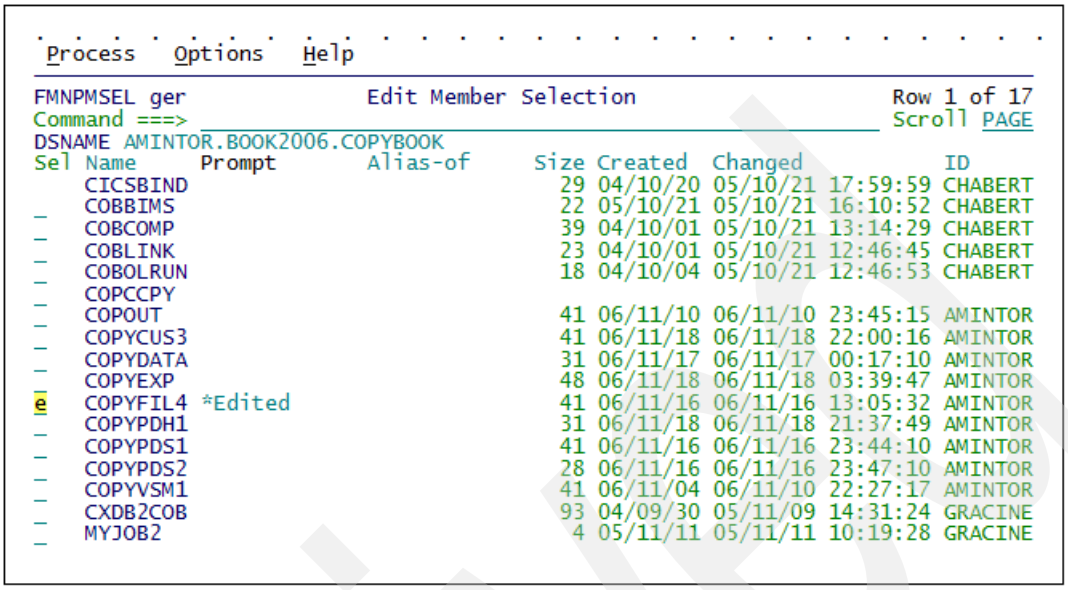


Figure 28-83 File Manager has been automatically invoked from IPT

The File Manager Partitioned Dataset has been invoked and you are now able to edit your member using the FM template features.

Part 8

Workload Simulator for z/OS and OS/390

In this part of the book, we discuss the major steps to install, customize, and use Workload Simulator.

29

Workload Simulator, Workload Simulator Test Manager settings, and customization

In this chapter we discuss the main steps to install and customize Workload Simulator (WSim) and Workload Simulator Test Manager (WTM).

Some steps in this process are usually performed by systems support personnel, and some are performed by WSim and WTM users.

29.1 Installing WSim

Step-by-step instructions for WSim installation from the product tape can be found in the program directory shipped with the tape.

To use the WSim/ISPF interface, which provides user-friendly access to most of the functions and utilities of WSim, perform the following steps:

1. Concatenate:
   - **.SITPPNL data set to ISPPLIB DD
   - **.SITPMSG data set to ISPMLIB DD
   - **.SITPEXEC data set to SYSEXEC DD
   - **.SITPTBL data set to ISPTLIB DD
   - **.SITPSKEL data set to ISPSLIB DD
2. Connect the WSim/ISPF interface to the ISPF system by including an option to invoke the WSim/ISPF interface from one of the ISPF panels.

   An example is shown in Figure 29-1.

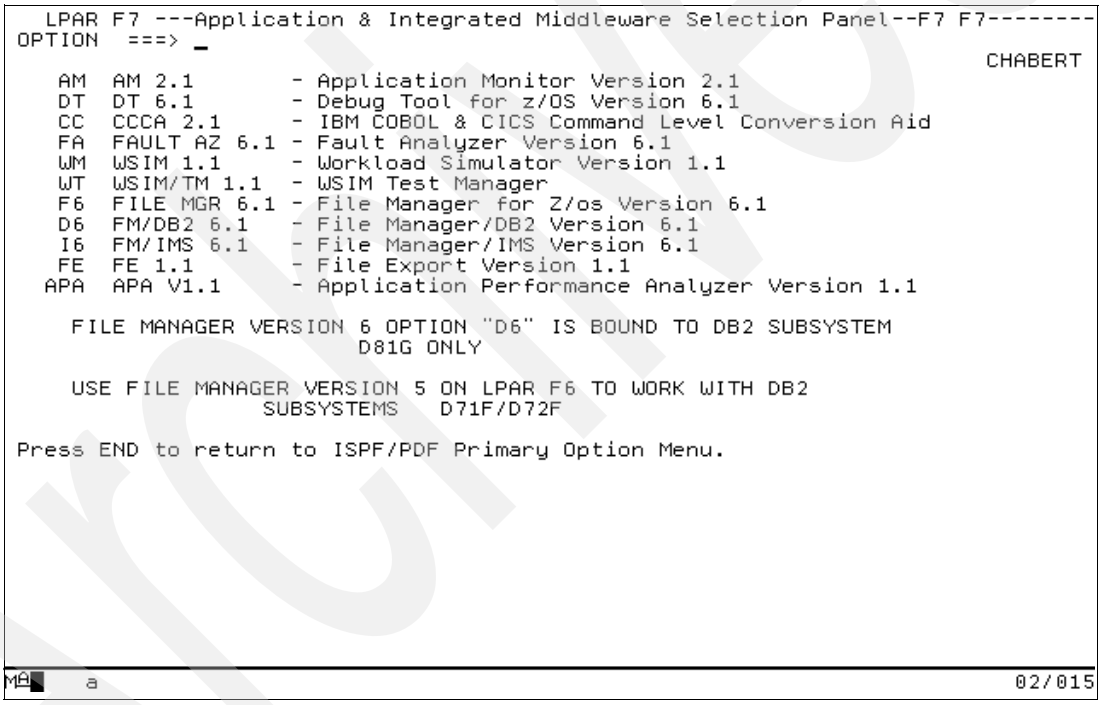


Figure 29-1 ISPF panel with WSim invocation

3. Allocate and catalog data sets to be used by WSim.

   The REXX procedure ITP0INST, which allocates all the data sets required to run the WSim/ISPF interface, is provided in the **.SITPEXEC data set. This procedure also sets up the default for an installation qualifier. If this procedure is not run or is not customized, each WSim user must repeatedly set some default values on the WSim/ISPF interface panels.

4. Update the load library name and models qualifier on the SETUP panel of the WSim/ISPF interface. If default name WSIM is used or they are defined in the REXX procedure ITP0INST in the previous step, there is no requirement to do this.

5. Authorize WSim to run under TSO if WSim simulations are to be run interactively.

Note: The models qualifier field on the SETUP panel of the WSim/ISPF interface accept a maximum of 17 characters, and its value cannot contain quotation marks. This must be taken into consideration when the local models qualifier value is defined.

The sample script INSTALL1 supplied with the product can be used for the verification of WSim installation.

29.2 Installing WTM

Two major tasks have to be performed after WTM data sets from the product tape are transferred:

1. Create a Virtual Telecommunications Access Method (VTAM) application major node.
2. Provide access to WTM.

These tasks are usually performed by systems support personnel.

Additional activities are required for each user of WTM. These activities can be performed by systems support personnel or, in some cases, by users themselves.

29.2.1 Creating a VTAM application major node

A VTAM application major node must be defined in SYS1.VTAMLST or its local equivalent. A local sample is shown in Example 29-1.

Example 29-1 VTAM application major node definition sample for WSim and WTM

```
* ------------------------------------------------------------------ *
*           WSIM TEST MANAGER MAJOR NODE
* ------------------------------------------------------------------ *
        VBUILD TYPE=APPL
*WTM DISPLAY MONITOR
WTMDM00  APPL  EAS=1,PARSESS=YES
WTMDM01  APPL  EAS=1,PARSESS=YES
WTMDM02  APPL  EAS=1,PARSESS=YES
   .
WTMDM09  APPL  EAS=1,PARSESS=YES
*INTERACTIVE DATA CAPTURE
ITPIDC0  APPL  EAS=1,SESSLIM=YES
ITPIDC1  APPL  EAS=1,SESSLIM=YES
ITPIDC2  APPL  EAS=1,SESSLIM=YES
   .
ITPIDC9  APPL  EAS=1,SESSLIM=YES
*WTM APPLICATIONS FOR LU SESSION SIMULATION
VAPPL00  APPL  EAS=1
VAPPL01  APPL  EAS=1
VAPPL02  APPL  EAS=1
   .
VAPPL09  APPL  EAS=1
*WTM APPLICATIONS FOR CPI-C SIMULATION
APPL0    APPL  APPC=YES
APPL1    APPL  APPC=YES
APPL2    APPL  APPC=YES
   .
APPL9    APPL  APPC=YES
```

The VTAM application major node must be activated.

Each WSim and WTM user must reference a unique name from the list of APPL statements for the display monitor (WTMDM0#) and interactive data capture (ITPIDC#).

29.2.2 Providing access to WTM

There are several ways to provide users with access to WTM. An entry point can be added to one of the ISPF menu panels to allow general access on the system, or individual access can be provided for the users defined to WTM.

A copy of the REXX procedure WSIMTM, which can be found in **. SITPEXEC, is used in both cases:

- For global access, concatenate a data set with a copy of the REXX procedure WSIMTM to SYSPROC DD.
- For individual access, make a copy of the REXX procedure WSIMTM into a CLIST with your own high-level qualifier.

The copy of WSIMTM must be updated with correct local WSim data set names. Otherwise, WTM does not work.

The WSIMTM concatenates the following data sets:

- **.SITPPNL to ISPPLIB DD
- **.SITPMSG to ISPMLIB DD
- **. SITPEXEC to SYSEXEC DD
- **. SITPTBL to ISPTLIB DD
- userid.WTMUSER.SKELS to ISPSLIB DD

29.2.3 Setting up a new WTM user

Perform the following steps for every new user of WTM:

1. Make a copy of the member ITMNUSER from the supplied **.SITPSAMP data set and modify it as follows:
   a. Add a job card.
   b. Change the user ID to match the user being set up.
   c. Change high level index to the correct local qualifier.
2. Submit the job to allocate the following four files:
   – userid.WTMUSER.IDCDFLTS - The WSim Interactive Data Capture (IDC) defaults file
   – userid.WTMUSER.IDCLOG - The WSim IDC log partitioned data set
   – userid.WTMUSER.SKELS - A copy of the WTM SKELS data set
   – userid.WTMUSER.TABLES - A data set that contains the tables used or created by WTM
3. Log on as the user being set up and go to the WTM main panel to finalize WTM settings. The WTM main panel is presented, as shown in Figure 29-2.

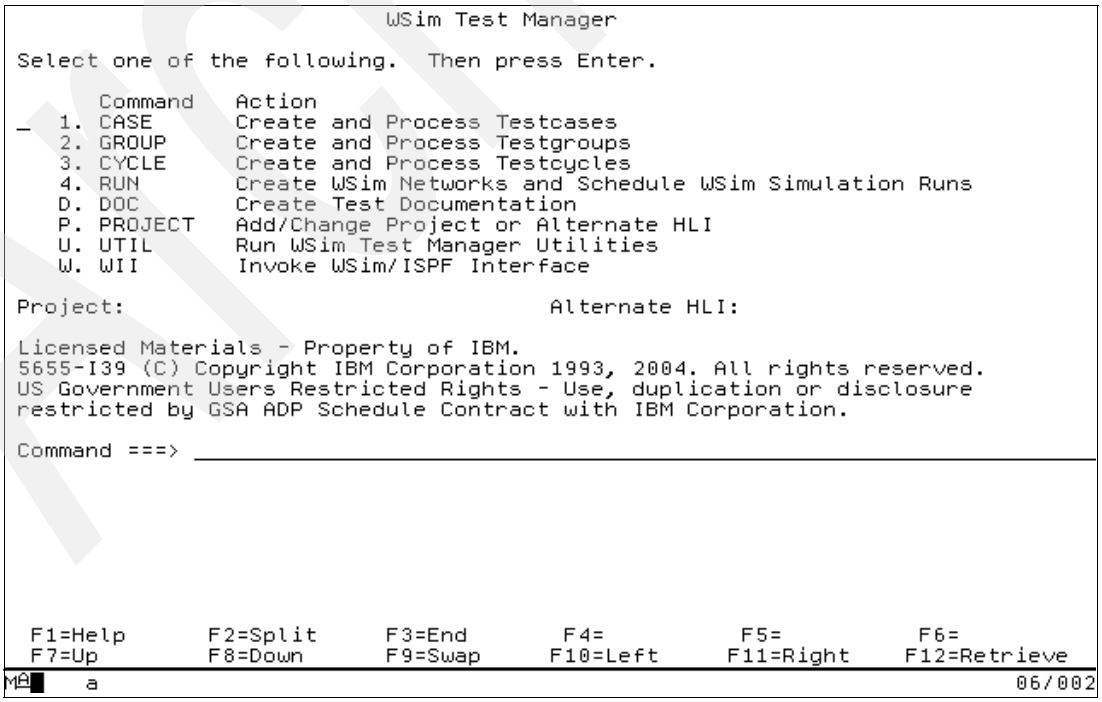


Figure 29-2 WTM main panel

From the WTM main panel:

– Move the cursor to the command line and enter the command **vars** or first select option U to get to the WTM Utilities panel, and there select option 1. On the presented panel, a number of default values are shown. Several missing fields must be populated to finish the setup.

  The WTM Variables and Options panel is shown in Figure 29-3.

– The values in the fields IDC VTAM APPL name and Display Monitor VTAM APPL name must be chosen from the corresponding values defined in the VTAM application major node definition. The value in the field VTAM Name Model must refer to the pool of values defined in the VTAM application major node definition.

– Another very important field on this panel is Automatic REFRESH. If set to Y, the REFRESH command runs every time you log on. This helps to ensure that shared projects are kept up-to-date.

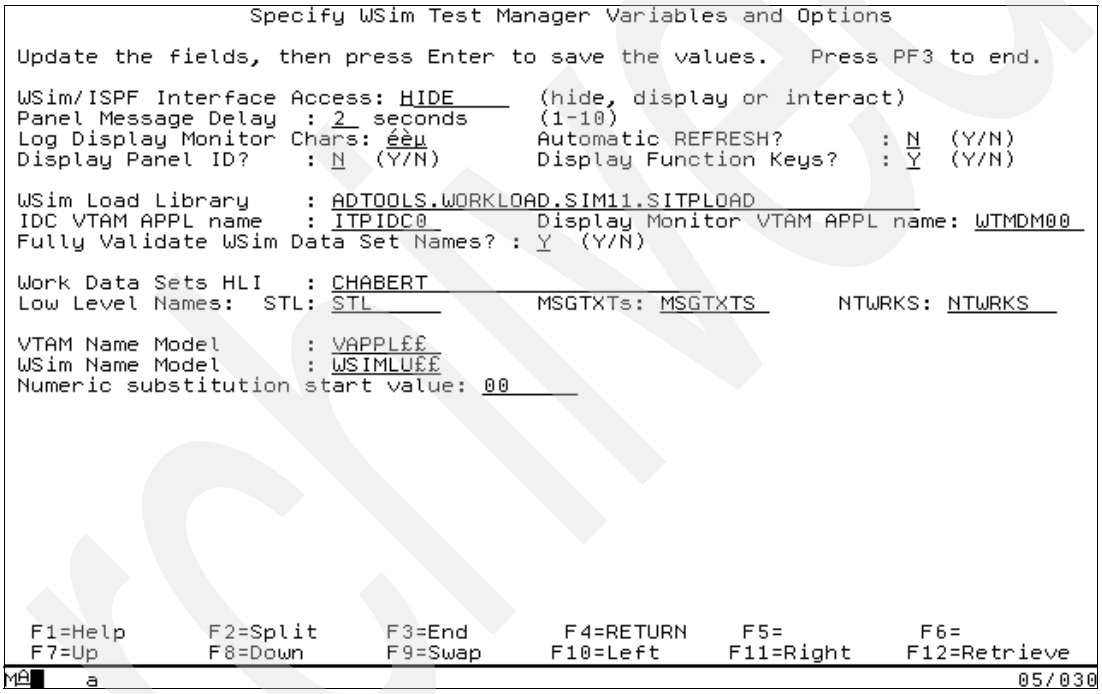


Figure 29-3 WTM Variables and Options panel

4. Press Enter to save the values. The basic WTM install is finished.

Additional values on the WTM Variables and Options panel must be supplied later.

29.2.4 Latest enhancements

To use the latest enhancements for the Workload Simulator, apply the PTF that fixes APAR PQ94132 (UQ93346).

Passwords are generally maintained on formatted 3270 screens in unprotected non-display fields. While not visible, the passwords are sent in the clear to host application programs. This enhancement masks potential passwords (it encrypts or hides them using asterisks) entered by users of the WSim data capture and script generation utilities, and masks their presence in generated WSim scripts, simulation data views, and output reports.

To use the 3270 password masking with WSim utilities and reports, run the job with the ITPGNKYZ utility, which generates a SMP/E USERMODE that sets a site-unique encryption key and initialization vector value.

To use the WSim Adapters for Rational TestManager, which allow WTM existing schedules and custom JCL scripts to be launched from the Rational TestManager running on a remote workstation, complete the following steps:

1. On the host system:
   a. Run the ITMSEQ CLIST member to apply changes to all schedules to be used by the Rational TestManager (for example, TSO EXEC 'HLI.SITPEXEC(ITMSEQ)').
   b. From the WTM main panel enter the command **vars**, and press PF3 when the Variables and Options panel is displayed. A file VARSINFO is created.
   c. From the WTM main panel enter the command **project**, and press PF3 when the Process Projects panel is displayed. A PROJLIST file is created.
2. On every remote workstation to be used:
   a. Run the install program WSimAdaptersRTMinstall.exe, which comes with the PTF.
   b. Register the WSim Adapters as new test script types with the Rational TestManager.
   c. Create new local test script sources for WTM schedules and JCL scripts.

30

Workload Simulator Test Manager

The Workload Simulator Test Manager (WSim Test Manager, or WTM) is a usability enhancement that provides guidance through the test process. WTM offers selectable modes of operation, test management services, automatic script generation, and task automation.

In this chapter we describe how WTM can be used. We step through the process of preparing simulations using Interactive Data Capture (IDC), running simulations, and viewing reports that are provided by WTM.

We also discuss ways to organize test cases into test groups, and how to employ user tables to modify user inputs.

30.1 Concepts

The primary concept of the WTM testing structure is a *project*, which is a set of libraries containing schedules and test scenarios. Projects can be archived and reused. A project must be created before any schedules or test scenarios can be created using WTM.

Test scenarios are organized into three levels: a test case, test group, and test cycle. A test group is an ordered list of test cases. Test cases can be reused within multiple test groups. A test cycle is an ordered group of test groups and test cases.

WTM offers various ways to automate the development of test cases, which are WSim scripts written in Structured Translator Language (STL).

Generated test cases are paired with network resource definitions as part of developing WTM schedules. A WTM schedule is a WSim network definition and the associated test scenario definition. The WTM schedule is used by WTM to define and control the WSim simulation run (test). WTM schedules can be archived and reused.

Figure 30-1 shows how WTM helps to manage and organize tests.

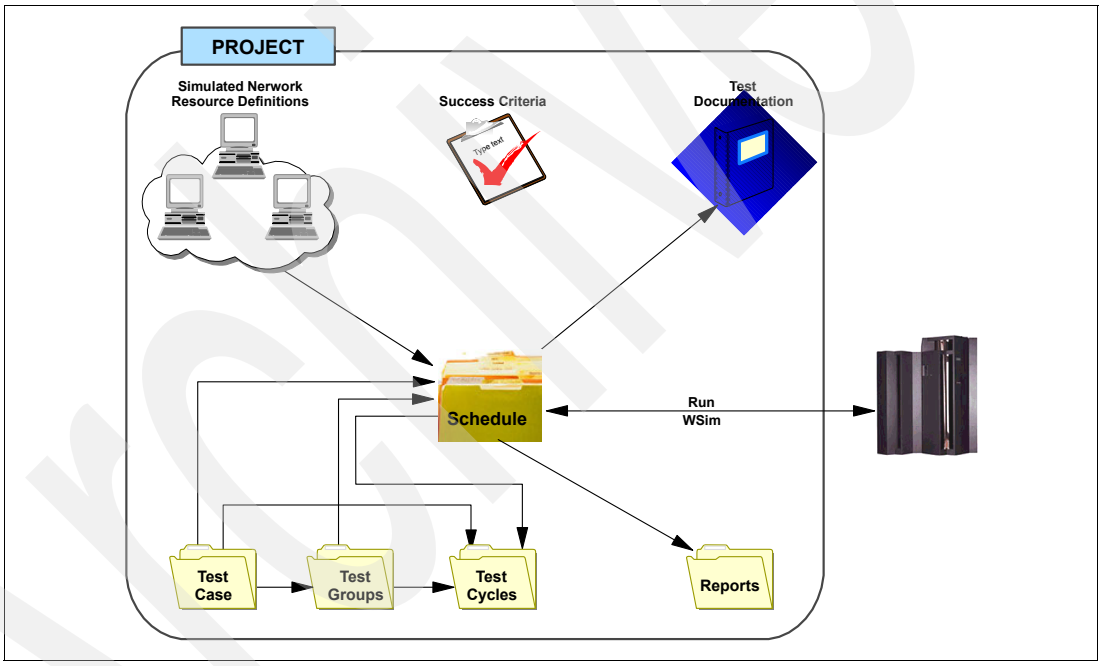


Figure 30-1 WTM helps to manage and organize tests

30.2 WTM operating modes

WTM has three modes of operation that control the amount of interaction you can have with WSim:

- **Hide mode (the default): Most of the WTM and WSim interactions are hidden.**
- **Display mode:** You can observe the entire WSim and WTM processes, but you have no direct control over them.
- **Interact mode:** You have control over WSim and WTM and must press Enter to start and execute various WSim utilities, and press the PF3 key to exit.

The operating mode can be entered on the command line (**hide**, **display**, or **interact**) of any WTM panel or by selecting the VARS option from the WTM Utilities panel shown in Figure 30-2.

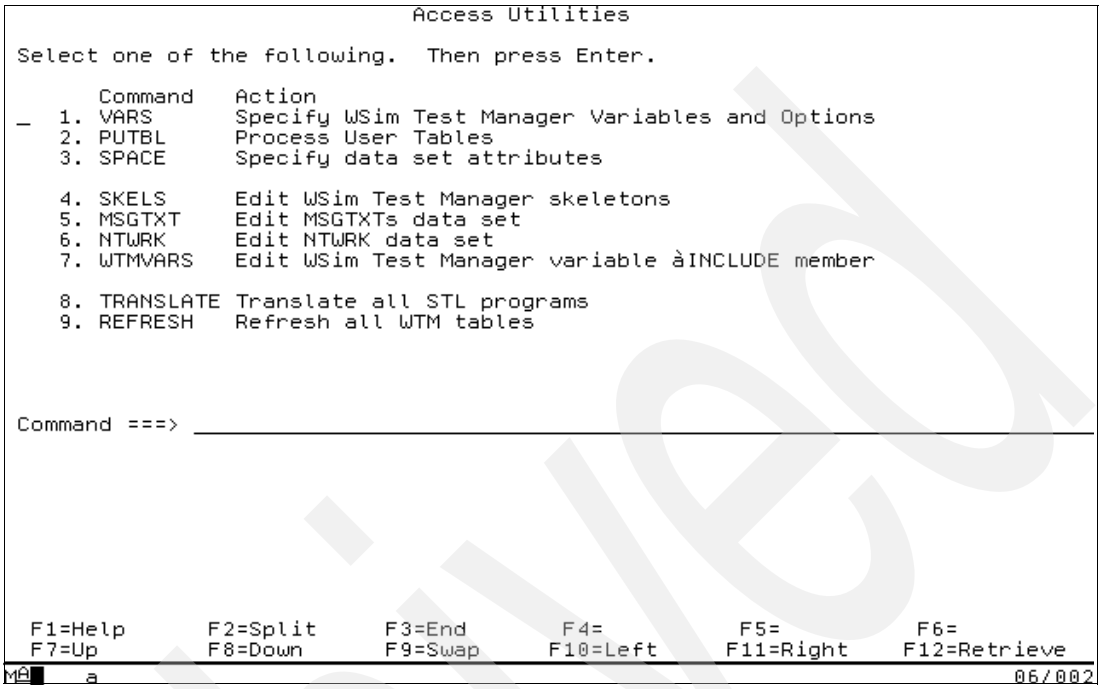


Figure 30-2 WTM Utilities panel

WTM also has a number of utilities for accessing and updating control information, skeletons, and data set attributes. STL translate and table refresh functions are also accessible. Select option **U** from the WTM main panel, or enter `UTILS` on the command line to access this panel.

There are two additional fields on the WTM Variables and Options panel that are important:

- Panel Message Delay: Specify the minimum time in seconds that a panel message is to be displayed.
- Automatic REFRESH: If set to Y, the REFRESH command runs every time you log on.

This helps to ensure that shared projects are kept up-to-date.

30.3 Creating a project

Perform the following steps to create a project:

1. To work with project definitions, select option **P** on the WTM main panel, as shown in Figure 30-3.

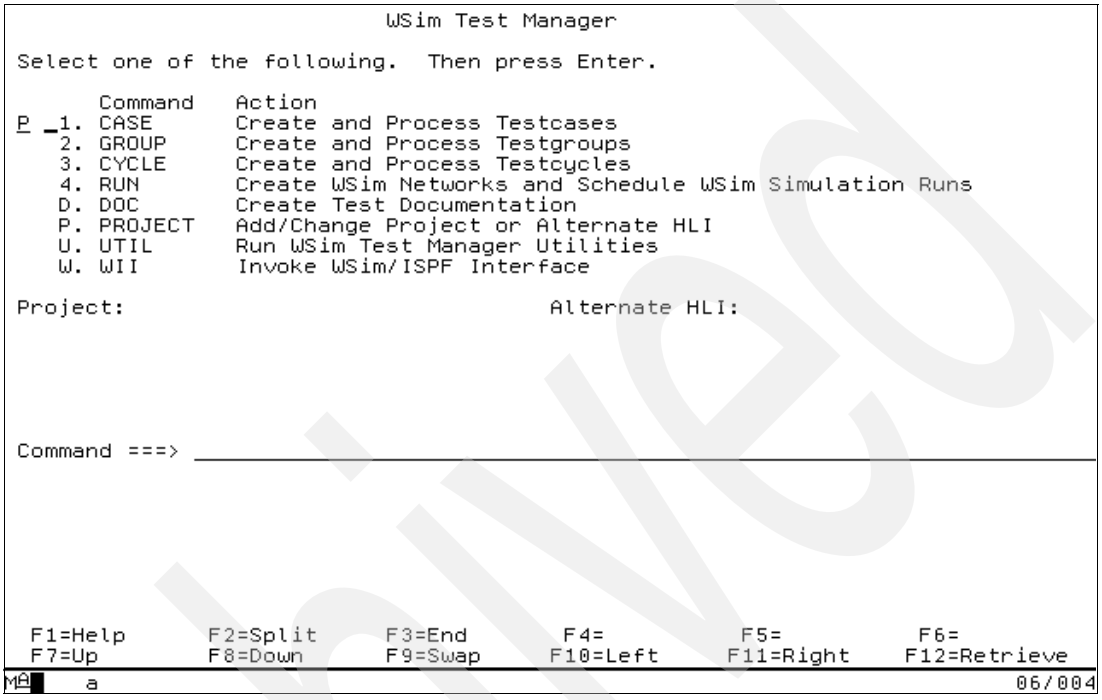


Figure 30-3 Option P to work with project definitions

2. To create a new project enter the command **add** on the next panel, as shown in Figure 30-4.

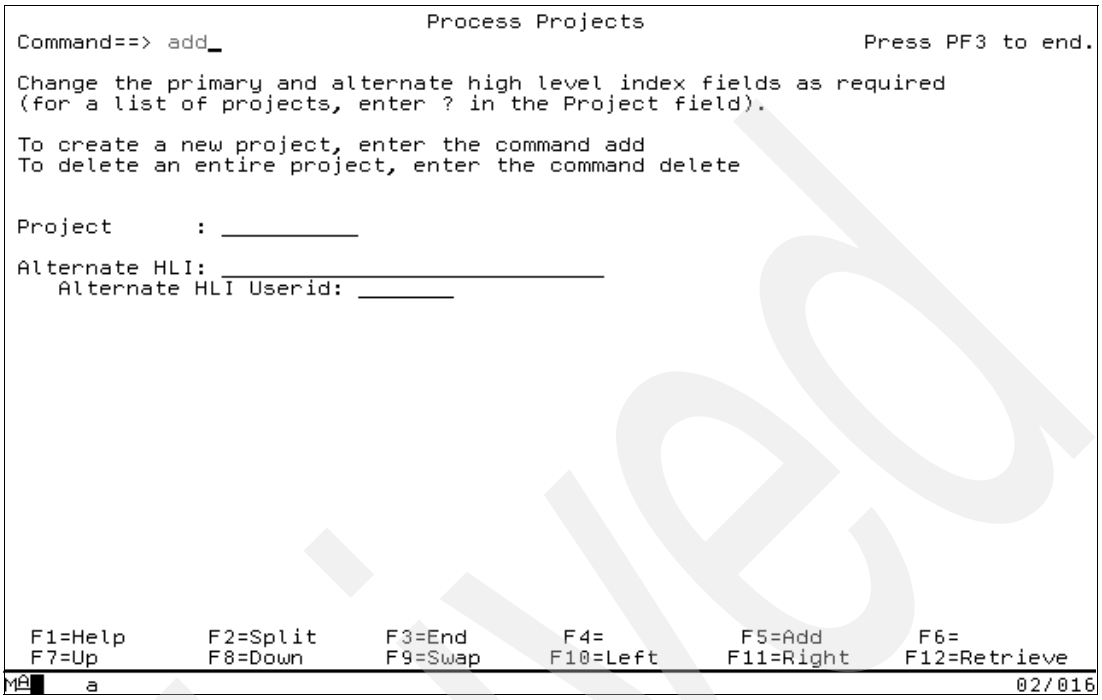


Figure 30-4 Command add new project

3. The next panel provides an opportunity to define the name for the project and high level index (HLI) and add the description, as shown in Figure 30-5.

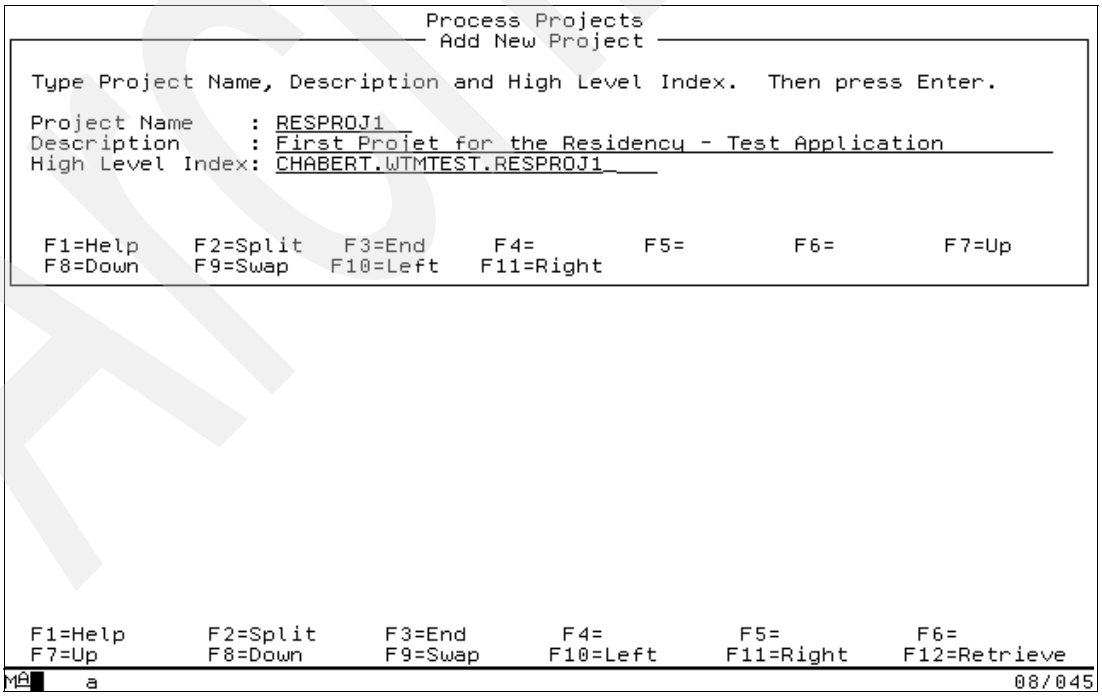


Figure 30-5 New project name, description, and HLI

4. Make sure that there are no existing data sets that have this high level index. Otherwise, WTM might write over the existing data sets. WTM creates project data sets and displays the confirmation, as shown in Figure 30-6.

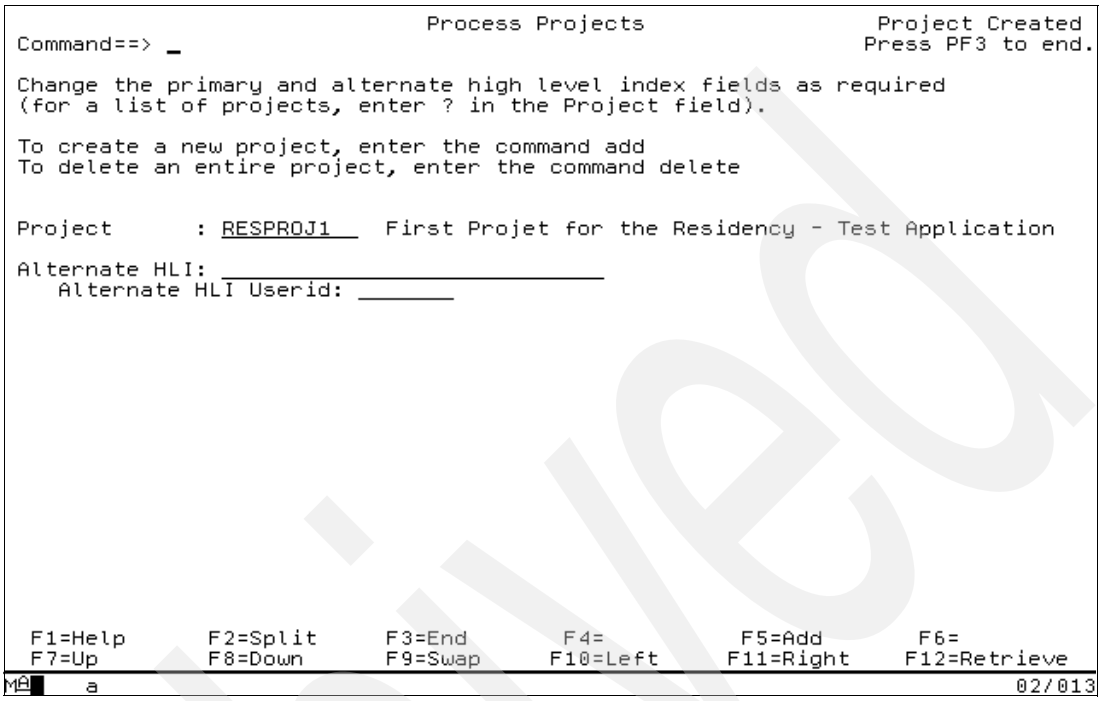
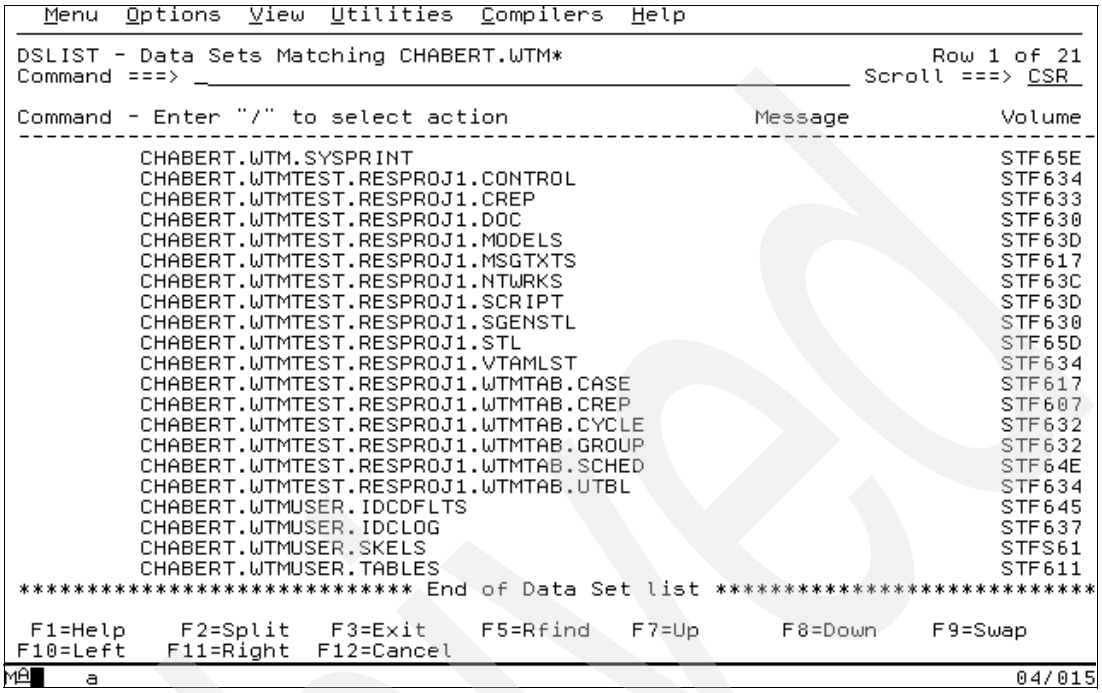


Figure 30-6 Project created

5. The alternate HLI (high level index) fields enable a user to use test items (test cases, MSGTXTS, and schedules) created under another high level index. These test assets can belong to the same user or somebody else. If an alternate high level index is specified, all test assets for that index are flagged with ALT. You cannot modify any of the ALT test assets since the data sets containing the ALT test assets are available in BROWSE mode only.

   Both the Alternate HLI and the Alternate HLI Userid fields must be completed to use this feature. Enter the high level index of the other project in the Alternate HLI field, and enter the owning user ID of the alternate high level index in the Alternate HLI Userid field.

6. Using ISPF option 3.4, you can see that WTM created 16 data sets (9 partitioned and 7 sequential) for the project RSPROJ1 with the names starting with the supplied HLI, as shown in Figure 30-7.


```
  Menu  Options  View  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
 DSLIST - Data Sets Matching CHABERT.WTM*                        Row 1 of 21
 Command ===> _____________________________________________ Scroll ===> CSR

 Command - Enter "/" to select action                 Message           Volume
 ------------------------------------------------------------------------------
          CHABERT.WTM.SYSPRINT                                          STF65E
          CHABERT.WTMTEST.RESPROJ1.CONTROL                              STF634
          CHABERT.WTMTEST.RESPROJ1.CREP                                 STF633
          CHABERT.WTMTEST.RESPROJ1.DOC                                  STF630
          CHABERT.WTMTEST.RESPROJ1.MODELS                               STF63D
          CHABERT.WTMTEST.RESPROJ1.MSGTXTS                              STF617
          CHABERT.WTMTEST.RESPROJ1.NTWRKS                               STF63C
          CHABERT.WTMTEST.RESPROJ1.SCRIPT                               STF63D
          CHABERT.WTMTEST.RESPROJ1.SGENSTL                              STF630
          CHABERT.WTMTEST.RESPROJ1.STL                                  STF65D
          CHABERT.WTMTEST.RESPROJ1.VTAMLST                              STF634
          CHABERT.WTMTEST.RESPROJ1.WTMTAB.CASE                          STF617
          CHABERT.WTMTEST.RESPROJ1.WTMTAB.CREP                          STF607
          CHABERT.WTMTEST.RESPROJ1.WTMTAB.CYCLE                         STF632
          CHABERT.WTMTEST.RESPROJ1.WTMTAB.GROUP                         STF632
          CHABERT.WTMTEST.RESPROJ1.WTMTAB.SCHED                         STF64E
          CHABERT.WTMTEST.RESPROJ1.WTMTAB.UTBL                          STF634
          CHABERT.WTMUSER.IDCDFLTS                                      STF645
          CHABERT.WTMUSER.IDCLOG                                        STF637
          CHABERT.WTMUSER.SKELS                                         STFS61
          CHABERT.WTMUSER.TABLES                                        STF611
 ****************************** End of Data Set list ****************************

  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
 MA    a                                                          04/015
```

Figure 30-7 List of project data sets

WTM requests a table refresh. A normal table refresh is sufficient in most cases. You must select an extended table refresh only if changes were made to the generated STL scripts or network definition statement.

30.4 Test cases

Test cases are STL programs with supporting documentation and User Data Tables (UTBLs). Select option **1** or enter the command `CASE` from the WTM main panel to see the list of test cases for the current project. This list consists of the following columns:

- Command code: A one-character command field (valid values are D, N, O, T, U, and V).
- Name: The name of the test case and the member name of the corresponding STL program.
- Type: The type of the test case. Filled by WTM. The values are V for VTAMAPPL, C for CPI-C, T for TCP/IP, and *-* for undefined, for example, a skeleton or an STL program created outside of WTM.
- UTBLs: The number of UTBLs for this test case.
- Notes: An asterisk (*) indicates that there are notes for this test case.
- Description: This is an optional free-form description of the test case.

For each test case, the following commands can be entered in the command field:

D Delete: Delete the test case and the associated members in WTM libraries.

N Notes: Edit the Notes data set for this test case.

O Open: Edit the STL source for this test case. WTM automatically translates the changed STL program into a message generation deck (MSGTXT).

T Translate: Translate the STL source code into MSGTXT.

U UTBLs: Display the list of UTBLs for this test case. The STL source for UTBLs in the list can be edited or browsed.

V View: If the original WSim or IDC log has not been overwritten, this option shows some or all of the screen images for this test case.

To create a new test case, enter the command **add** or press PF5, as shown in Figure 30-8.

```
                              Process Testcases
Command==> add_
Enter a line command or add to create a new Testcase.          Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
****************************** Bottom of data ********************************




 F1=Help      F2=Split     F3=End       F4=          F5=Add       F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA  a                                                                 02/016
```

Figure 30-8 Adding a new test case

There are six ways to create a new test case, as shown in Figure 30-9:

- Use the Interactive Data Capture (IDC) utility to actually log on to the application and capture your session.
- Use a System Network Architecture (SNA) trace file that was previously saved.
- Use a log file that was generated during one of the previous runs of the IDC utility.
- Use one of the WSim model scripts.
- Use one of the STL skeleton scripts.
- Use an SNA trace file that was previously saved into a Common Programming Interface for Communications (CPI-C) script.

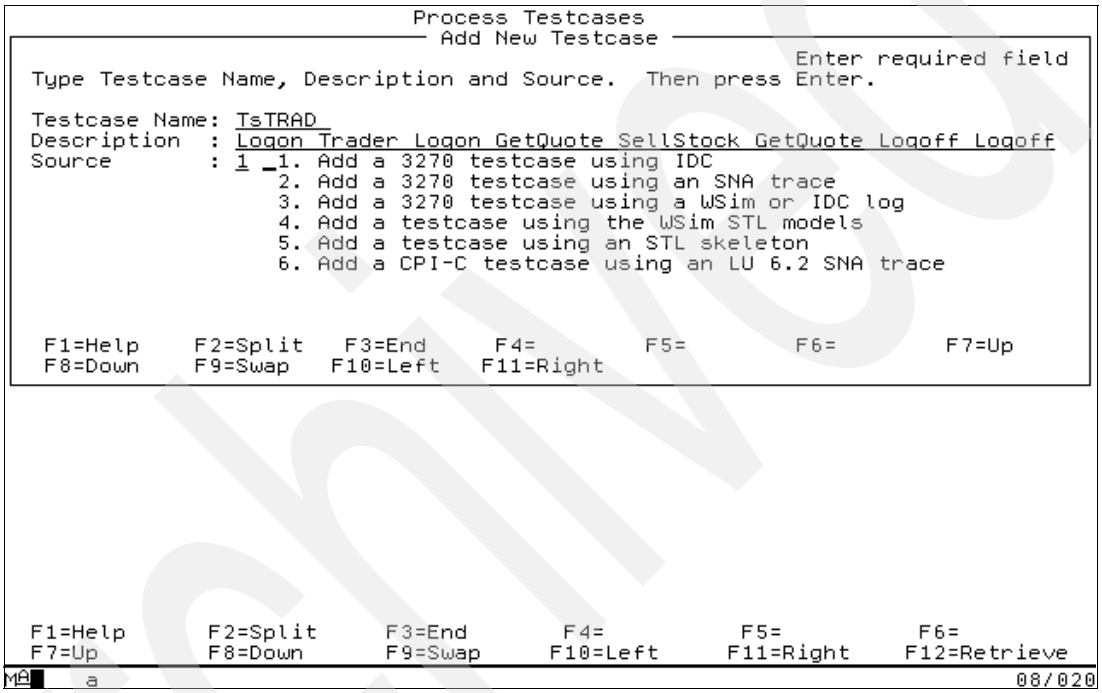


Figure 30-9 Add New Testcase panel

Now we describe how to create a new test case, TSTTRAD, using the IDC utility. The test case does these tasks:

1. Connect to CICS region CICSC23G. This is the corresponding VTAM name. Refer to the parameter APPLID of the CICS system.
2. Go to CICS native mode.
3. Start COBOL/DB2/VSAM transaction TDB2.
4. Log on to the trader application.
5. Select a company.
6. Get a quote.
7. Buy some stock.
8. Get a quote.
9. Log off from the trader application.
10. Log off from CICS using the cesf transaction.

During the capture process, WTM displays some “Milestones” panels to show steps and the status of WSim tasks. The selection of the WTM operating mode affects what WTM displays:

- Hide: Only “Milestones” panels.
- Display: WSim panels also.
- Interactive: Interaction with the WSim panels is possible.

The capture described in this section is performed in the Hide mode.

Use the following steps to create a test case using the IDC utility, as shown in Figure 30-10.

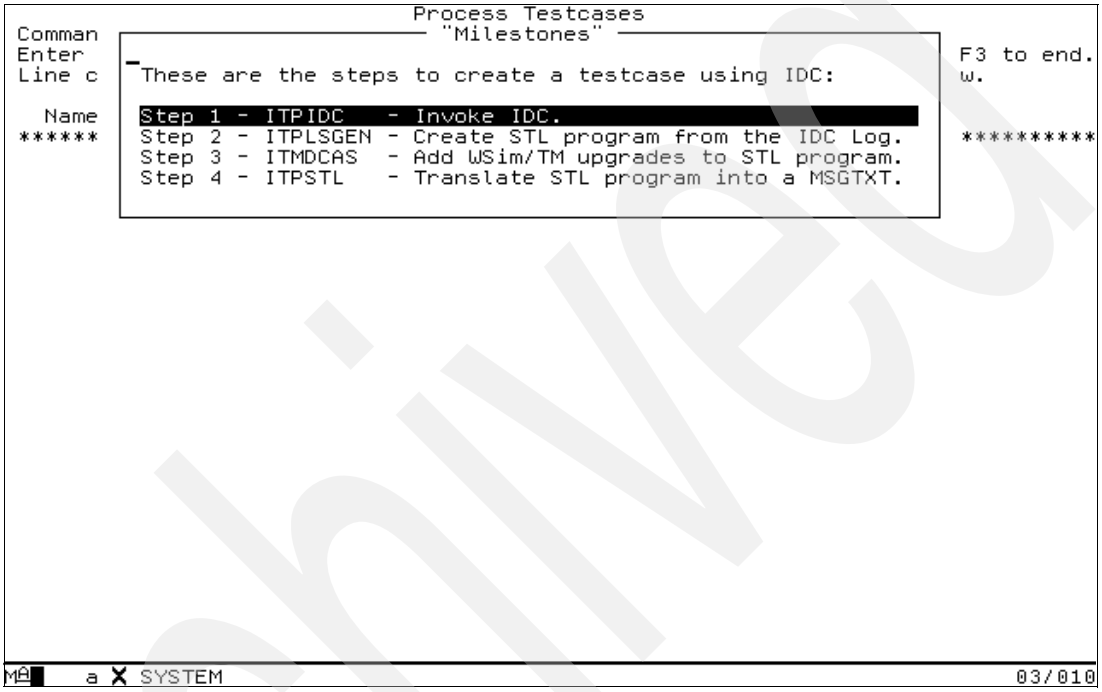


Figure 30-10 Test case creation steps when using IDC utility

The steps are:

1. Start a session with a host application to capture data, as shown in Figure 30-11.

```
IDCMAIN          WSim Interactive Data Capture (IDC) Utility

Select one of the following, then press Enter.

1 1. Start a session with a host application and capture data

  2. Generate an STL program from captured data
  3. Generate a message generation deck from captured data

  4. End the IDC utility program




WSim Version 1 Release 1.0.1  Program Number 5655-I39

Licensed Materials - Property of IBM
5655-I39 (C) Copyright IBM Corporation 1976, 2004.  All Rights Reserved.
US Government Users Restricted Rights - Use, duplication or disclosure
restricted by GSA ADP Schedule Contract with IBM Corporation.
F1=Help  F3=Exit  F12=Cancel
MA    a                                                        05/004
```

Figure 30-11 Starting the first step to capture data

2. Press Enter and the WSim IDC start panel is displayed, as shown in Figure 30-12.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . CICSC23G
  Logon mode name . . . . . . . . . . LSX32703 (Optional)
  Logon user data . . . . . . . . . . ________________________ (Optional)


IDC log data set name . . . . . . . . CHABERT.WTMUSER.IDCLOG(TSTRAD)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . Y        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1      (PAn, PFnn, CLEAR, or ATTN)









F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
MA    a                                                        07/040
```

Figure 30-12 WSim IDC start panel

3. Connect to the application CICSC22F. The IDC log is captured in the member TSTTRAD (which is replaced if it existed earlier), data capture starts immediately, and an IDC escape key is defined as PA1. Press Enter. The CICS sign-on panel is displayed, as shown in Figure 30-13.

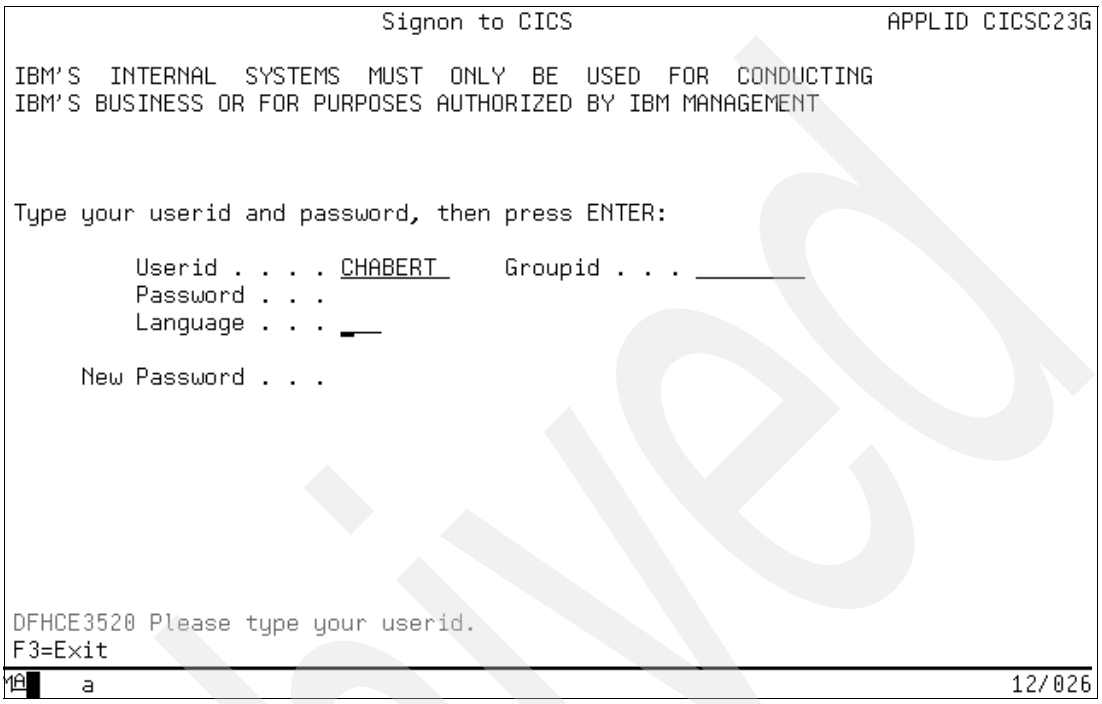


Figure 30-13 CICS sign-on panel

4. The user ID and the password are keyed in and sign on is completed, as shown in Figure 30-14.

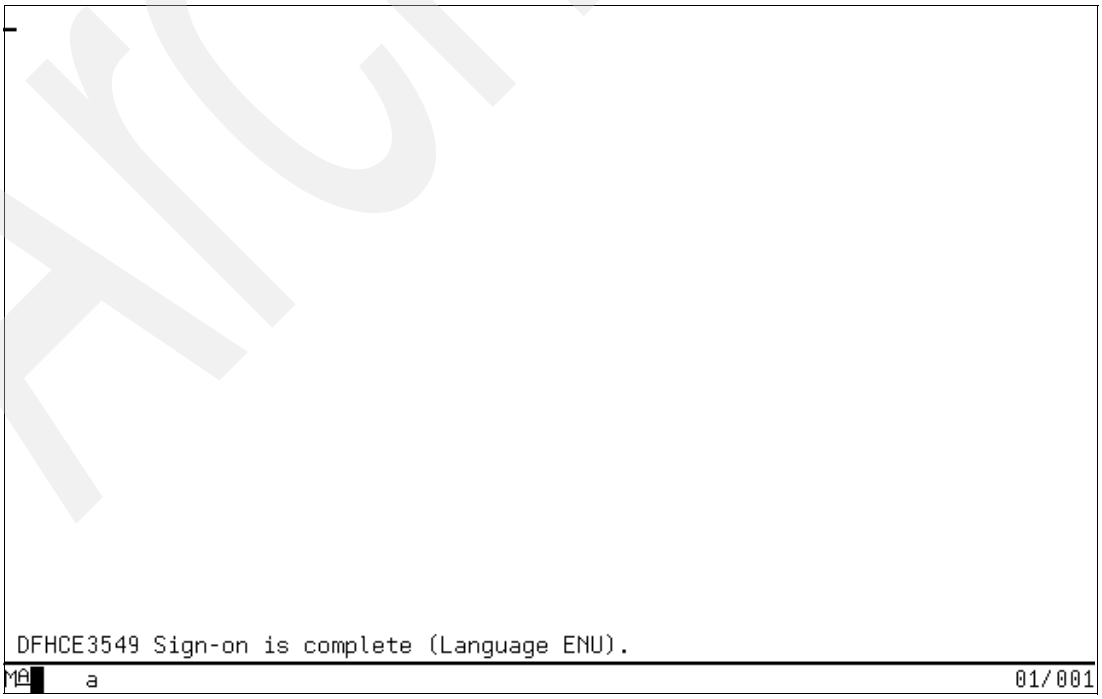


Figure 30-14 CICS sign-on is complete

5. Start the TDB2 transaction, which provides you with several options, as shown in Figure 30-15.

Figure 30-15 MYTD transaction entered

6. The trader application is started. Sign on, supplying a user name and password, as shown in Figure 30-16.

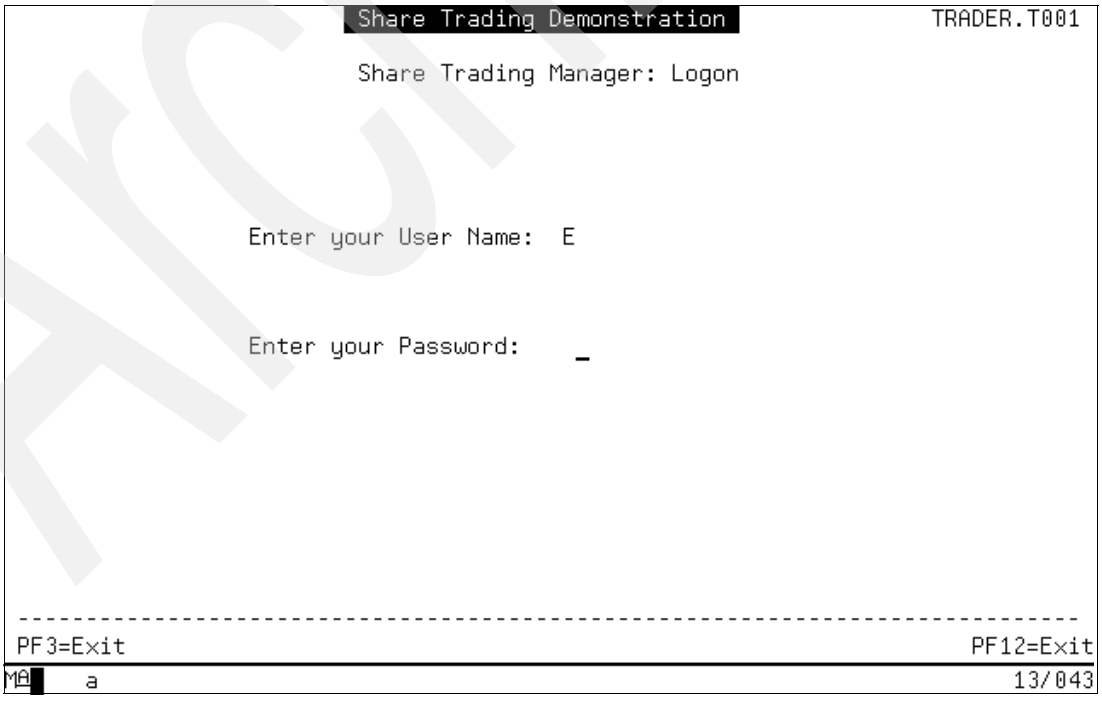


Figure 30-16 Trader application logon panel

7. The company selection panel of the trader application is displayed, as shown in Figure 30-17. Company number 1 is selected.

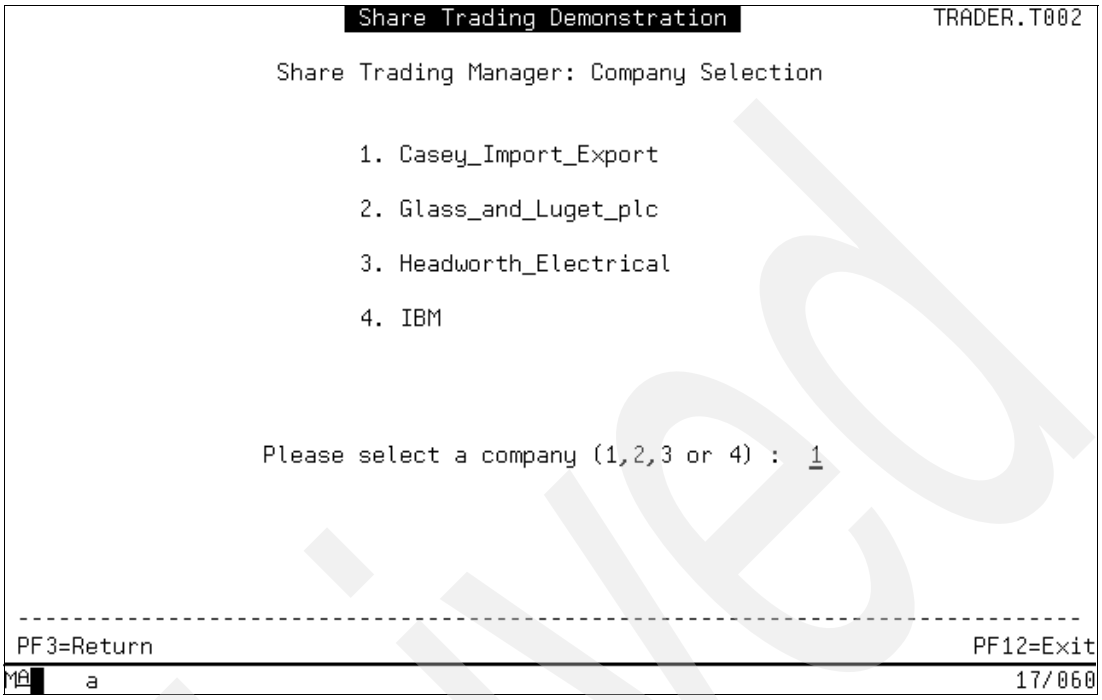


Figure 30-17 Trader application company selection panel

8. Option 1 for a real-time quote is chosen, as shown in Figure 30-18.

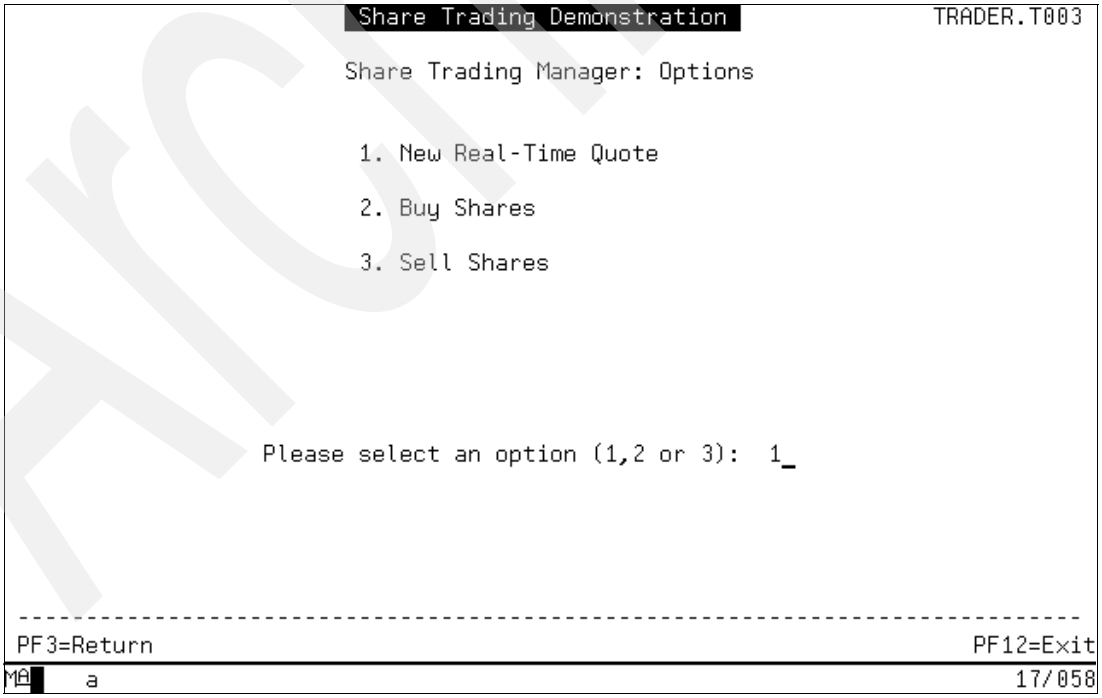


Figure 30-18 Trader application options panel

9. The panel with the results of the query is presented, as shown in Figure 30-19.

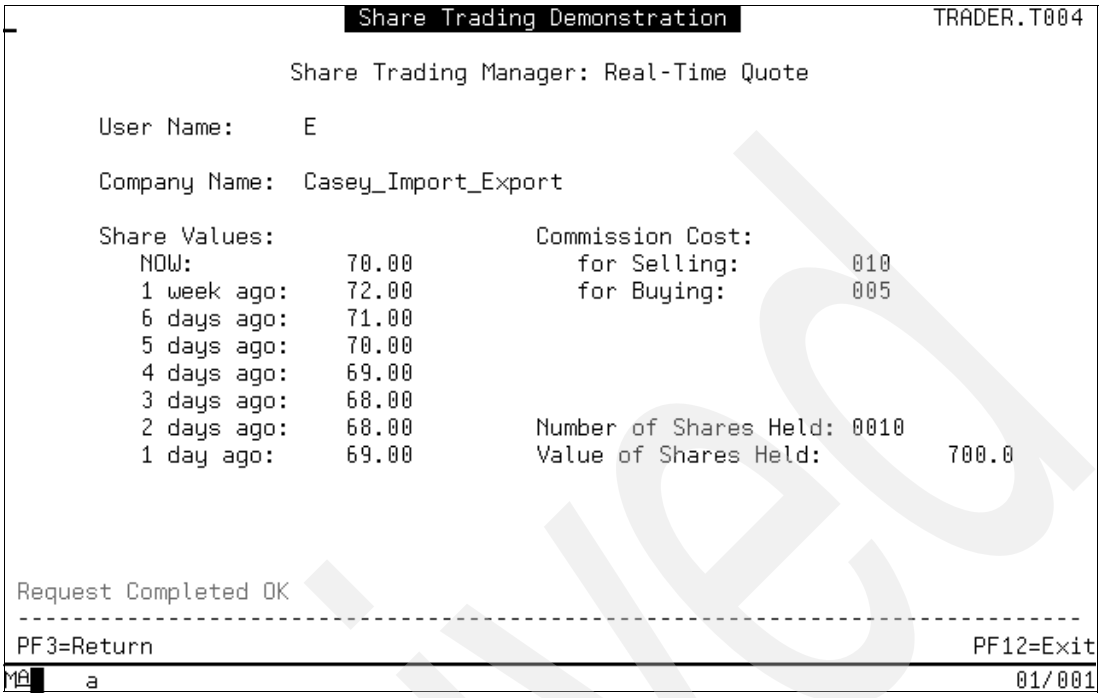


Figure 30-19 Trader application quote results

10. After pressing PF3 to return to the Options panel of the trader application, select the option to buy some shares, as shown in Figure 30-20.

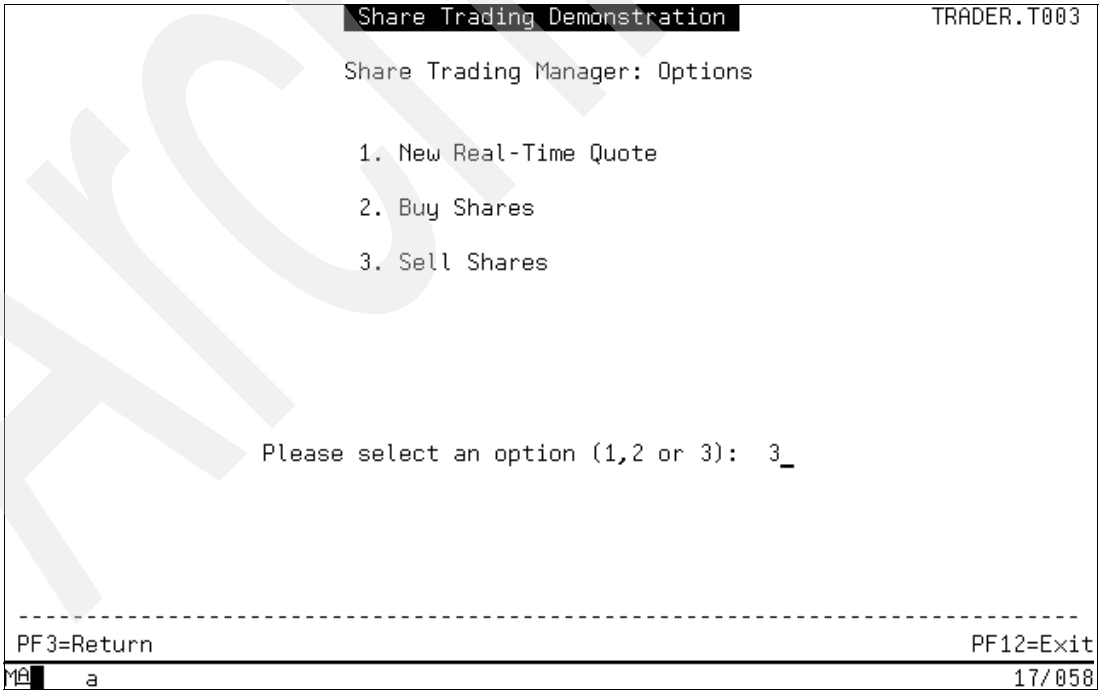


Figure 30-20 Trader application option 3 selected

For this example, this is 1 share, as shown in Figure 30-21.

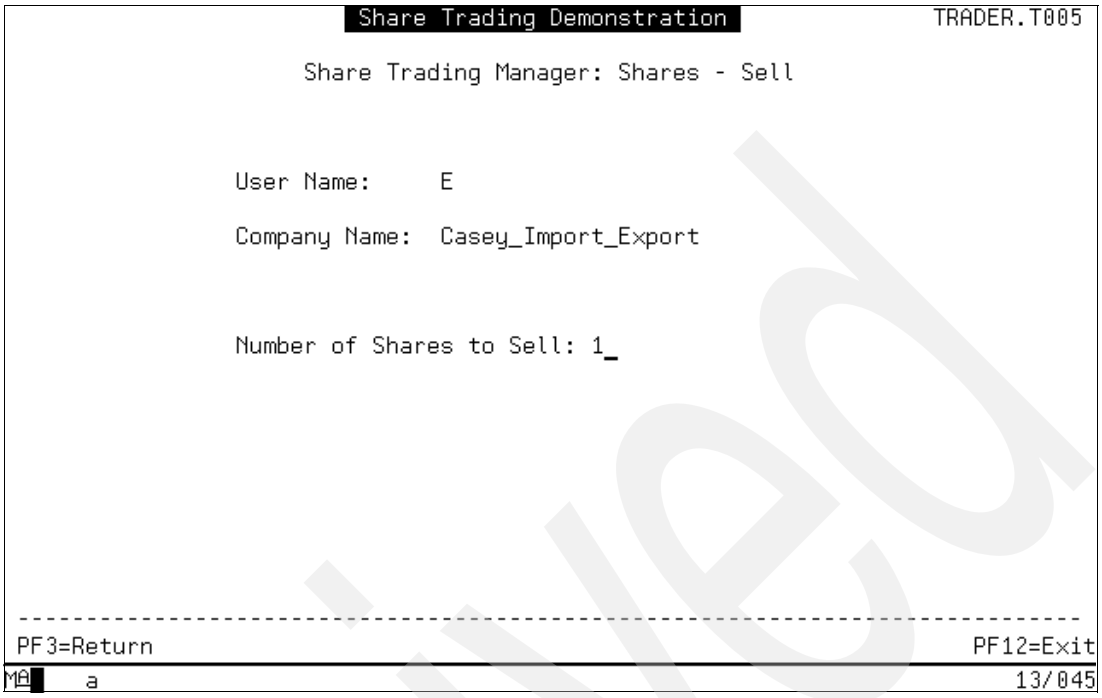


Figure 30-21 Trader application selling shares

11. Issue a second request for a real-time quote, as shown in Figure 30-22.

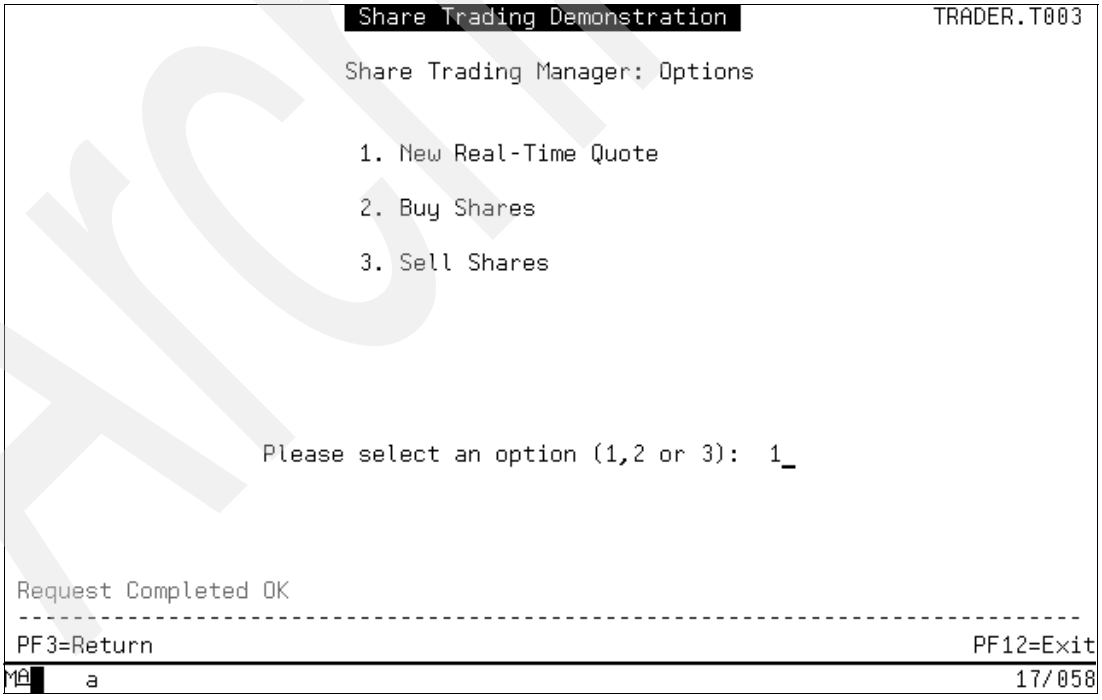


Figure 30-22 Trader application quote request

12. The results confirm and the number of shares held is now 9, as shown in Figure 30-23.

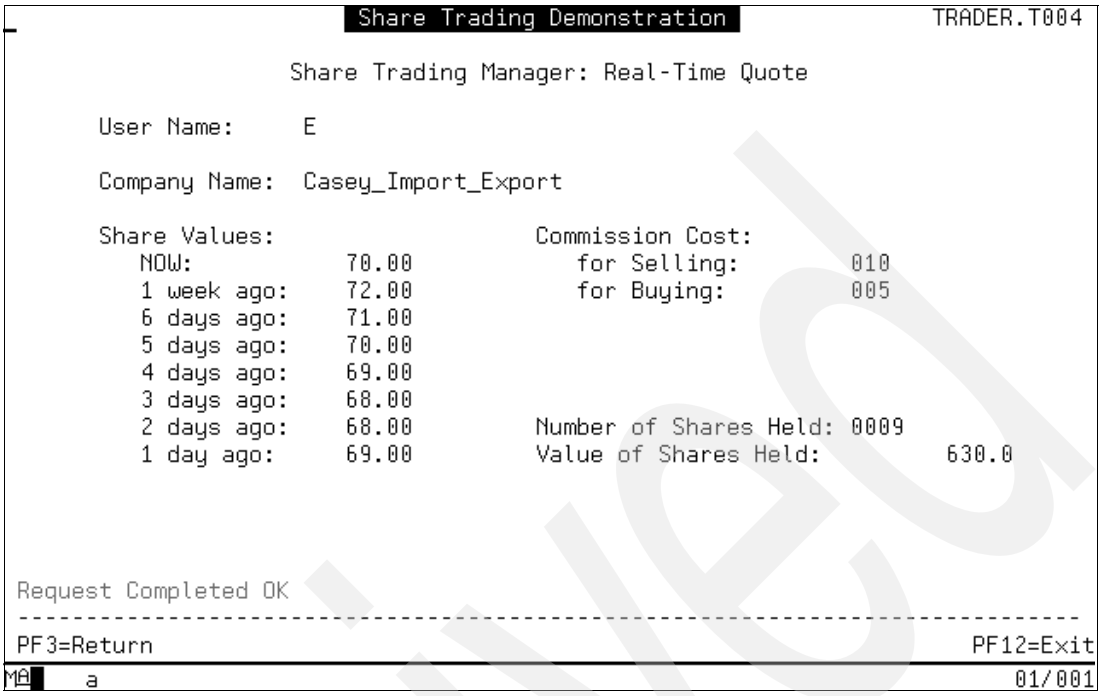


Figure 30-23 Trader application new quote results

13. To finish working with the trader application press PF12. The session is over, as shown in Figure 30-24.

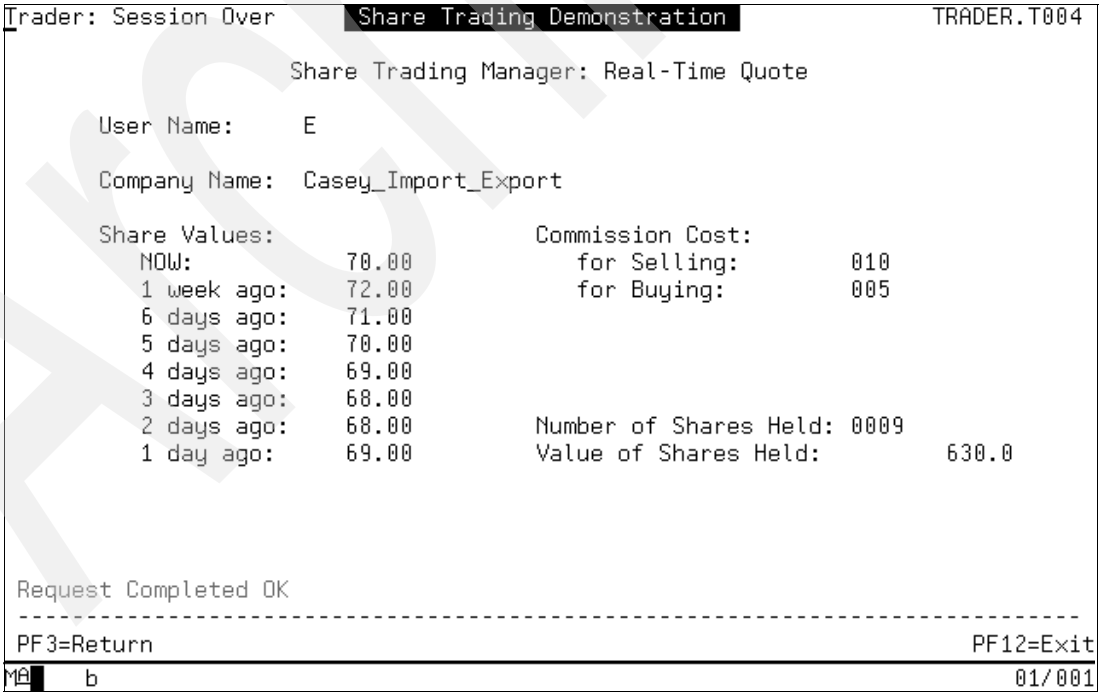


Figure 30-24 Trader application session over

14. Transaction cesf is entered to exit CICS, as shown in Figure 30-25.

Figure 30-25 CICS transaction CESF entered

15. The CICS session is finished with the message shown in Figure 30-26.

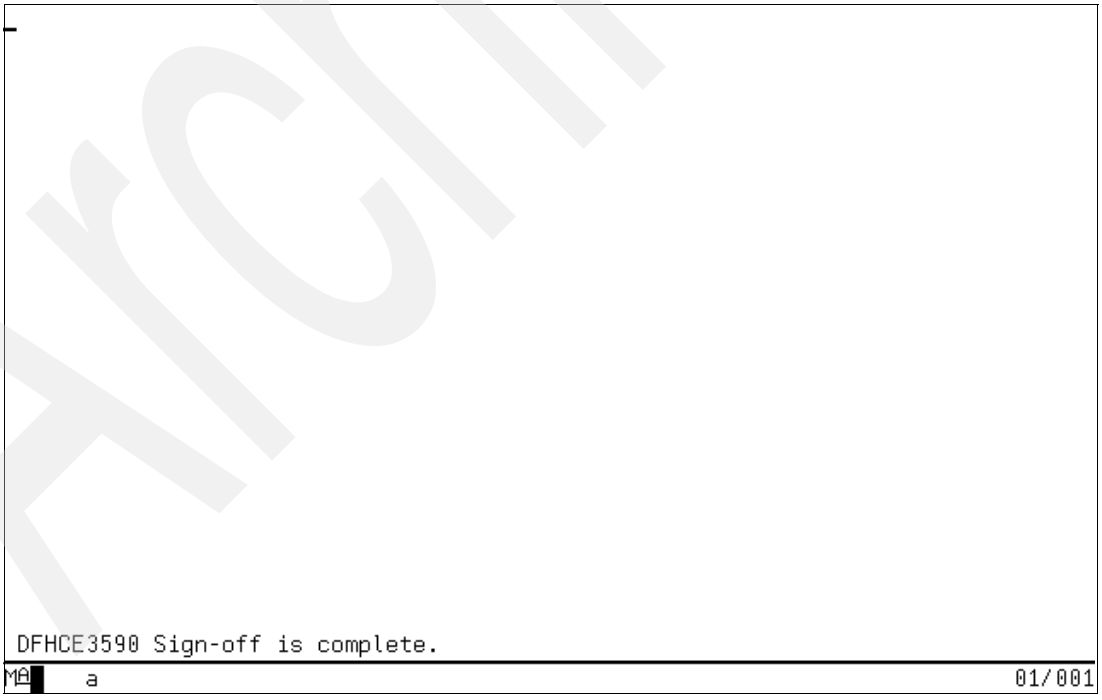


Figure 30-26 CICS session ended

16. By pressing PA1, which was defined as the IDC control key, you are returned to the WSim IDC utility Escape Actions panel. The choice is made to finish the session with the host application, as shown in Figure 30-27.

```
IDCESCA             WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

3 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : ON
Current IDC log data set: CHABERT.WTMUSER.IDCLOG(TSTRAD)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
MA    a                                                        06/004
```

Figure 30-27 Escape actions panel option 3 selected

17. The WSim IDC utility confirms the end of the session with CICSC22F, as shown in Figure 30-28.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . CICSC23G
  Logon mode name . . . . . . . . . . LSX32703 (Optional)
  Logon user data . . . . . . . . . . ________________________ (Optional)


IDC log data set name . . . . . . . . CHABERT.WTMUSER.IDCLOG(TSTRAD)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . Y        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1      (PAn, PFnn, CLEAR, or ATTN)









ITP1508I SESSION ENDED WITH APPLICATION CICSC23G
F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
MA    a                                                        06/040
```

Figure 30-28 Session ended message

18. WTM completes the remaining steps, as shown on the Milestones panel (Figure 30-10 on page 960) and the new test case TSTRAD is created. The line command **v** is entered to view the test case, as shown in Figure 30-29.

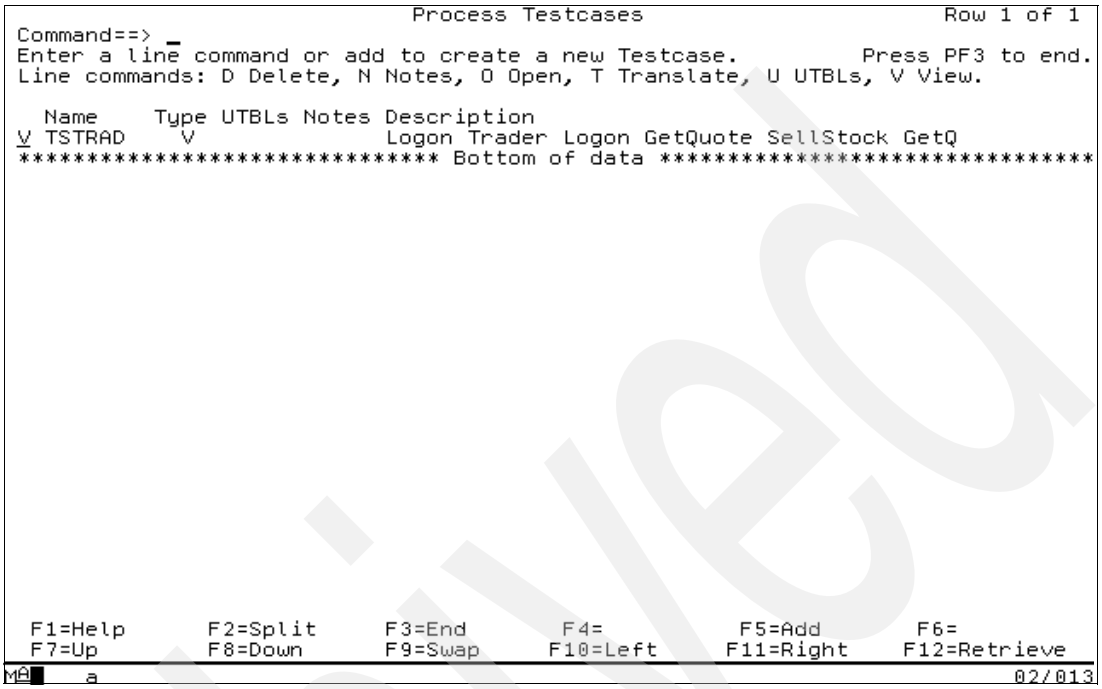


Figure 30-29 View test case command

The test case TSTRAD contains 29 display panels. Figure 30-30 shows the WTM panel that presents the choice to view the test case as a slide show with a logged or fixed time for each panel to be displayed, or to view just the index of the panels. The index has been selected in this example.

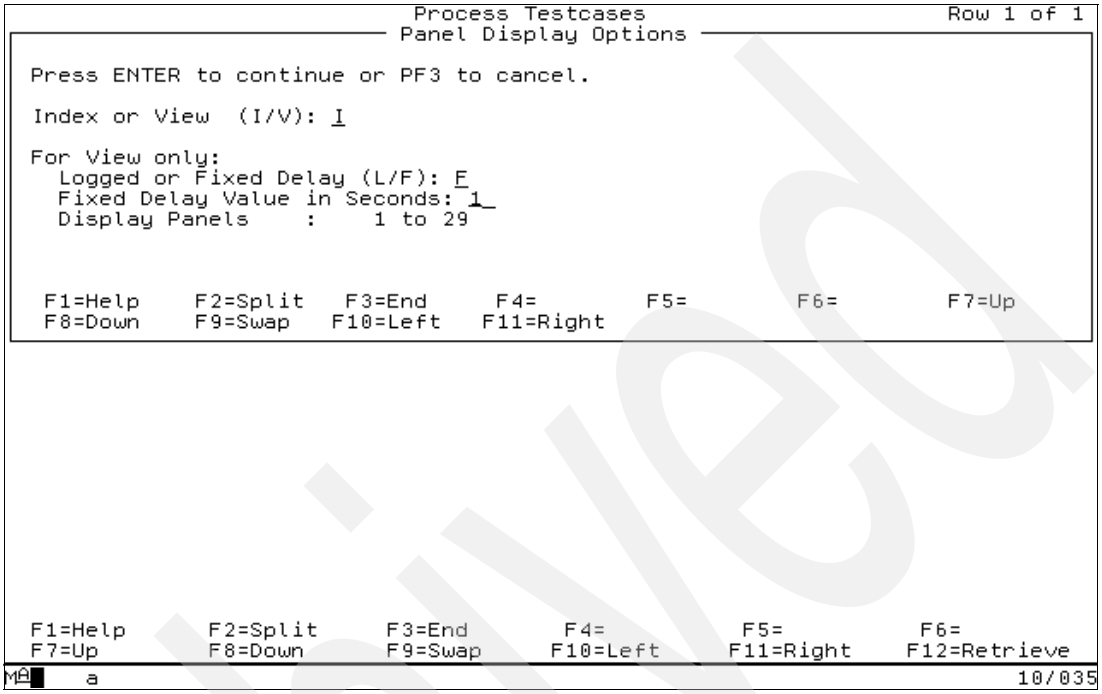


Figure 30-30 Index of display panels requested

19. The index of panels in the test case TSTRAD is presented in two parts, shown in Figure 30-31 and Figure 30-32.

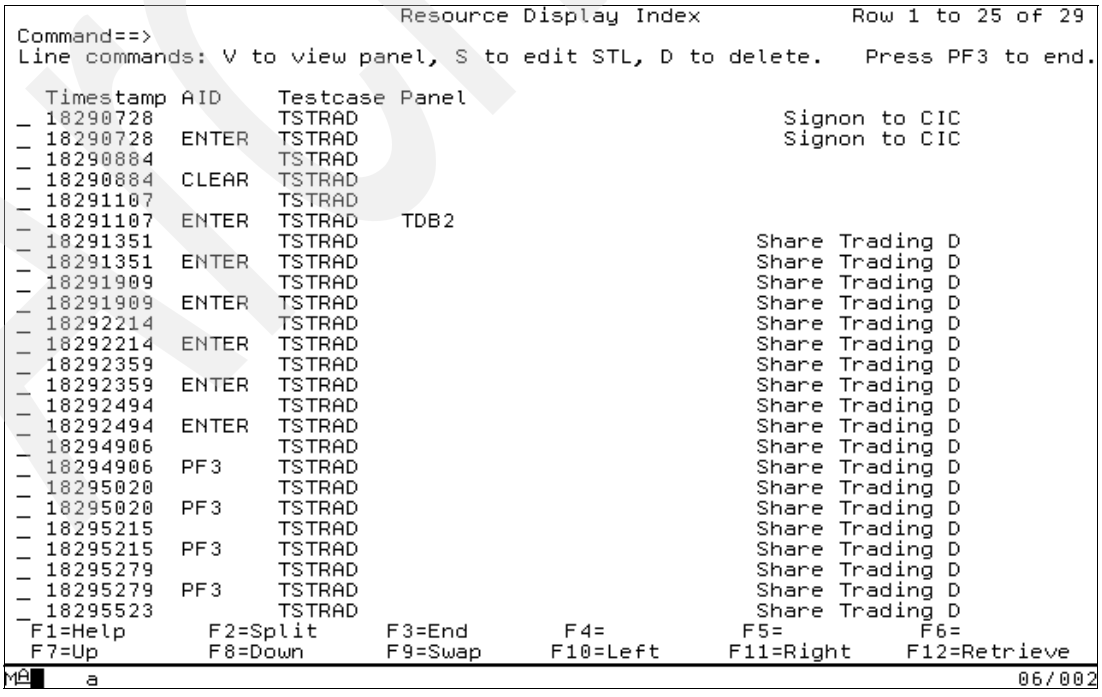


Figure 30-31 Index panels for test case (1 of 2)

```
                          Resource Display Index          Row 26 to 29 of 29
Command==> _
Line commands: V to view panel, S to edit STL, D to delete.   Press PF3 to end.

  Timestamp AID     Testcase Panel
_ 18295523  CLEAR   TSTRAD
_ 18300720          TSTRAD
_ 18300720  ENTER   TSTRAD   cesf
_ 18310919          TSTRAD
****************************** Bottom of data ********************************

 F1=Help      F2=Split     F3=End       F4=          F5=          F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA    a                                                              02/013
```

Figure 30-32 Index of panels for test case (2 of 2)

20. You can view the panels or the generated STL program using line commands. The beginning of the STL program is shown in Figure 30-33.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
EDIT       CHABERT.WTMTEST.RESPROJ1.STL(TSTRAD) - 01.01    Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
000001 àprogram=TSTRADT
000002 àinclude wtmvars
000003 TSTRAD: msgtxt
000004 /*-------------------------------------------------------------------------*/
000005 /* ITPIDC: DISPLAY=TSOCON   APPLICATION=CICSC23G 18:28:59.11 11/11/05*/
000006 /*   ---------- DISPLAY CHARACTERISTICS AND FEATURES ------------   */
000007 /*   ALTCSET=APL                APLCSID=(963,310)                  */
000008 /*   BASECSID=(695,1147)        CCSIZE=(8,12)     COLOR=MULTI      */
000009 /*   DBCS=NO                                                       */
000010 /*   DISPLAY=(24,80,32,80)      DLOGMOD=LSX32703  EXTFUN=YES       */
000011 /*   FLDOUTLN=NO                FLDVALID=NO       HIGHLITE=YES     */
000012 /*   MAXNOPTN=0                 PS=NONE           UOM=INCH         */
000013 /*-------------------------------------------------------------------------*/
000014 /* ITPLSGEN: SCRIPT GENERATION PARAMETERS        18:31:54.19 11/11/05*/
000015 /*   INPUT    CHABERT.WTMUSER.IDCLOG(TSTRAD)                     */
000016 /*   OUTPUT   CHABERT.WTMTEST.RESPROJ1.STL                       */
000017 /*   MSGTXT   TSTRAD                                             */
000018 /*   NODELAY                                                     */
000019 /*   GENERATE CHANGED                                            */
000020 /*   LU       IDCSLU-1                                           */
000021 /*   STL      TRACE=TSTRADT                                      */
000022 /*   NOVERIFY                                                    */
000023 /*-------------------------------------------------------------------------*/
000024
000025 /*------------------------------------------------- 18285911 00001 */
000026 onin0001: onin substr(ru,1,1) = 'F5'x,
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
MA    a                                                              05/015
```

Figure 30-33 Fragment of STL program

21. The STL program generated for the TSTRAD test case is too long to present screen-by-screen, so a portion is presented in Example 30-1.

Example 30-1 The STL program generated for TSTRAD (password field is encrypted)

```
àprogram=TSTRADT
àinclude wtmvars
TSTRAD: msgtxt
/*--------------------------------------------------------------------*/
/* ITPIDC: DISPLAY=TSOCON   APPLICATION=CICSC23G 18:28:59.11 11/11/05*/
/*   ----------- DISPLAY CHARACTERISTICS AND FEATURES ------------   */
/*   ALTCSET=APL               APLCSID=(963,310)                      */
/*   BASECSID=(695,1147)       CCSIZE=(8,12)     COLOR=MULTI          */
/*   DBCS=NO                                                          */
/*   DISPLAY=(24,80,32,80)     DLOGMOD=LSX32703  EXTFUN=YES           */
/*   FLDOUTLN=NO               FLDVALID=NO       HIGHLITE=YES         */
/*   MAXNOPTN=0                PS=NONE           UOM=INCH             */
/*--------------------------------------------------------------------*/
/* ITPLSGEN: SCRIPT GENERATION PARAMETERS       18:31:54.19 11/11/05*/
/*   INPUT    CHABERT.WTMUSER.IDCLOG(TSTRAD)                        */
/*   OUTPUT   CHABERT.WTMTEST.RESPROJ1.STL                          */
/*   MSGTXT   TSTRAD                                                */
/*   NODELAY                                                        */
/*   GENERATE CHANGED                                               */
/*   LU       IDCSLU-1                                              */
/*   STL      TRACE=TSTRADT                                         */
/*   NOVERIFY                                                       */
/*--------------------------------------------------------------------*/

/*------------------------------------------------- 18285911 00001 */
onin0001: onin substr(ru,1,1) = 'F5'x,
               then found = on
found = off
initself('CICSC23G','LSX32703')
do while found = off            /* wait for onin0001 data received */
 wait until onin
end
deact onin0001

/* 18:28:59.13 ITP1507I SESSION STARTED WITH APPLICATION CICSC23G */

/*------------------------------------------------- 18290728 00001 */
WTM_panel_ID = 'PNL00001'
log 'WTM_panel_ID' WTM_panel_ID
cursor(10,26)
ereof
charset 'field'
type 'CHABERT'
cursor(11,26)
upnd = 'ACDE436701F74B6F900C81093F0B97DD267CFCB536'x
userexit('ITPUMNDX',upnd)
cursor(12,26)
transmit using enter

/*------------------------------------------------- 18290884 00003 */
WTM_panel_ID = 'PNL00002'
log 'WTM_panel_ID' WTM_panel_ID
transmit using clear

/*------------------------------------------------- 18291107 00005 */
WTM_panel_ID = 'PNL00003'
```

```
log 'WTM_panel_ID' WTM_panel_ID
cursor(1,1)
charset 'field'
type 'TDB2'
transmit using enter

/*------------------------------------------------- 18291351 00007 */
WTM_panel_ID = 'PNL00004'
log 'WTM_panel_ID' WTM_panel_ID
cursor(9,42)
ereof
charset 'field'
type 'E'
cursor(13,42)
ereof
upnd = '90B925F901E7D9201D4A483E3D'x
userexit('ITPUMNDX',upnd)
transmit using enter

/*------------------------------------------------- 18291909 00009 */
WTM_panel_ID = 'PNL00005'
log 'WTM_panel_ID' WTM_panel_ID
cursor(17,60)
charset 'field'
type '1'
cursor(17,60)
transmit using enter

/*------------------------------------------------- 18292214 00011 */
WTM_panel_ID = 'PNL00006'
log 'WTM_panel_ID' WTM_panel_ID
cursor(17,57)
ereof
charset 'field'
type '3'
transmit using enter

/*------------------------------------------------- 18292359 00013 */
WTM_panel_ID = 'PNL00007'
log 'WTM_panel_ID' WTM_panel_ID
cursor(13,44)
ereof
charset 'field'
type '1'
transmit using enter

/*------------------------------------------------- 18292494 00015 */
WTM_panel_ID = 'PNL00008'
log 'WTM_panel_ID' WTM_panel_ID
cursor(17,57)
ereof
charset 'field'
type '1'
transmit using enter

/*------------------------------------------------- 18294906 00017 */
WTM_panel_ID = 'PNL00009'
log 'WTM_panel_ID' WTM_panel_ID
transmit using pf3
```

```
/*------------------------------------------------- 18295020 00019 */
WTM_panel_ID = 'PNL00010'
log 'WTM_panel_ID' WTM_panel_ID
transmit using pf3

/*------------------------------------------------- 18295215 00021 */
WTM_panel_ID = 'PNL00011'
log 'WTM_panel_ID' WTM_panel_ID
transmit using pf3

/*------------------------------------------------- 18295279 00023 */
WTM_panel_ID = 'PNL00012'
log 'WTM_panel_ID' WTM_panel_ID
transmit using pf3

/*------------------------------------------------- 18295523 00025 */
WTM_panel_ID = 'PNL00013'
log 'WTM_panel_ID' WTM_panel_ID
transmit using clear

/*------------------------------------------------- 18300720 00027 */
WTM_panel_ID = 'PNL00014'
log 'WTM_panel_ID' WTM_panel_ID
cursor(1,1)
charset 'field'
type 'cesf'
transmit using enter

/*------------------------------------------------- 18310919 00072 */
WTM_panel_ID = 'PNL00015'
log 'WTM_panel_ID' WTM_panel_ID
onin0002: onin substr(rh,1,5) = '6B80003201'x,
               then found = on
found = off
termself()
do while found = off             /* wait for onin0002 data received */
 wait until onin
end
deact onin0002

/* 18:31:09.20 ITP1508I SESSION ENDED WITH APPLICATION CICSC23G */

say 'WTM:' msgtxtid() 'Finished'
endtxt
```

30.5 Scheduling and running a simulation

In this section we walk through a process to schedule and run simulations using WTM.

1. Select option **4** (Run) on the WTM main panel. The WTM Process WSim Schedules panel is displayed showing all test schedules defined for the current project. A schedule is a WSim network definition with supporting documentation and reporting facilities. There are no existing schedules in this case, so use the **add** command to create a new schedule, as shown in Figure 30-34.

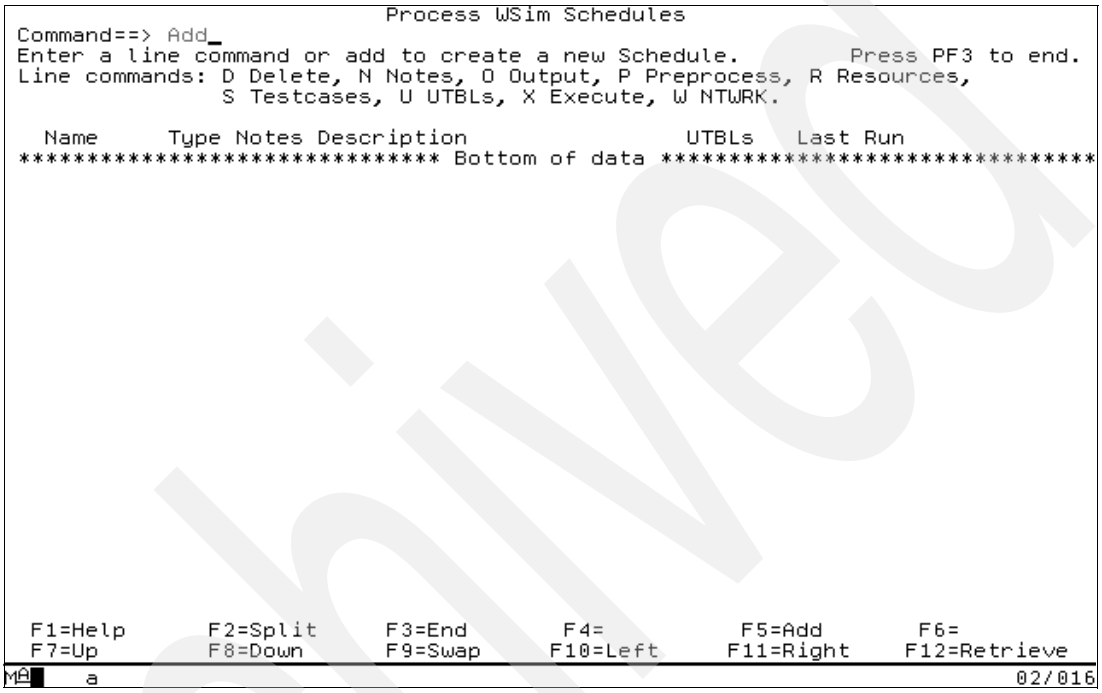


Figure 30-34 Adding new schedule

2. Provide a name, description, and type for this new schedule on the next panel, as shown in Figure 30-35.

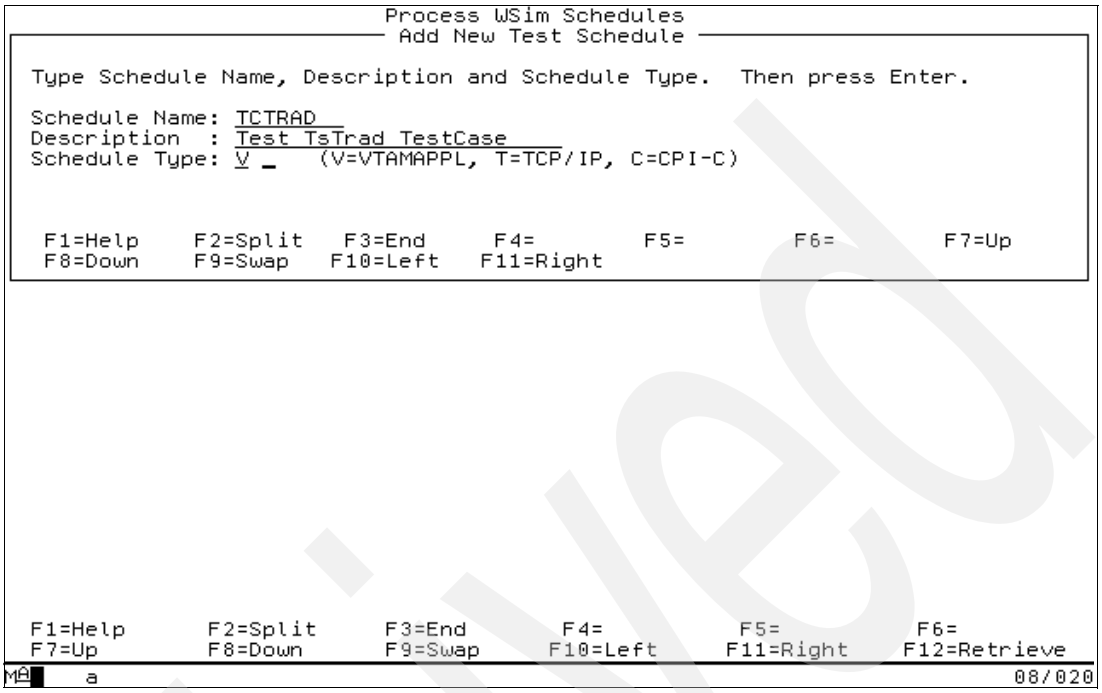


Figure 30-35 New schedule name, description, and type

3. WTM displays the “Milestones” panel with the steps to be performed, as shown in Figure 30-36.

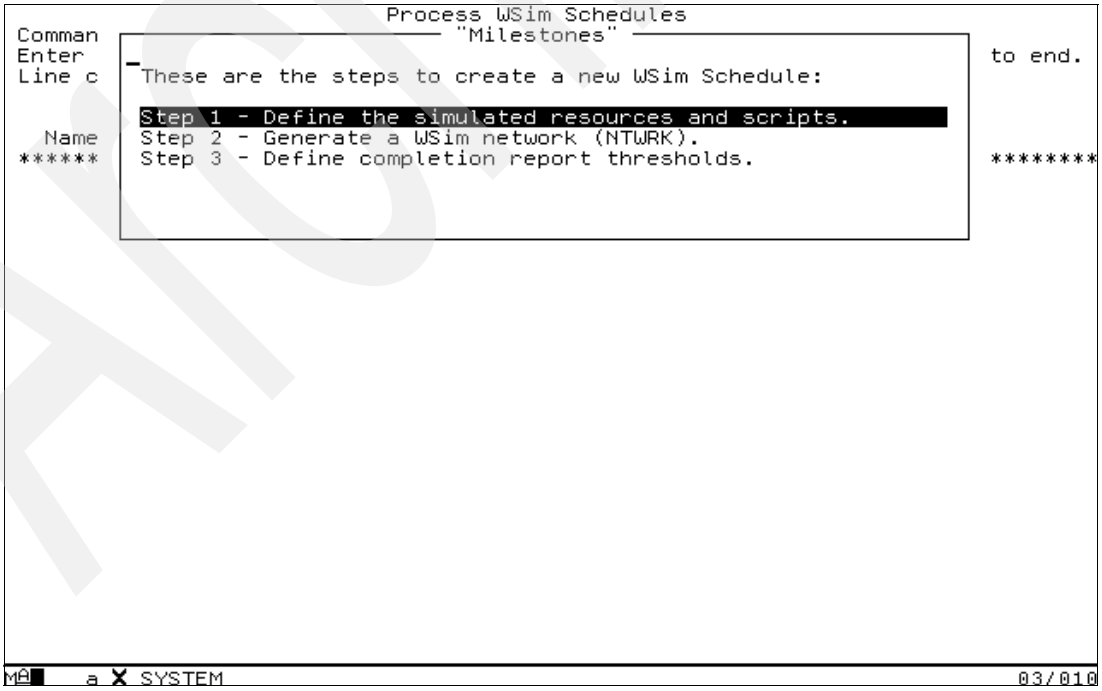


Figure 30-36 “Milestones” panel with steps to create schedule

4. Selection of the network resources is done on the next panel, shown in Figure 30-37.

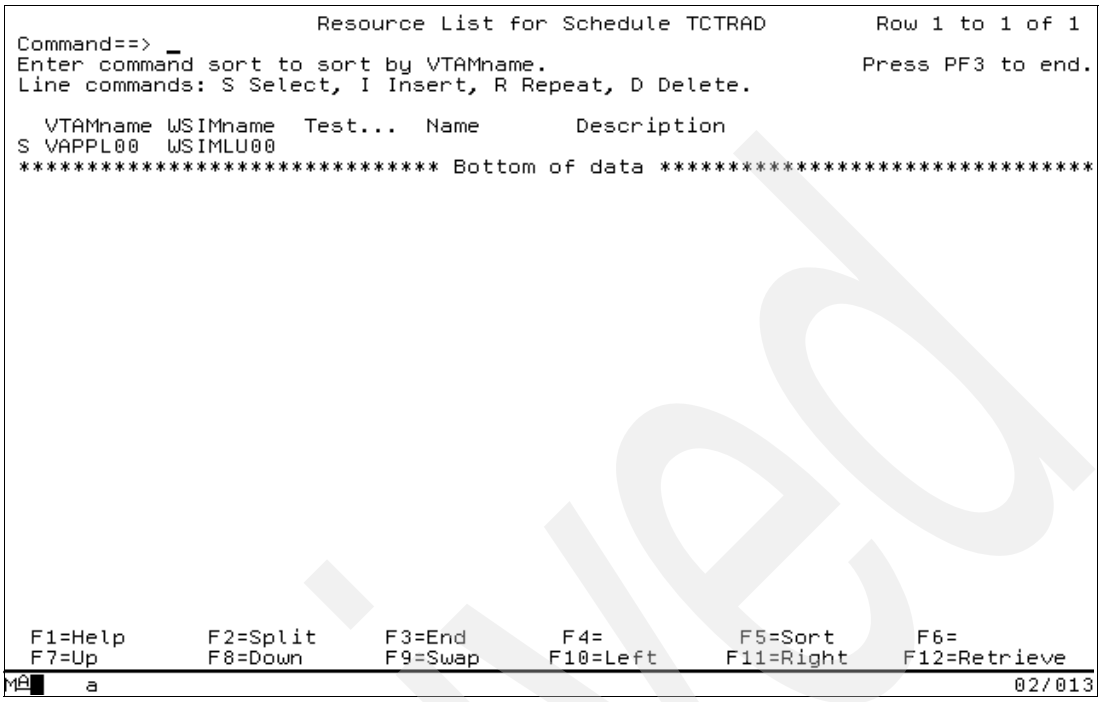


Figure 30-37 Selecting resource for simulation

5. The list of the test cases (also test groups and test cycles, if applicable) available to be used in this schedule is presented. You can define the order in which they are executed. Each test case can be used more than once. In this case multiple order positions should be specified, for example, 1 2 3, as shown in Figure 30-38.

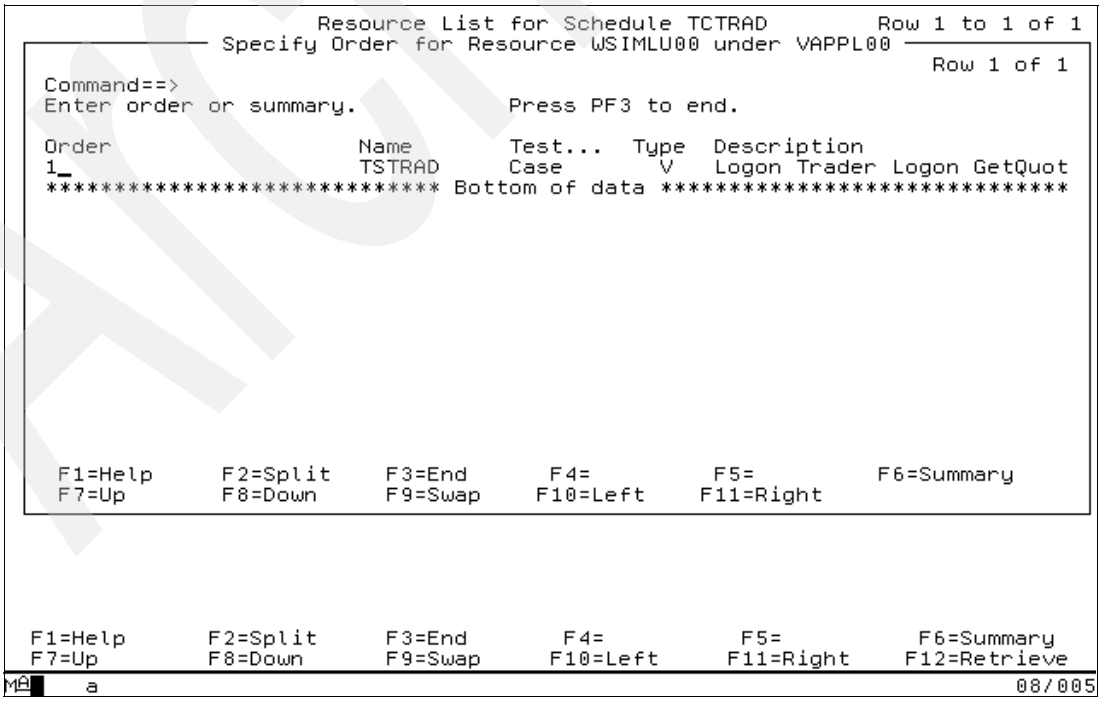


Figure 30-38 Specifying order for resource

6. After selecting the order, press PF3 for the panel shown in Figure 30-39. WTM presents (for the first time only) the Completion Report Response Time Thresholds panel, where you can enter response time mean, medium, mode, and high values in seconds and percentiles of 95%, 90%, 80%, and 70% in seconds for success criteria to be used in the Response Time Report.

> **Note:** These values can be set up later using option **4** (RUN) from the main panel, line command **0** for the corresponding schedule, PF8 to get to the lower part of the next panel, and option **18** THRESH.

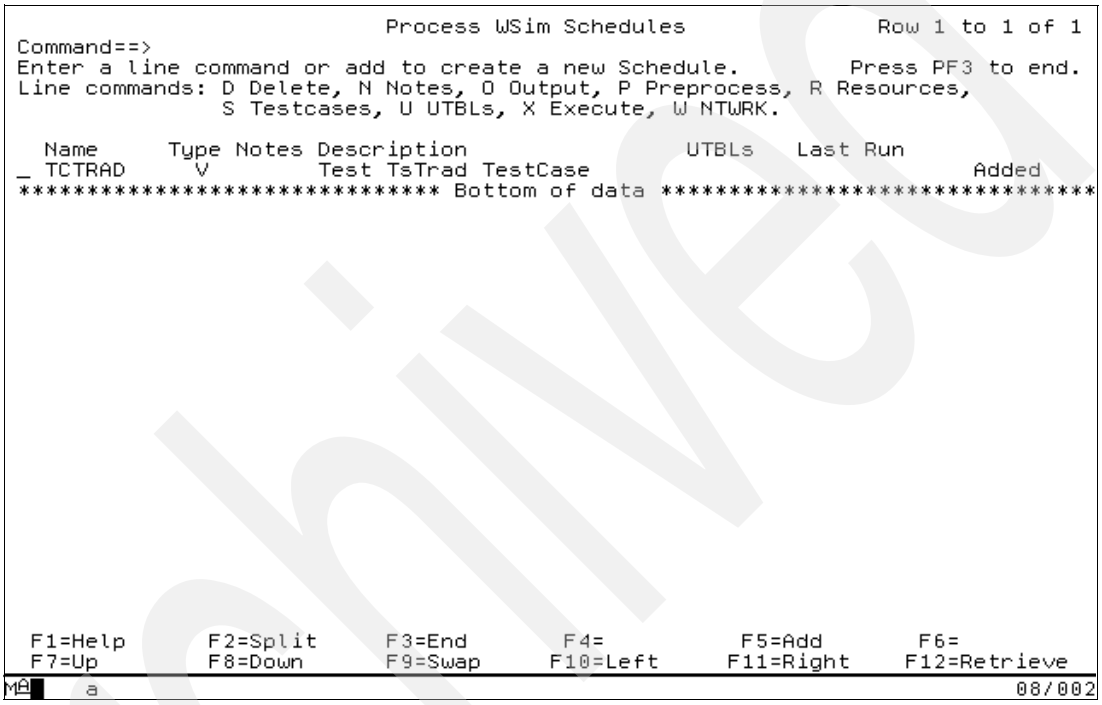


Figure 30-39 Test case assigned to resource

7. Now you are ready to execute the newly created schedule TCTRAD. The line command **X** is used to start a simulation, as shown in Figure 30-40.

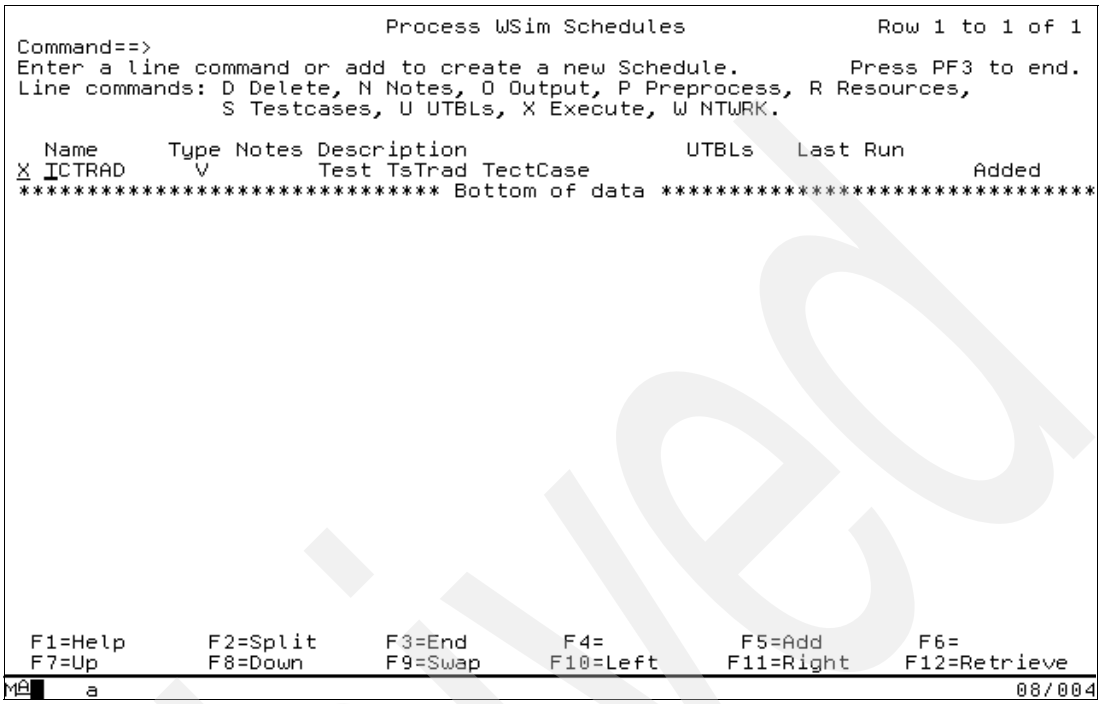


Figure 30-40 Starting TCTRAD

8. WTM provides an opportunity to specify the data set names of one or more WSim logs that is written during the test and provides the generated ones, which can be changed if you want, as shown in Figure 30-41.

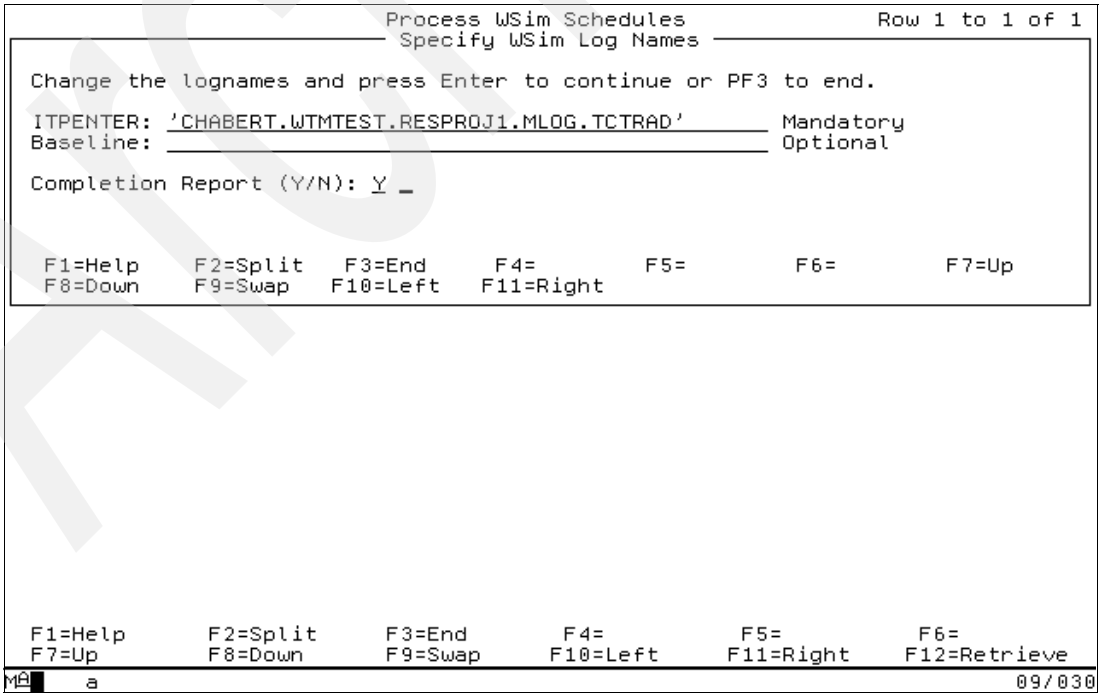


Figure 30-41 Specifying lag data sets names

9. When performing a test in batch mode, WSim directs generated console messages to the MVS log. When performing a test from TSO, WSim sends the console messages to the TSO terminal, as shown in Figure 30-42.

```
ITP016I Workload Simulator (WSim) Version 1 Release 1.0.1, Feb. 10 2005 06:50
ITP003I WSim INITIALIZATION COMPLETE
ITP200I DISPLAY MONITOR FACILITY ACTIVE USING APPL WTMDM00
ITP029I INITIALIZATION COMPLETE FOR NETWORK TCTRAD
ITP006I NETWORK TCTRAD STARTED
ITP137I TCTRAD   WSIMLU00-00001 - WTM: TSTRAD Finished
ITP137I TCTRAD   WSIMLU00-00001 - WTMEND: 1 out of 1 finished.
ITP137I TCTRAD   WSIMLU00-00001 - WTMEND: All scripts run.  Issuing ZEND.
ITP002I TCTRAD   WSIMLU00-00001 - ZEND
ITP201I DISPLAY MONITOR FACILITY IS CLOSED DOWN
ITP079I WSim IS CLOSED DOWN
*** _
MA    a                                                          12/006
```

Figure 30-42 WSim simulation messages

10. The generated completion report from the simulation is presented in Figure 30-43.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CHABERT.WTMTEST.RESPROJ1.CREP(R3) - 01.00       Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ********************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
==MSG> -CAUTION- Profile is set to STATS ON. Statistics did not exist for
==MSG>           this member, but will be generated if data is saved.
000001 Report for TCTRAD - Test TsTrad TectCase.
000002 Run completed at 14:53:29 on 14 Nov 2005.
000003 No Response Time Thresholds selected.
000004 Loglist Extract:
000005 RUN  TIME 14.52.55,  NOVEMBER 14, 2005  VERSION  1  RELEASE  1.0.1
000006 14532765 ITP137I TCTRAD   WSIMLU00-00001 - WTM: TSTRAD Finished
000007 Comments:
000008 _
000009
****** **************************** Bottom of Data ******************************
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
MA    a                                                          17/009
```

Figure 30-43 Completion report

The test is finished.

30.6 Simulation reports

Several output reports are accessible from the list of schedules. Enter the line command **o** to view outputs for the corresponding schedule, as shown in Figure 30-44 and Figure 30-45.

```
                          Process WSim Schedules              Row 1 to 1 of 1
Command==> _
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                UTBLs   Last Run
O TCTRAD     V        Test TsTrad TectCase             11/14/05 14:40 Reports
****************************** Bottom of data ********************************
 F1=Help      F2=Split     F3=End       F4=RETURN    F5=Add       F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA    a                                                              02/013
```

Figure 30-44 Output of TCTRAD simulation to be viewed

The Report panel is presented, as shown in Figure 30-45.

```
                        Reports for schedule TCTRAD

Select one of the following.  Then press Enter.

      Command    Action                                                More:  +
1_  1. TLOG      View loglist report from the latest WSim run
    2. TRESP     View response time report from the latest WSim run
    3. TDM       Log display monitor for the latest WSim run
    4. TSP       View SYSPRINT from the latest WSim run

    5. MLOG      View baseline loglist report
    6. MRESP     View baseline response time report
    7. MDM       Log display monitor for the baseline log

    8. COMP      View screen compare report
    9. CDM       Log display comparator
   10. RTCOMP    Edit response time compare report

   11. COMPREP   Edit completion reports

Command===> ____________________________________________________________

 F1=Help      F2=Split     F3=End       F4=RETURN    F5=          F6=
 F7=          F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA    a                                                              06/003
```

Figure 30-45 WTM Reports panel, part 1

Press PF8 to see the second part of the panel, as shown in Figure 30-46.

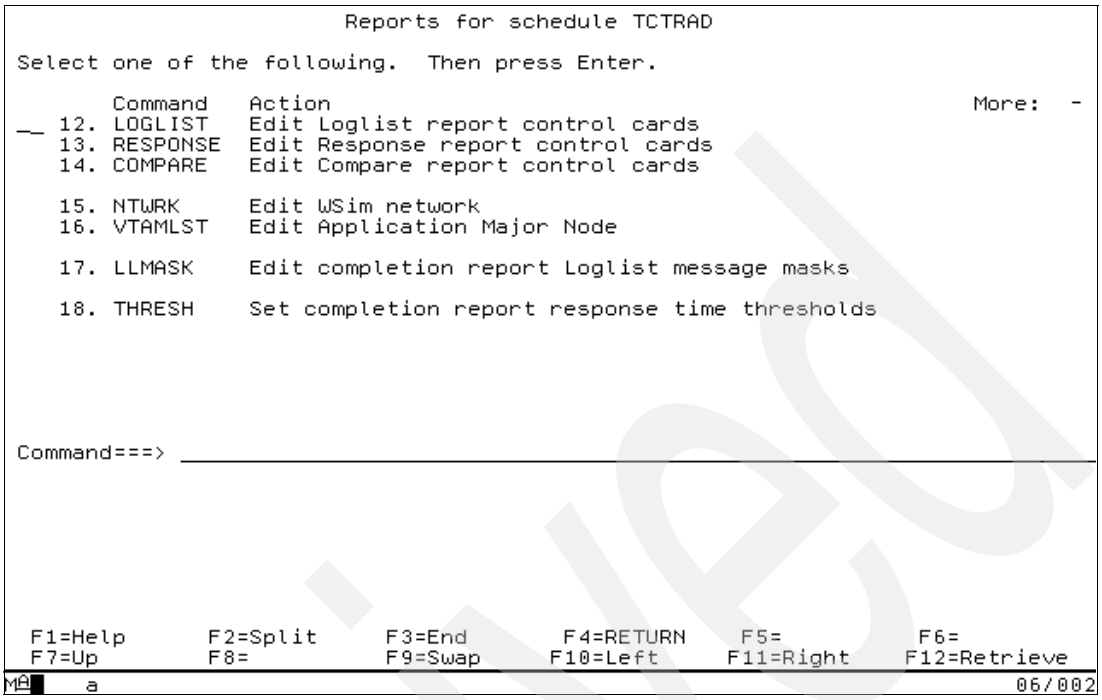


Figure 30-46 WTM Reports panel, part 2

The choices on this panel have the following meanings:

TLOG WTM invokes ITPLL on the WSim/ISPF Interface for the WSim log from the latest simulation run. The loglist report is browsed. If the baseline log exists, this log is known as the test WSim log.

TRESP WTM invokes ITPRESP on the WSim/ISPF Interface for the WSim log from the latest simulation run. The response time report is browsed.

TDM The screen images for the WSim log from the latest simulation run are displayed.

TSP The SYSPRINT file from the latest simulation run is browsed.

MLOG WTM invokes ITPLL on the WSim/ ISPF Interface for the master WSim log. The loglist report is browsed.

MRESP WTM invokes ITPRESP on the WSim/ISPF Interface for the master WSim log. The response time report is browsed.

MDM The screen images for the master WSim log are displayed.

COMP If both the master and test WSim logs exist, WTM invokes ITPCOMP on the WSim/ISPF Interface. The output is browsed.

CDM If both the master and test WSim logs exist, WTM invokes the Log Display Monitor to compare screen images.

RTCOMP If both the master and test WSim logs exist, WTM creates a report that compares the response times on both the logs. You are placed in an edit session on this report.

COMPREP This displays a list of the completion reports that can be selected for edit.

LOGLIST Edit the control cards used when the WSim loglist utility ITPLL is run.

RESPONSE Edit the control cards used when the WSim response time utility ITPRESP is run.

COMPARE Edit the control cards used when the WSim screen compare utility ITPCOMP is run.

NTWRK Edit the WSim network definition for this schedule.

VTAMLST Edit the VTAM application major node generated by WTM for this schedule.

LLMASK When the completion report is created, certain messages are not shown on the report panel and some messages are not written to the completion report itself. These masks can be edited using this option.

THRESH Use this option to change the response time thresholds. The response time thresholds are useful in determining the rate of traffic — what messages were sent and received within a specified criteria. The percentile values are extracted from the response time skeleton. The top four values (out of 10 maximum) of the PERCENT command are used for the percentiles.

For our example, the only report of interest would be the loglist report. Select option **1** (TLOG). Figure 30-47 appears, giving the ability to specify control information for the report.

```
                    WSim: Control Analysis of Logged Data

Type information.  Then press Enter.
                                                                    More:  +
  Message decks/STL procedures _________ , ________ , ________ , ________
  Networks . . . . . . . . . . ________ , ________ , ________ , ________
  Group name and group type    ________ , V (V=VTAMAPPL, T=TCP/IP,
                                             A=APPCLU)
    Term, LU, or TP name . . . ________________ , ________________ , ________________
                               ________________ , ________________ , ________________
    Include terms, LUs, TPs    Y              (Y=Yes or N=No)
  Time Limits  . . . . . . . . ______ - ______

  Select one or more record types by typing a '/' or 'S'.
    / Console                / Informational          / Message generation trace
    / Display                / Data                   / STL trace
      _ Extended attributes  _ Header-only records    _ Verification logs
      _ Non-displayables     _ CPI-C trace data       / Log
      _ Separate partitions
      _ Scripting records


 F1=Help              F2=Split   F5=Refresh  F7=Bkwd  F8=Fwd  F9=Swap
F10=Additional input  F11=Save   F12=Cancel

MA  a                                                          05/033
```

Figure 30-47 Control for loglist report, part 1

Press PF8 to see the lower part of this panel, as shown in Figure 30-48.

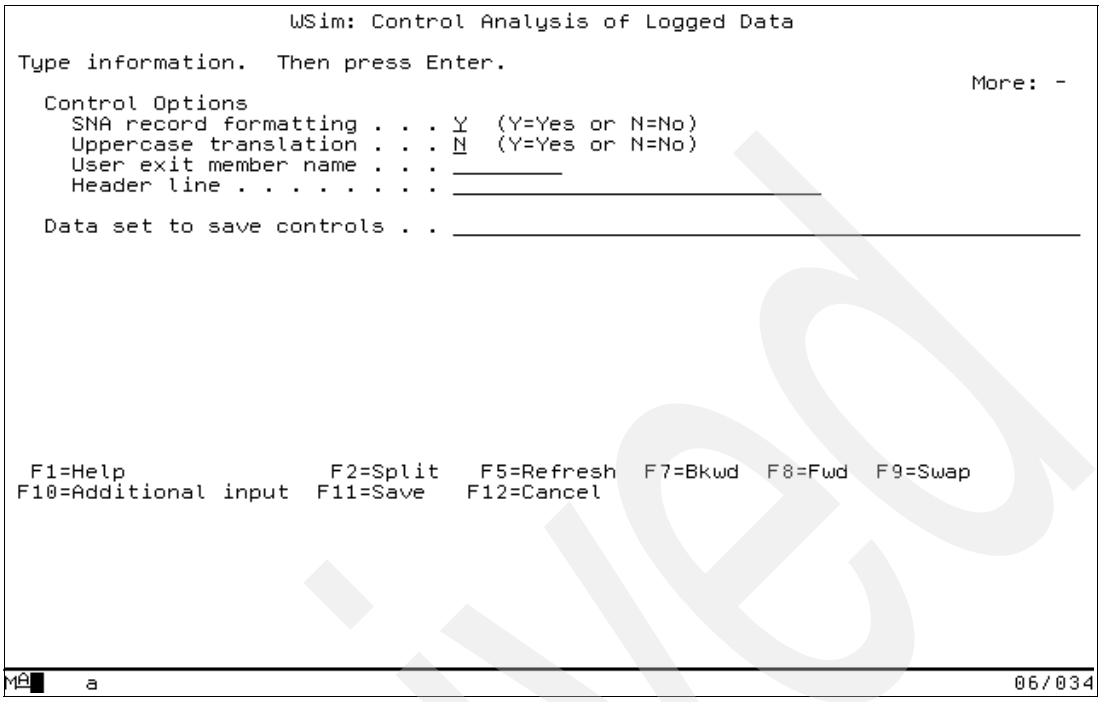


Figure 30-48 Control for loglist report, part 2

Additional parameters can be added on the second panel, which is accessible by pressing PF10, as shown in Figure 30-49.

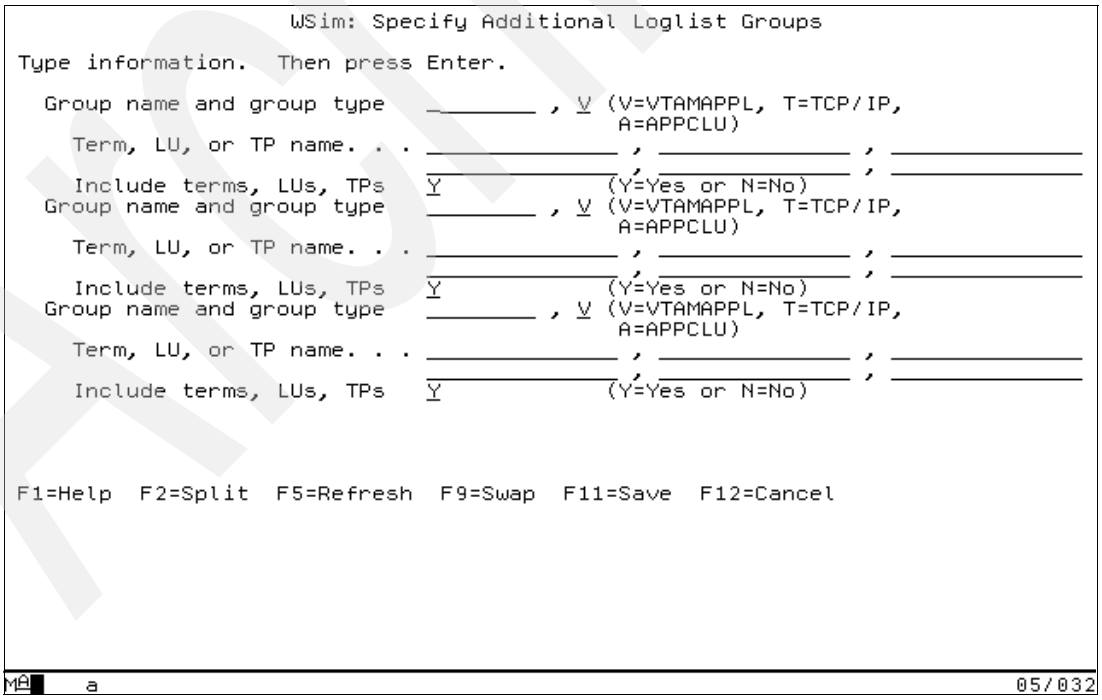


Figure 30-49 Additional parameters for loglist report

When WTM finishes the analysis, the informational message appears, and you can browse the report shown in Figure 30-50.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CHABERT.WTM.SYSPRINT                                 Columns 00001 00072
Command ===> _                                                   Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 1WSim LOGLIST OUTPUT
000002 0CNSL
000003  DSPLY
000004  LOG
000005  INFO
000006  DATA
000007  CDLOG
000008  NOHDR
000009  NORR
000010  MTRC
000011  STRC
000012  NOVRPAC
000013  FMTSNA
000014  NOCTRC
000015  RUN
000016 1WSim LOGLIST OUTPUT
000017  RUN  TIME 14.52.55,  NOVEMBER 14, 2005  VERSION  1  RELEASE  1.0.1
000018
000019  NETWORK   APPCLU/TCPIP/VTAMAPPL   DEV/LU/TP        START    STOP     RE
000020   NAME             NAME              NAME          TIME     TIME     TI
000021 0                                                 14525545 0105318  1101
000022 0  Workload Simulator (WSim) Version 1 Release 1.0.1, Feb. 10 2005 06:50
000023 0-------------------------------------------------------------------------
000024 0                                                 14525550 0105318  1101
000025 0  ITP029I INITIALIZATION COMPLETE FOR NETWORK TCTRAD
000026 0-------------------------------------------------------------------------
000027 0                                                 14525550 0105318  1101
MA    a                                                                  04/015
```

Figure 30-50 Loglist report, part 1

This report can be very long. In this case it contains almost 1850 lines, as shown in Figure 30-51.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CHABERT.WTM.SYSPRINT                                 Columns 00001 00072
Command ===> _                                                   Scroll ===> CSR
001823          DIMENSIONS: ( 32, 80)
001824 0-------------------------------------------------------------------------
001825 0TCTRAD          VAPPL00          WSIMLU00-1     14532765 0105318  1101
001826 0  ITP3515I PROGRAM=TSTRADT  STMT£=00613 PROCEDURE=TSTRAD  : EXECUTION R
001827    ITP3512I PROGRAM=TSTRADT  STMT£=00613 PROCEDURE=TSTRAD  : DO WHILE CO
001828 0-------------------------------------------------------------------------
001829 1WSim LOGLIST OUTPUT
001830  RUN  TIME 14.52.55,  NOVEMBER 14, 2005  VERSION  1  RELEASE  1.0.1
001831
001832  NETWORK   APPCLU/TCPIP/VTAMAPPL   DEV/LU/TP        START    STOP     RE
001833   NAME             NAME              NAME          TIME     TIME     TI
001834 0                                                 14532765 0105318  1101
001835 0  ITP137I TCTRAD   WSIMLU00-00001 - WTM: TSTRAD Finished
001836 0-------------------------------------------------------------------------
001837 0                                                 14532765 0105318  1101
001838 0  ITP137I TCTRAD   WSIMLU00-00001 - WTMEND: 1 out of 1 finished.
001839 0-------------------------------------------------------------------------
001840 0                                                 14532765 0105318  1101
001841 0  ITP137I TCTRAD   WSIMLU00-00001 - WTMEND: All scripts run.  Issuing Z
001842 0-------------------------------------------------------------------------
001843 0                                                 14532765 0105318  1101
001844 0  ITP002I TCTRAD   WSIMLU00-00001 - ZEND
001845 0-------------------------------------------------------------------------
001846 0                                                 14532766 0105318  1101
001847 0  ITP201I DISPLAY MONITOR FACILITY IS CLOSED DOWN
001848 1WSim LOGLIST OUTPUT
001849 0END
****** ***************************** Bottom of Data ***************************
MA    a                                                                  04/015
```

Figure 30-51 Loglist report, part 2

30.7 Organizing test cases

There are several options available in WTM to organize test cases and make their reuse possible.

A *test group* is an ordered list of test items (test cases and MSGTXTs). The purpose of using test groups is to encourage modularity of test cases, such as a special logon test case, developed once and reused in multiple test groups. As a result, if the logon process changes, only the logon test case has to be changed, even if it is used in several different test groups.

A *test cycle* is also an ordered list of test items (test cases, MSGTXTs, and test groups). The test cycles represent another level of organization of test items and allow the creation of highly complex structures. The WTM test cycle management panels are very similar to the ones for the test groups.

There are several methods to create modular scripts:

- Capture a single test case starting from the beginning of an application session.
- Capture a single test case by starting and stopping capture interactively.
- Capture multiple test cases from a single application session by switching to different IDC log files.

30.7.1 Creating modular scripts

Use the following steps to start an IDC session and capture test cases:

1. Select option **1** (CASE) or enter the command **case** from the WTM main panel. Use the **add** command to start with a new test case, as shown in Figure 30-52.

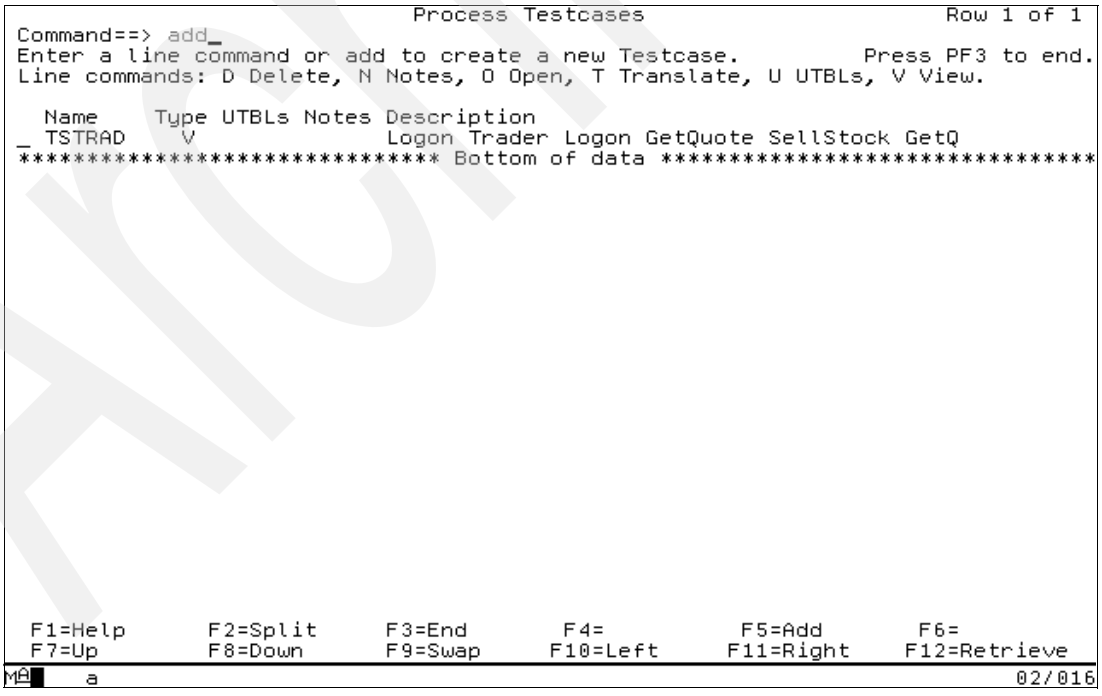


Figure 30-52 Adding another test case

2. On the next panel, shown in Figure 30-53, provide the name and description for the new test case and select option **1** to create the test case using the IDC utility.

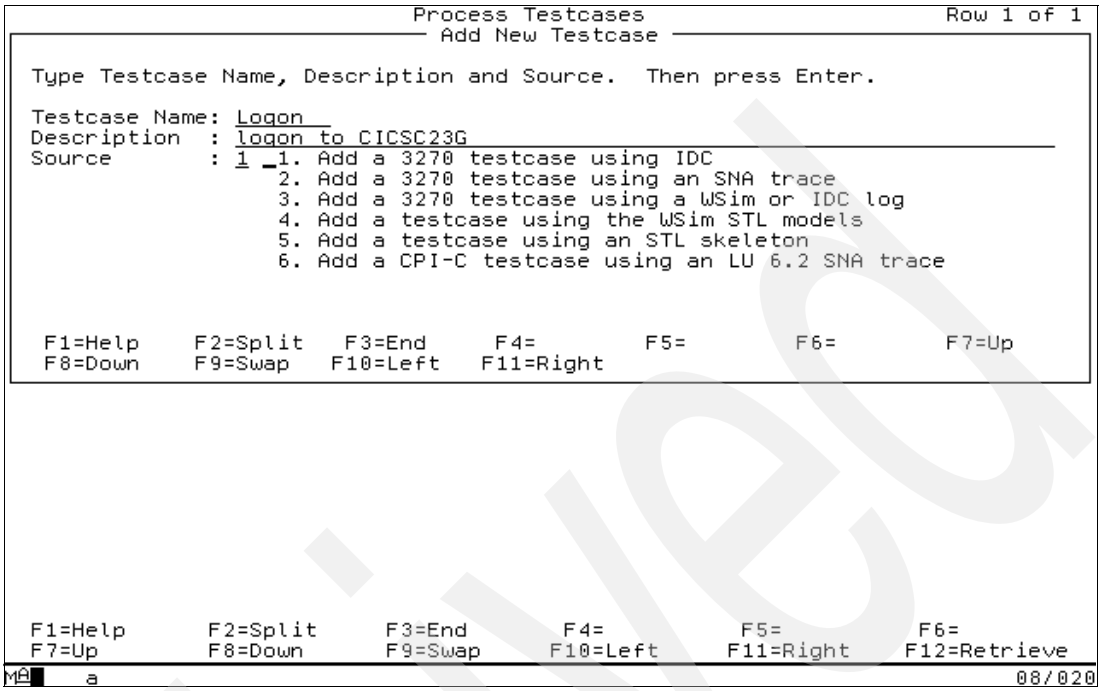


Figure 30-53 Adding test case Logon

3. Press Enter. When the IDC utility panel appears, select option **1** to start a session with a host application. In our case this was the CICS system, as shown in Figure 30-54.

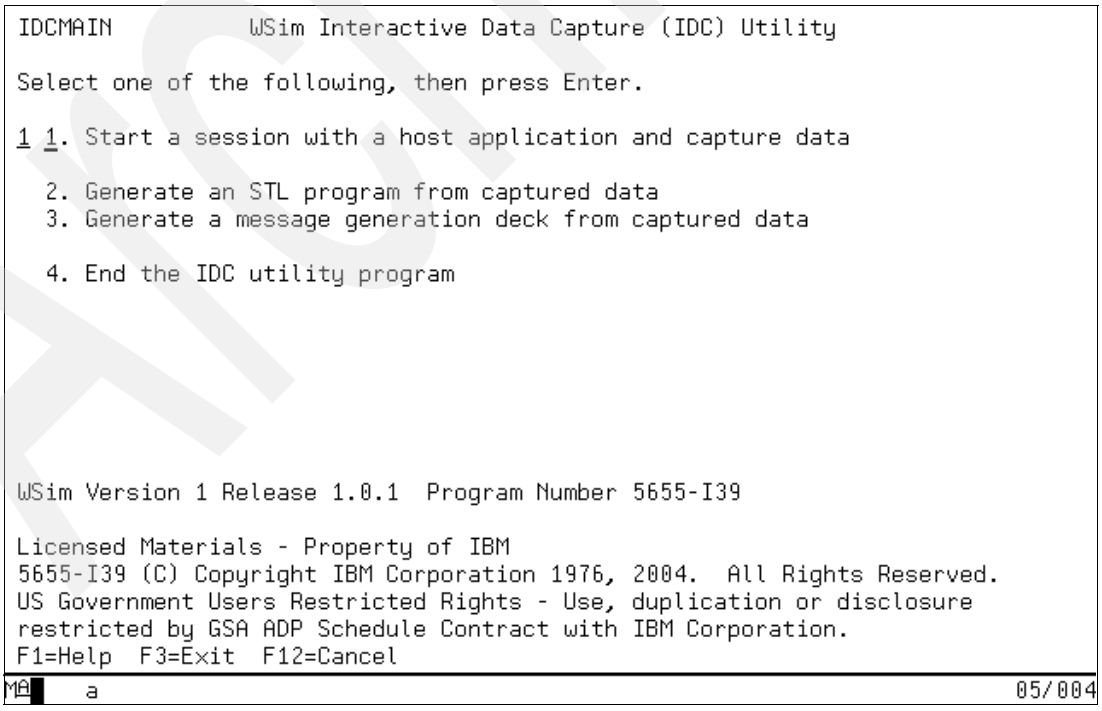


Figure 30-54 WSim IDC panel, start session

4. Press Enter. On the next panel supply the host application name (the corresponding VTAM APPLID), the IDC utility log data set name, the option to replace or append the log data set, the IDC escape key, and whether data capture should start immediately, as shown in Figure 30-55.

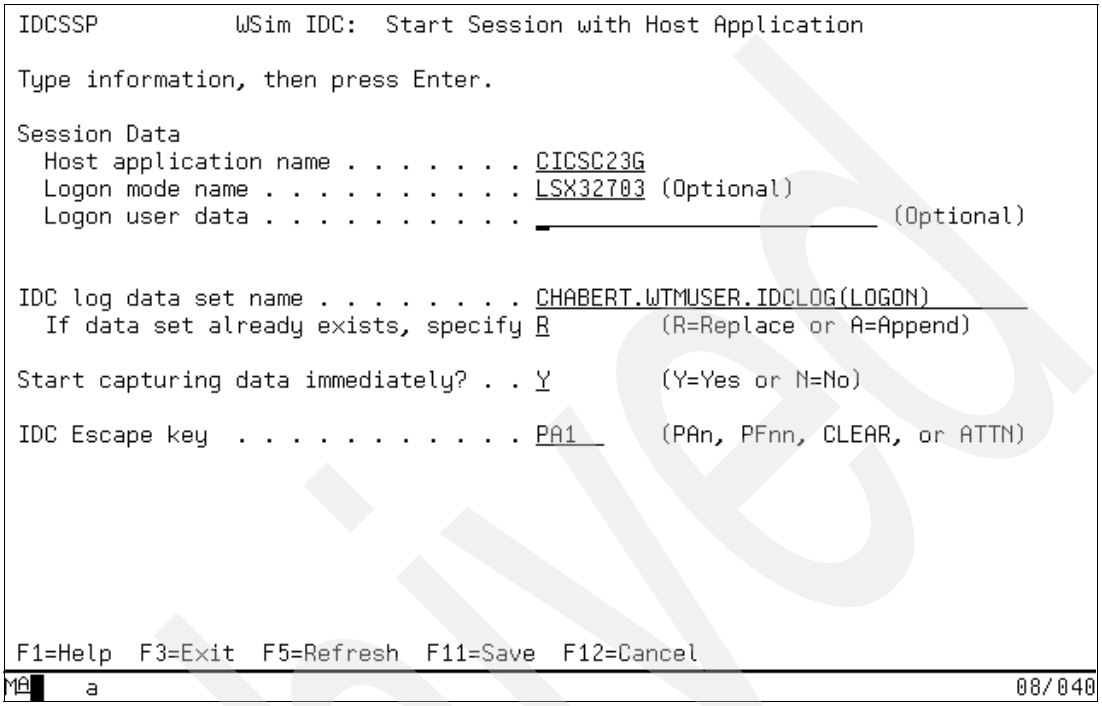


Figure 30-55 IDC controls

5. Press Enter. WTM connects to the host application, in our case CICSC22F, as shown in Figure 30-56.

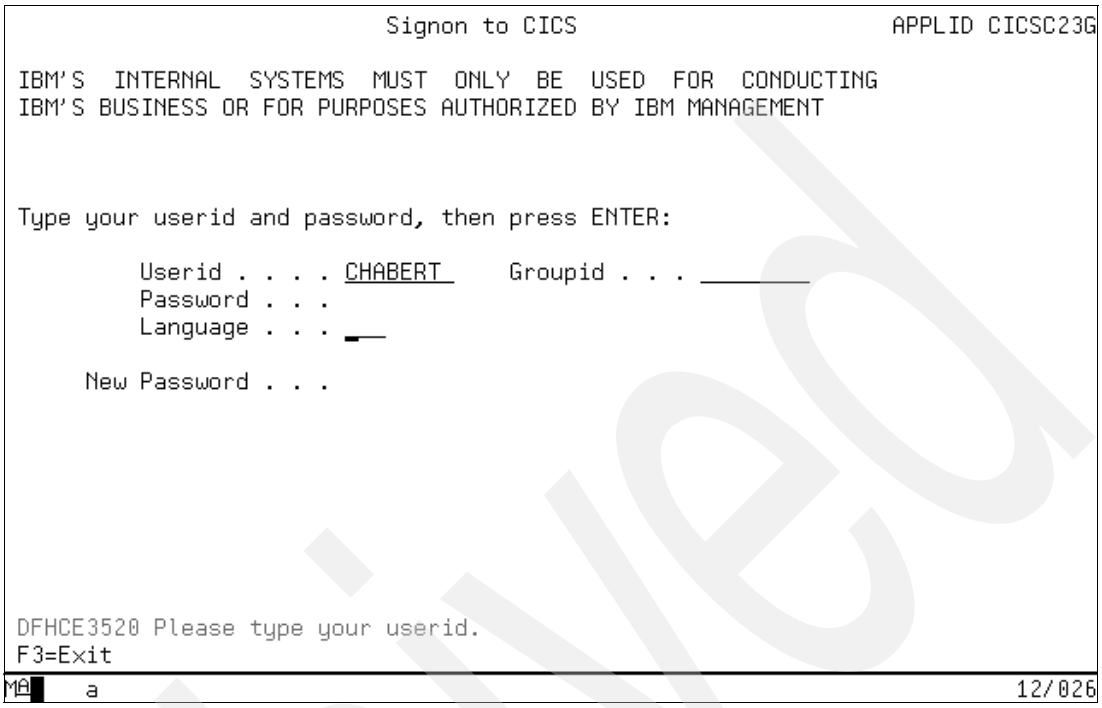


Figure 30-56 CICS sign-on panel

6. Enter the user ID and password and press Enter. Logon to CICS is completed, as shown in Figure 30-57.

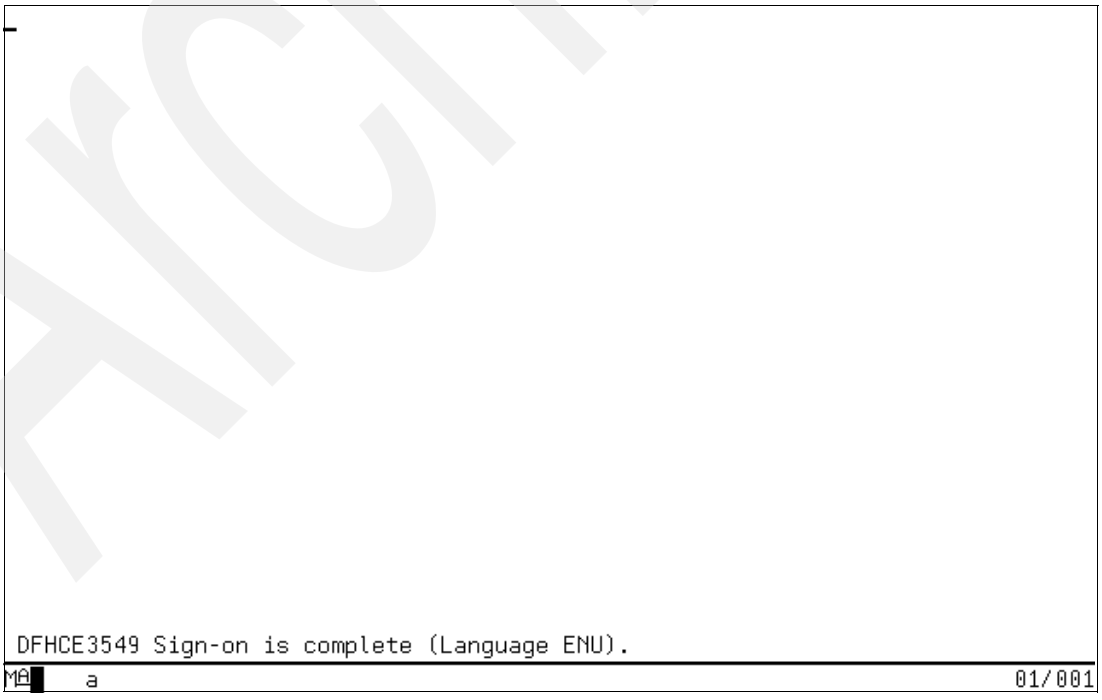


Figure 30-57 CICS sign-on complete

7. Since the test case under creation is just a log on to CICS, press the IDC escape key (PA1), and WTM brings up the Escape Actions panel, where you select option **2** to stop the capture, as shown in Figure 30-58.

```
IDCESCA              WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

2 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : ON
Current IDC log data set: CHABERT.WTMUSER.IDCLOG(LOGON)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
MA    a                                                        06/004
```

Figure 30-58 Stopping data capture

8. Press Enter and make sure that the data capture status indicator is OFF before proceeding further.

> **Important:** If option 3 to end the session with the host application is selected and Enter is pressed when data capture is ON, this event becomes a part of the captured data.

9. Press PF3 to return to the host session and issue the transaction **`cesf logoff`**. The IDC utility ends the session with CICSC23G and displays the confirmation message, as shown in Figure 30-59. The IDC log data set for the test case LOGON is created.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . CICSC23G
  Logon mode name . . . . . . . . . . LSX32703 (Optional)
  Logon user data . . . . . . . . . . _____________________ (Optional)


IDC log data set name . . . . . . . . CHABERT.WTMUSER.IDCLOG(LOGON)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . Y        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1      (PAn, PFnn, CLEAR, or ATTN)




ITP1508I SESSION ENDED WITH APPLICATION CICSC23G
F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
MA    a                                                        06/040
```

Figure 30-59 Session ended message

10. Press PF3. WTM performs the steps to create the test case from the IDC log and returns to the list of test cases with a confirmation message, as shown in Figure 30-60.

```
                          Process Testcases                    Row 1 of 2
Command==>
Enter a line command or add to create a new Testcase.         Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
_ LOGON     V              Logon to CICSC23G                          Added
_ TSTRAD    V              Logon Trader Logon GetQuote SellStock GetQ
****************************** Bottom of data ********************************









 F1=Help      F2=Split     F3=End       F4=          F5=Add       F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA    a                                                        07/002
```

Figure 30-60 Test case LOGON added

11. Enter the command **add** and press Enter to continue with the creation of the next test case. This test case involves the start of the TDB2 transaction to start the trader application, as shown in Figure 30-61.

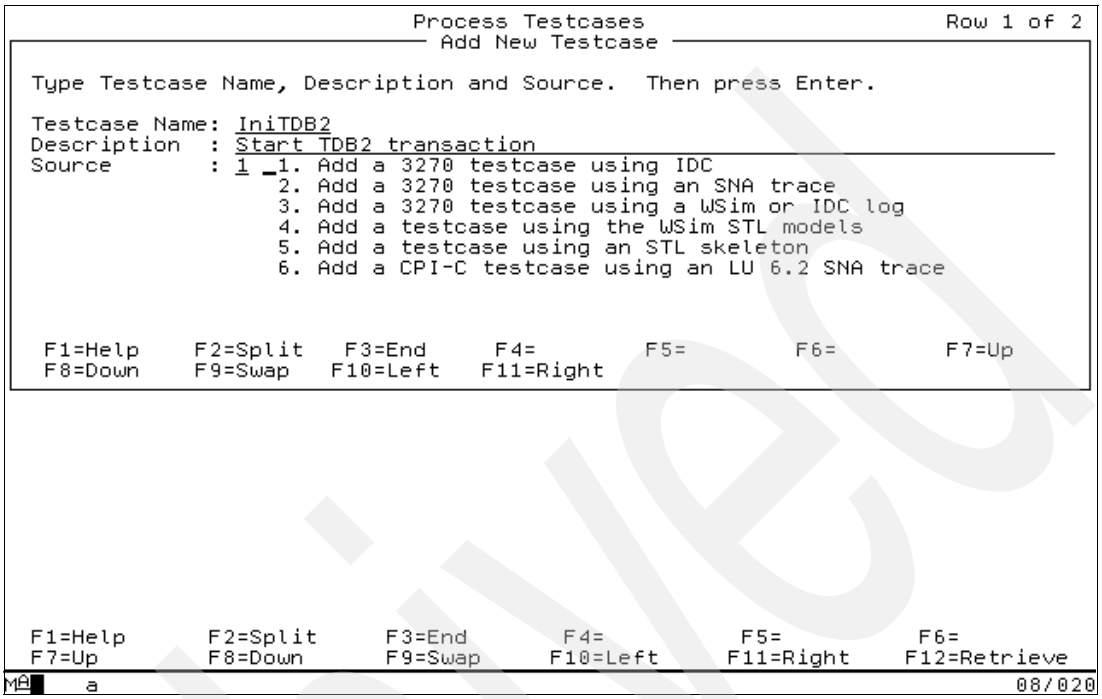


Figure 30-61 Adding test case IniTDB2

12. The panel to enter data required to start a host application is displayed. Since the LOGON test case is already created, we do not want to start capturing data immediately, as shown in Figure 30-62.

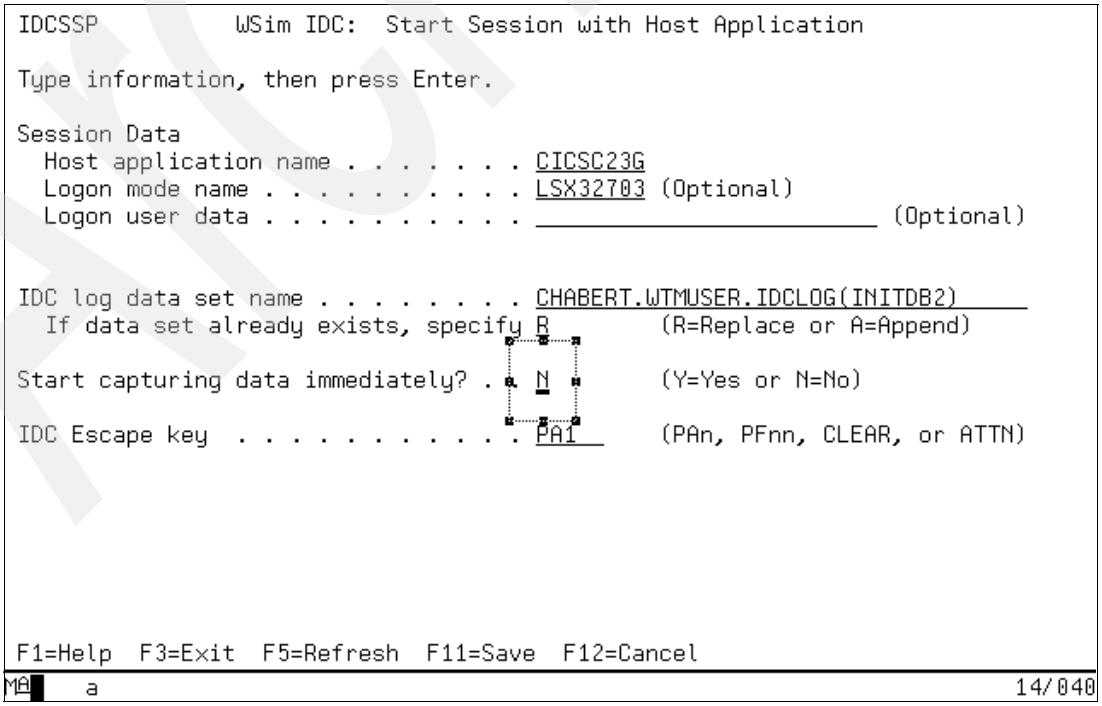


Figure 30-62 Data capture delayed

13. After you complete the steps to log on to application CICSC22F, at the screen shown in Figure 30-57 on page 990, press the IDC escape key (PA1). This brings up the already familiar IDC utility Escape Actions panel. Note that the data capture status is OFF, as shown in Figure 30-63.

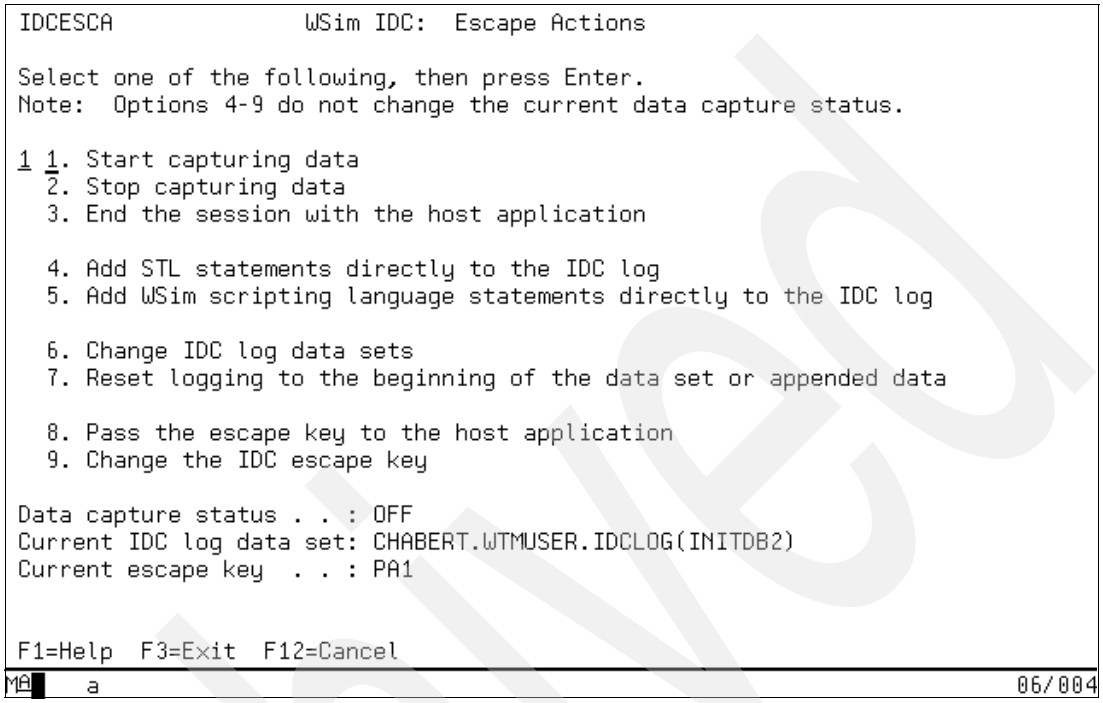


Figure 30-63 Starting delayed data capture

14. Select option **1** and press Enter. The data capture status is now ON, as shown in Figure 30-64.

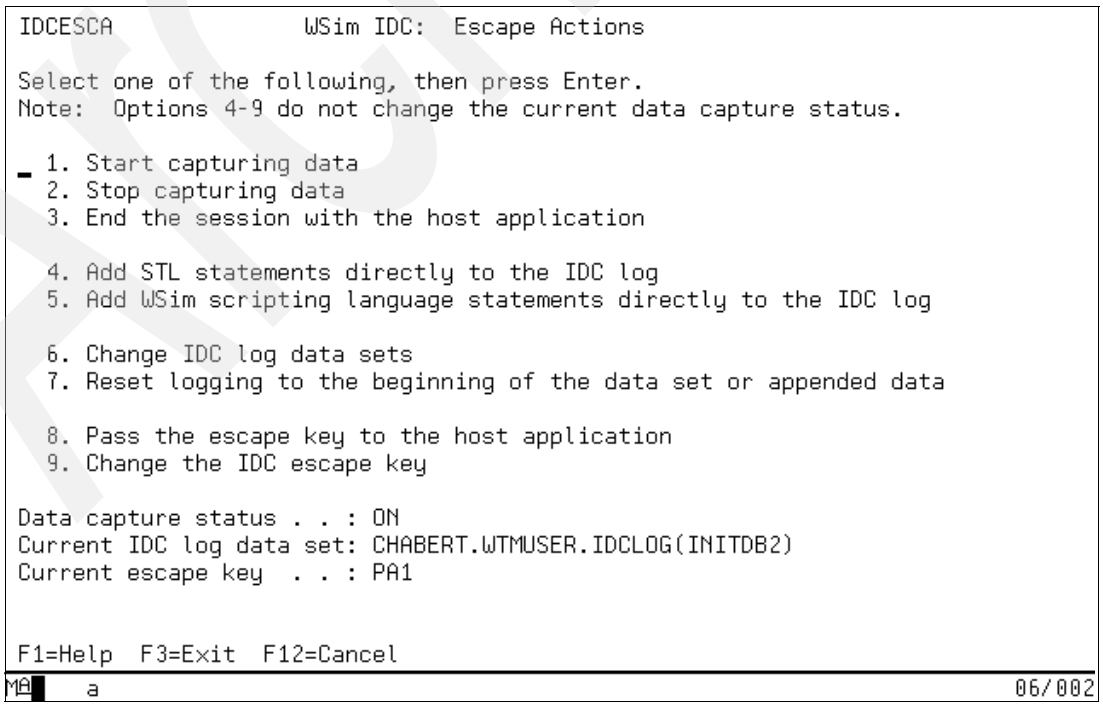


Figure 30-64 Data capture is now ON

15. Press PF3 and WTM brings you back to the application screen. Start the transaction TDB2, as shown Figure 30-65.

Figure 30-65 Transaction TDB2 entered

16. On the screen brought up by the TDB2 transaction select option **7** and press Enter to start the trader application. The trader application logon screen, shown in Figure 30-16 on page 963, appears. Do not enter any values in any fields, and press the IDC escape key (PA1) to get to the IDC utility Escape Actions panel. Select option **2** to stop capturing data and press Enter, as shown Figure 30-66.

```
IDCESCA              WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

2 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : ON
Current IDC log data set: CHABERT.WTMUSER.IDCLOG(INITDB2)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
MA    a                                                        06/004
```

Figure 30-66 Stopping data capture

17. The data capture status indicator changes to OFF. Select option **3** to end the session and press Enter, as shown in Figure 30-67.

```
IDCESCA             WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

3 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : OFF
Current IDC log data set: CHABERT.WTMUSER.IDCLOG(INITDB2)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
MA    a                                                        06/004
```

Figure 30-67 Data capture is OFF - session end requested

18. WTM returns to the IDC utility panel and displays the session end confirmation message, as shown in Figure 30-68.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . CICSC23G
  Logon mode name . . . . . . . . . . LSX32703 (Optional)
  Logon user data . . . . . . . . . . ____________________ (Optional)


IDC log data set name . . . . . . . . CHABERT.WTMUSER.IDCLOG(INITDB2)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . N        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1__    (PAn, PFnn, CLEAR, or ATTN)









ITP1508I SESSION ENDED WITH APPLICATION CICSC23G
F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
MA    a                                                        06/040
```

Figure 30-68 Session ended message

19. Press PF3. WTM goes through the test case creation steps, as shown in Figure 30-69.

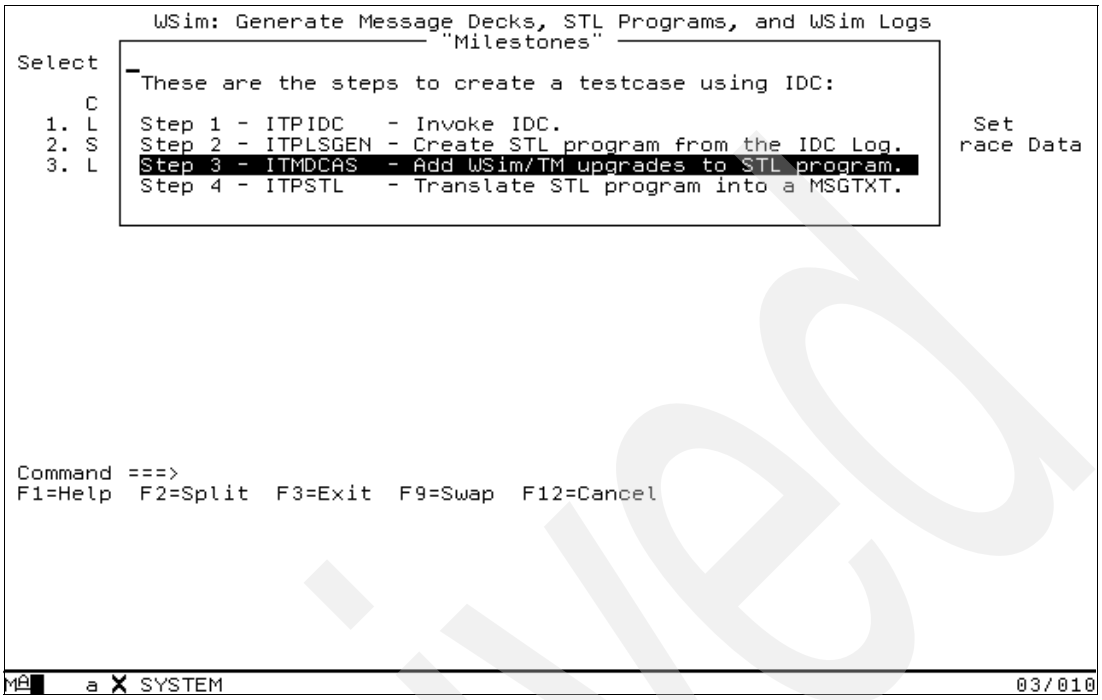


Figure 30-69 Milestones panel with test case creation steps

20. The test case INITDB2 is created and the corresponding message is displayed. Enter the command **add** again to continue to create new test cases, as shown in Figure 30-70. Press Enter.

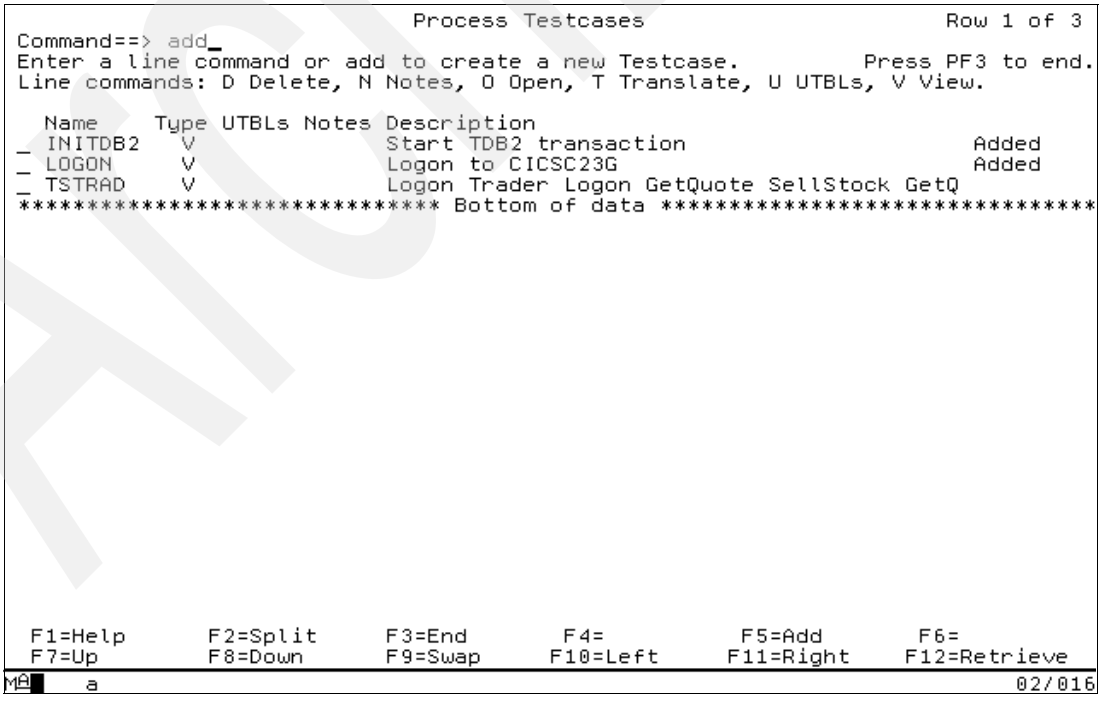


Figure 30-70 Test case INITDB2 added - continue with next test case

21. The next test case involves logging on to the trader application and is called TRADLOG, as shown in Figure 30-71.

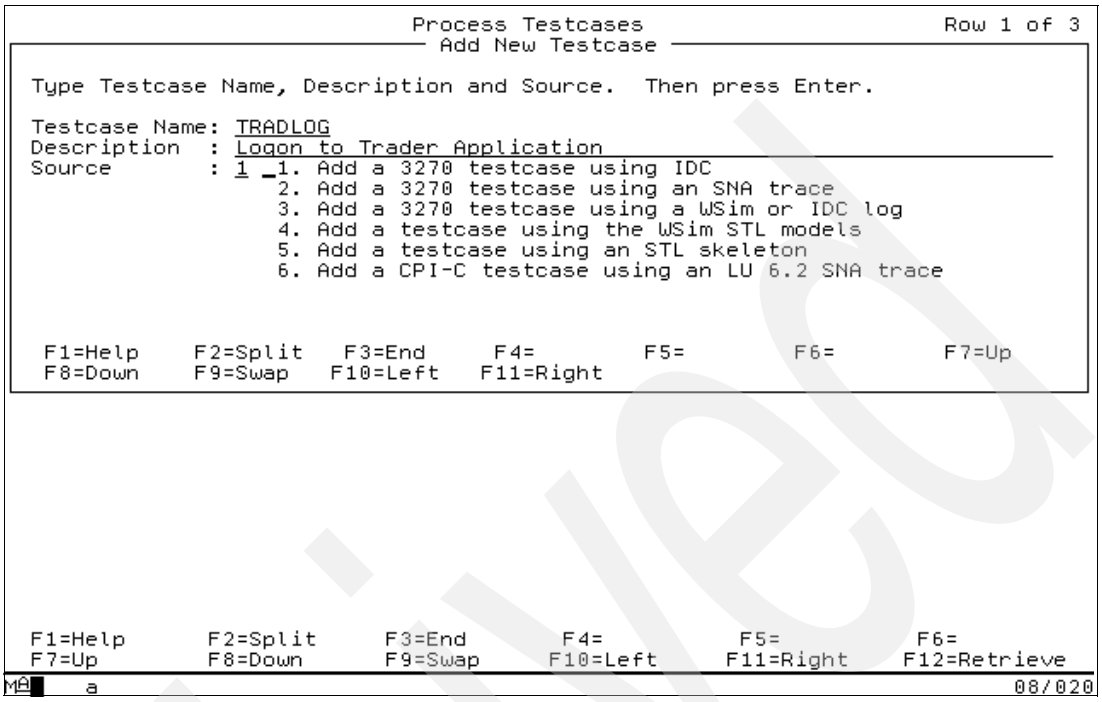


Figure 30-71 Adding test case TRADLOG

22. Start a session with the CICSC23G again and do not start data capture immediately, as shown in Figure 30-72.

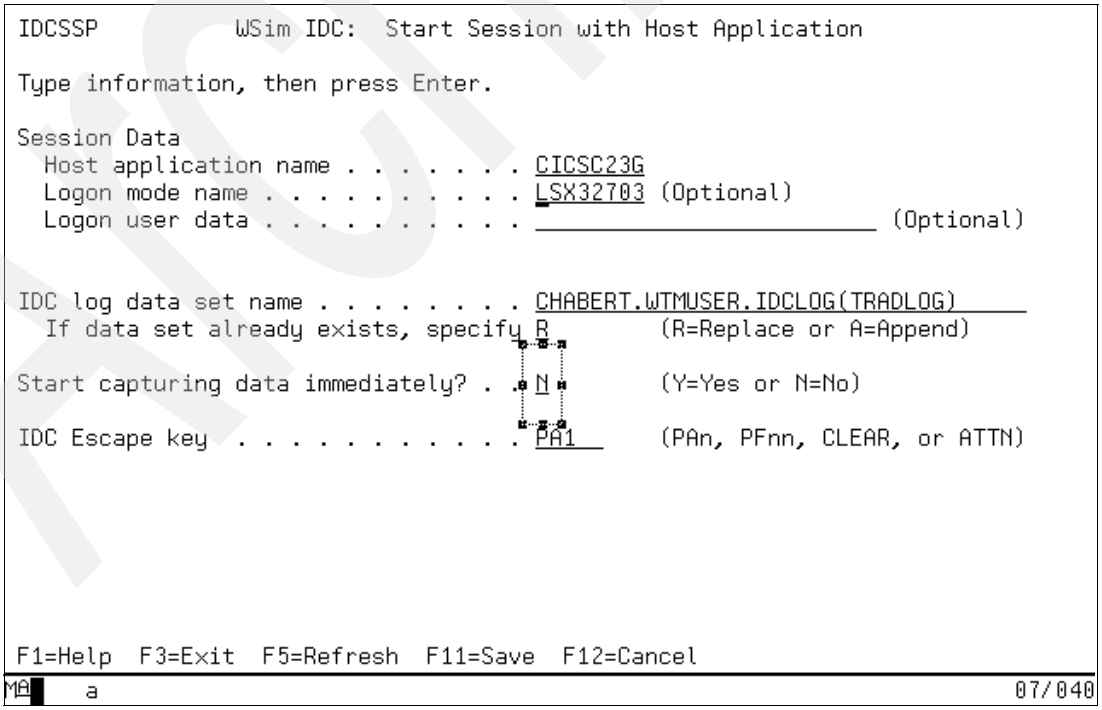


Figure 30-72 Data capture delayed for test case TRADLOG

23. Repeat all the steps to log on to the CICSC23G. Start the TDB2 transaction. At the data trader application logon screen, shown in Figure 30-73, press the IDC escape key (PA1).

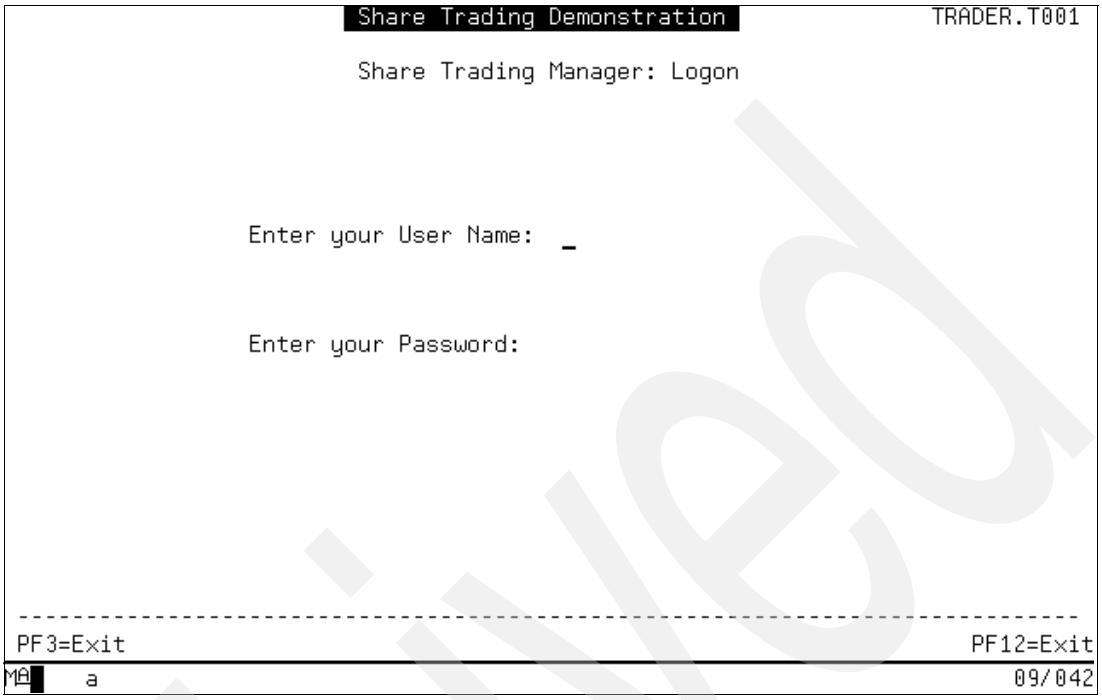


Figure 30-73 Trader application logon screen

24. Select option **1** on the IDC utility Escape Actions panel to start data capturing, as shown in Figure 30-74. Press Enter and PF3 to return to the CICS session.

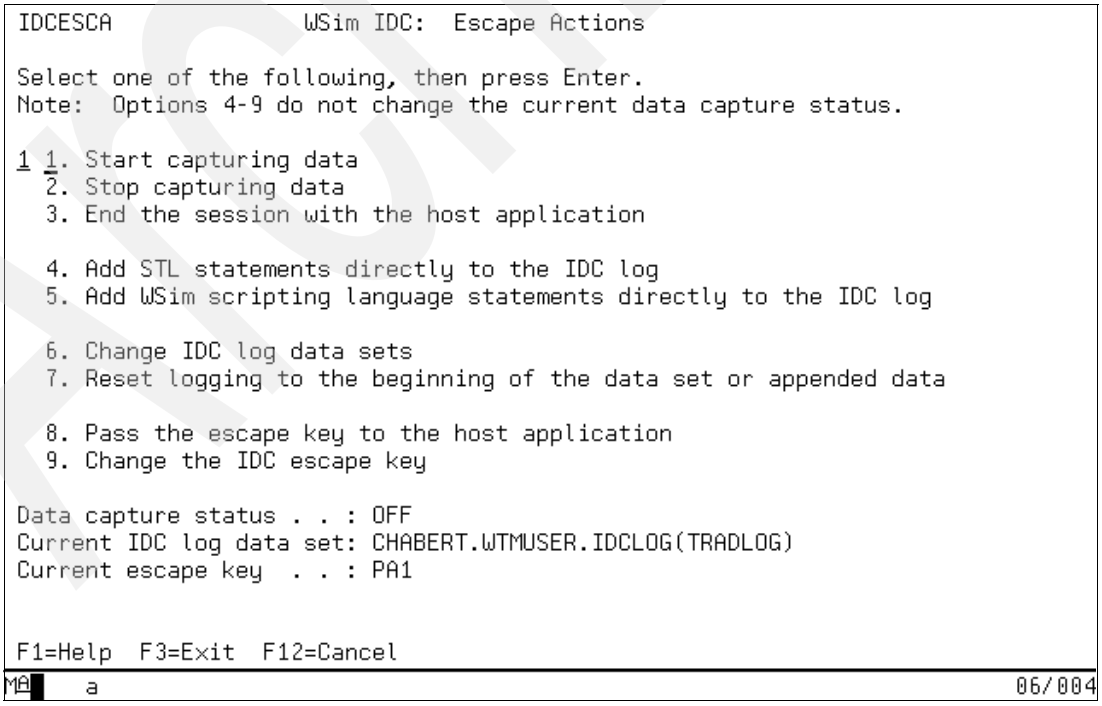


Figure 30-74 Starting data capture for test case TRADLOG

25. Continue with the trader application workflow. Press the IDC escape key (PA1) when at the Company Selection screen, shown in Figure 30-75.

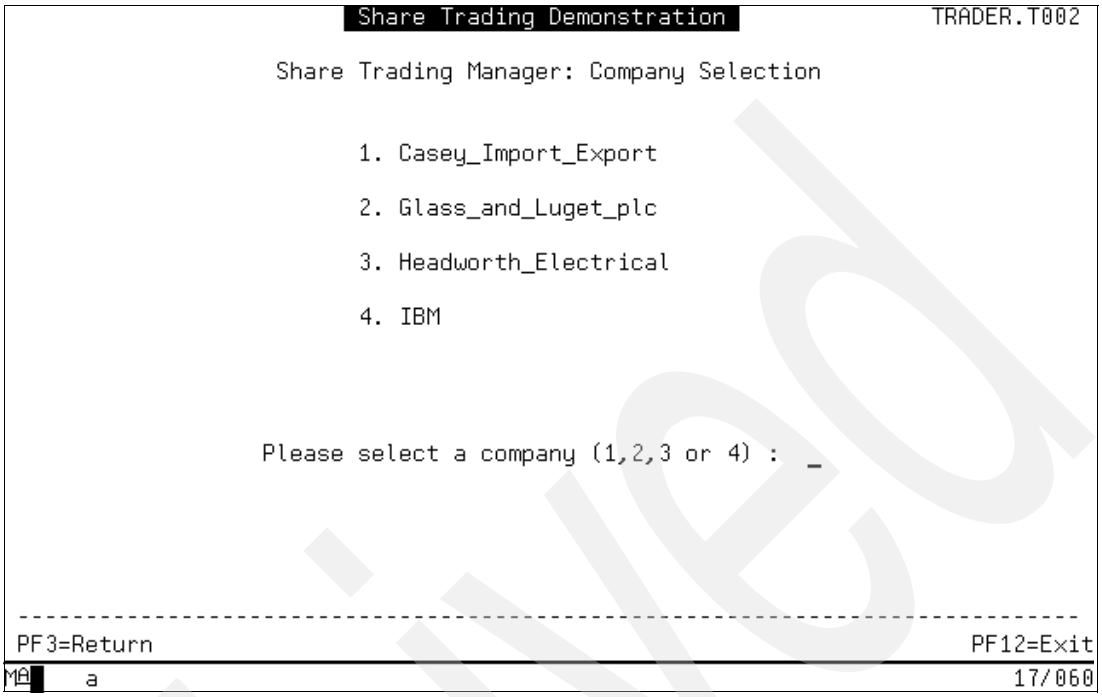


Figure 30-75 Trader application company selection screen

26. On the familiar IDC Escape Actions panel, shown in Figure 30-76, select option **6** and press Enter.

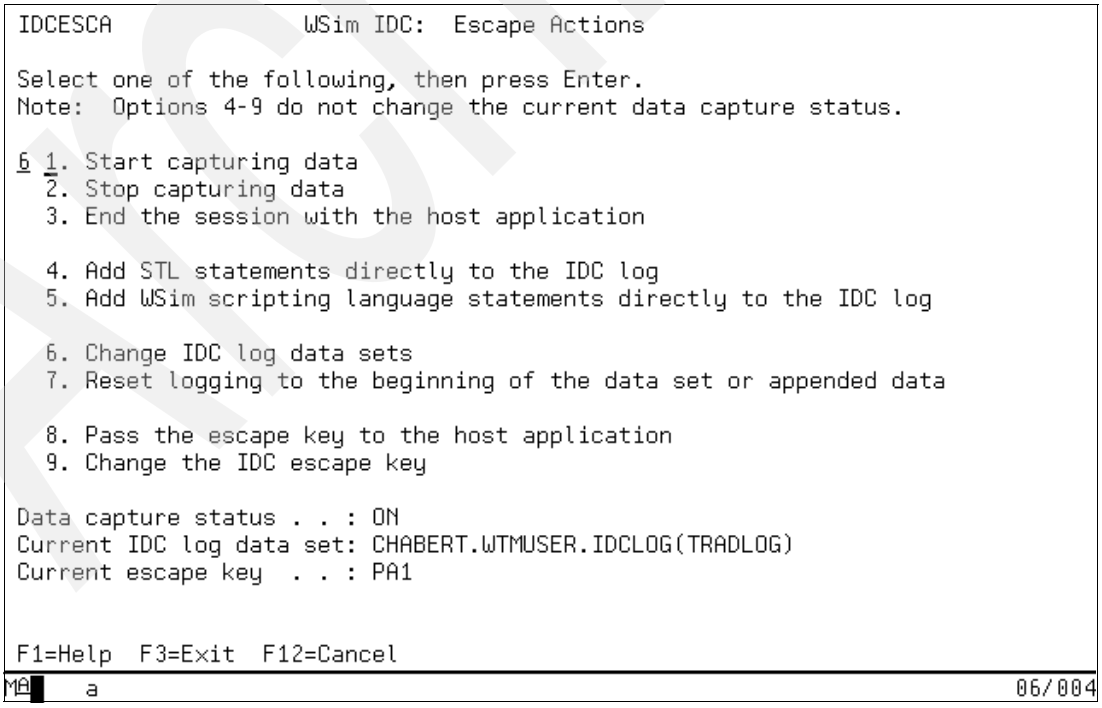


Figure 30-76 IDC Escape Actions panel

27. The IDC utility Change IDC Log Data Sets panel appears. Overtype the log data set name in the new IDC log data set name field with a new name, for example, selcomp (select company), as shown in Figure 30-77, and press Enter.

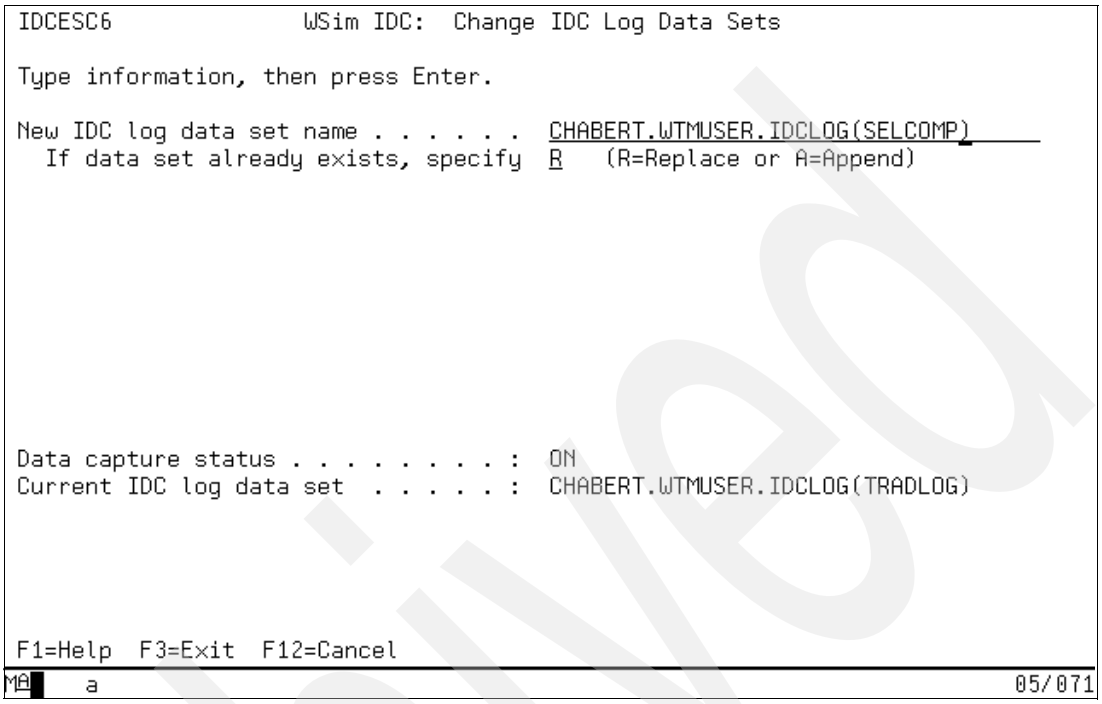


Figure 30-77 New log data set to be used

28. IDC displays the message confirming that the previous IDC log file is closed and the new one is opened, as shown in Figure 30-78.

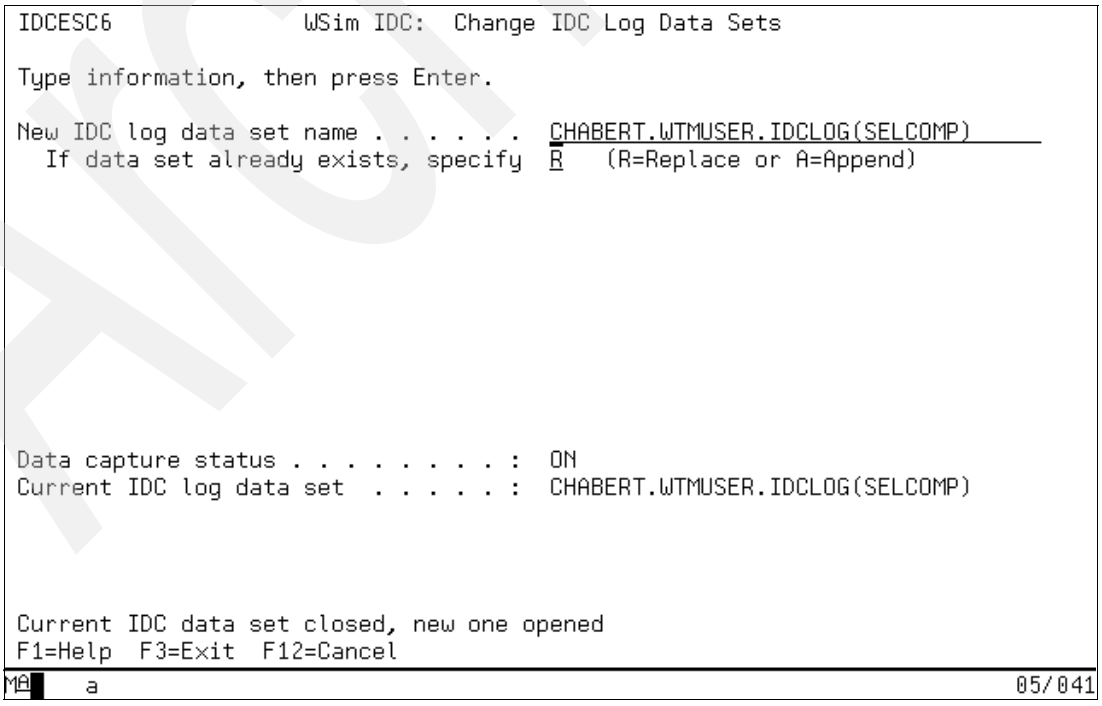


Figure 30-78 IDC log data sets switched

29. Press PF3 to return to the host application session. Continuing to work with the trader application, select a company and press Enter. When presented with the trader application screen for options selection, shown in Figure 30-79, press the IDC escape key (PA1).

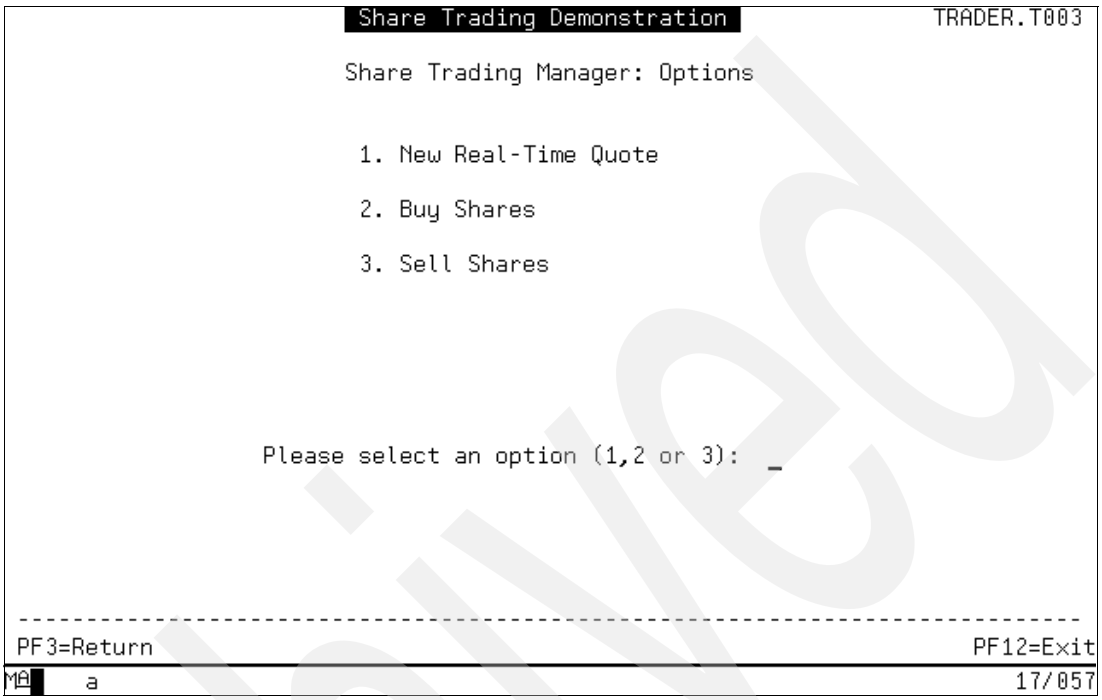


Figure 30-79 Trader application option selection screen

30. On the IDC utility Escape Actions panel select option **6**, as shown in Figure 30-80, to switch to the next IDC log file. Press Enter.

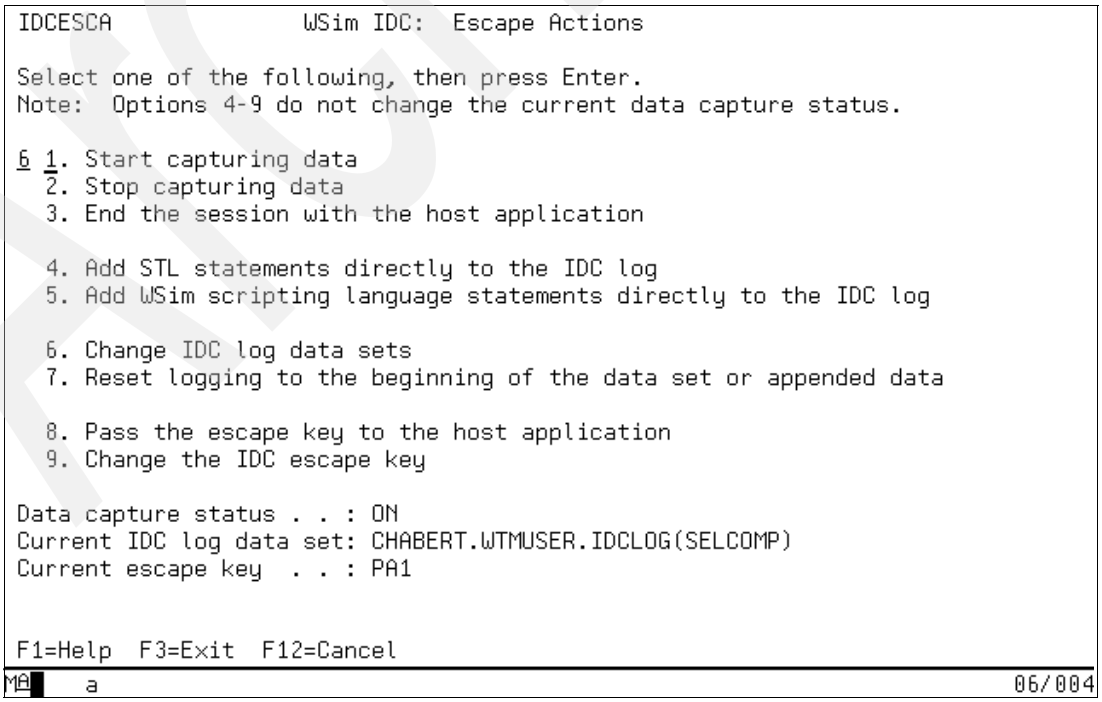


Figure 30-80 Changing log data sets

31. Overtype the current IDC log data set name with the new one and press Enter, as shown in Figure 30-81.

```
IDCESC6              WSim IDC:  Change IDC Log Data Sets

Type information, then press Enter.

New IDC log data set name . . . . . .  CHABERT.WTMUSER.IDCLOG(SELL1)
  If data set already exists, specify  R   (R=Replace or A=Append)









Data capture status . . . . . :  ON
Current IDC log data set  . . . . . :  CHABERT.WTMUSER.IDCLOG(SELCOMP)




F1=Help  F3=Exit  F12=Cancel
MA    a                                                        05/069
```

Figure 30-81 New log data set name entered

32. The old IDC log file is closed and the new one is opened, as shown in Figure 30-82.

```
IDCESC6              WSim IDC:  Change IDC Log Data Sets

Type information, then press Enter.

New IDC log data set name . . . . . .  CHABERT.WTMUSER.IDCLOG(SELL1)
  If data set already exists, specify  R   (R=Replace or A=Append)









Data capture status . . . . . :  ON
Current IDC log data set  . . . . . :  CHABERT.WTMUSER.IDCLOG(SELL1)




Current IDC data set closed, new one opened
F1=Help  F3=Exit  F12=Cancel
MA    a                                                        05/041
```

Figure 30-82 Log data sets switched

33. Press PF3 to continue working with the trader application. Sell 1 share, as shown in Figure 30-83.

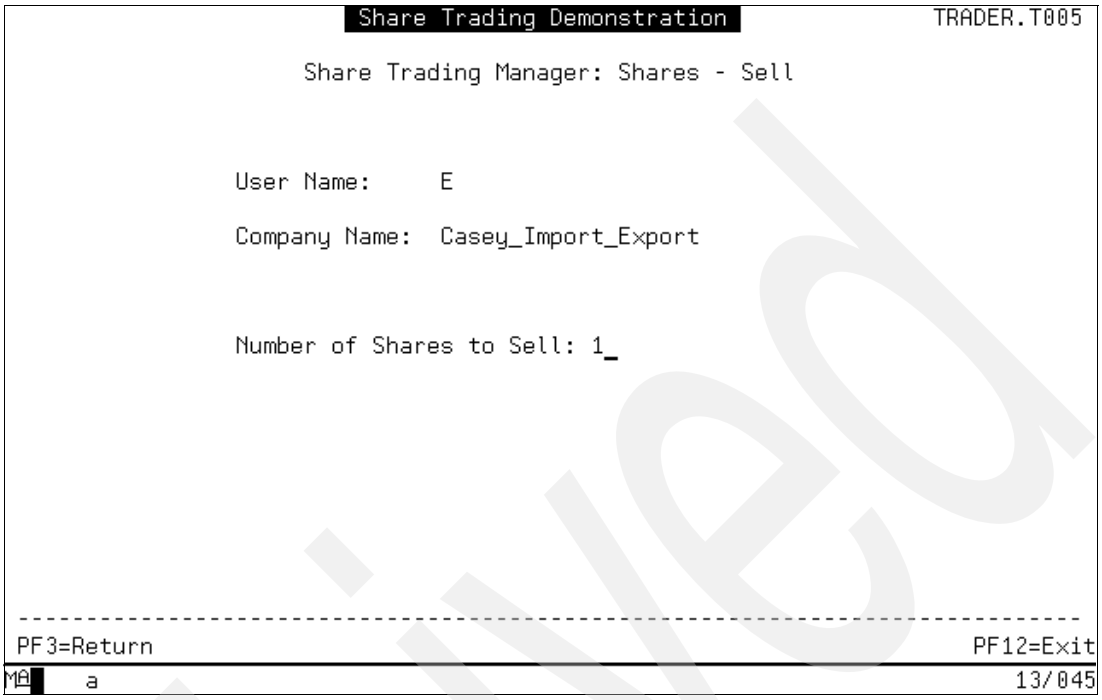


Figure 30-83 Trader application: selling 1 share

34. When done, press the IDC escape key PA1, select option **6** on the IDC utility Escape Actions panel to switch to the new IDC log file, press Enter, and type over the name of the current IDC log data set with the new one and press Enter, as shown in Figure 30-84.

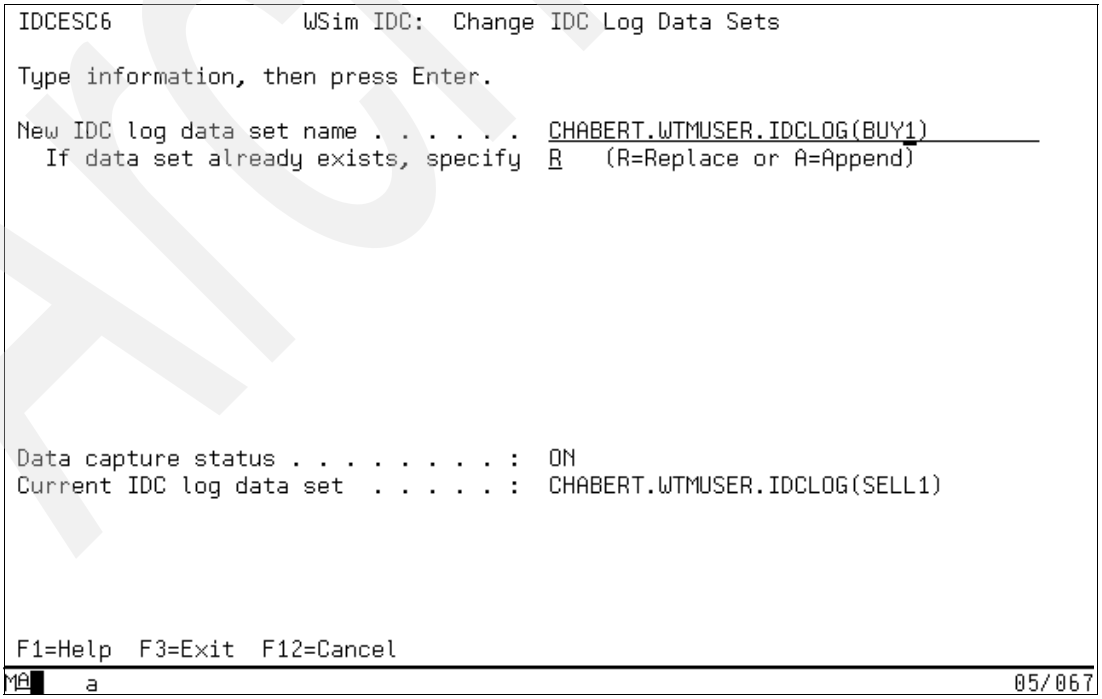


Figure 30-84 New data set log name BUY1 entered

35. The IDC utility closed the previous IDC log file and opened the new one, as shown in Figure 30-85.

```
IDCESC6              WSim IDC:  Change IDC Log Data Sets

Type information, then press Enter.

New IDC log data set name . . . . . .  CHABERT.WTMUSER.IDCLOG(BUY1)
  If data set already exists, specify  R   (R=Replace or A=Append)









Data capture status . . . . . :  ON
Current IDC log data set  . . . . . :  CHABERT.WTMUSER.IDCLOG(BUY1)



Current IDC data set closed, new one opened
F1=Help  F3=Exit  F12=Cancel
MA    a                                                        05/041
```

Figure 30-85 Old log data set closed, new opened

36. Press PF3 to return to the trader application and select option **2** to buy shares, as shown in Figure 30-86. Press Enter.

```
                    Share Trading Demonstration                TRADER.T003

                    Share Trading Manager: Options

                     1. New Real-Time Quote

                     2. Buy Shares

                     3. Sell Shares




                Please select an option (1,2 or 3):  2_



Request Completed OK
--------------------------------------------------------------------------------
PF3=Return                                                          PF12=Exit
MA    a                                                          17/058
```

Figure 30-86 Trader application: buy shares selected

37. The number of share to buy is 1, as shown in Figure 30-87.

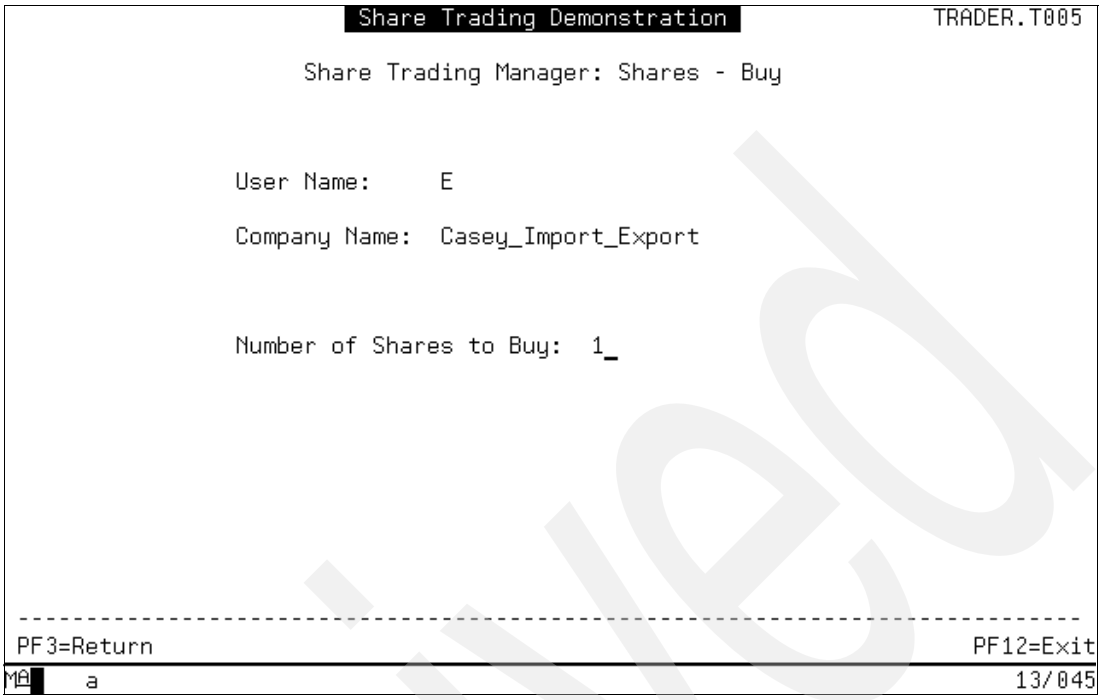


Figure 30-87 Trader application: buy 1 share

38. The request to buy is completed, as shown in Figure 30-88.

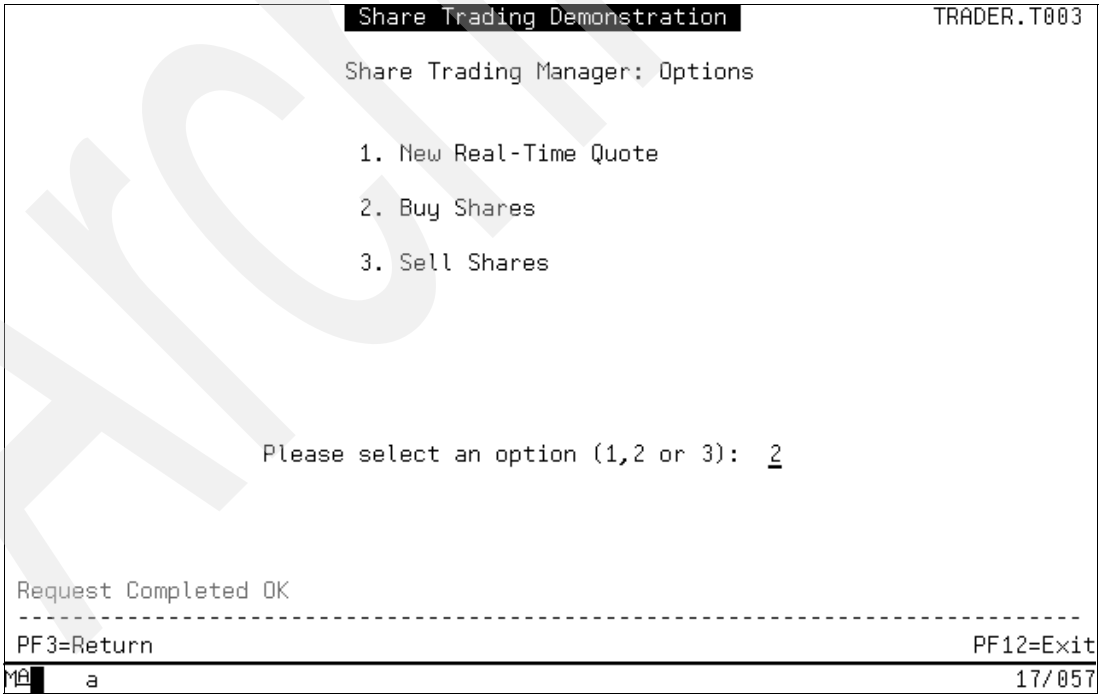


Figure 30-88 Trader application: request completed

39. Press the IDC escape key (PA1), select option **6** on the IDC utility Escape Actions panel (as shown in Figure 30-89), and press Enter.

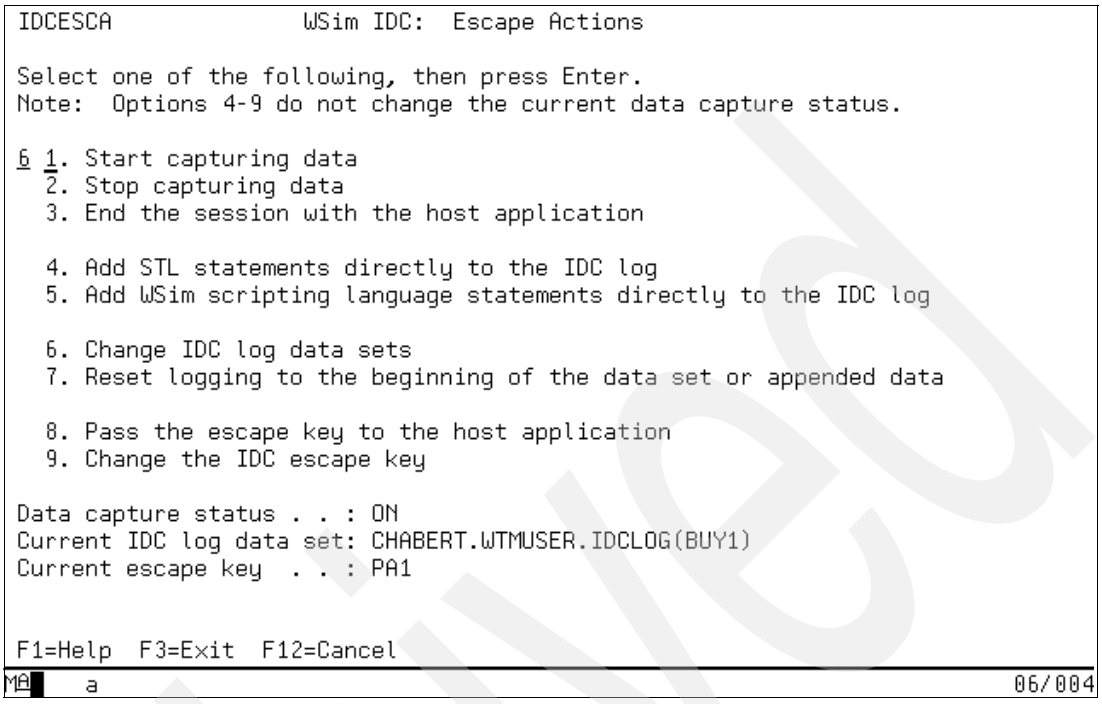


Figure 30-89 Change of log data sets selected

40. Type over the name of the current IDC log data set to start a new one and press Enter, as shown in Figure 30-90.

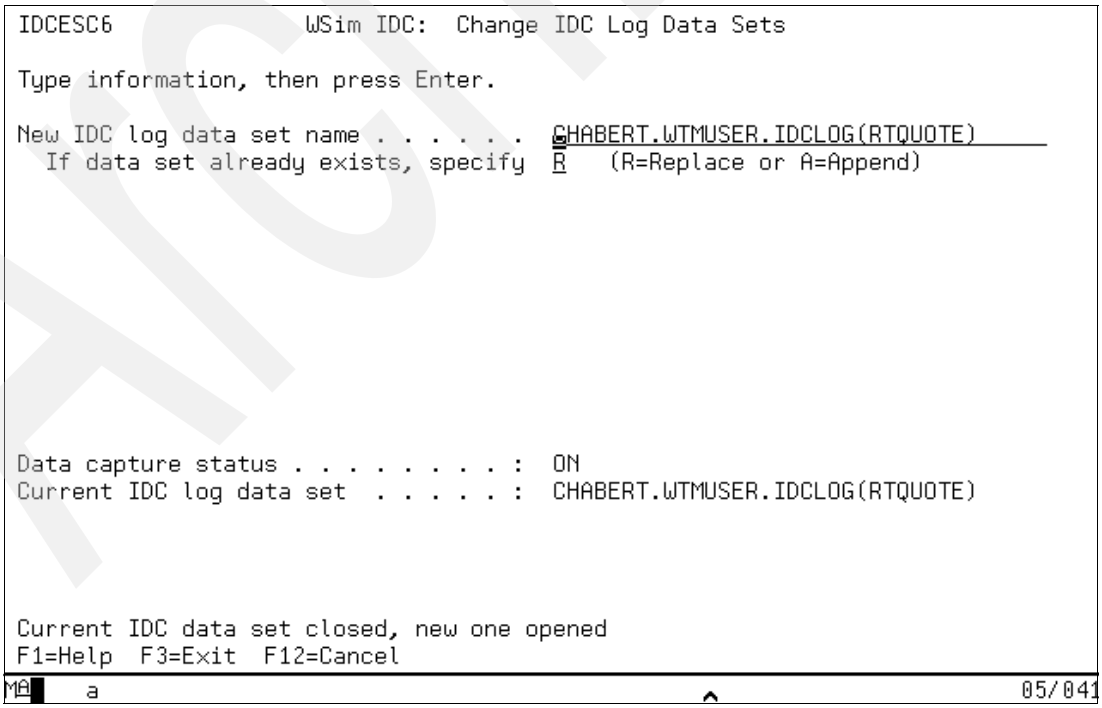


Figure 30-90 Log data set for RQUOTE is opened

41. When the switch of the IDC log files is confirmed, press PF3 to return to the host session. Select option **1** to request a quote, as shown in Figure 30-91.

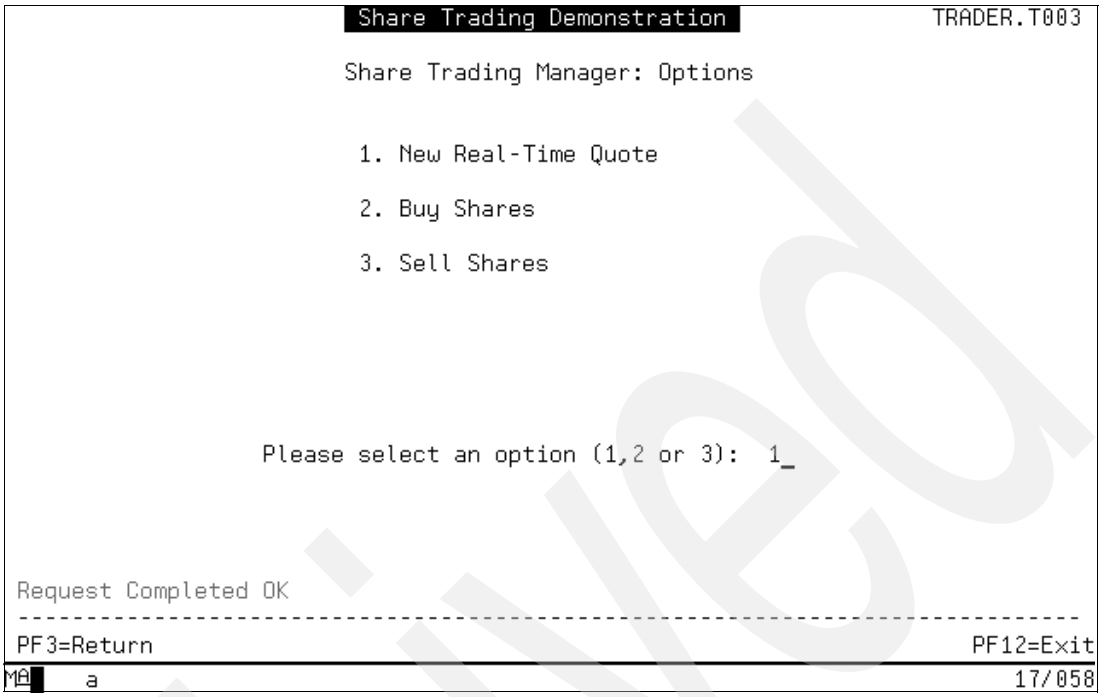


Figure 30-91 Trader application: quote requested

42. The results are presented in Figure 30-92.

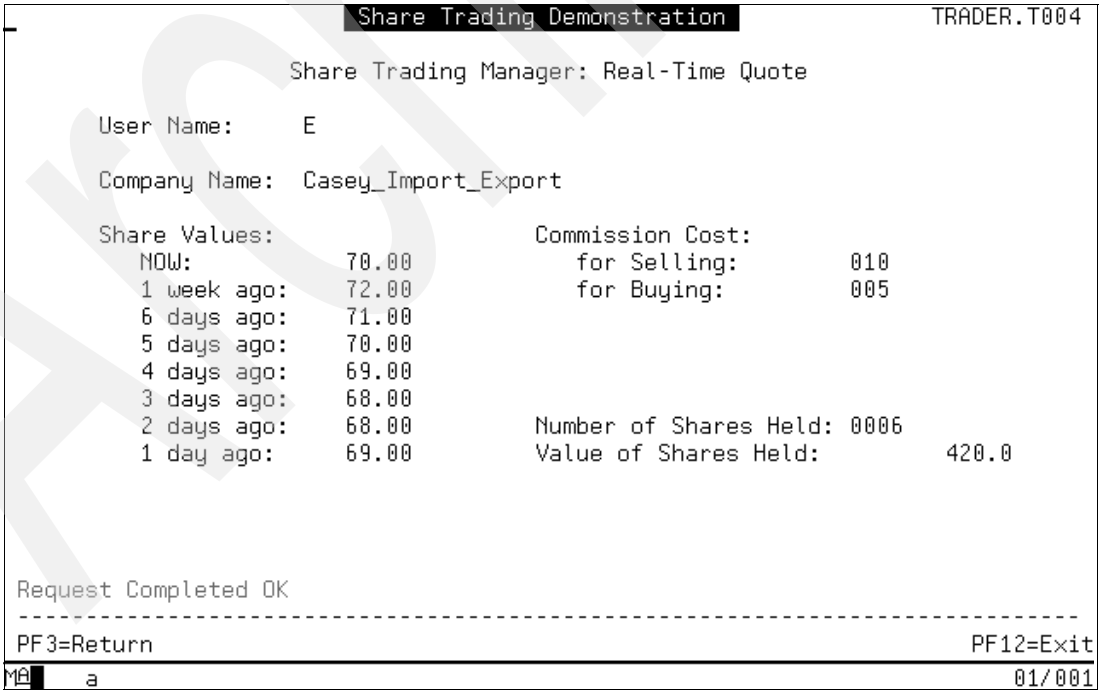


Figure 30-92 Trader application: real-time quote

43. Press PF12 to exit from the trader application. The trader application session is over, as shown in Figure 30-93.

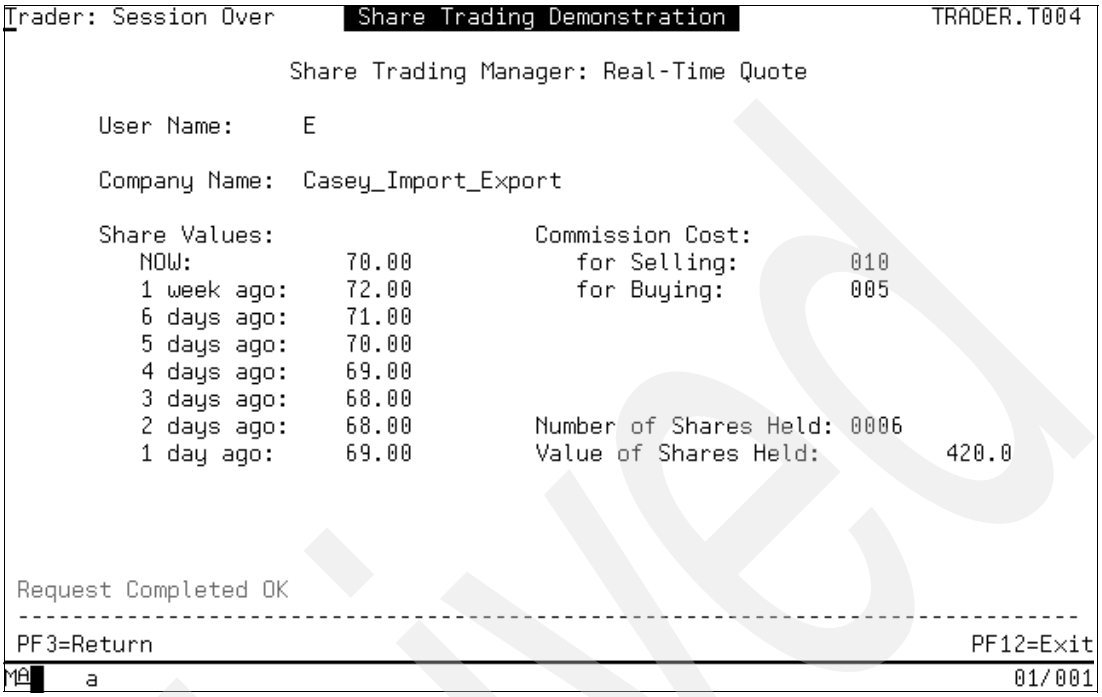


Figure 30-93 Trader application: session over

44. Press the IDC escape key (PA1) to switch IDC log files again, as shown in Figure 30-94. Select option **6**, type over the current name with the new one, and press Enter.

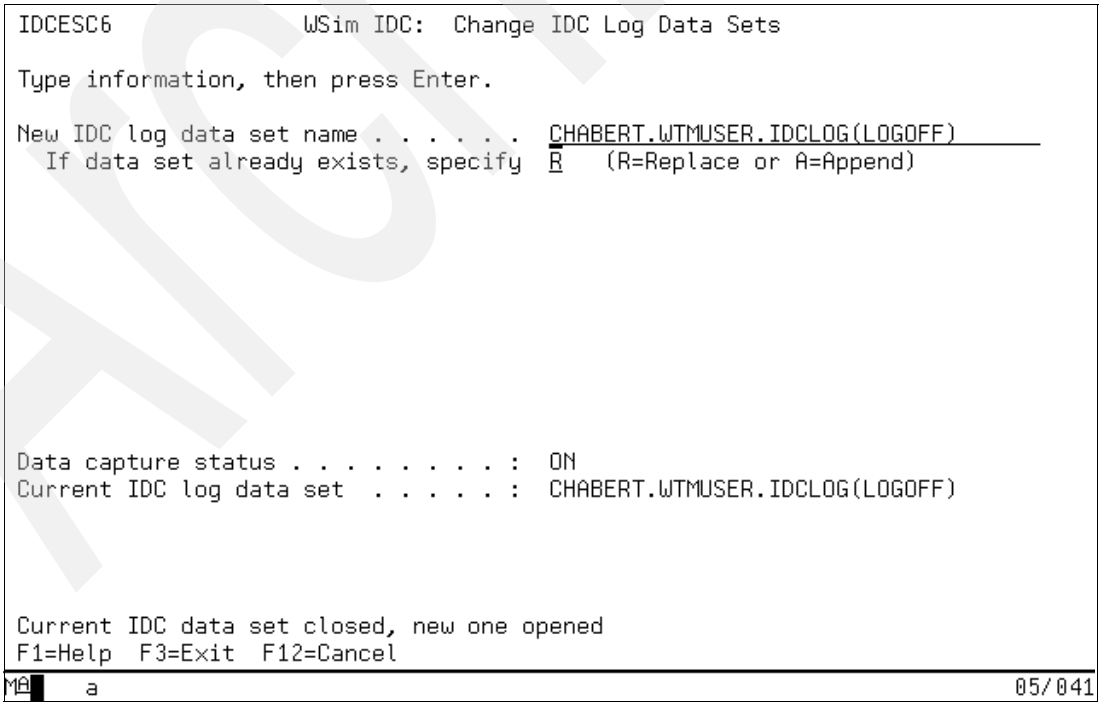


Figure 30-94 Log data set for LOGOFF opened

45. Press PF3 to return to the host application session and start the CESF LOGOFF transaction to end the session with CICSC23G, as shown in Figure 30-95.

Figure 30-95 CICS logoff initiated

46. Note the difference in the results of the transactions CESF and CESF LOGOFF. In the case of the former, the IDC utility Escape Actions panel was presented and option **3** had to be selected to finish the host application session. In the case of the latter, the host application session is ended immediately and the IDC utility presents the confirmation message, as shown in Figure 30-96.

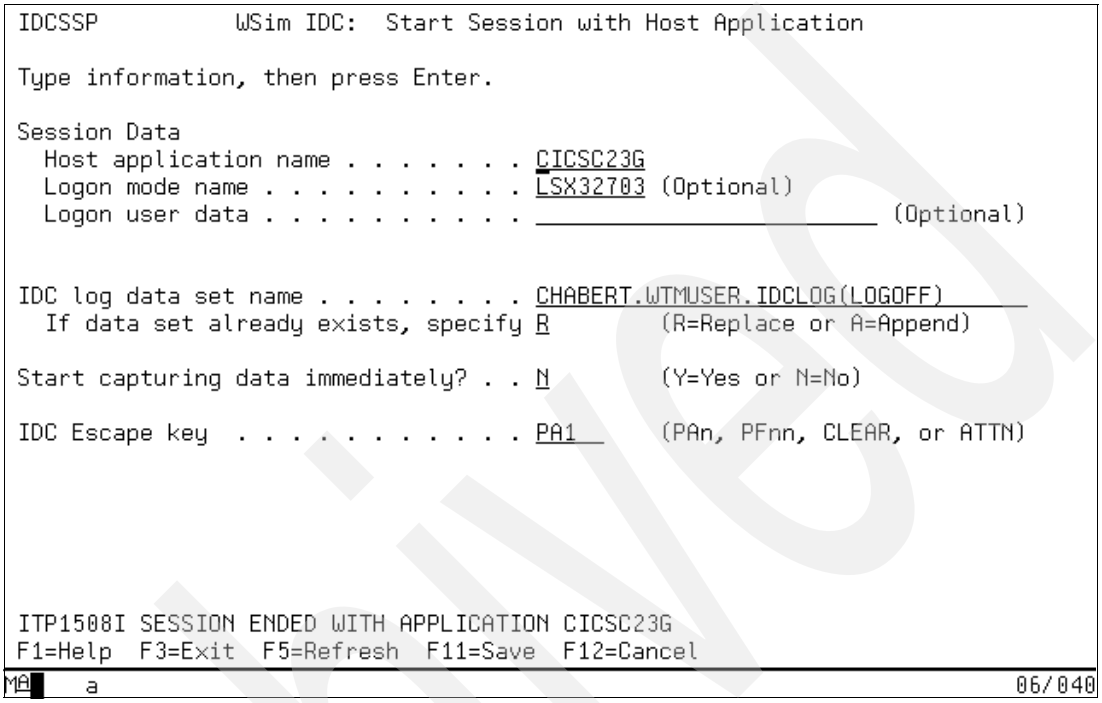


Figure 30-96 CICS session ended

47. Press PF3. WTM performs the steps required to generate a test case and presents the list of test cases confirming that the test case TRADLOG is added. What happened to all of the other test cases we were preparing when changing IDC log files? WTM automatically generated only the first of them, as shown in Figure 30-97.

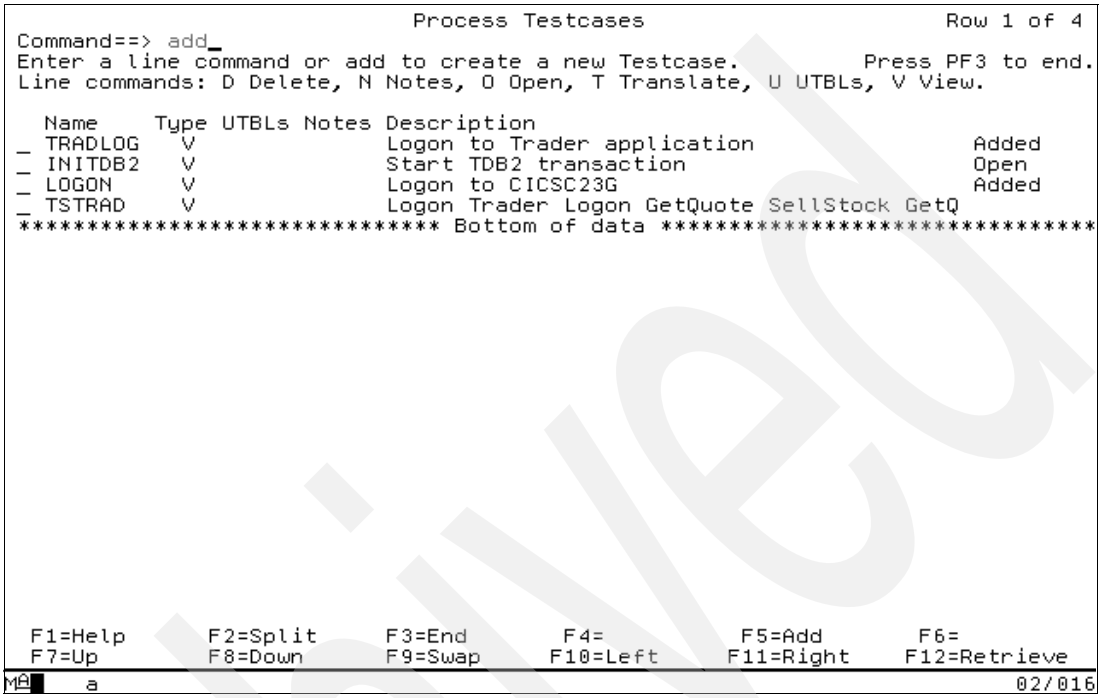


Figure 30-97 Process test cases

48. We have to add them one-by-one using source option 3 on the WTM Add New Test case panel, shown in Figure 30-98.

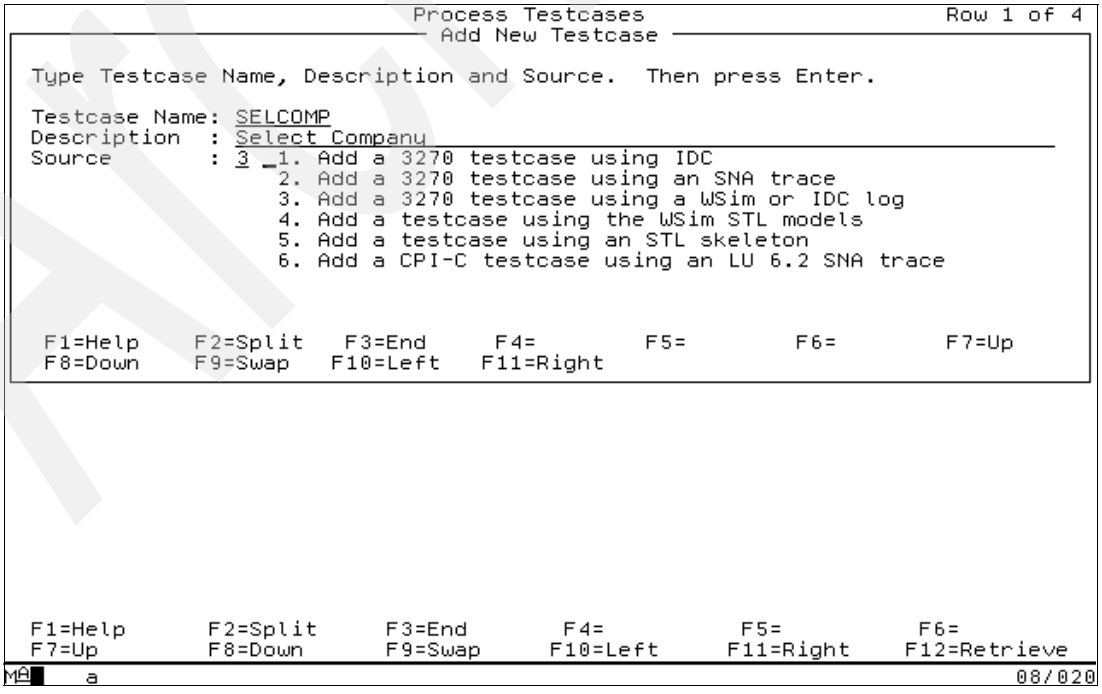


Figure 30-98 Adding new test case SELCOMP

49. Provide the test case name and description and press Enter. On the next WTM panel type over he name of the IDC log file for the corresponding test case and press Enter, as shown in Figure 30-99.

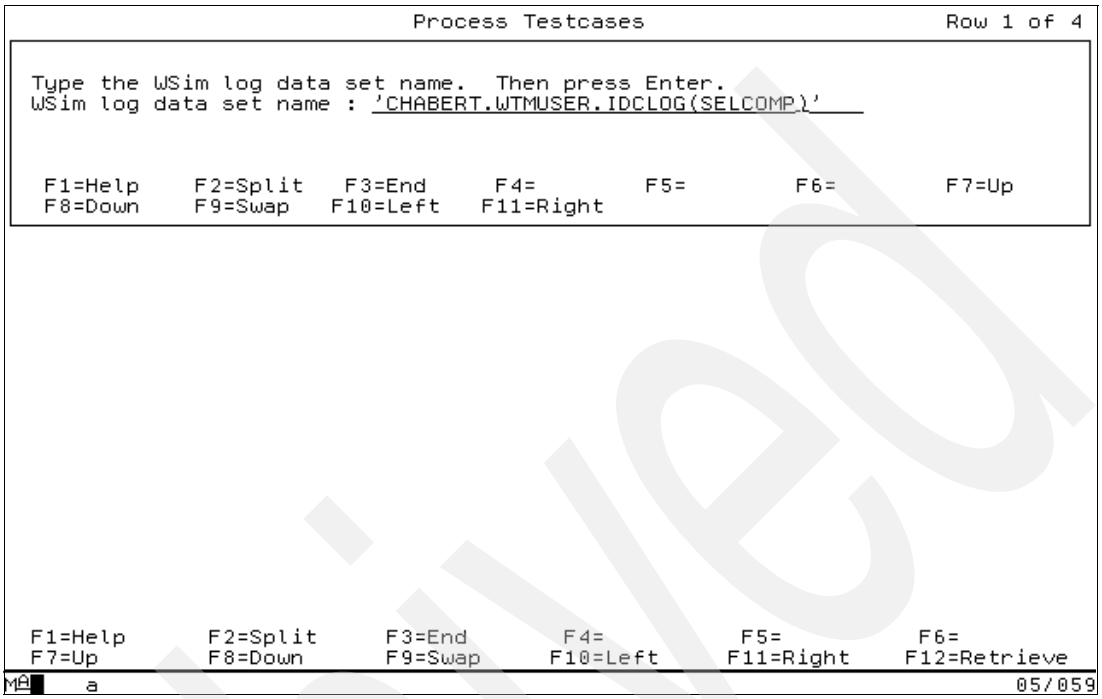


Figure 30-99 Log data set for test case SELCOMP

50. WTM generates the test case SELCOMP and displays the confirmation message. Enter the command **add** to continue to add new test cases from the captured IDC log files, as shown in Figure 30-100.

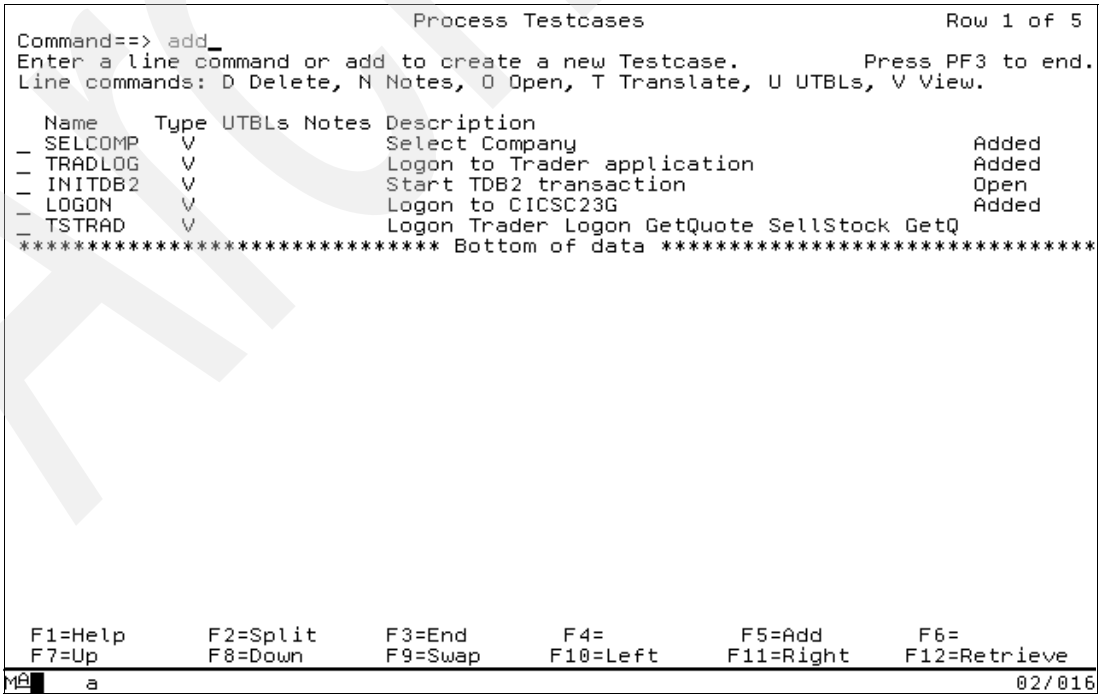


Figure 30-100 Test case SELCOMP added

51. The next test case is SELL 1, as shown in Figure 30-101.

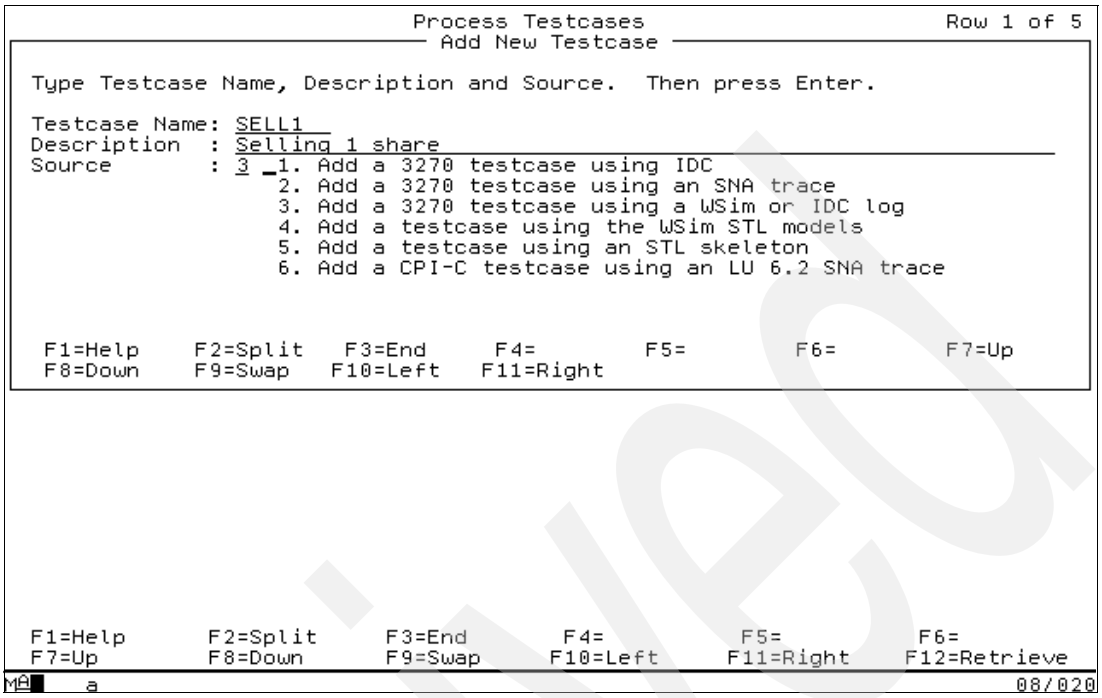


Figure 30-101 Adding test case SELL 1

52. Type over the name of the corresponding IDC log files, as shown in Figure 30-102.

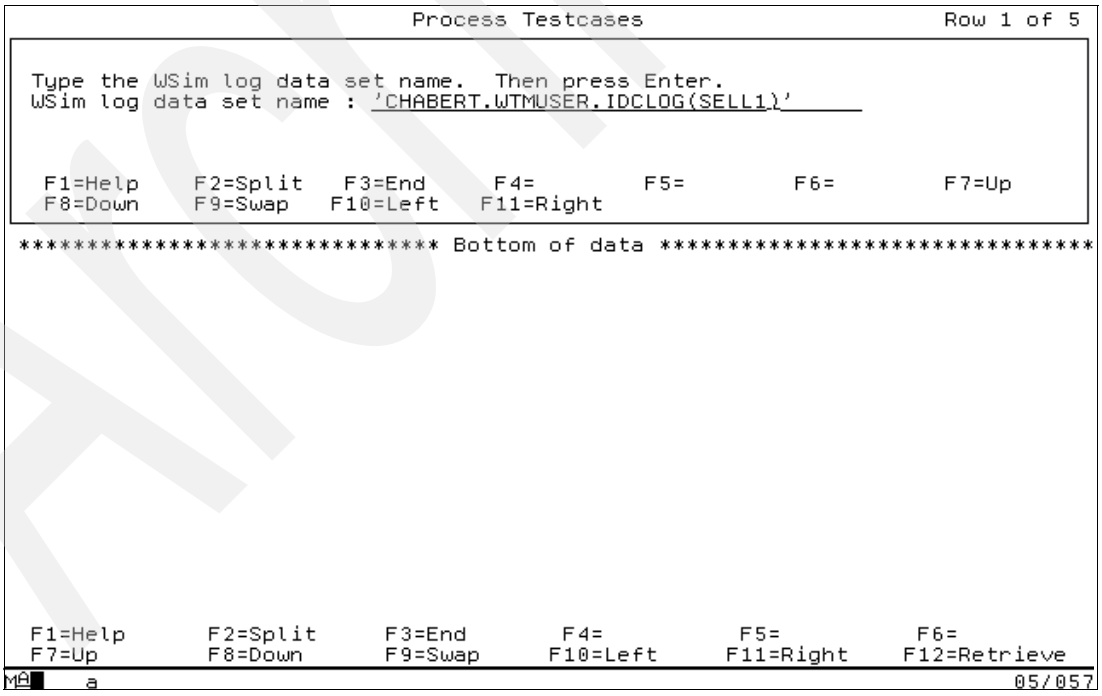


Figure 30-102 Log data set for test case SELL 1

53. The test case SELL 1 is added. Continue to add test cases, as shown in Figure 30-103.

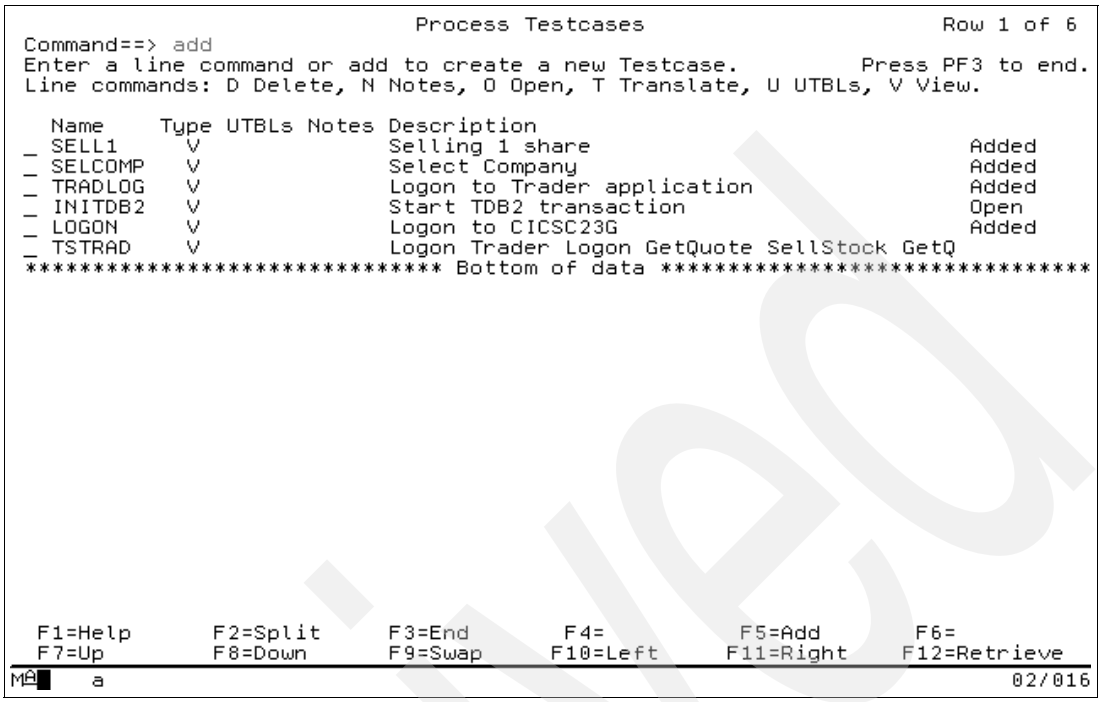


Figure 30-103 Test case SELL 1 added

54. The next test case is BUY 1, as shown in Figure 30-104.

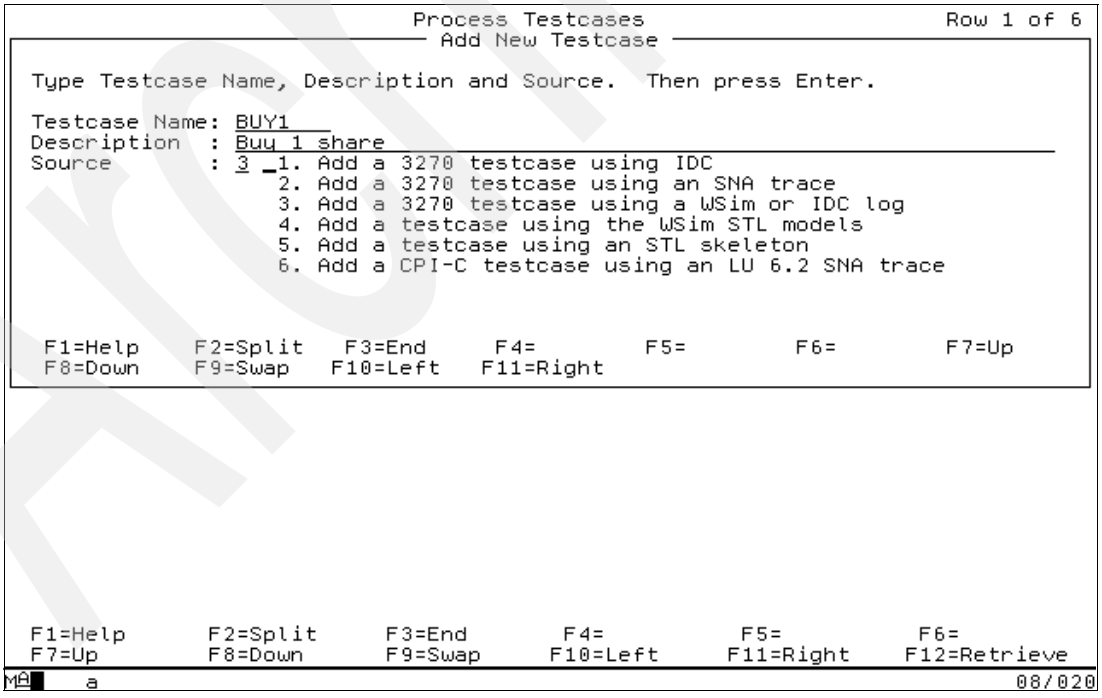


Figure 30-104 Adding test case BUY 1

55. Type over the name of the corresponding IDC log file, as shown in Figure 30-105.

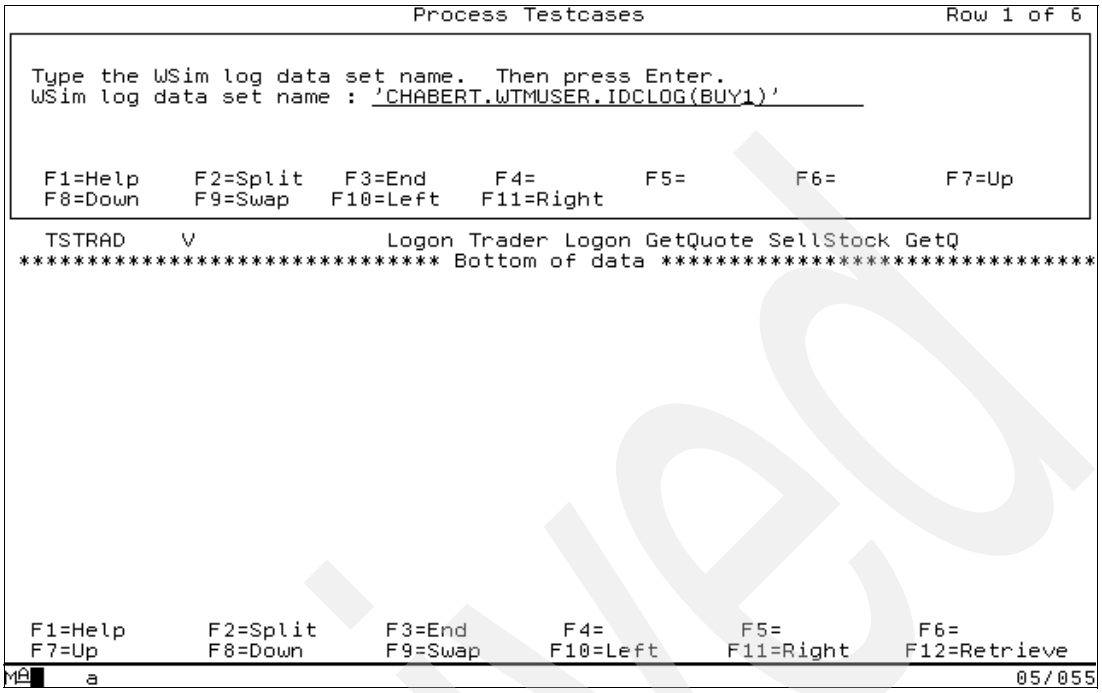


Figure 30-105 Log data set for test case BUY 1

56. Test case BUY 1 is successfully added. Continue adding test cases, as shown in Figure 30-106.

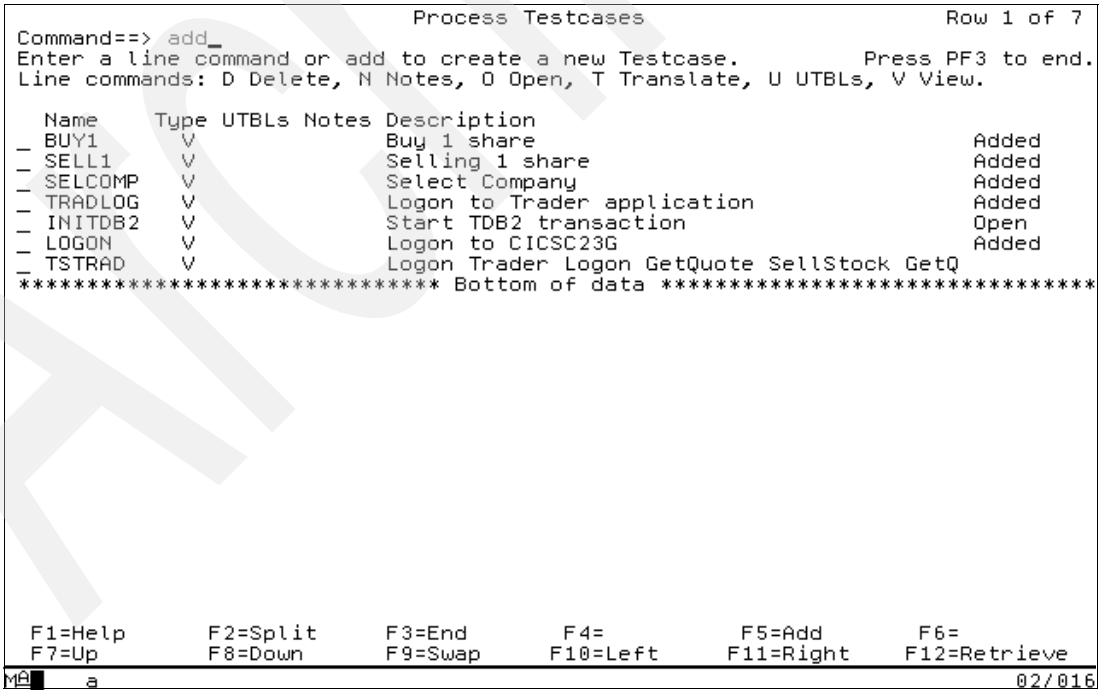


Figure 30-106 Test case BUY 1 added

57. The next test case is RTQUOTE, as shown in Figure 30-107.

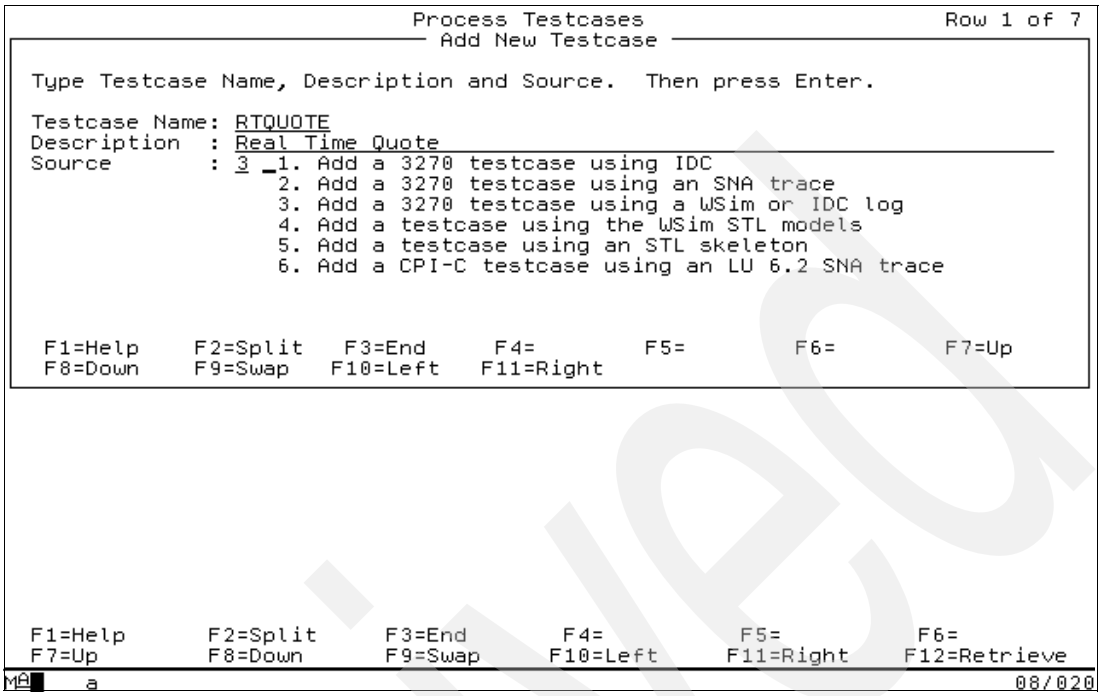


Figure 30-107 Adding test case RTQUOTE

58. Once again, type over the name of the IDC of the data set for the test case RTQUOTE, as shown in Figure 30-108.

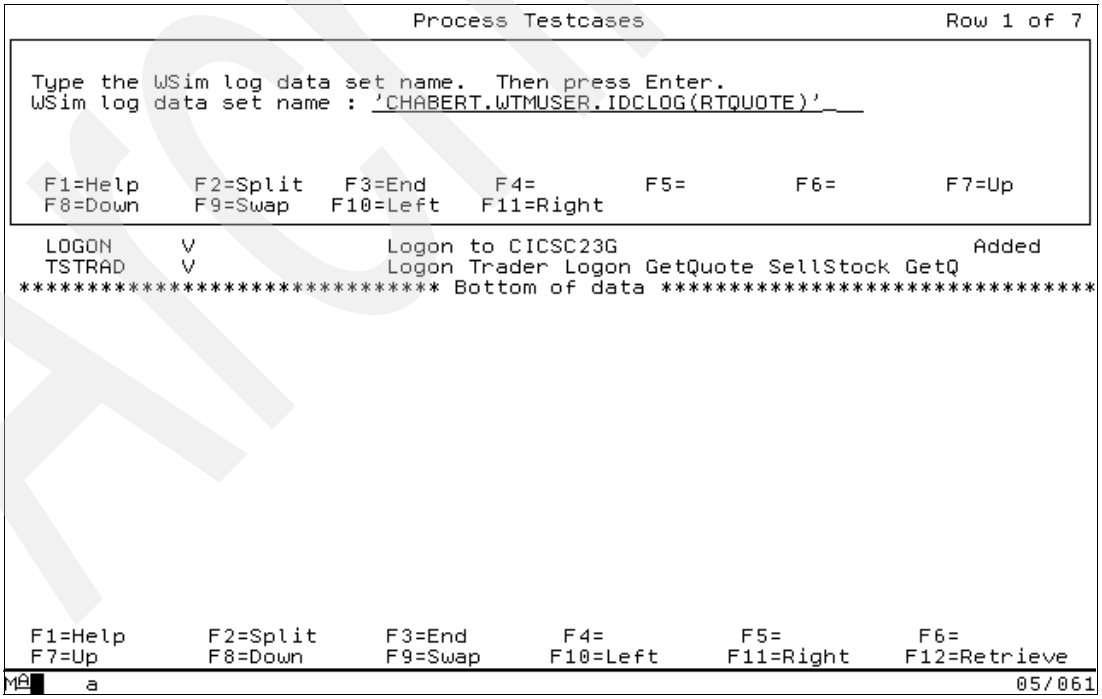


Figure 30-108 Log data set for test case RTQUOTE

59. WTM generates the test case, going through the required steps, as shown in Figure 30-109.

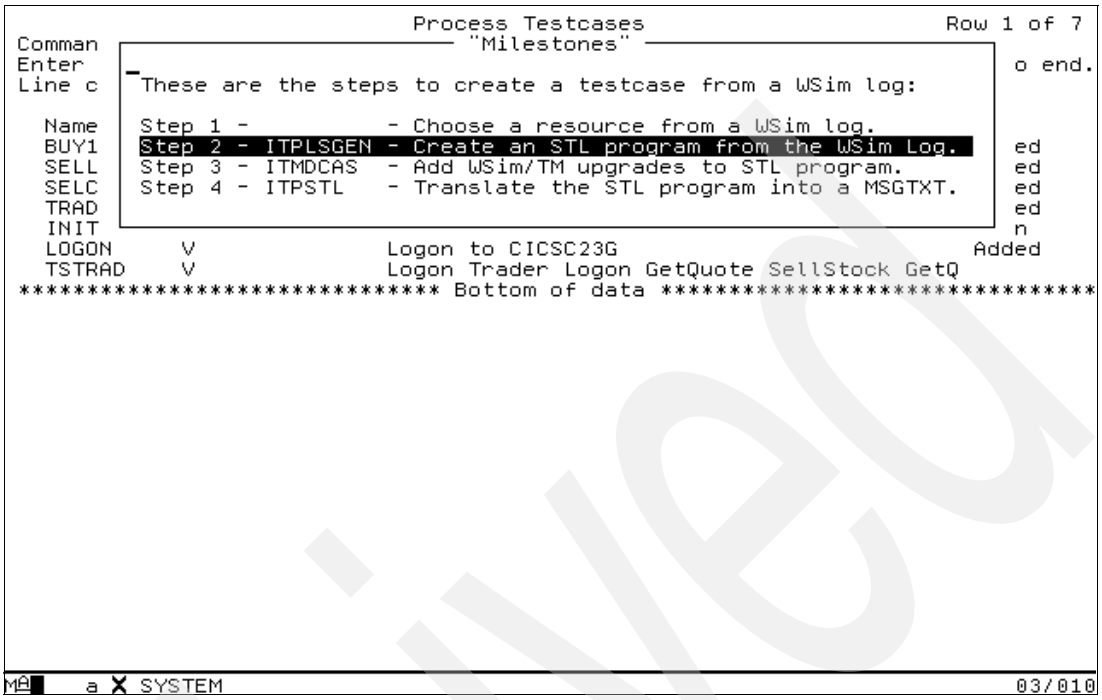


Figure 30-109 "Milestones" panel presents test case creation steps

60. The test case RTQUOTE generation is finished and is added to the list. One test case still has to be generated, as shown in Figure 30-110.

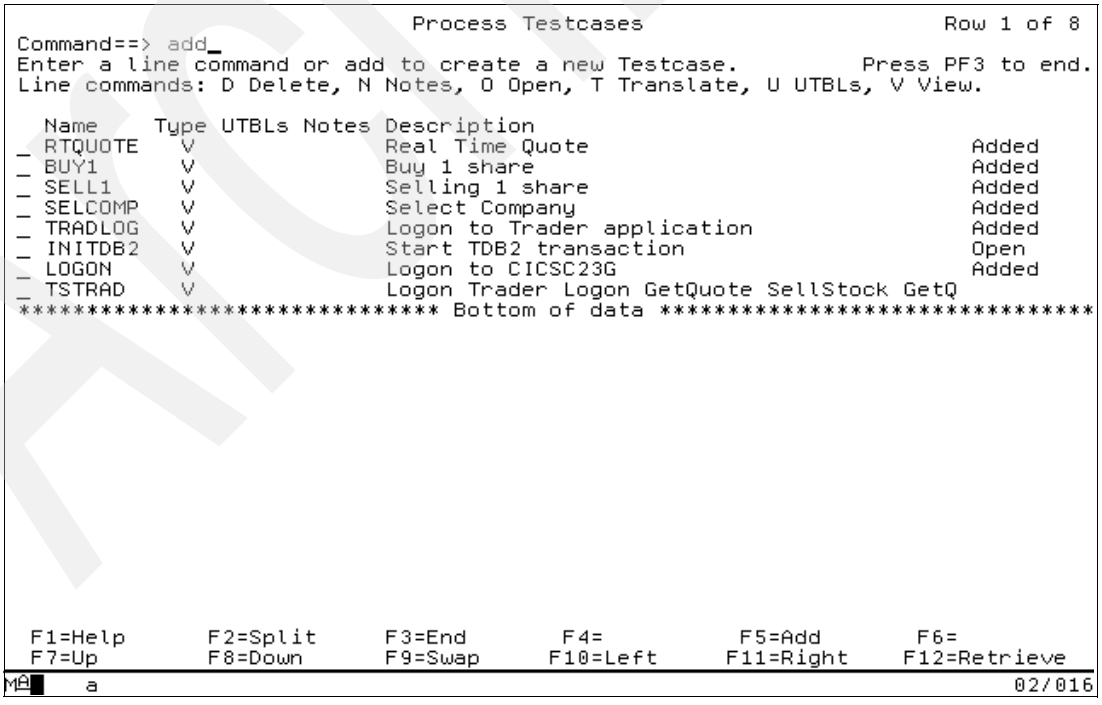


Figure 30-110 Test case RTQUOTE added

61. The test case LOGOFF to log off from CICS is generated from the log file, as shown in Figure 30-111.

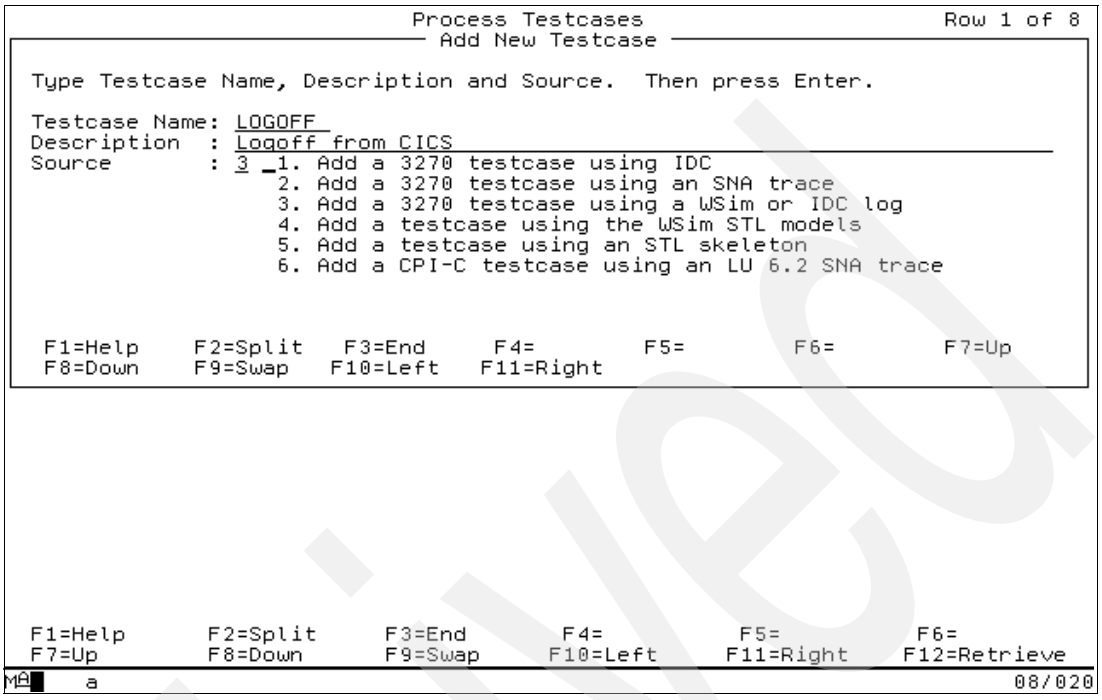


Figure 30-111 Adding test case LOGOFF

62. WTM asks for the name of the corresponding log data set. Type over the previous name with the new one, as shown in Figure 30-112, and press Enter.

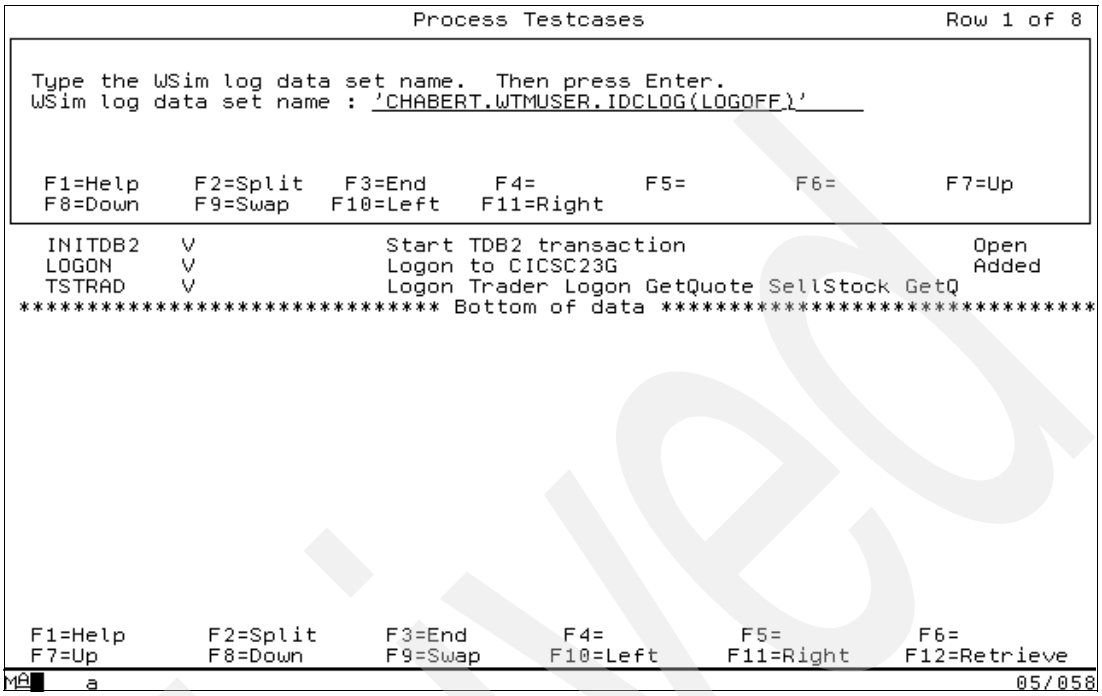


Figure 30-112 Log data set for test case LOGOFF

63. All captured IDC logs are now processed. The final list of the generated test cases is presented in Figure 30-113.

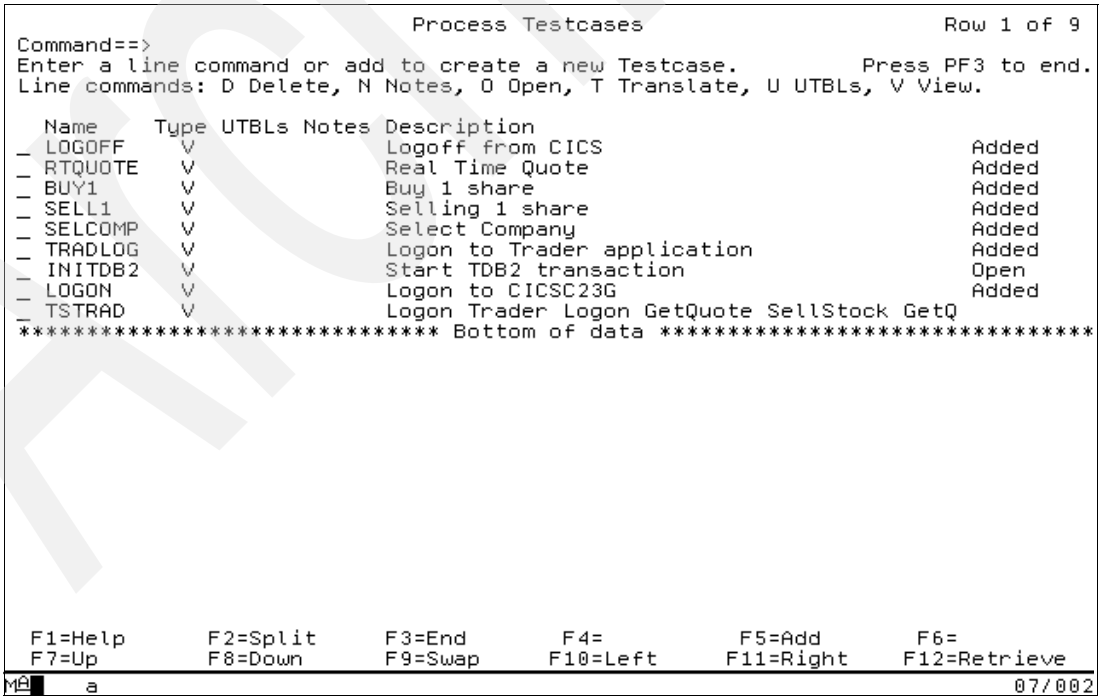


Figure 30-113 All test cases added

30.7.2 Creating a test group

A *test group* is an ordered list of test items (test cases and MSGTXTs). This section describes the steps for creating a test group:

1. Select option **2** or enter the GROUP command from the WTM main menu. The WTM panel Process Test groups is presented, as shown in Figure 30-114. There are no test groups currently, so issue the ADD command to create one.

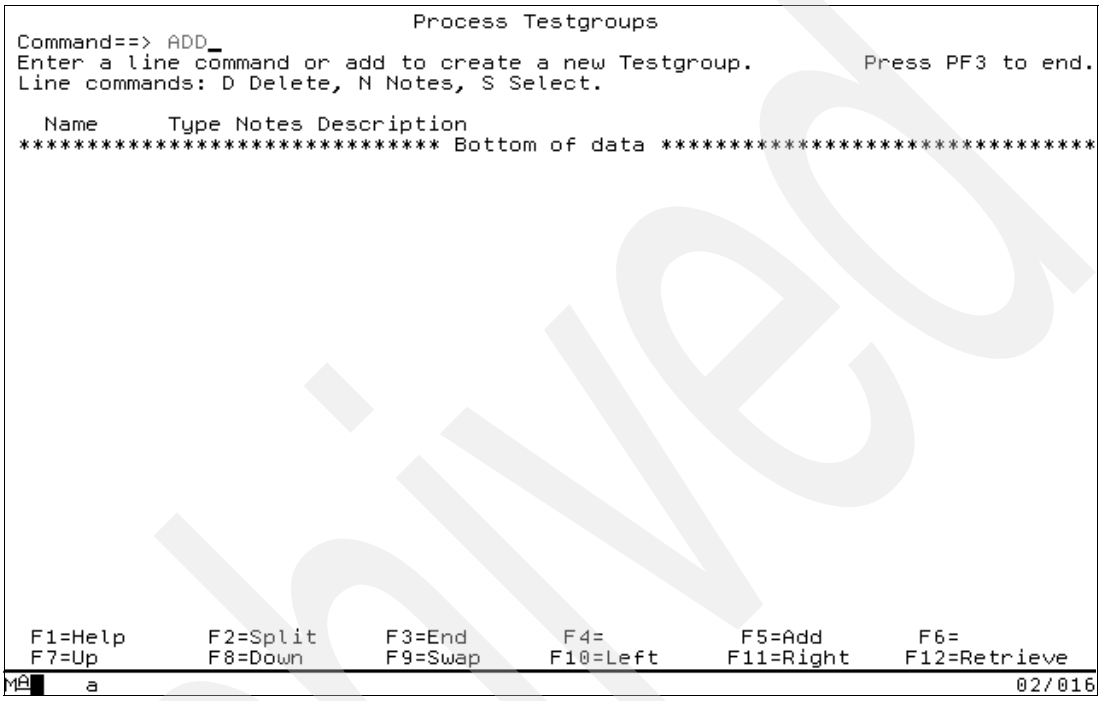


Figure 30-114 WTM Process Testgroups panel

The Type column has the following valid values (which are specified when a test group is added):

V VTAMAPPL
T TCP/IP
C CPI-C

2. A new test group TRADFULL, which includes a full session with the trader application, is being added, as shown in Figure 30-115.

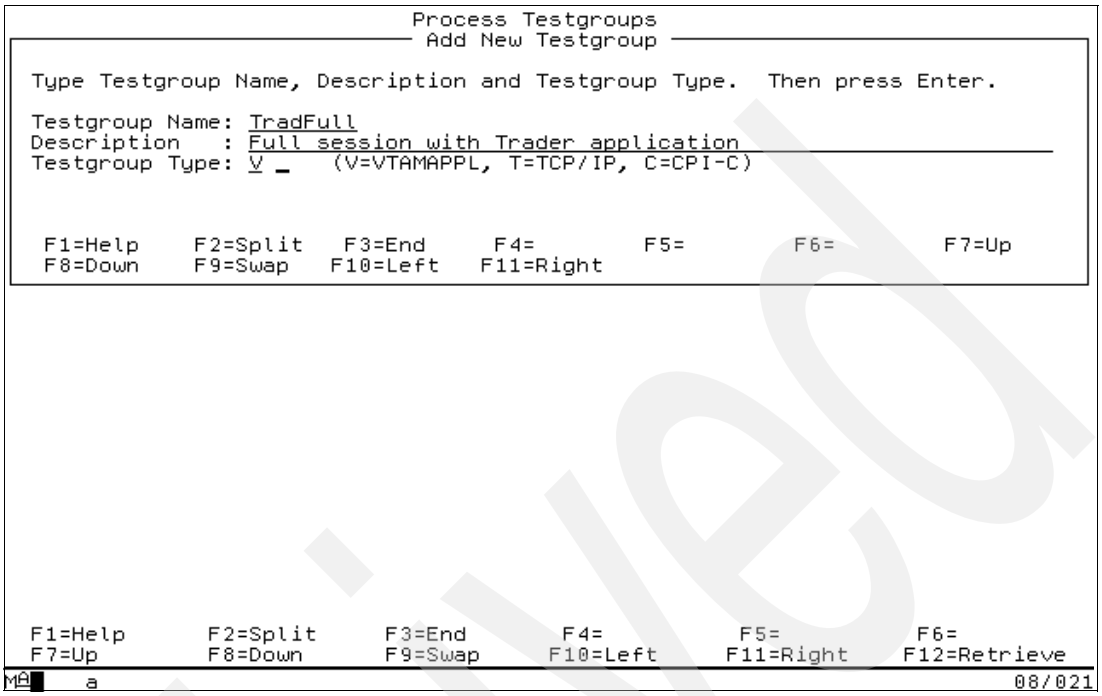


Figure 30-115 New test group TradFull

3. The order for test cases to be included in the test group, TradFull is defined on the next panel, shown in Figure 30-116. It is possible to use the same test case several times. In this exercise the test cases BUY1 and SELL1 are used more than once in the sequence.

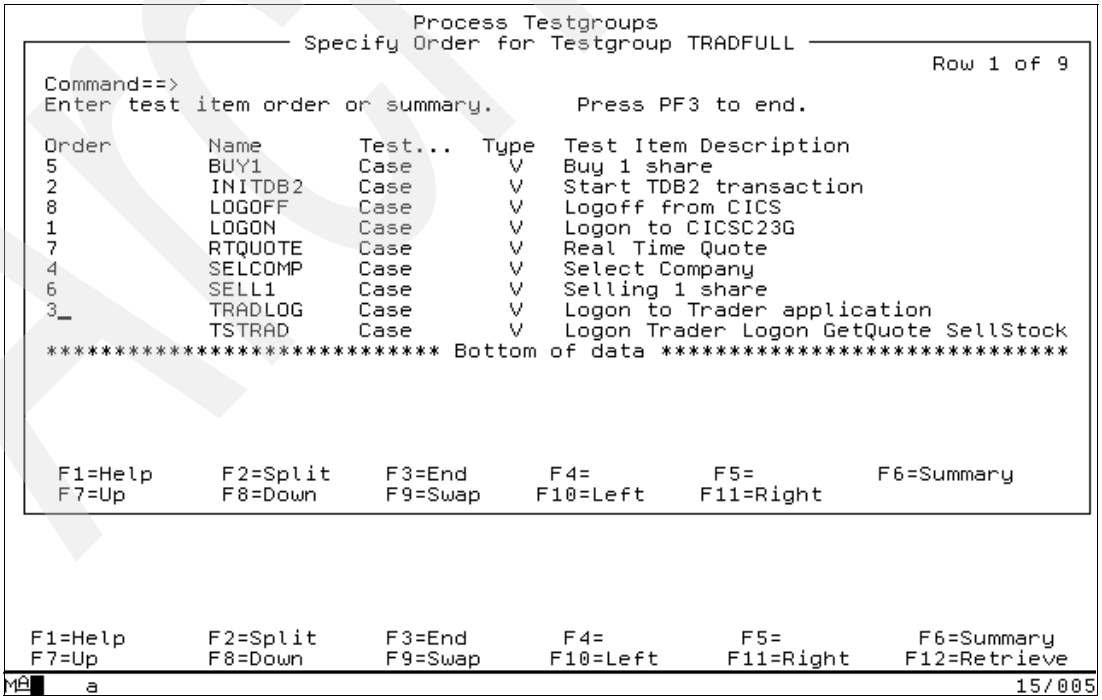


Figure 30-116 Order of test cases in testgroup TradFull

4. The test group TradFull is added. Enter the line command **S** to see its content, as shown in Figure 30-117. The ordered list of the test items for the test group TradFull is presented in Figure 30-118.

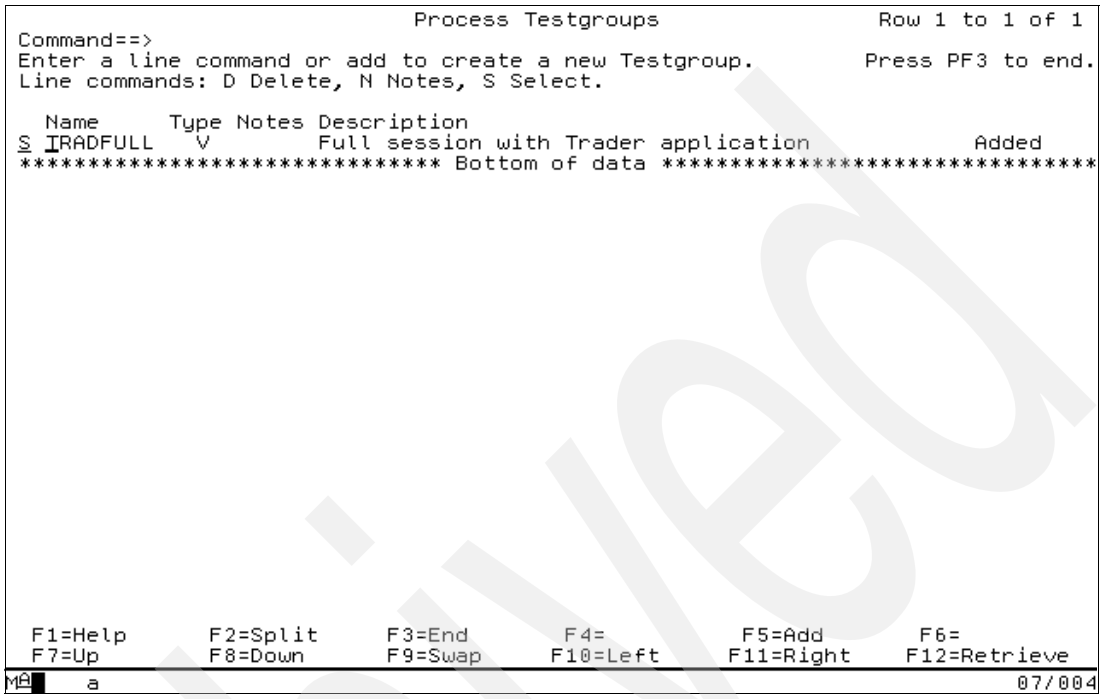


Figure 30-117 Testgroup TradFull created

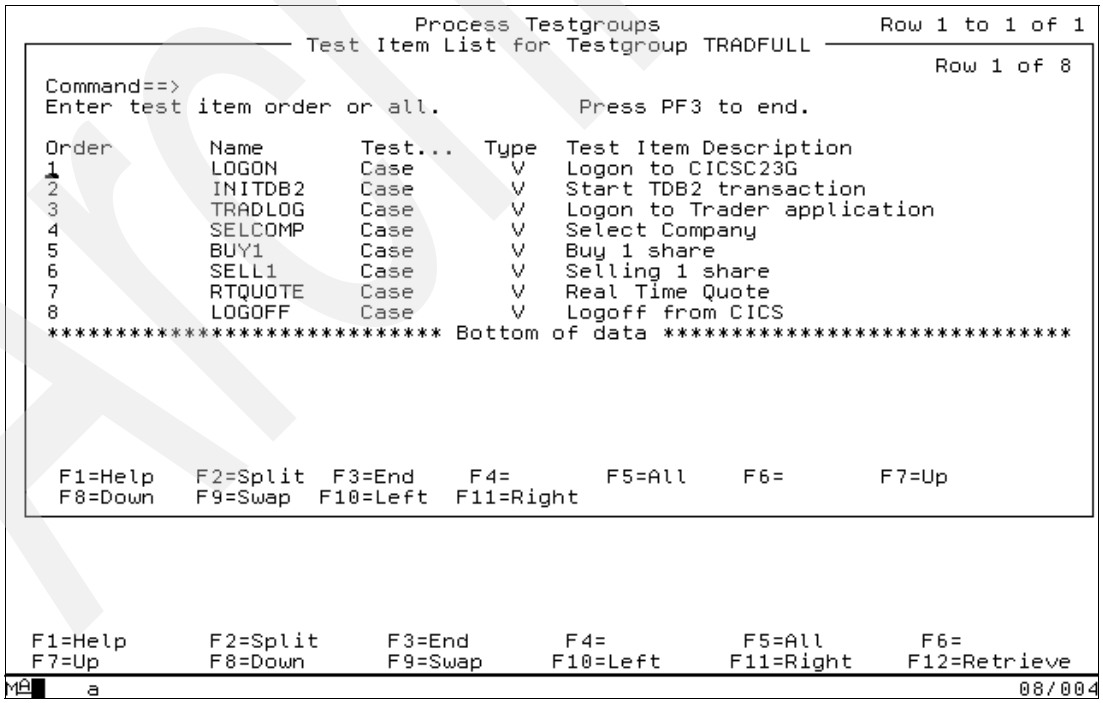


Figure 30-118 Ordered list of test cases in TradFull

5. To run a simulation for the newly created test group, a schedule must be added. To do this, select option **4** or issue the RUN command from the WTM main panel, as shown in Figure 30-119.

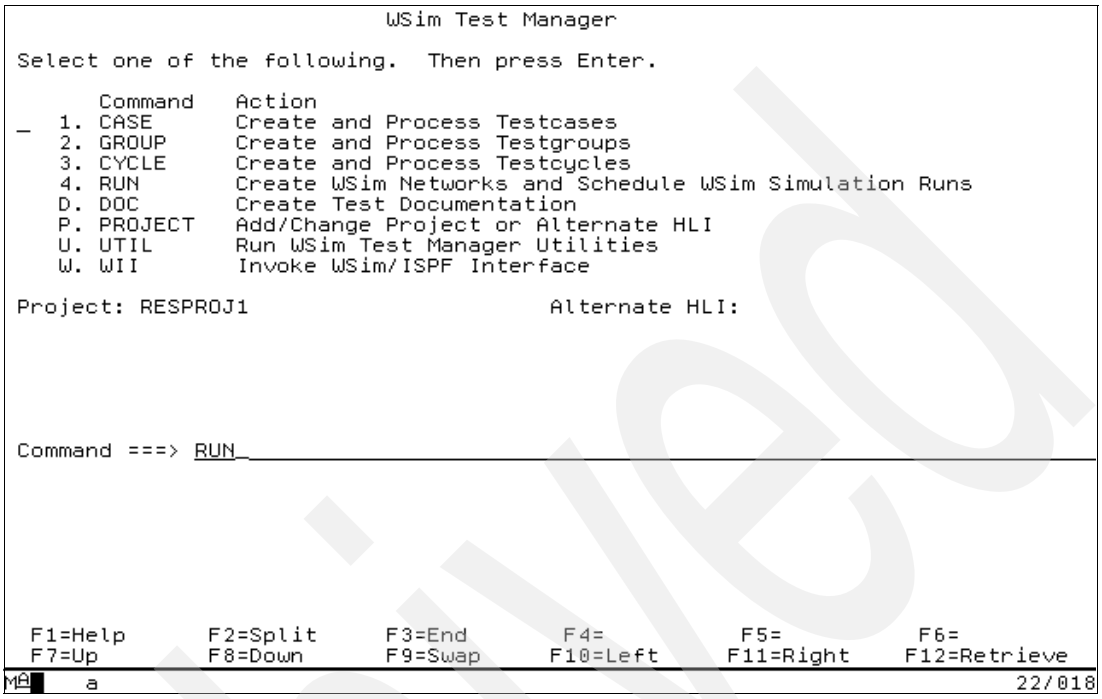


Figure 30-119 Starting to work with schedules

6. On the WTM Process Schedules panel enter the ADD command to create a new schedule to run the test group, as shown in Figure 30-120.

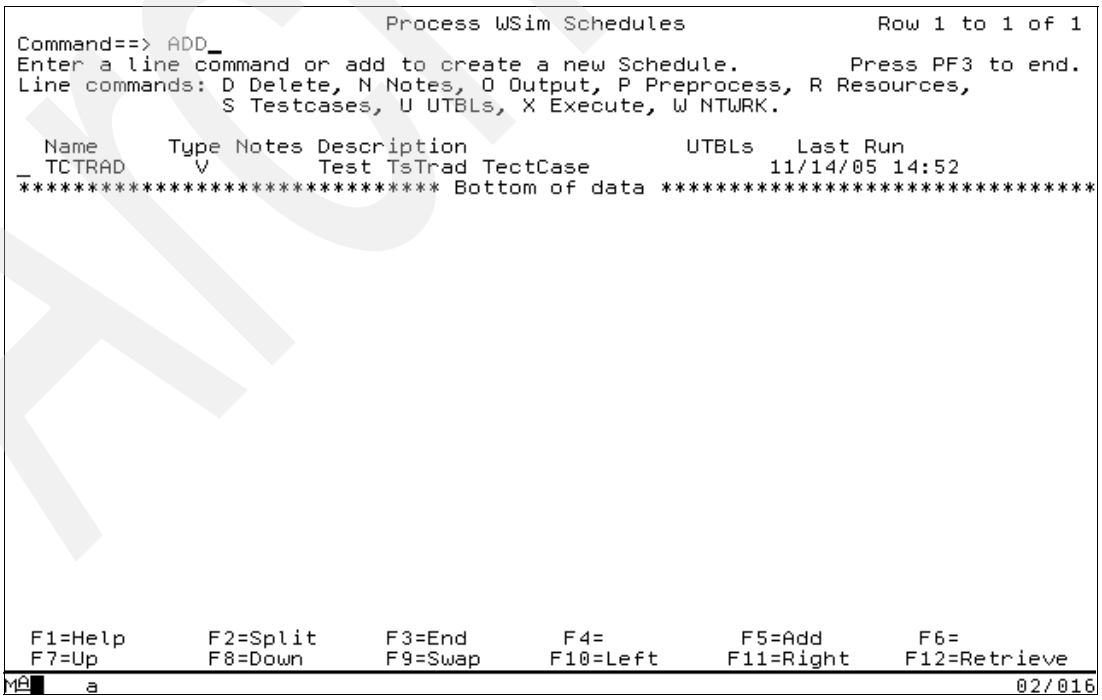


Figure 30-120 Adding schedule

7. Enter a name for the new schedule, then add a description. Identify its type and press Enter, as shown in Figure 30-121.

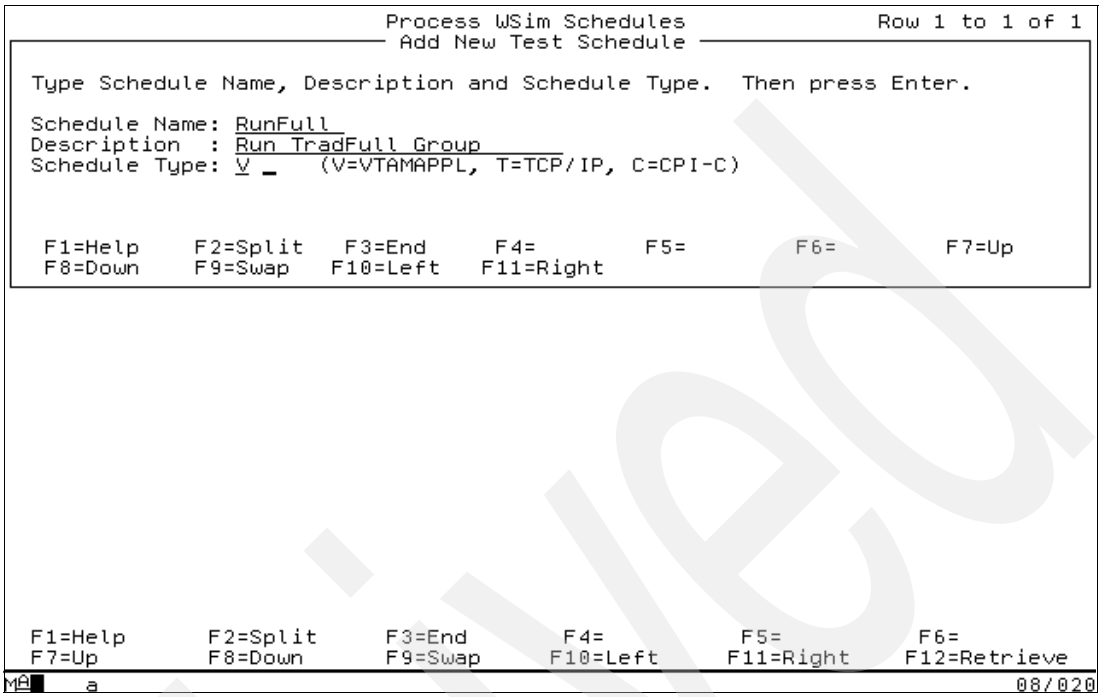


Figure 30-121 Creating schedule RunFull

8. On the next panel, Resource List for Schedule, enter the line command **S** to select the resources in the list, as shown in Figure 30-122.

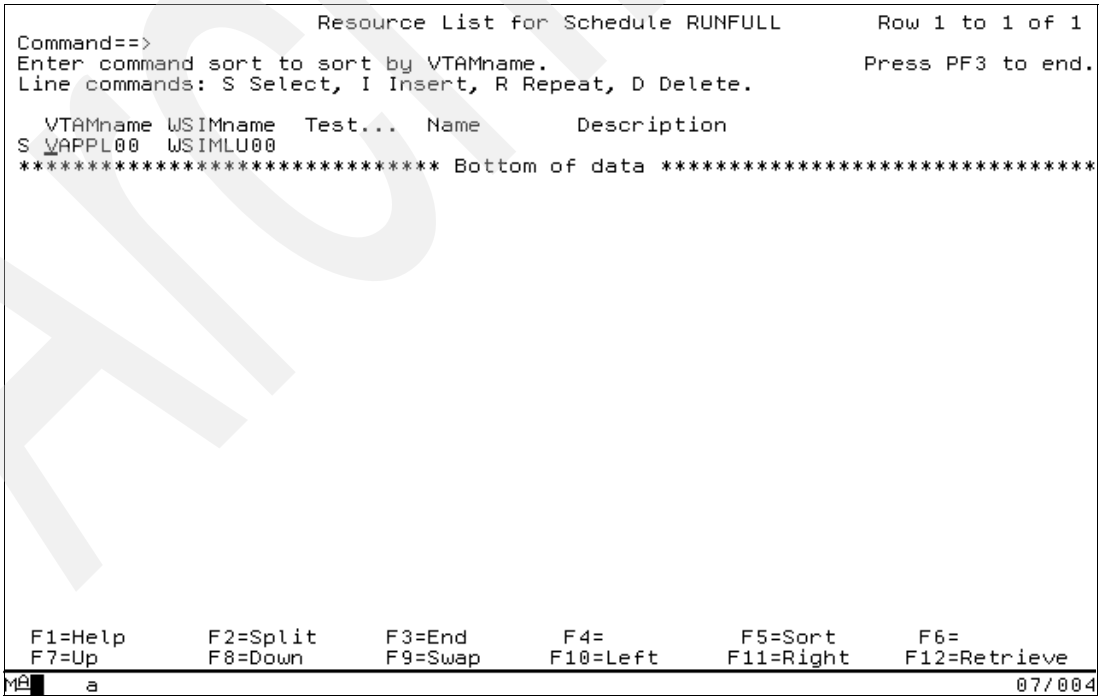


Figure 30-122 Simulated display WSIMLU00 is selected to specify test items

9. The only test item to be used in this simulation is the test group TradFull, so its order **1** is entered, as shown in Figure 30-123.

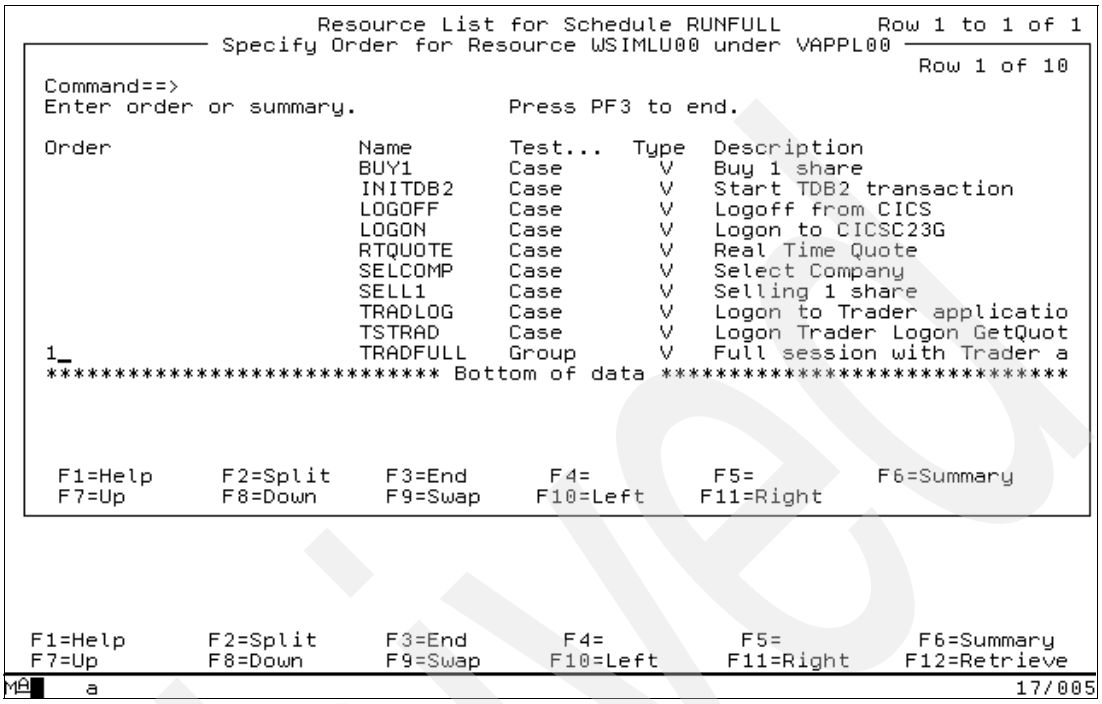


Figure 30-123 Testgroup assigned to WSIMLU00

10. The resource list for the schedule RunFull is now defined as shown in Figure 30-124.

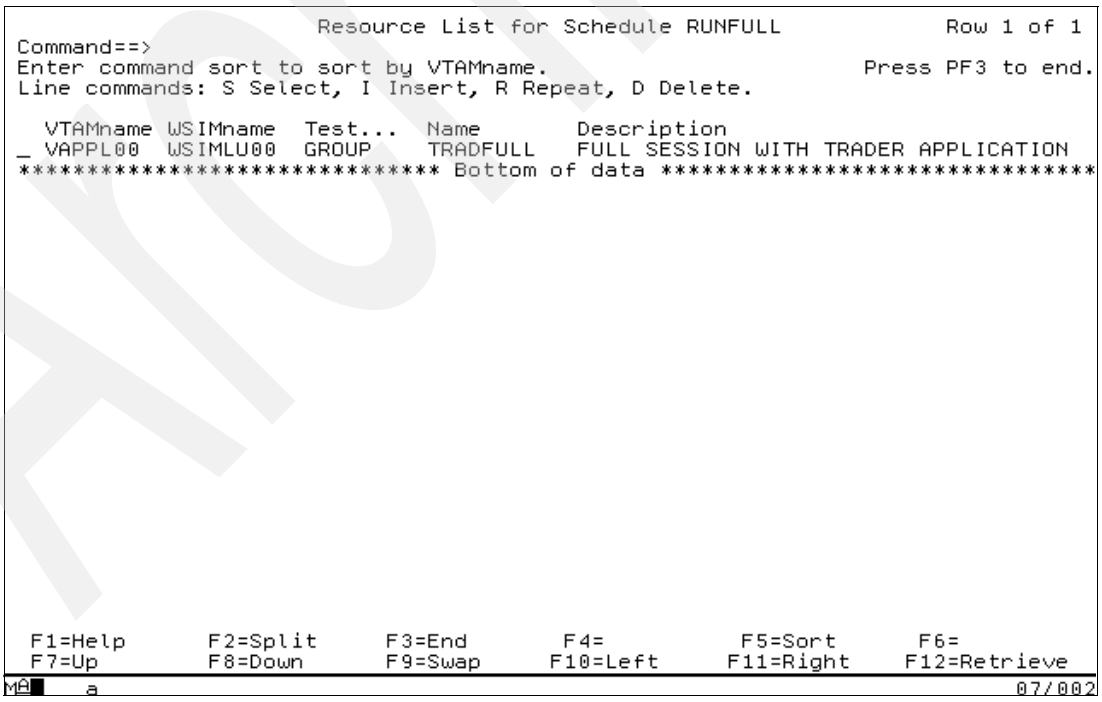


Figure 30-124 Test items assigned to schedule RunFull

11. Press PF3. Since this is the schedule that is being added, WTM presents the Completion Report Response Time Thresholds panel to enter threshold values, as shown in Figure 30-125.

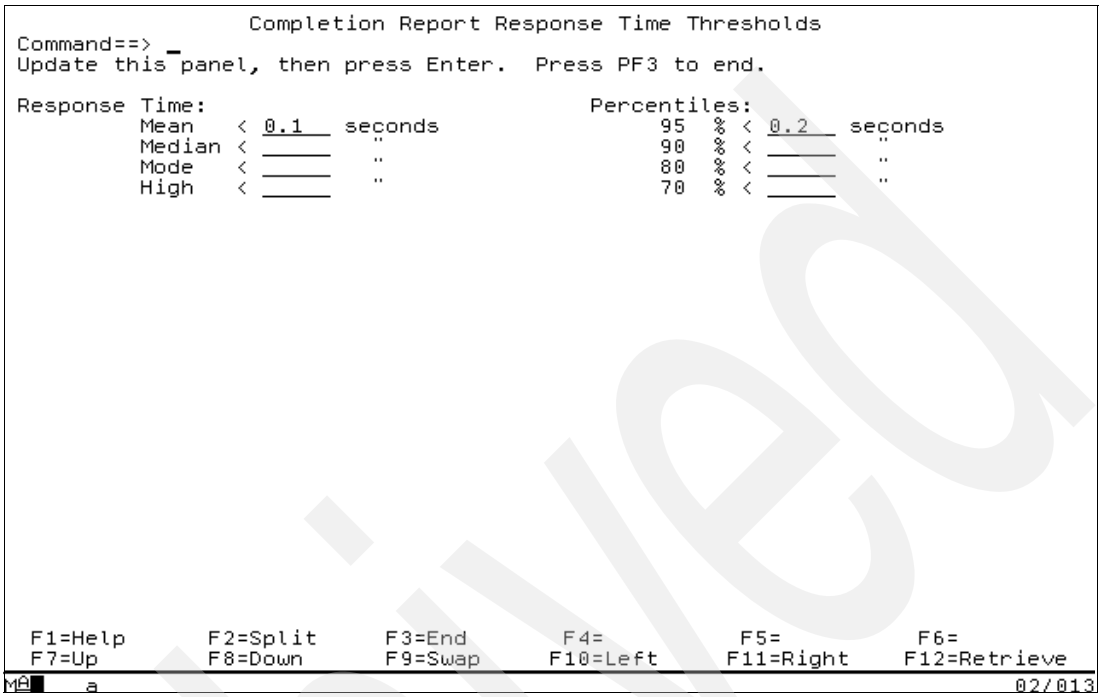


Figure 30-125 Completion reports time thresholds panel

12. The new schedule RunFull is created and is ready to be run. Enter the line command **X** to initiate the simulation, as shown in Figure 30-126.

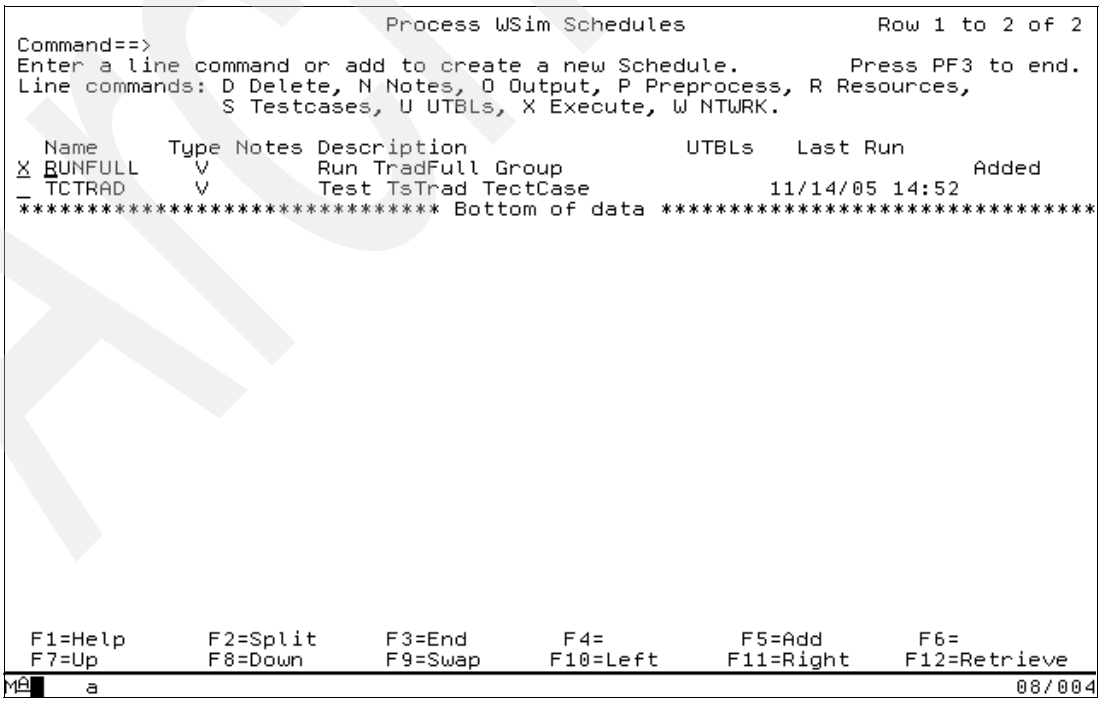


Figure 30-126 Executing schedule RunFull

13. On the next panel, shown in Figure 30-127, WTM gives you an opportunity to define the log data set names to be used for this simulation. Press Enter to proceed with the simulation.

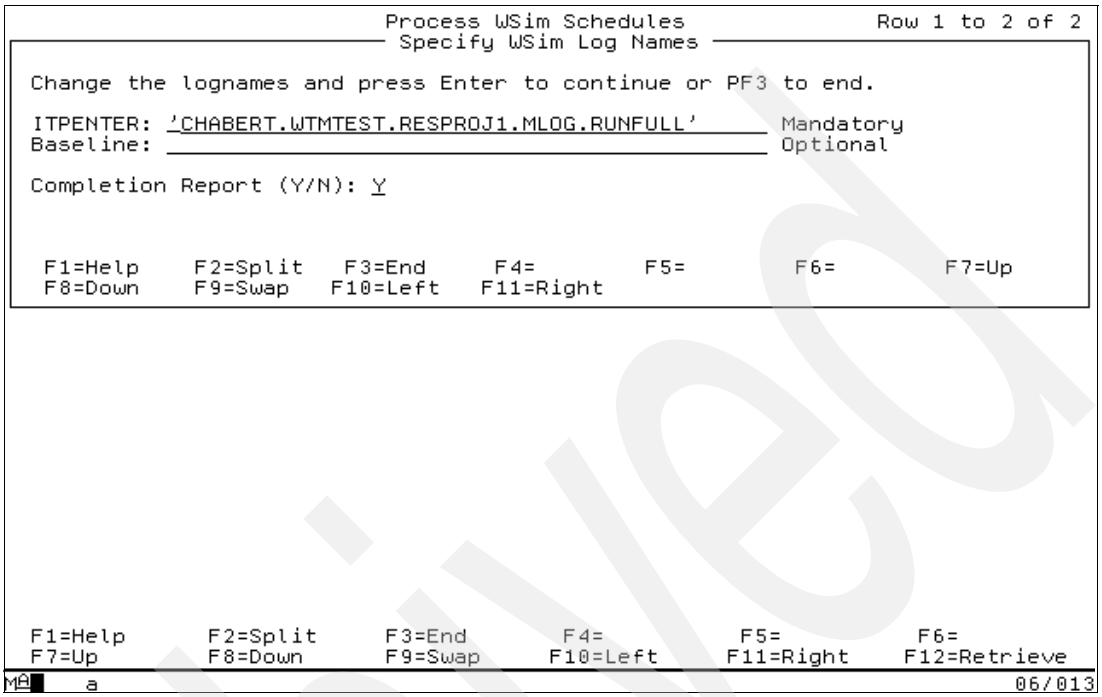


Figure 30-127 Simulation log data set name

14. The WTM starts the simulation and displays the WSim run-time progress messages (the DISPLAY mode was specified). The simulation is successfully completed, as shown in Figure 30-128.

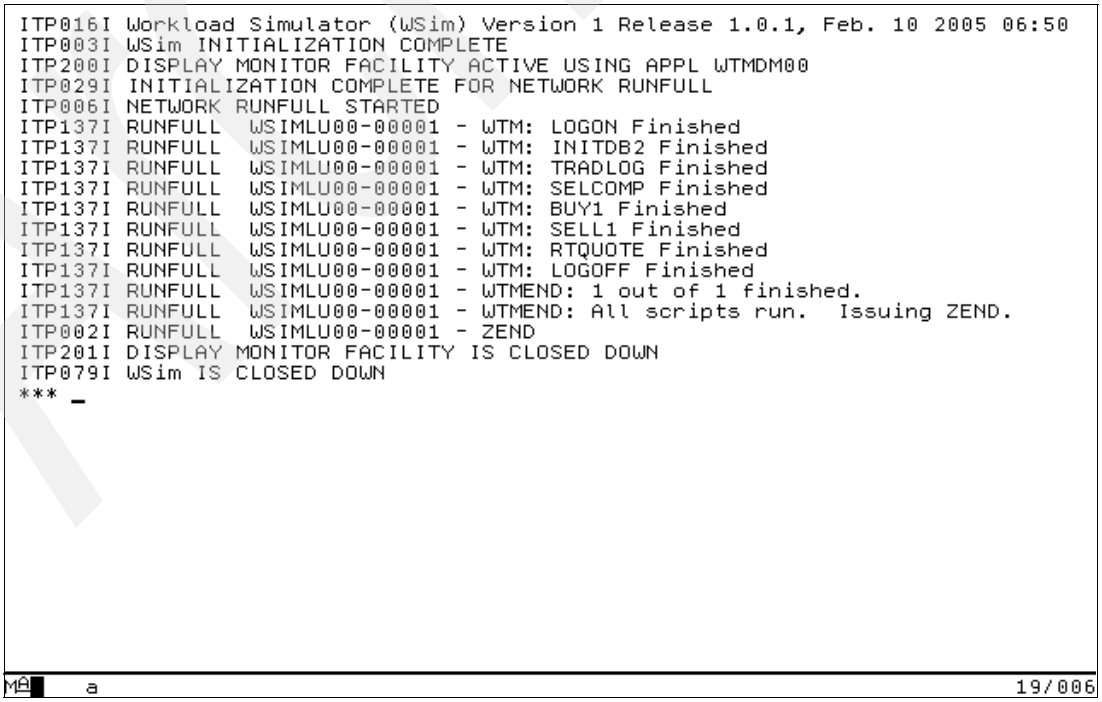


Figure 30-128 Simulation messages for RunFull

15. The requested optional completion report for the simulation is shown in Figure 30-129.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CHABERT.WTMTEST.RESPROJ1.CREP(R4) - 01.00       Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 Report for RUNFULL - Run TradFull Group.
000002 Run completed at 10:31:14 on 15 Nov 2005.
000003 This run successfully met the response time targets:
000004   95%_Percentile actual (0.00)  target(0.2)
000005   Mean actual (0.00)  target(0.1)
000006 Loglist Extract:
000007 RUN  TIME 10.30.43,  NOVEMBER 15, 2005  VERSION  1  RELEASE  1.0.1
000008 10304995 ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGON Finished
000009 10305195 ITP137I RUNFULL  WSIMLU00-00001 - WTM: INITDB2 Finished
000010 10305396 ITP137I RUNFULL  WSIMLU00-00001 - WTM: TRADLOG Finished
000011 10305597 ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELCOMP Finished
000012 10310005 ITP137I RUNFULL  WSIMLU00-00001 - WTM: BUY1 Finished
000013 10310407 ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELL1 Finished
000014 10310809 ITP137I RUNFULL  WSIMLU00-00001 - WTM: RTQUOTE Finished
000015 10311210 ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGOFF Finished
000016 Comments:
000017
000018
****** ***************************** Bottom of Data ***************************


 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
MA    a                                                          04/015
```

Figure 30-129 Simulation report

16. Press PF3 to return to the list of the available schedules, shown in Figure 30-130.

```
                        Process WSim Schedules              Row 1 to 2 of 2
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                UTBLs   Last Run
0 RUNFULL    V        Run TradFull Group               11/15/05 10:30 Execute
_ TCTRAD     V        Test TsTrad TectCase             11/14/05 14:52
******************************* Bottom of data ********************************


 F1=Help      F2=Split     F3=End       F4=          F5=Add       F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA    a                                                          08/004
```

Figure 30-130 RunFull simulation executed

17. To see the output from the simulation enter the line command **0**. Available output reports from simulations were discussed previously.

30.8 User data tables

A User Data Table (UTBL) is a list of string constants, for example, a list of client or account numbers, or a list of user IDs and passwords. The WSim Test Manager organizes UTBLs into fields and each line can contain one or more fields.

WTM can automatically generate UTBLs and the STL programs to use them. This is done by editing the source of STL programs in the WTM and using the command UTBL.

There are three options available when the user table function is invoked in WTM:

- Create a new UTBL.
- Create a new field in the existing UTBL.
- Use an existing field or UTBL.

There are three types of access to data in an UTBL:

- Random: STL code is accessing UTBL randomly.
- Single sequential: STL code is accessing UTBL in strictly sequential order with only one pass. When the end of the UTBL is reached, further access is not allowed.
- Single sequential repeated: STL code is accessing UTBL in strictly sequential order in multiple passes from the first to the last record.

In this section we take you through the process of creating a UTBL, so instead of using hard-coded data, a script can access a UTBL.

1. Start with the CASE command from the WTM main panel shown in Figure 30-131.

```
                          WSim Test Manager

Select one of the following.  Then press Enter.

      Command   Action
_  1. CASE      Create and Process Testcases
   2. GROUP     Create and Process Testgroups
   3. CYCLE     Create and Process Testcycles
   4. RUN       Create WSim Networks and Schedule WSim Simulation Runs
   D. DOC       Create Test Documentation
   P. PROJECT   Add/Change Project or Alternate HLI
   U. UTIL      Run WSim Test Manager Utilities
   W. WII       Invoke WSim/ISPF Interface

Project: RESPROJ1                      Alternate HLI:




Command ===> CASE_____________________________________________________________




 F1=Help      F2=Split     F3=End       F4=          F5=          F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA    a                                                          22/019
```

Figure 30-131 Command CASE to work with test case

2. Enter the line command **o** to open the corresponding test case STL code and edit it. This example is working with the LOGON test case, as shown in Figure 30-132.

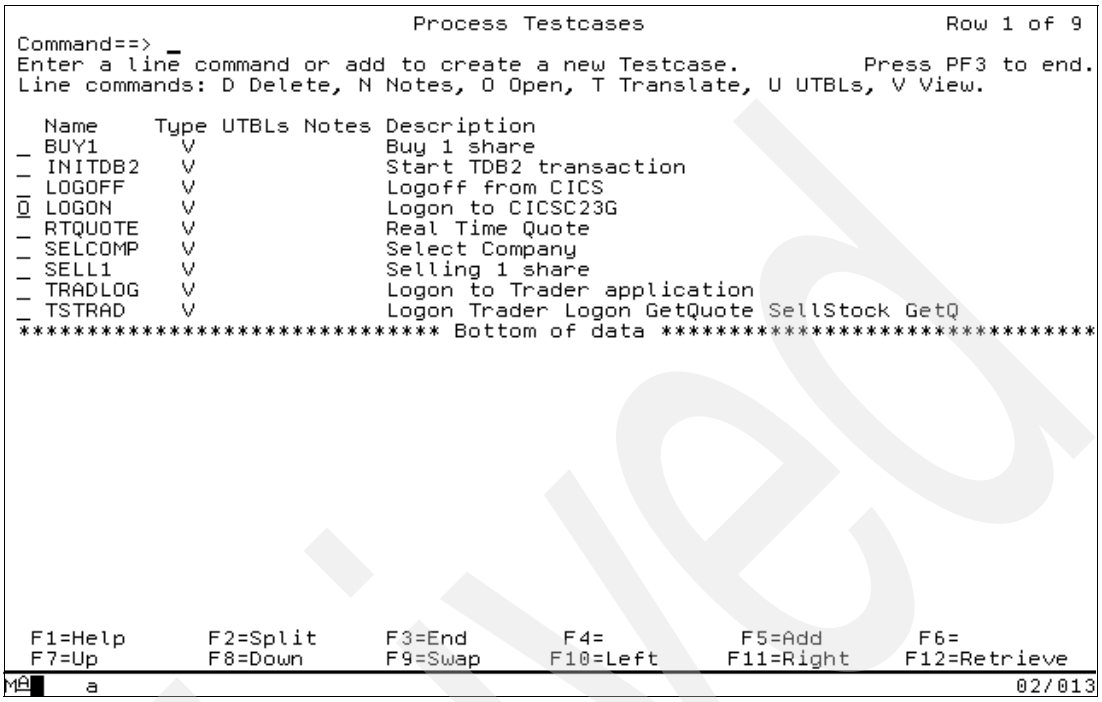


Figure 30-132 Opening test case LOGON

3. The source of the STL program for the test case LOGON is opened in an ISPF editing session, as shown in Figure 30-133.

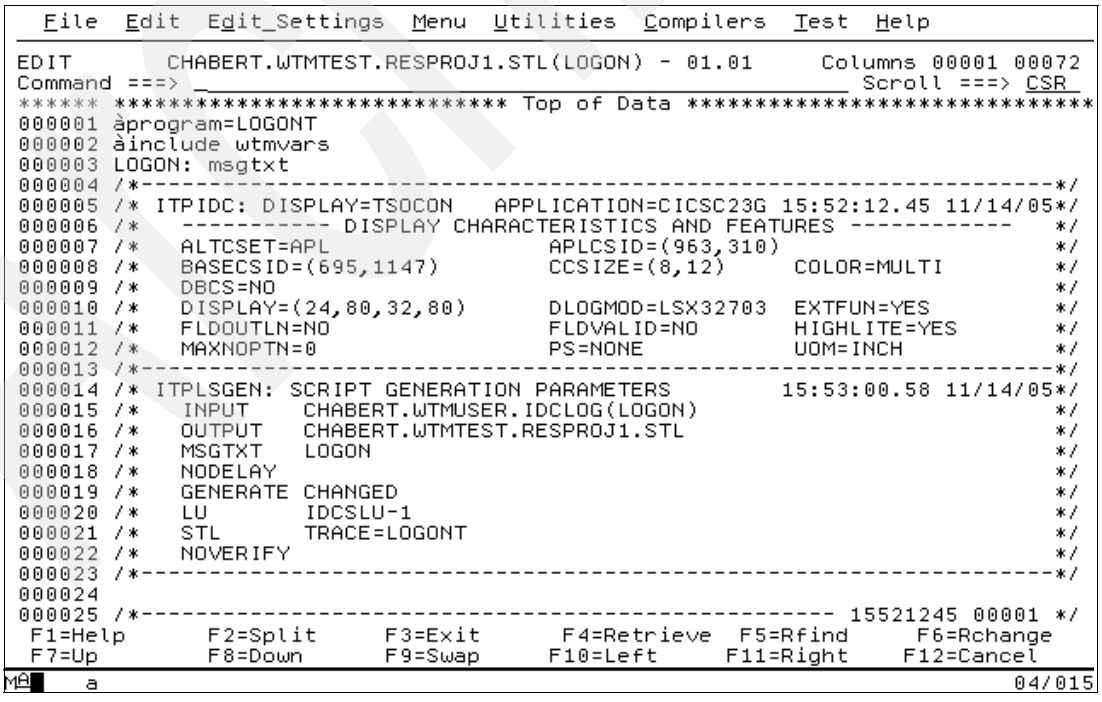


Figure 30-133 STL program for test case LOGON

4. Scroll down using the PF8 key and find the STL statement `type`. Enter the UTBL command on the ISPF command line and place the cursor on the line where a UTBL is to be used. Press Enter. The WTM pop-up panel Create WSim User Table appears, as shown in Figure 30-134.
5. Provide the information about the UTBL, such as the name and description, field (there can be more than one) name and description, field delimiter (use the default), and access type. Press Enter.

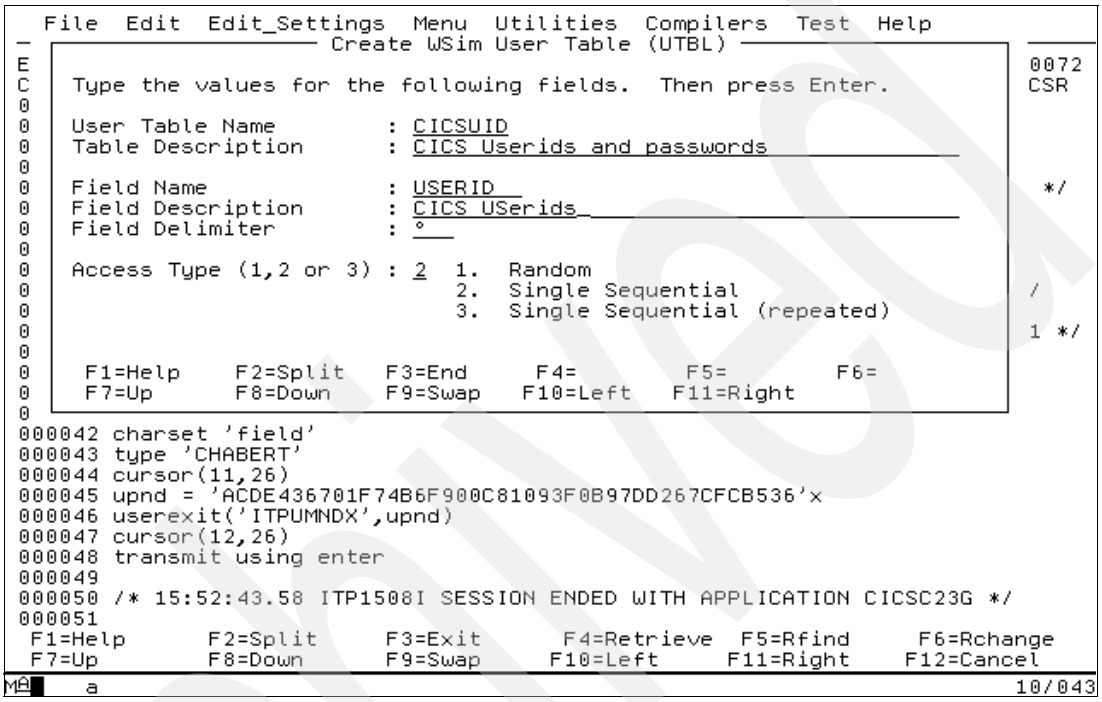


Figure 30-134 User table: CICSUID

Attention: The access type selected is **2** (Single Sequential), which allows just one pass through the UTBL CICSUIS. This choice has an impact on the load test, which we discuss later in this chapter and in which we reuse the test case LOGON.

6. WTM makes the appropriate changes to the STL code and presents the updated program, as shown in Figure 30-135.

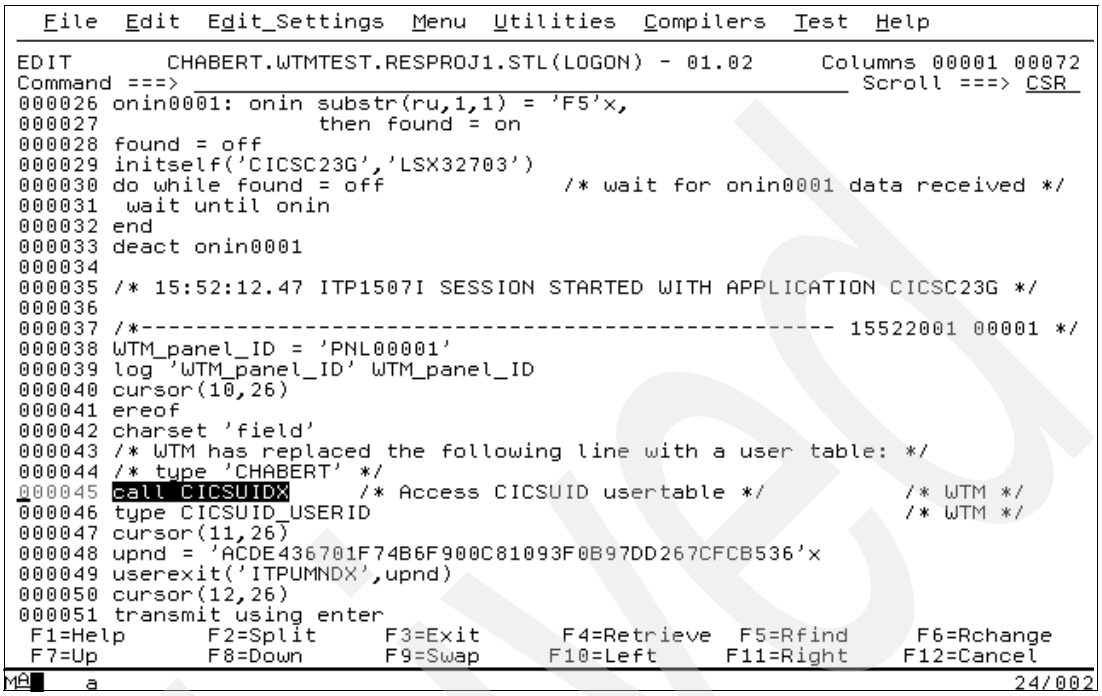


Figure 30-135 Changes to STL program for LOGON

7. When you use the latest level of WSim, the password can be protected. They are coded by the IDC. You see `upnd` instead of `type` in the generated script, followed by a call to the protection exit.
8. Add another field to the just created CICSUIS UTBL. Enter the command UTBL on the ISPF command line and place the cursor at the next upnd STL statement. Press Enter.

9. On the pop-up panel that appears, select option **2** to create a new field in the existing UTBL and press Enter, as shown in Figure 30-136.

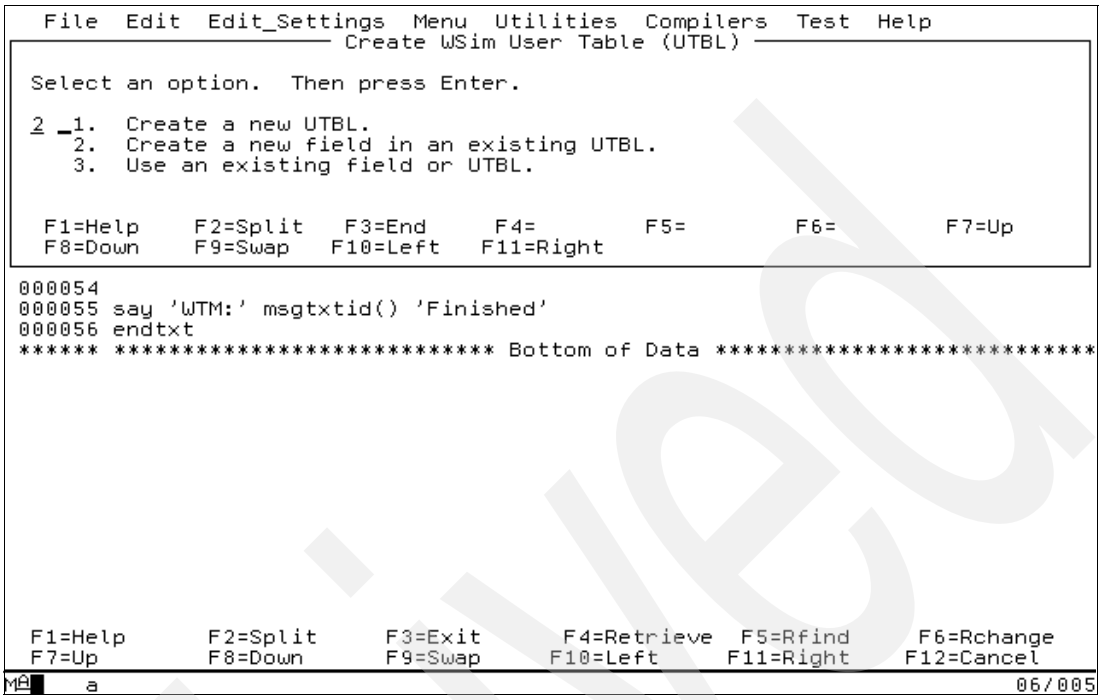


Figure 30-136 Adding new field to existing user table

10. The WTM UTBL selection panel appears, as shown in Figure 30-137. Enter the line command **S** in front of the appropriate UTBL and press Enter.

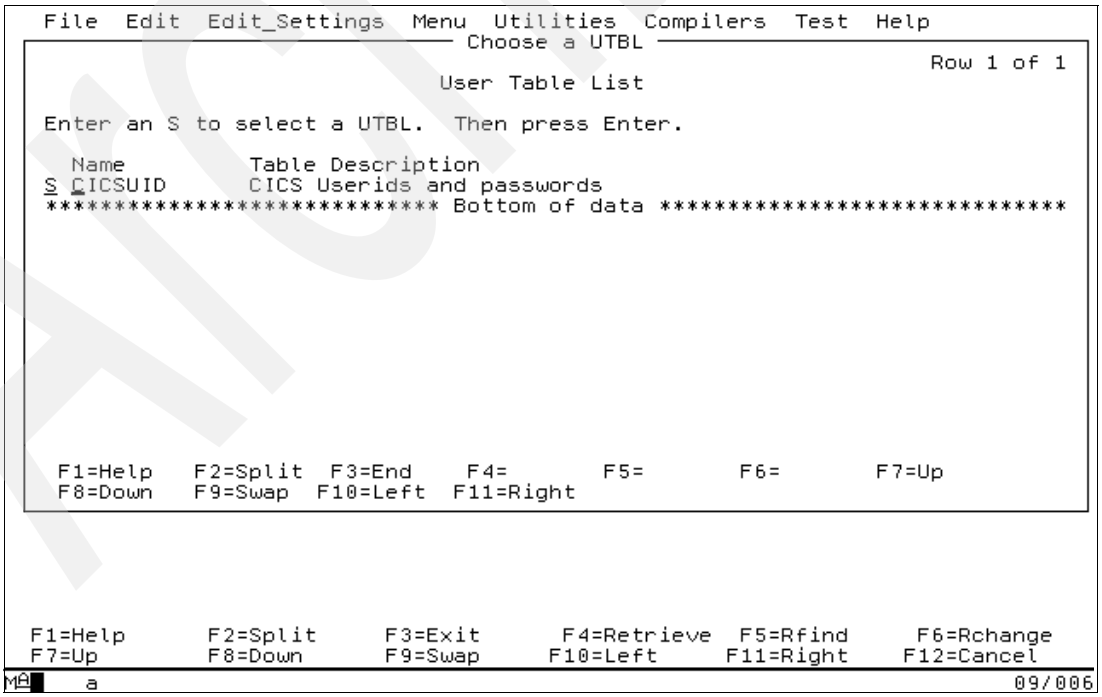


Figure 30-137 User table: CICSUID selected

11. Provide the information for the new field in the CICSUIS UTBL, such as name, description, and delimiter. Press Enter, as shown in Figure 30-138.

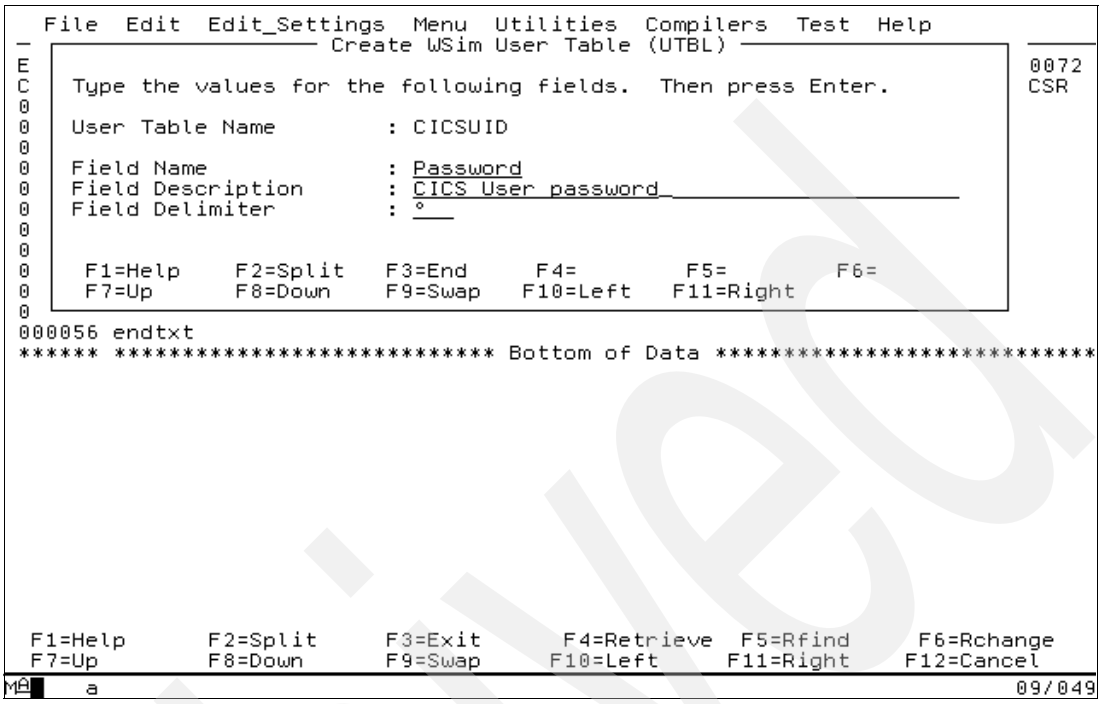


Figure 30-138 New field in user table CICSUID

12. WTM added a new statement to the STL program, as shown in Figure 30-139. Press PF3 to save the changes made by WTM and exit from the ISPF editing session.

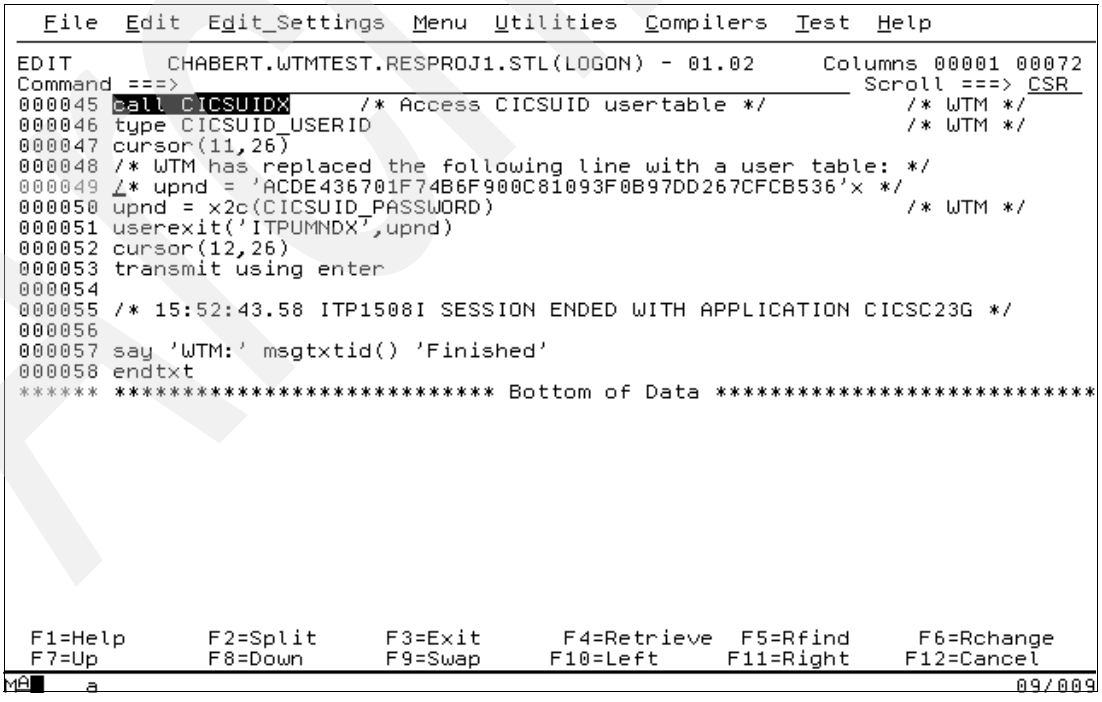


Figure 30-139 More changes in STL program LOGON

13. WTM returns to the Process Testcases panel and shows that one UTBL is to be used by the test case LOGON. Enter the line command **u** to begin working with UTBLs, as shown in Figure 30-140.

```
                              Process Testcases                    Row 1 of 9
Command==> _
Enter a line command or add to create a new Testcase.          Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name     Type UTBLs Notes Description
_ BUY1       V              Buy 1 share
_ INITDB2    V              Start TDB2 transaction
_ LOGOFF     V              Logoff from CICS
U LOGON      V      1       Logon to CICSC23G                          Open
_ RTQUOTE    V              Real Time Quote
_ SELCOMP    V              Select Company
_ SELL1      V              Selling 1 share
_ TRADLOG    V              Logon to Trader application
_ TSTRAD     V              Logon Trader Logon GetQuote SellStock GetQ
****************************** Bottom of data ********************************




 F1=Help      F2=Split     F3=End       F4=          F5=Add       F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA    a                                                            02/013
```

Figure 30-140 Starting work with UTBL used in LOGON

14. On the WTM User Table List panel enter the line command **s** to start working with the STL code corresponding to the UTBL used in the test case LOGON (there is only one at this time), as shown in Figure 30-141, and press Enter.

```
                              User Table List                      Row 1 of 1
Command==> _
Enter a line command or all.                 Press PF3 to end.
Line commands: S STL, M MSGTXT, D Delete.

  Name          Table Description
S CICSUID       CICS Userids and passwords
******************************** Bottom of data ********************************




 F1=Help      F2=Split     F3=End       F4=          F5=All       F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA    a                                                            02/013
```

Figure 30-141 User table: CICSUID is selected

15. An ISPF editing session is opened for the STL code of the UTBL CICSUSIS. The actual user ID and password are on line 7, as shown in Figure 30-142.

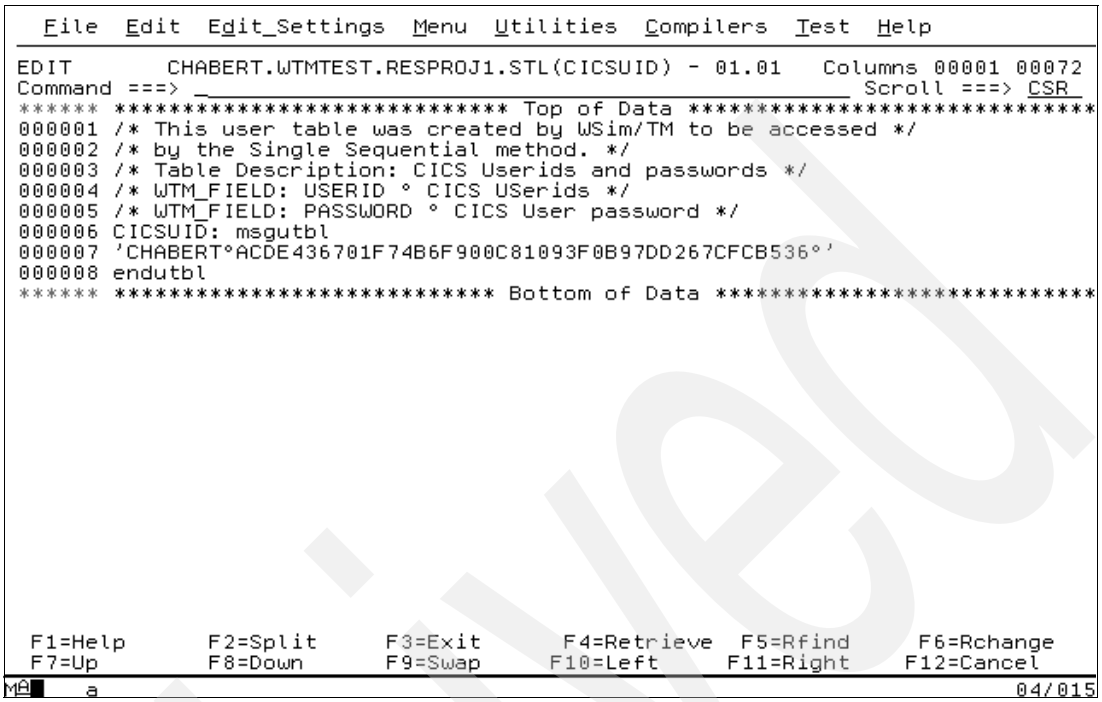


Figure 30-142 Editing CICSUID

16. Add one more line with the second user ID and the password (you must know the hexadecimal encrypted value of the new password), as shown in Figure 30-143. Press PF3 to save the changes to the UTBL CICSUIS.

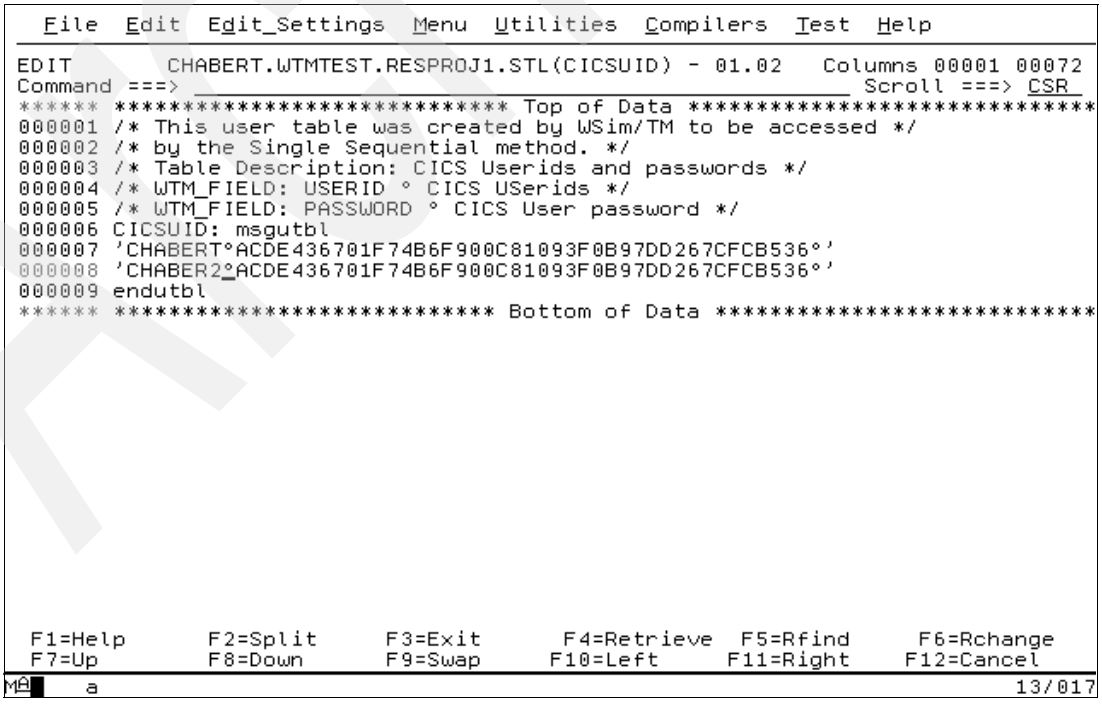


Figure 30-143 Second line with user ID and password added

17. WTM returns to the User Table List panel shown in Figure 30-144. Press PF3.

```
                              User Table List                          Row 1 of 1
Command==>
Enter a line command or all.                 Press PF3 to end.
Line commands: S STL, M MSGTXT, D Delete.

  Name          Table Description
_ CICSUID       CICS Userids and passwords                                  STL
******************************** Bottom of data ********************************
 F1=Help       F2=Split      F3=End        F4=           F5=All        F6=
 F7=Up         F8=Down       F9=Swap       F10=Left      F11=Right     F12=Retrieve
MA    a                                                                  07/002
```

Figure 30-144 STL program for CICSUID saved

18. WTM returns to the test cases list, as shown in Figure 30-145.

```
                              Process Testcases                        Row 1 of 9
Command==>
Enter a line command or add to create a new Testcase.          Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name     Type UTBLs Notes Description
_ BUY1       V               Buy 1 share
_ INITDB2    V               Start TDB2 transaction
_ LOGOFF     V               Logoff from CICS
_ LOGON      V      1        Logon to CICSC23G                              UserTable
_ RTQUOTE    V               Real Time Quote
_ SELCOMP    V               Select Company
_ SELL1      V               Selling 1 share
_ TRADLOG    V               Logon to Trader application
_ TSTRAD     V               Logon Trader Logon GetQuote SellStock GetQ
******************************** Bottom of data ********************************
 F1=Help       F2=Split      F3=End        F4=           F5=Add        F6=
 F7=Up         F8=Down       F9=Swap       F10=Left      F11=Right     F12=Retrieve
MA    a                                                                  10/002
```

Figure 30-145 List of all test cases

19. Press PF3 again to return to the WTM main panel. Enter the RUN command or select option **4** to run a simulation to test the new version of the LOGON test case, as shown in Figure 30-146.

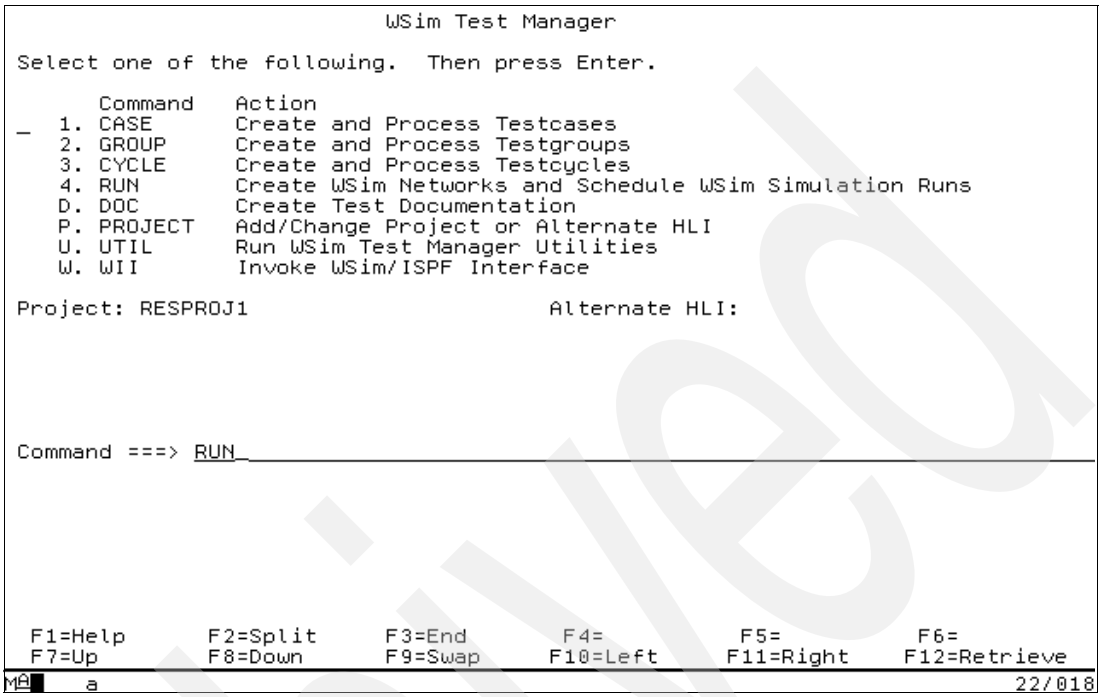


Figure 30-146 RUN command from WTM main panel

20. Enter the line command **r** to work with resources for the schedule RunAll, as shown in Figure 30-147.

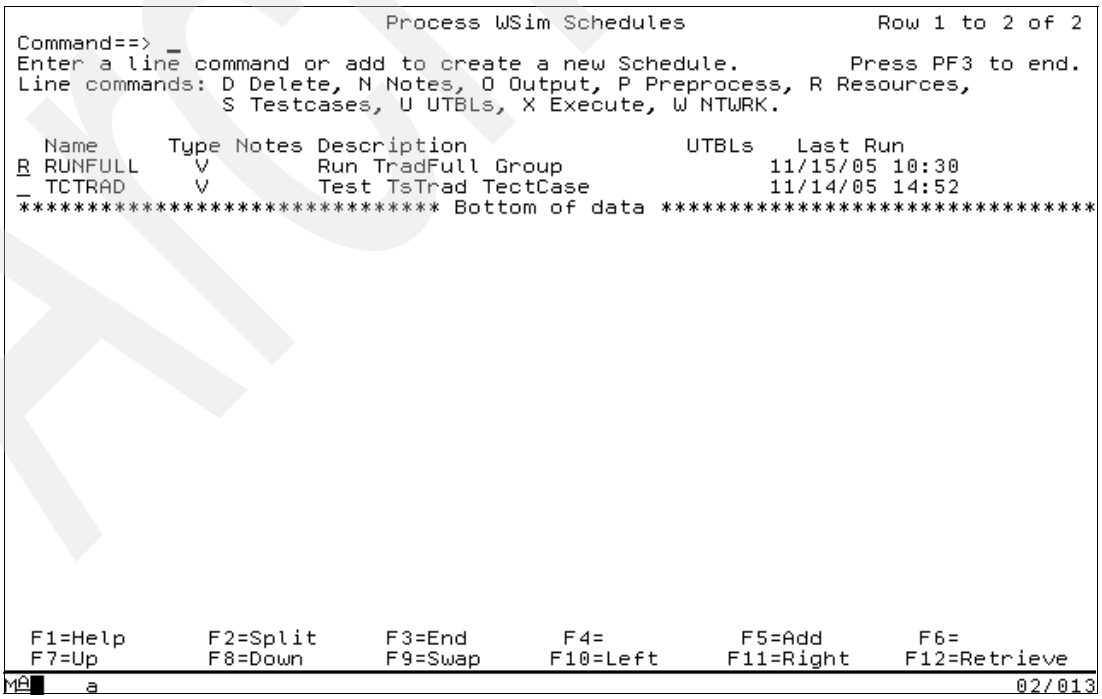


Figure 30-147 Working with resources for RunFull

21. Enter the line command **s** to work with the network resource, as shown in Figure 30-148.

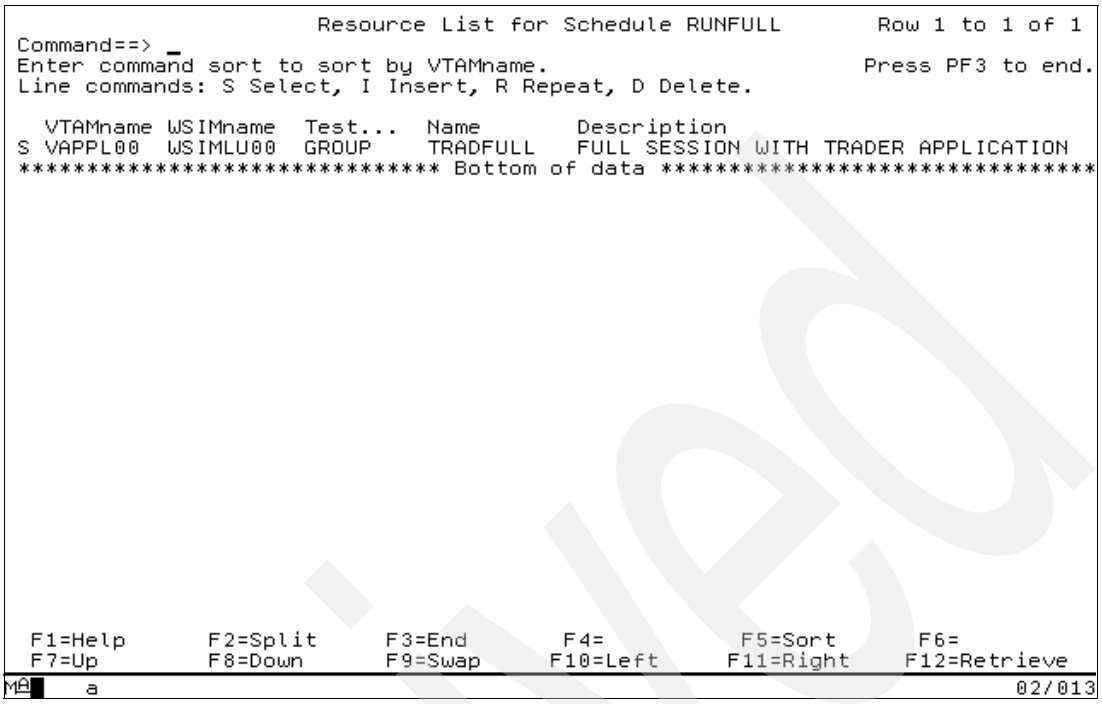


Figure 30-148 Simulated resource (terminal) WSIMLU00 is selected

22. You want the test group TradFull to be executed twice in a row on this simulated terminal, as shown in Figure 30-149.

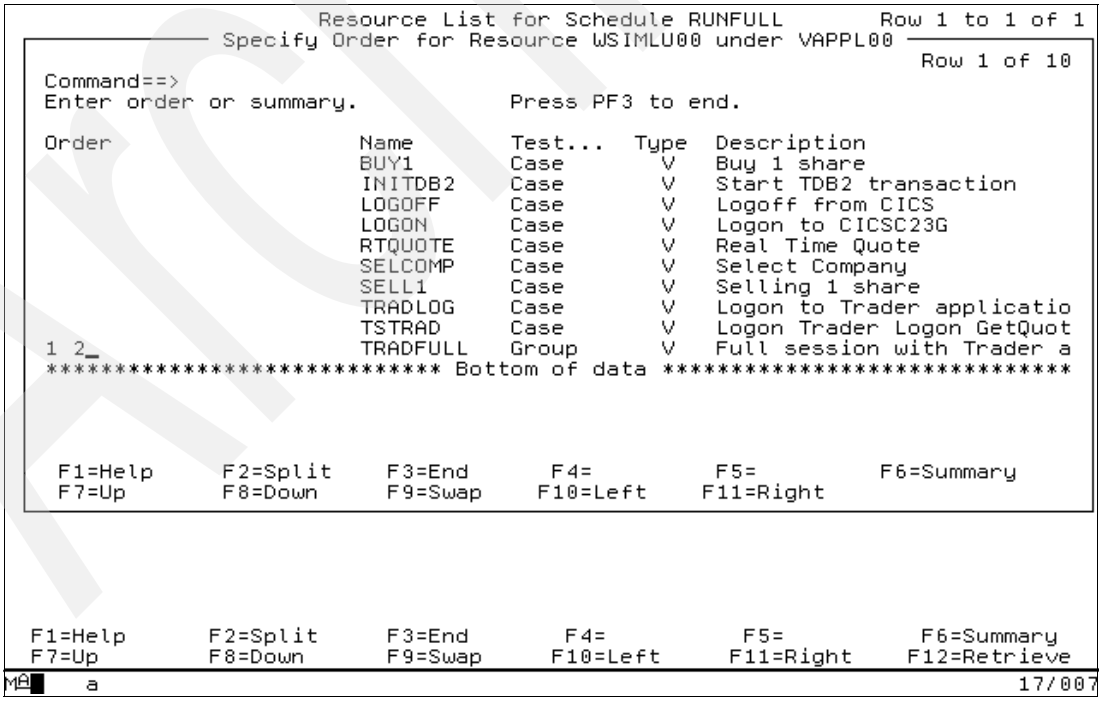


Figure 30-149 TradFull to be executed twice

23. WTM repeats the network definition to make the script run two times, as shown in Figure 30-150. Press PF3.

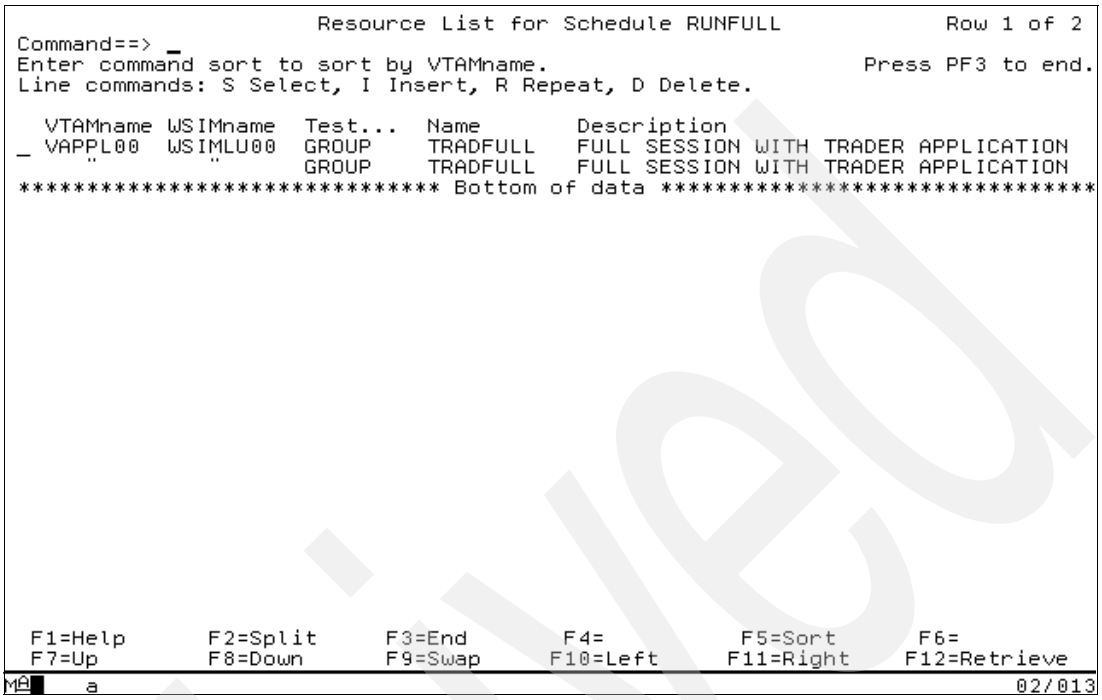


Figure 30-150 WSIMLU00 is to be used twice

24. WTM displays a warning about forthcoming changes to the network definition for the schedule RUNFULL, shown in Figure 30-151. We did not make any manual changes to the generated source code in this case so there is no reason for concern. Press Enter to continue.

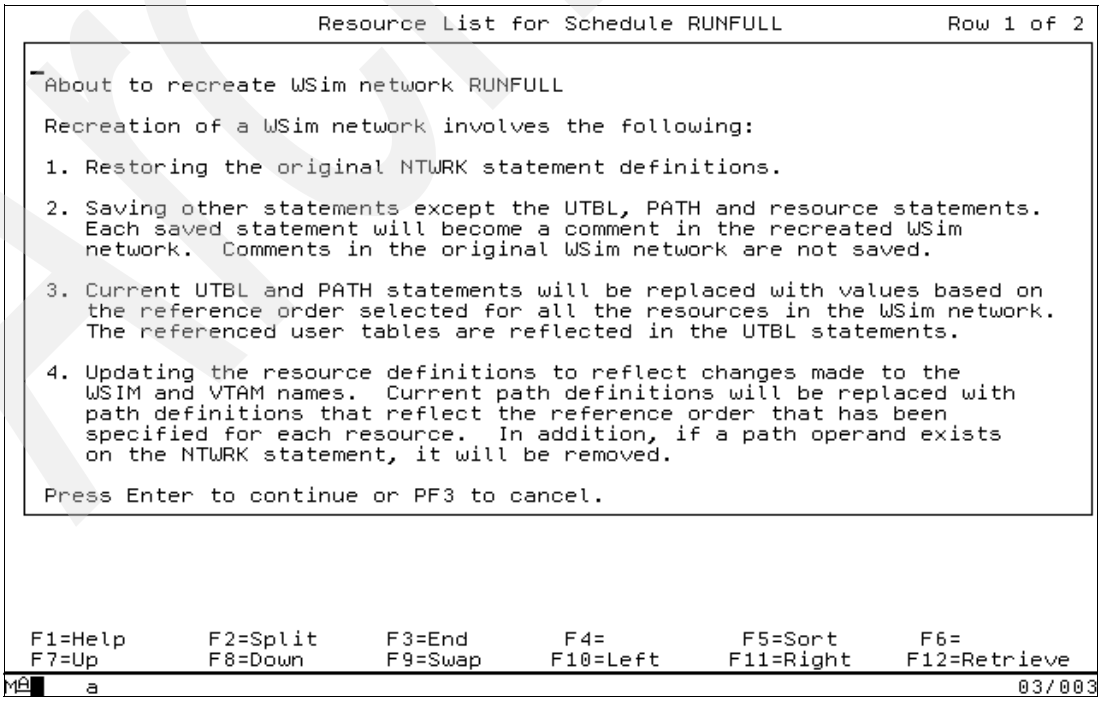


Figure 30-151 Warning messages about changes to RunFull

25. The schedule RUNFULL is updated and ready to run. Enter the line command **X** to execute it, as shown in Figure 30-152.

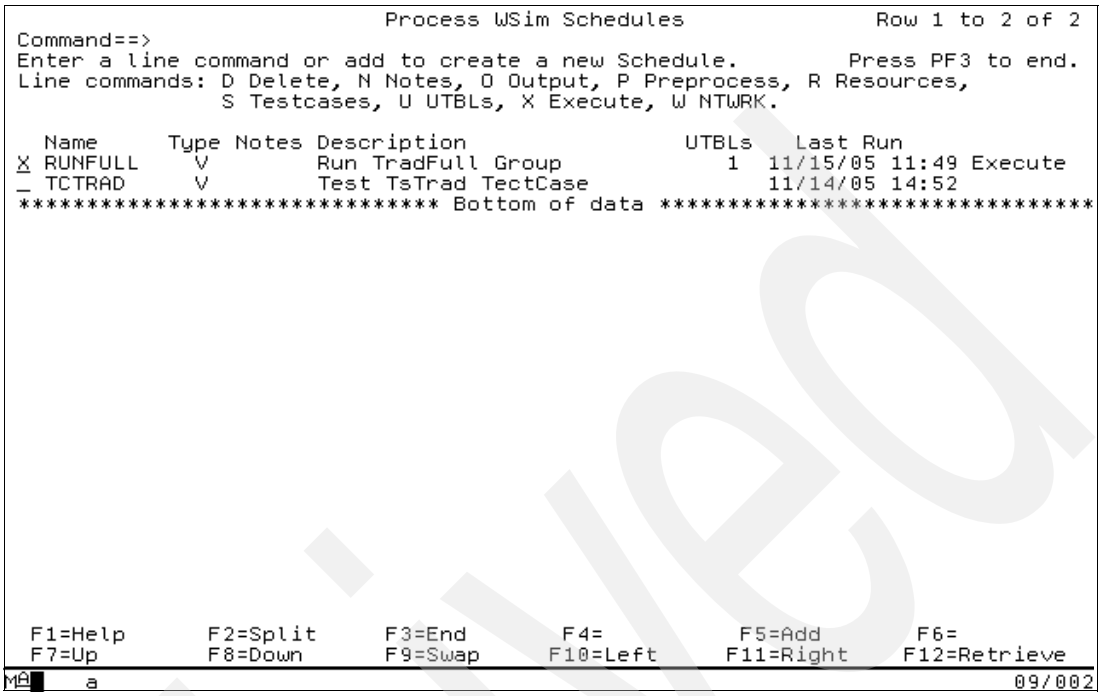


Figure 30-152 RunFull run requested

26. Note that one user table is used.

27. On the next WTM panel, change the log names if desired and press Enter to continue with the execution. WSim starts the simulation and executes all the test cases twice, as shown in the Figure 30-153.

```
ITP016I Workload Simulator (WSim) Version 1 Release 1.0.1, Feb. 10 2005 06:50
ITP003I WSim INITIALIZATION COMPLETE
ITP200I DISPLAY MONITOR FACILITY ACTIVE USING APPL WTMDM00
ITP029I INITIALIZATION COMPLETE FOR NETWORK RUNFULL
ITP006I NETWORK RUNFULL STARTED
ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGON Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: INITDB2 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: TRADLOG Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELCOMP Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: BUY1 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELL1 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: RTQUOTE Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGOFF Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGON Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: INITDB2 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: TRADLOG Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELCOMP Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: BUY1 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELL1 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: RTQUOTE Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGOFF Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTMEND: 1 out of 1 finished.
ITP137I RUNFULL  WSIMLU00-00001 - WTMEND: All scripts run.  Issuing ZEND.
ITP002I RUNFULL  WSIMLU00-00001 - ZEND
ITP201I DISPLAY MONITOR FACILITY IS CLOSED DOWN
ITP079I WSim IS CLOSED DOWN
*** _

MA    a                                                          27/006
```

Figure 30-153 RunFull simulation messages

28. Here we skip presenting the optional completion report and proceed to the simulation output, as shown in Figure 30-154.

```
                          Process WSim Schedules              Row 1 to 2 of 2
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                UTBLs   Last Run
O RUNFULL    V        Run TradFull Group            1  11/15/05 11:49 Execute
_ TCTRAD     V        Test TsTrad TectCase             11/14/05 14:52
****************************** Bottom of data ********************************

 F1=Help      F2=Split     F3=End       F4=          F5=Add       F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA    a                                                          08/004
```

Figure 30-154 RunFull executed

29. Select option **3** to see the Display Monitor log, as shown in Figure 30-155.

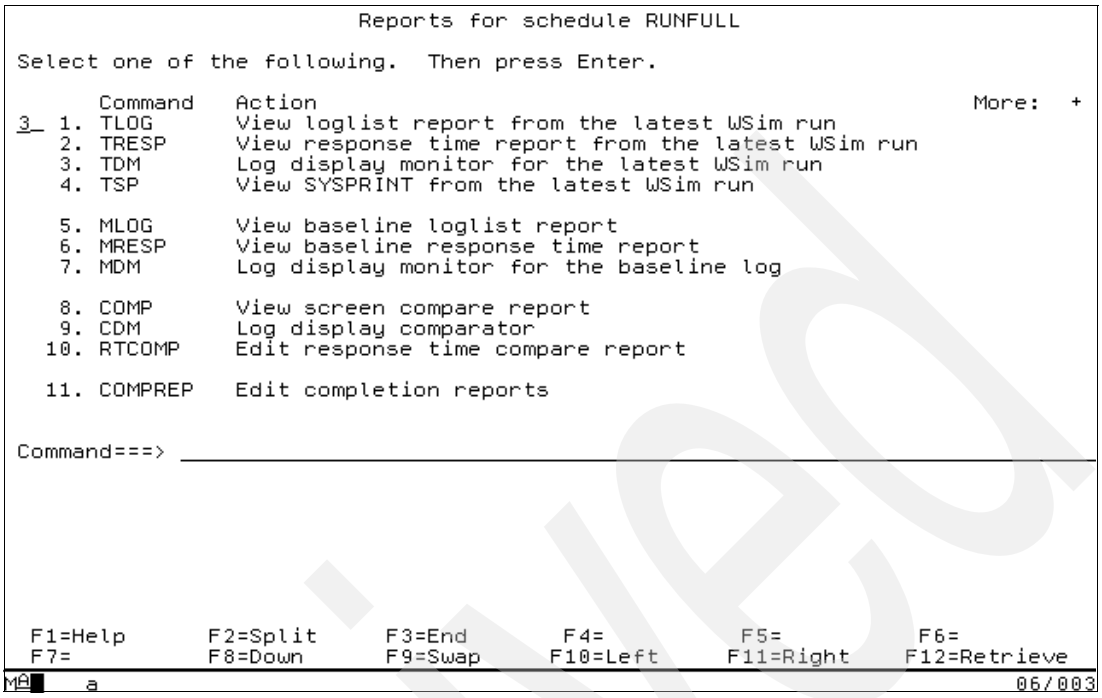


Figure 30-155 WTM Reports panel

30. Select to view the index, as shown in Figure 30-156.

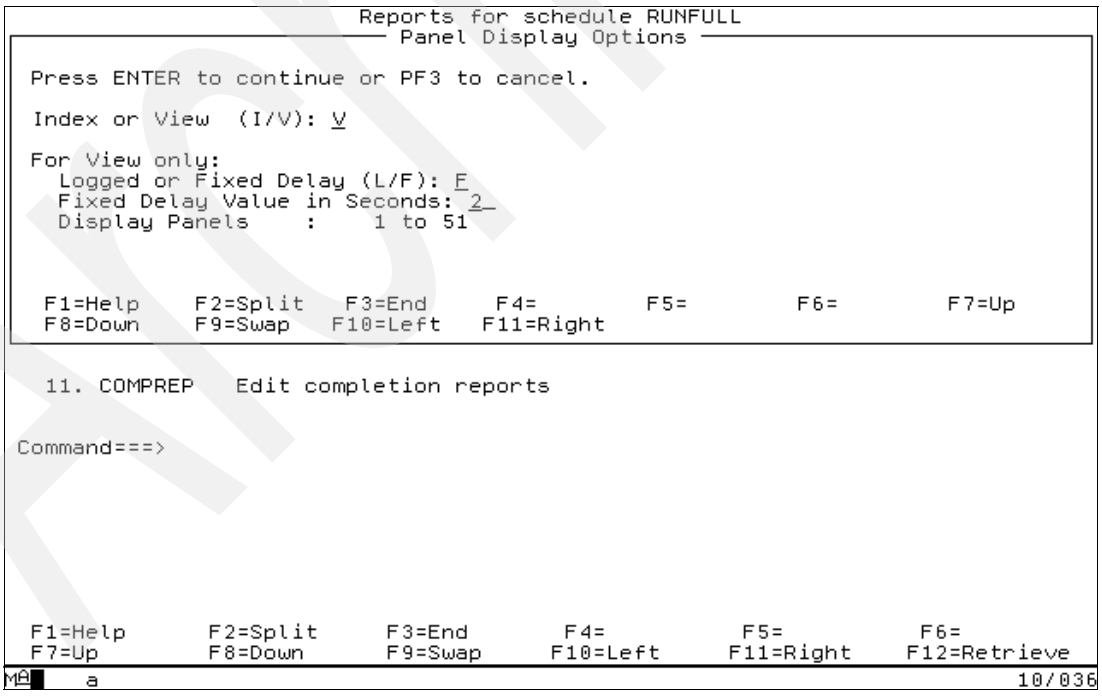


Figure 30-156 Options for log display monitor

31. Enter the line command **V** to view the screen captured when executing the test case LOGON when presented with the index. The user ID is the first one from the UTBL CICSUIS, as shown in the Figure 30-157.

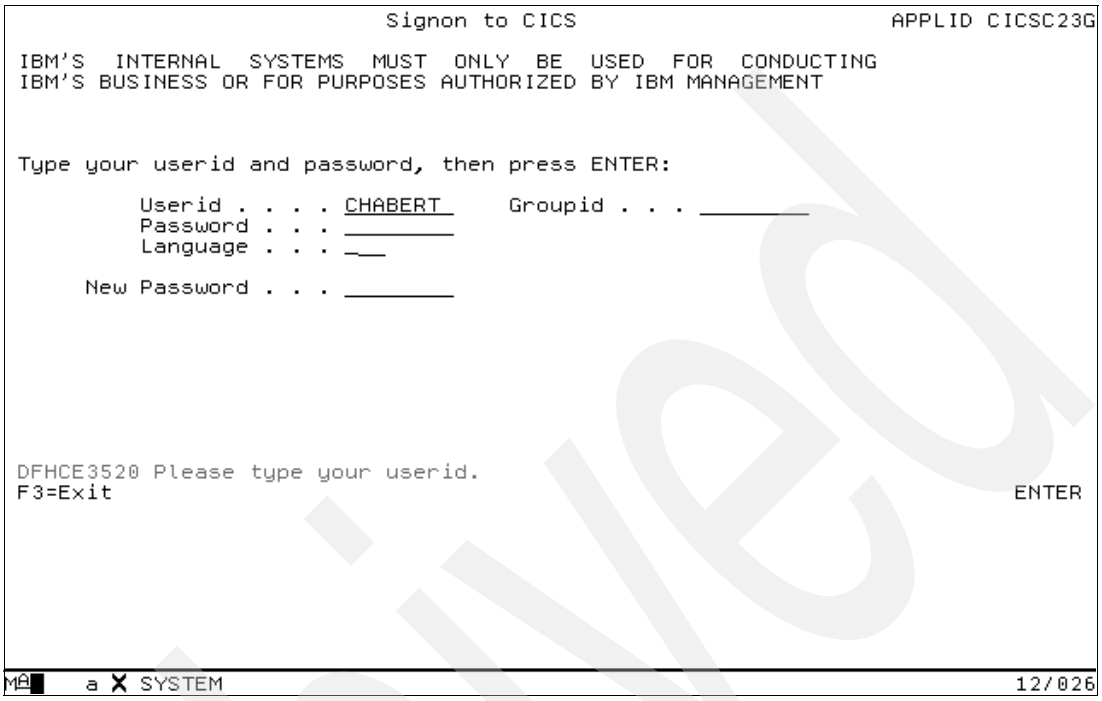


Figure 30-157 First captured Trader application logon screen

32. Press PF3 to return to the index of screens, scroll down to the next logon screen, and use the line command **V** to view it, too, as shown Figure 30-158.

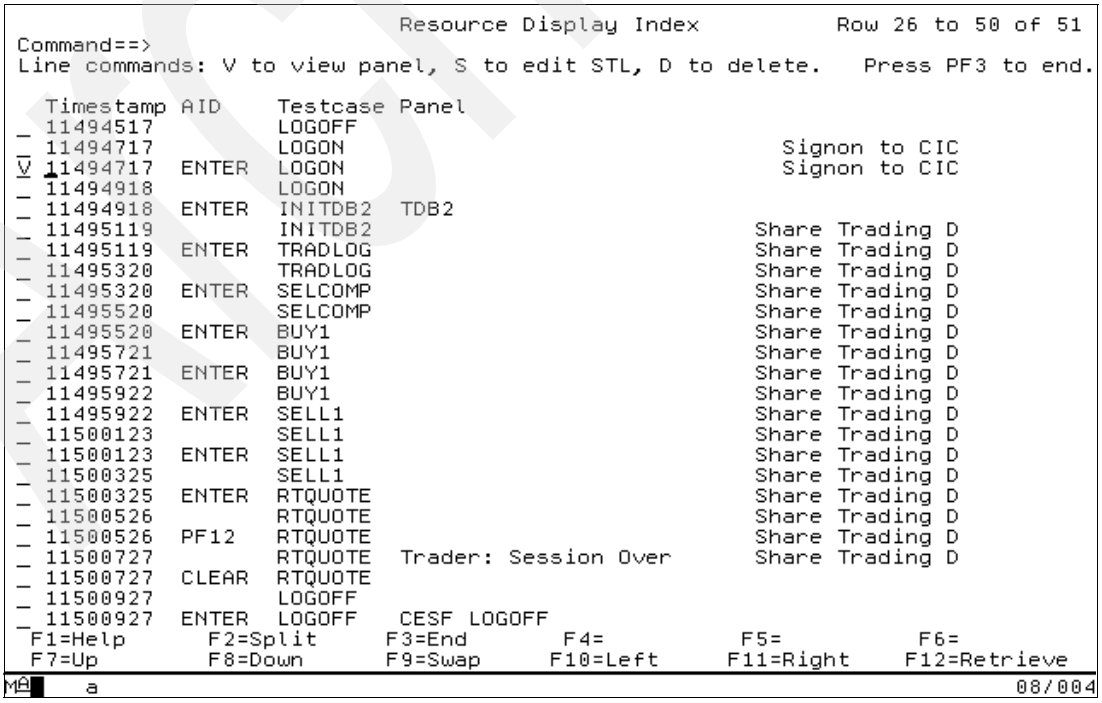


Figure 30-158 Capture screen selected for view

33. The user ID used this time is the second one from the UTBL used, as shown in Figure 30-159.

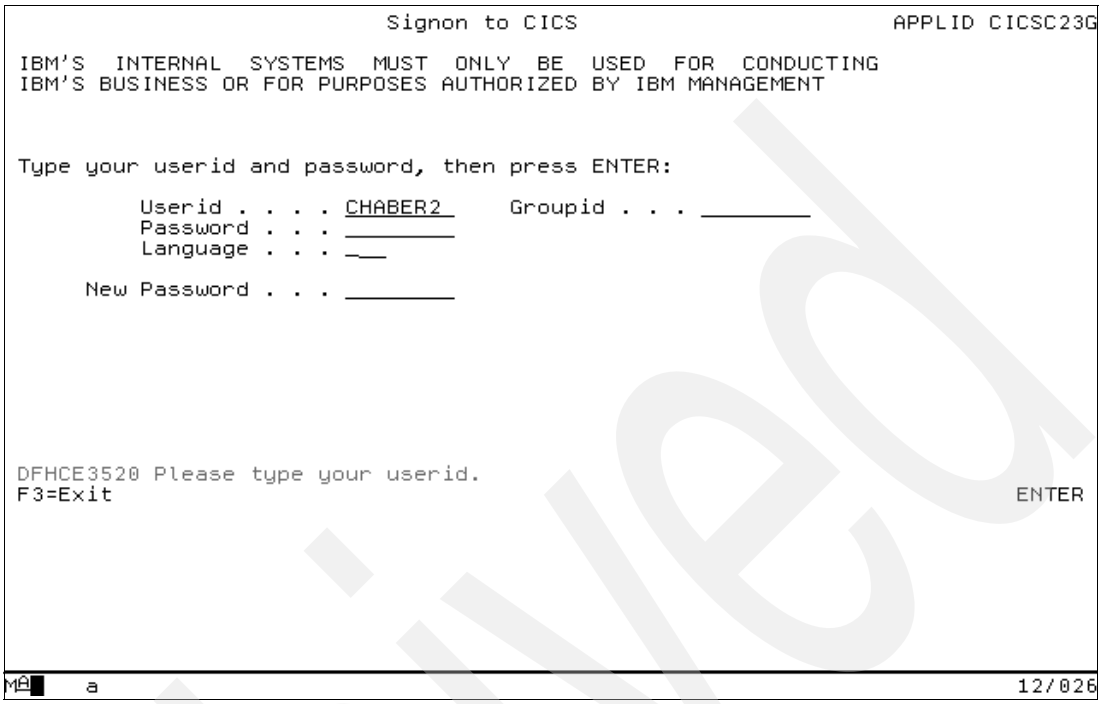


Figure 30-159 Second captured Trader application logon screen

30.9 WSim in batch mode

As we already mentioned, WTM can run in HIDE, DISPLAY, and INTERACT modes. In the INTEACT mode WTM displays WSim panels and allows you to manually change values, which gives you more control. In this mode WTM can generate JCL for submitting for batch processing.

1. Enter the INTERECT command from the WTM main panel to directly switch WTM into the INTERACT mode, as shown in Figure 30-160, or use any other way to navigate to the WTM Variables and Options panel and change the value of the Interface Access field.

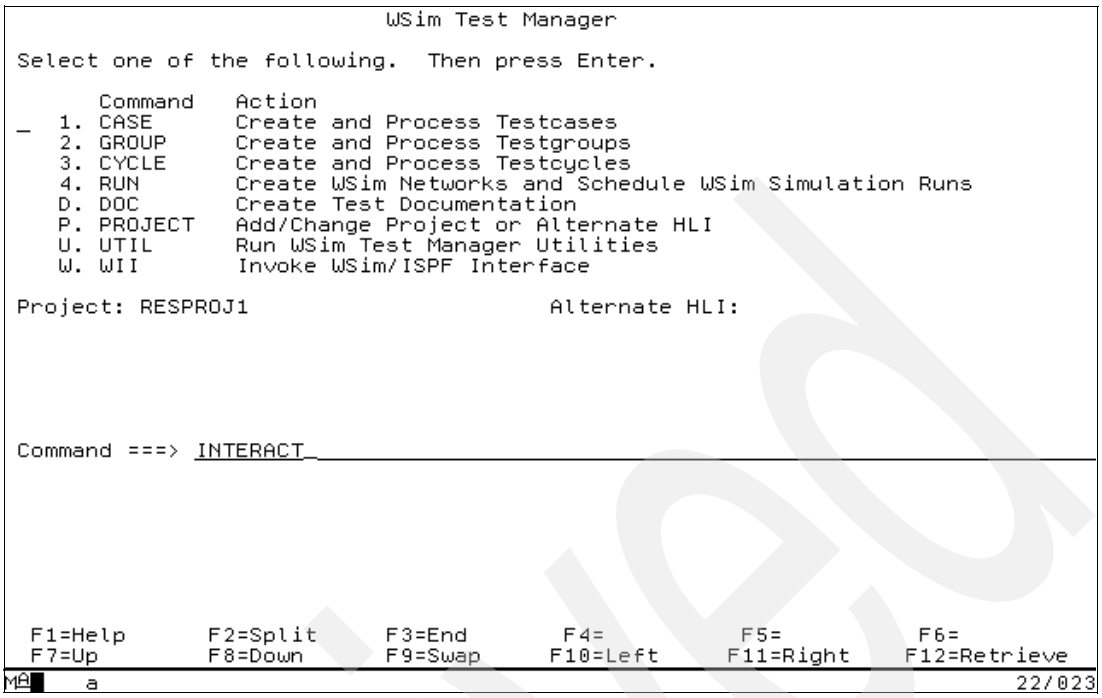


Figure 30-160 Switching to INTERACT mode

2. To run a simulation, select option **4** or enter the command **run** on the WTM main panel, as shown in Figure 30-161.

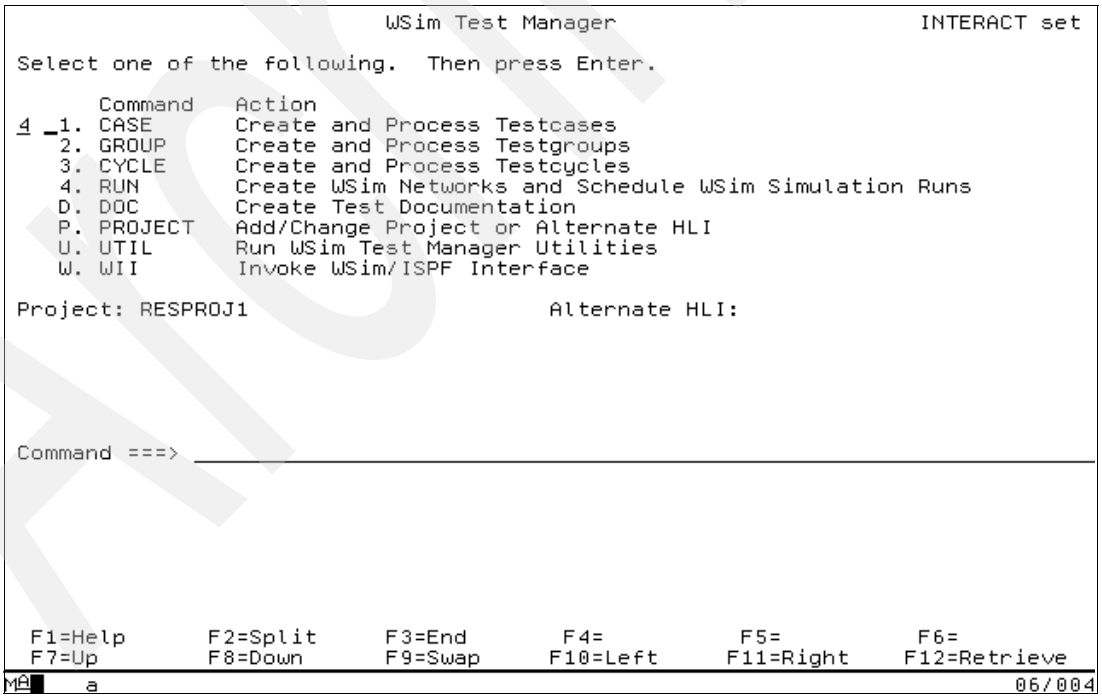


Figure 30-161 Run simulation selected

3. Enter the line command **X** to run the RunFull schedule, as presented in Figure 30-162. Press Enter.

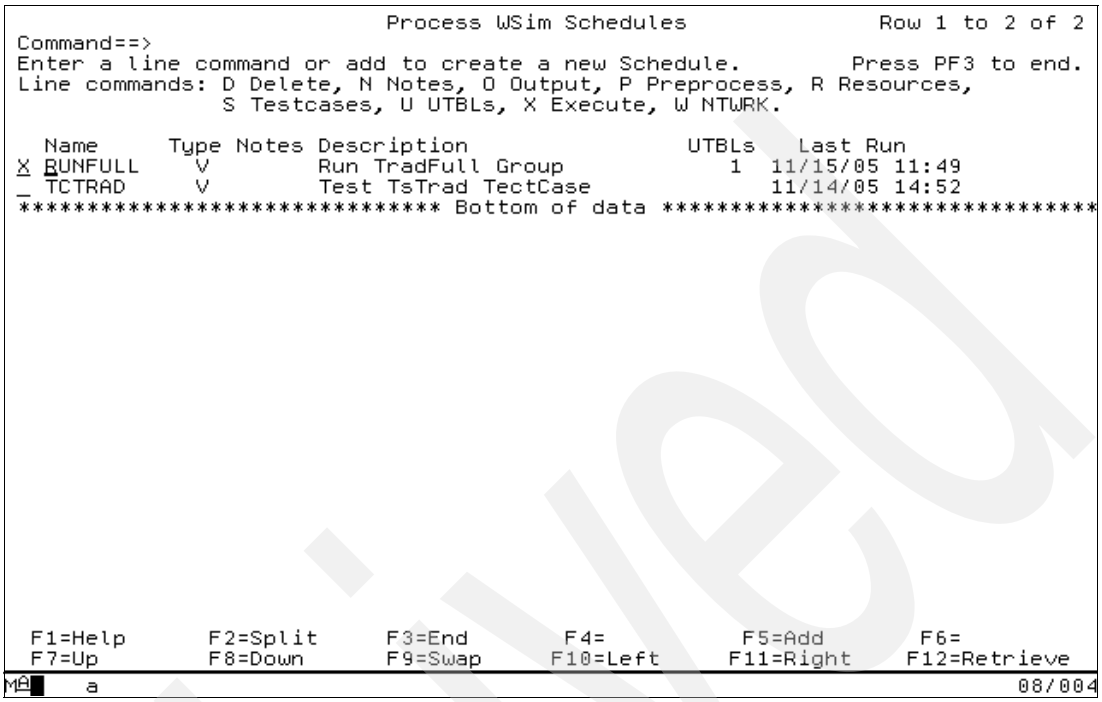


Figure 30-162 Running schedule RunFull

4. WTM provides an opportunity to specify names for log data sets. Accept the defaults or change names if preferred. A completion report is optional, as shown in Figure 30-163. Press Enter.

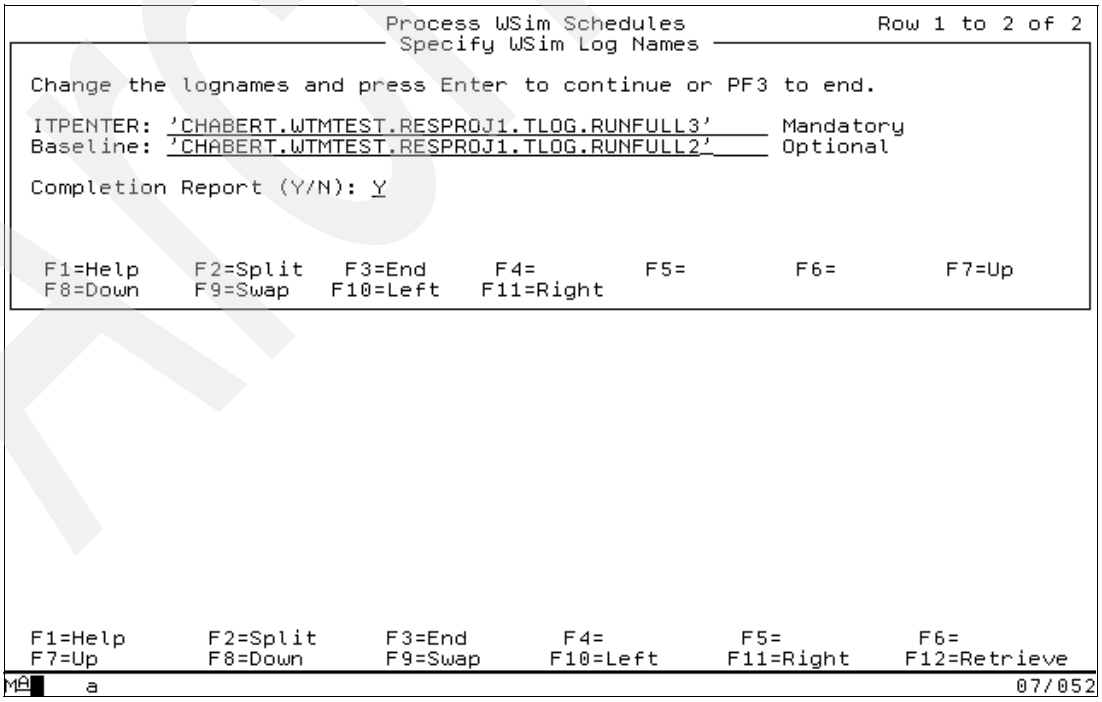


Figure 30-163 Simulation log names

5. The next panel is a WSim ISPF interface and not a WTM panel. It would not be displayed in HIDE mode. DISPLAY mode allows you to see the panel, and the INTERACT mode allows you to work with it. Part 1 is shown in Figure 30-164. The batch run mode is specified by entering B in the corresponding field.

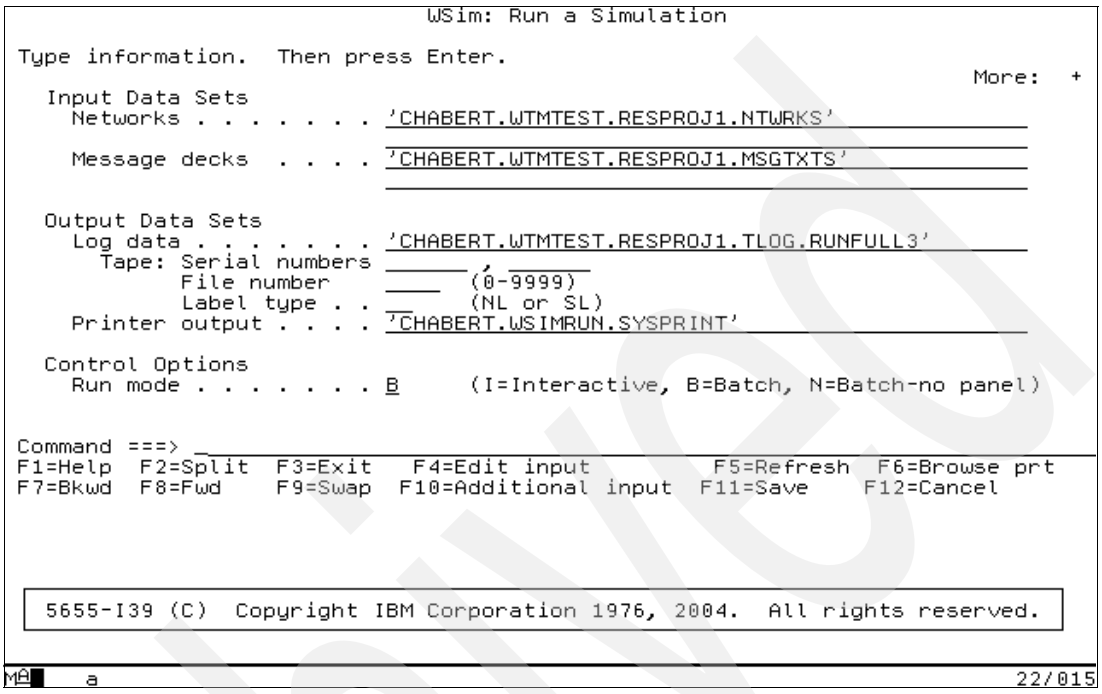


Figure 30-164 Batch simulation selected

6. Part 2 (scroll down using PF8) is shown in Figure 30-165. Press Enter.

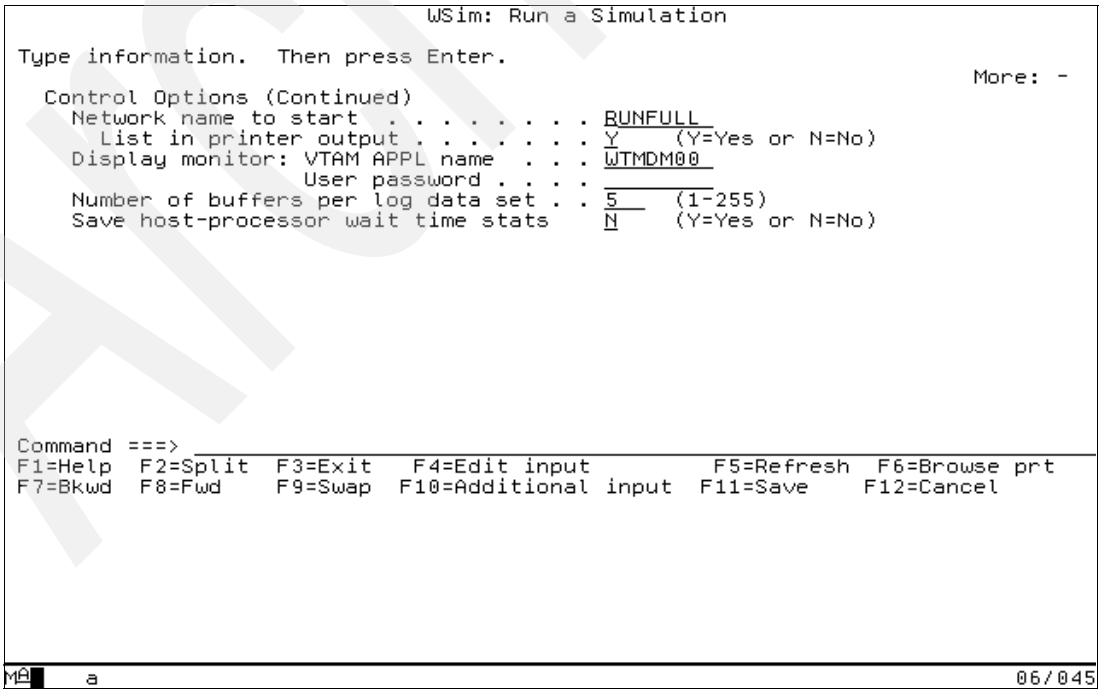


Figure 30-165 Second part of Run simulation panel

7. On the next panel, shown in Figure 30-166, you can modify the JOB card as required and optionally specify the data set to save the generated JCL. This JCL can later be resubmitted without going through either WSim or WTM.

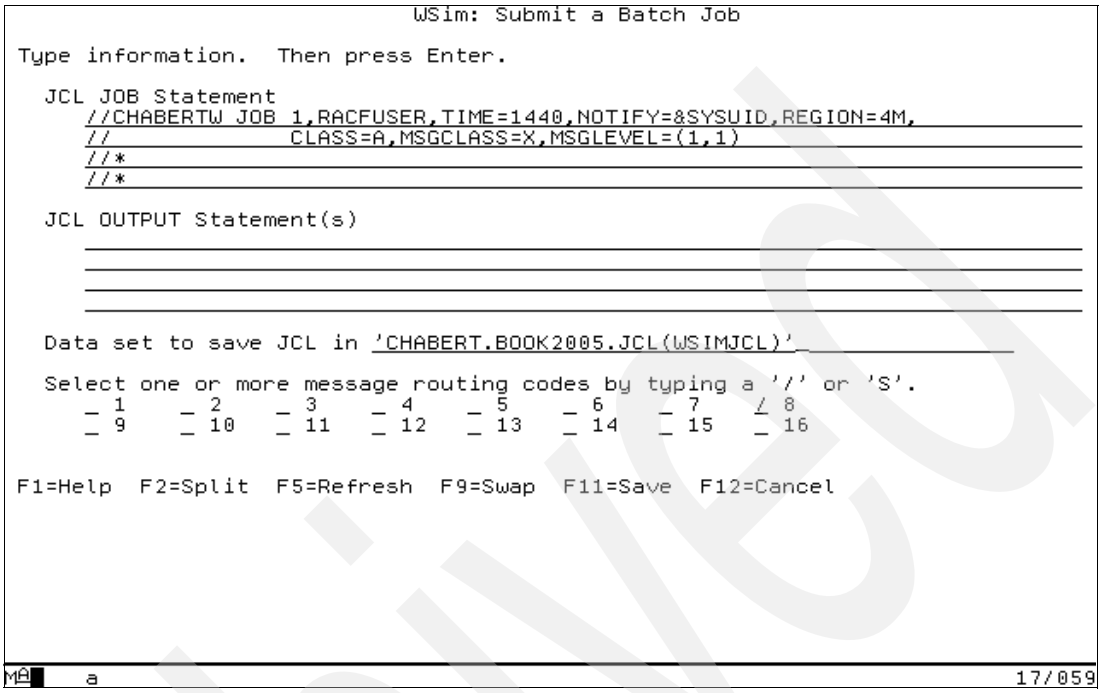


Figure 30-166 JOB card for batch simulation

8. The batch job is submitted and WSim displays the message shown in Figure 30-167.

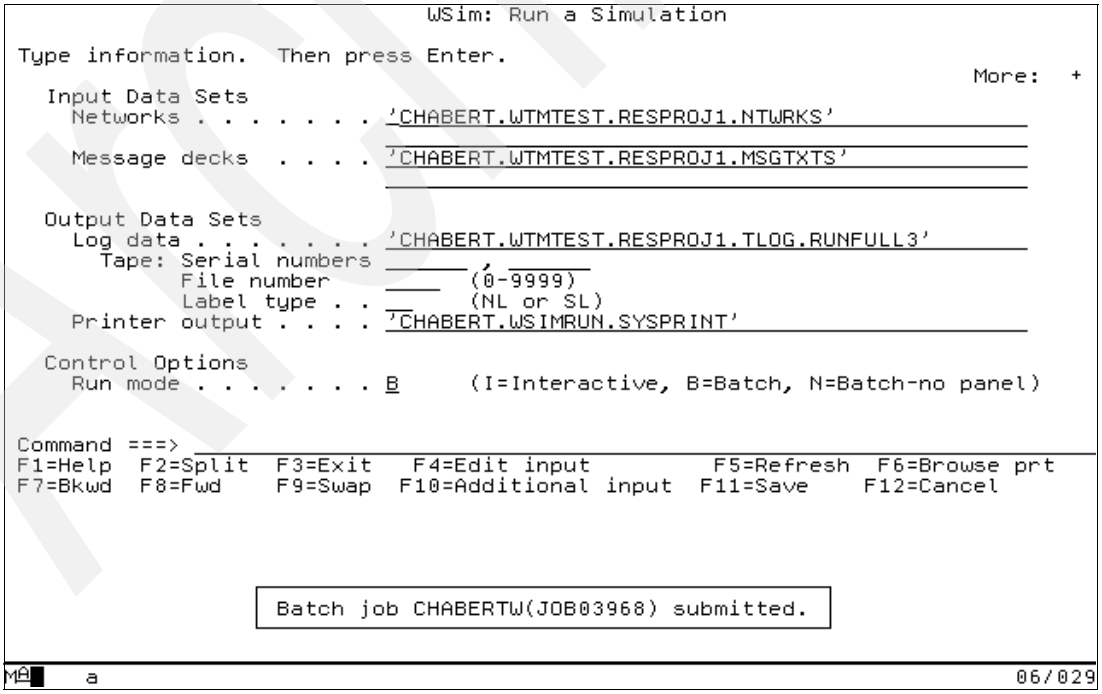


Figure 30-167 Job is submitted

9. The job runs successfully, as shown in Figure 30-168.

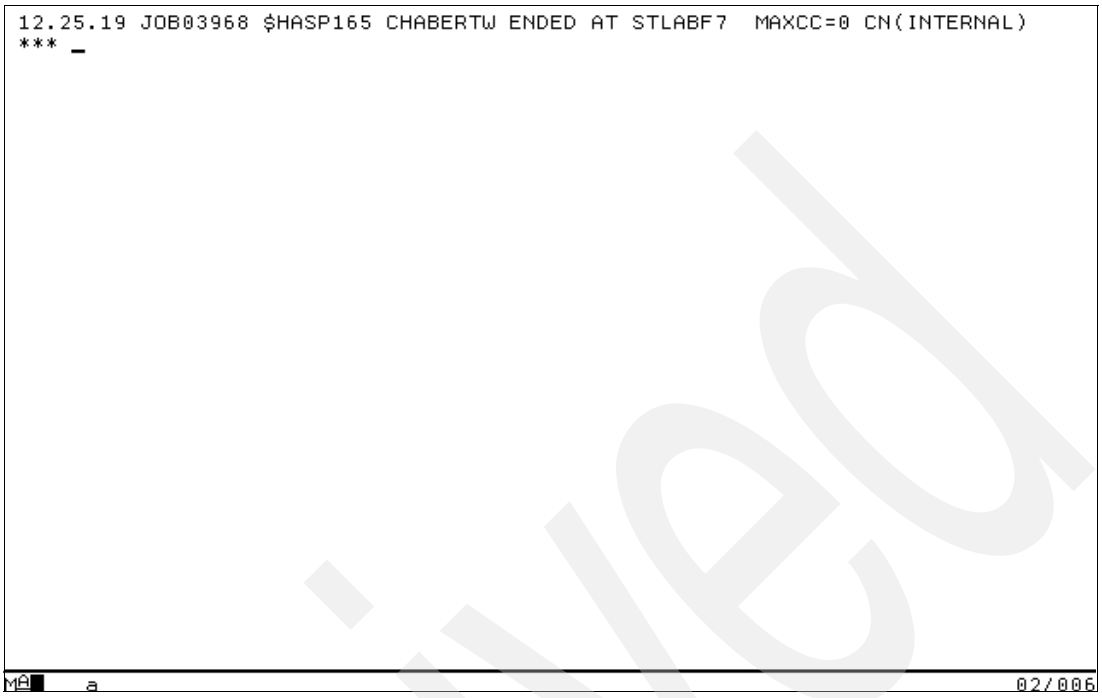


Figure 30-168 Successful run

You can look at the output in the system job log or use the online WTM panels to review reports.

The generated JCL is presented in Figure 30-169. It can be reused as required.

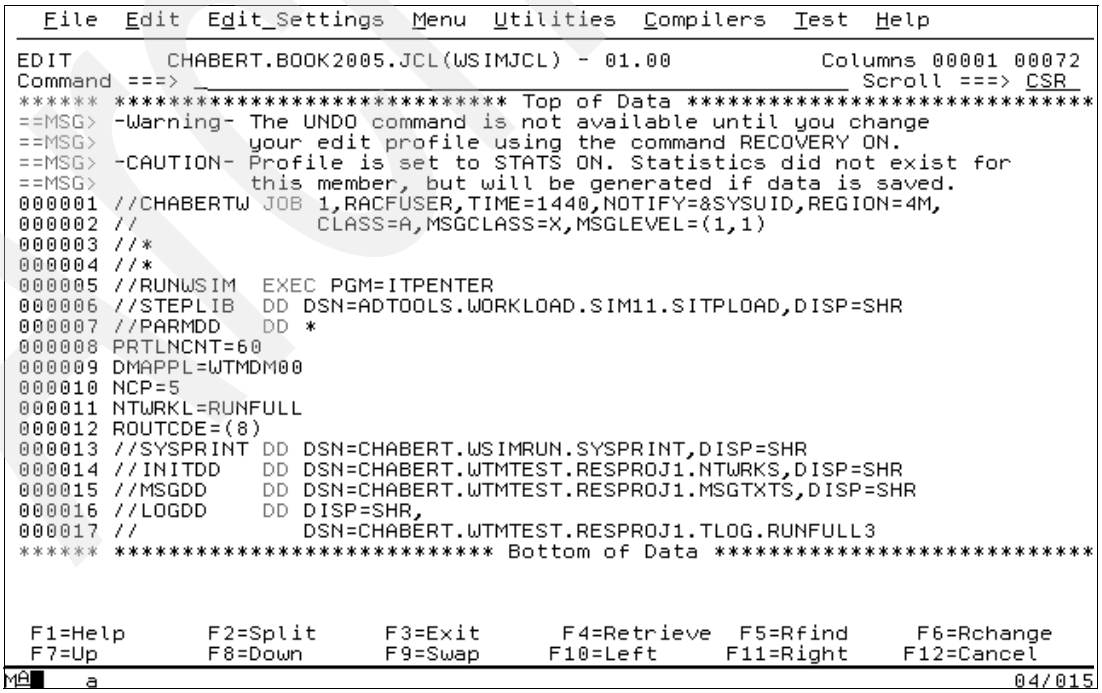


Figure 30-169 Generated JCL

30.10 Regression test

When preparing for the simulation batch run, since it was not the first time we were running the RunFull simulation, we specified names for new and baseline simulation logs on the WTM panel used to specify the names of the log data sets, as shown in Figure 30-163 on page 1049.

We now compare the results of these two runs and see if there are any differences.

1. Enter the line command **0** to work with the output from the RunFull simulation, as shown in Figure 30-170.

```
                          Process WSim Schedules              Row 1 to 2 of 2
Command==>
Enter a line command or add to create a new Schedule.         Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                UTBLs   Last Run
0 RUNFULL    V        Run TradFull Group            1  11/15/05 13:03
_ TCTRAD     V        Test TsTrad TectCase             11/14/05 14:52
****************************** Bottom of data ********************************
 F1=Help      F2=Split     F3=End       F4=          F5=Add       F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA  a                                                                  08/004
```

Figure 30-170 Working with schedule RunFull output

2. On the WTM Reports panel select option **8** to view the screen compare report, as shown in Figure 30-171. Press Enter.

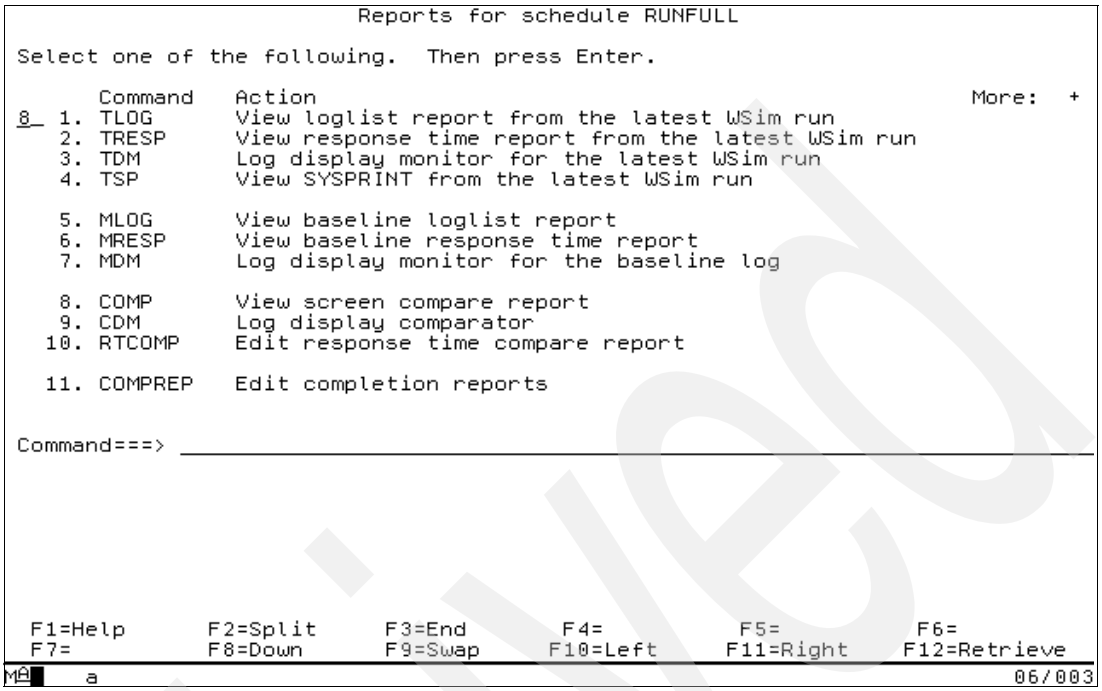


Figure 30-171 Viewing screen compare report

3. WTM displays the panel with the names of the log data sets to be compared and where to put the results, as shown in Figure 30-172. Press Enter.

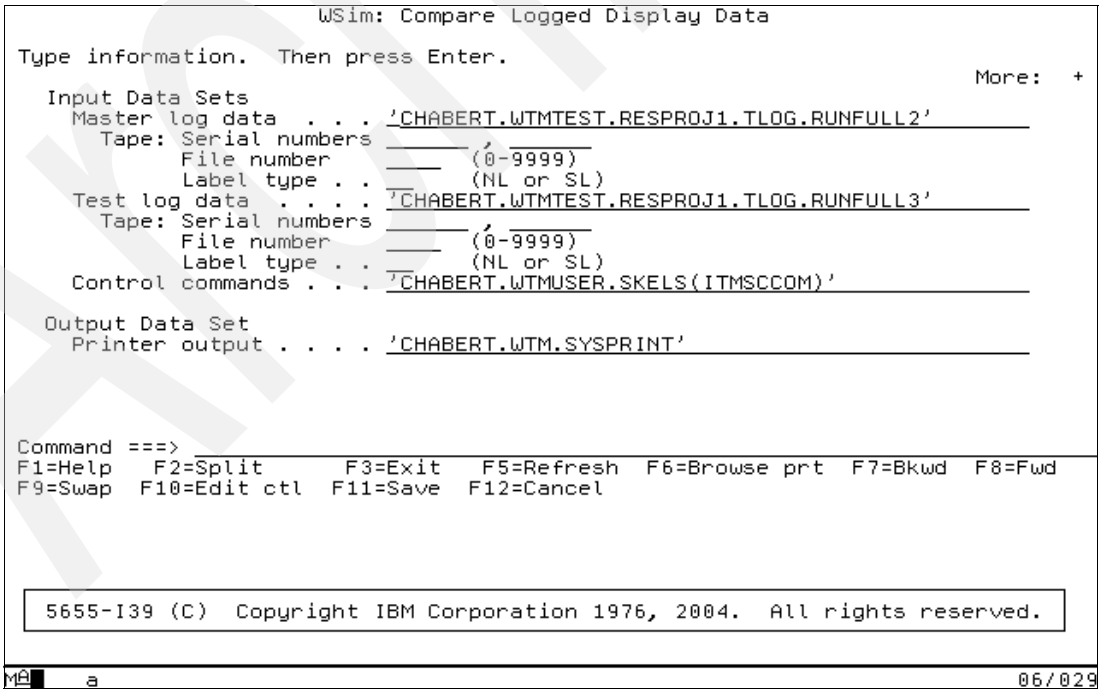


Figure 30-172 Log data sets to compare

4. WTM reports that a difference was found for at least one resource, as shown in Figure 30-173.

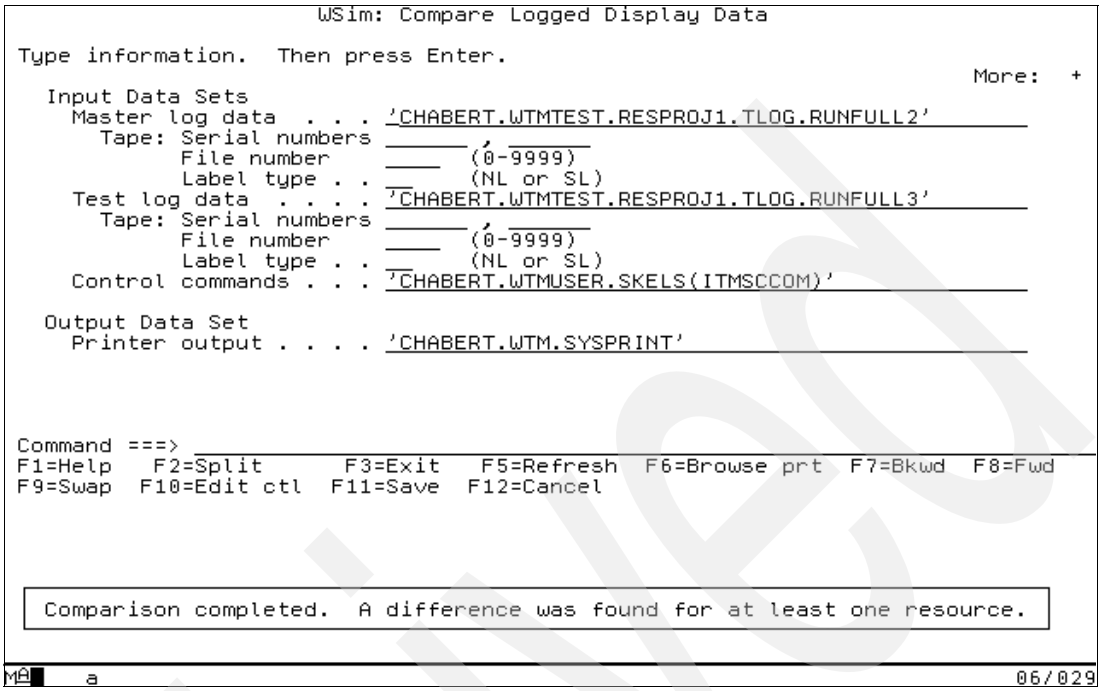


Figure 30-173 Difference found

5. To see it press PF6. Scroll down to see lines 148 and 149. They contain information about the master and the test runs, as shown in Figure 30-174.

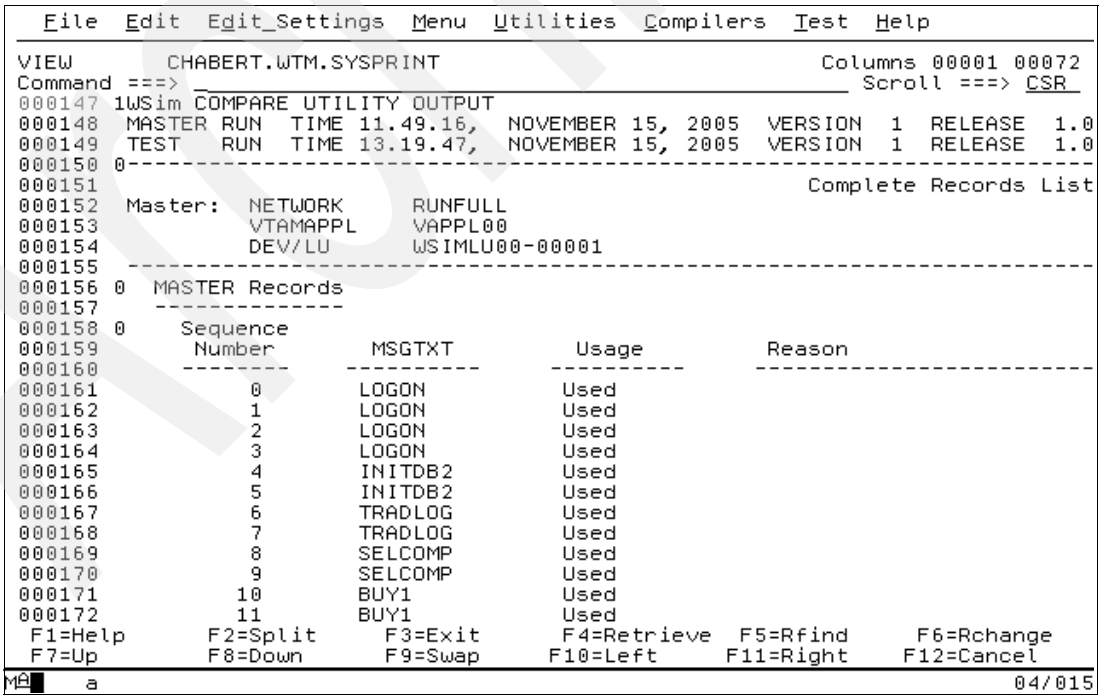


Figure 30-174 Master and test run information

6. The master (baseline) screens (the corresponding MSGTXT programs) are listed starting from line 156, as shown in Figure 30-175.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
VIEW       CHABERT.WTM.SYSPRINT                          Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
000156 0  MASTER Records
000157    --------------
000158 0    Sequence
000159       Number       MSGTXT         Usage          Reason
000160       --------    ----------     ----------     -------------------------
000161            0       LOGON          Used
000162            1       LOGON          Used
000163            2       LOGON          Used
000164            3       LOGON          Used
000165            4       INITDB2        Used
000166            5       INITDB2        Used
000167            6       TRADLOG        Used
000168            7       TRADLOG        Used
000169            8       SELCOMP        Used
000170            9       SELCOMP        Used
000171           10       BUY1           Used
000172           11       BUY1           Used
000173           12       BUY1           Used
000174           13       BUY1           Used
000175           14       SELL1          Used
000176           15       SELL1          Used
000177           16       SELL1          Used
000178           17       SELL1          Used
000179           18       RTQUOTE        Used
000180           19       RTQUOTE        Used
000181           20       RTQUOTE        Used
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
MA    a                                                            05/002
```

Figure 30-175 Master run records listed

7. Scroll down to line 224. Starting from this line the screens used in the test run are listed, as shown in Figure 30-176.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
VIEW       CHABERT.WTM.SYSPRINT                          Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
000224 -  TEST Records
000225    ------------
000226 0    Sequence
000227       Number       MSGTXT         Usage          Reason
000228       --------    ----------     ----------     -------------------------
000229            0       LOGON          Used
000230            1       LOGON          Used
000231            2       LOGON          Used
000232            3       LOGON          Used
000233            4       INITDB2        Used
000234            5       INITDB2        Used
000235            6       TRADLOG        Used
000236            7       TRADLOG        Used
000237            8       SELCOMP        Used
000238            9       SELCOMP        Used
000239           10       BUY1           Used
000240           11       BUY1           Used
000241           12       BUY1           Used
000242           13       BUY1           Used
000243           14       SELL1          Used
000244           15       SELL1          Used
000245           16       SELL1          Used
000246           17       SELL1          Used
000247           18       RTQUOTE        Used
000248           19       RTQUOTE        Used
000249           20       RTQUOTE        Used
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
MA    a                                                            04/015
```

Figure 30-176 Test run records listed

8. Continue scrolling down. There are differences reported (for more clarity, we passed some differences found earlier in the report), as shown in Figure 30-177.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
VIEW       CHABERT.WTM.SYSPRINT                          Columns 00001 00072
Command ===> _                                            Scroll ===> CSR
000301 0 MASTER    TEST                                   Screen
000302  Sequence Sequence                  All             Compare
000303   Number   Number  Checkonly Mask  Mask  Result   Differences Reason For
000304  -------- -------- --------- ----- ---- --------- ----------- ----------
000305        0        0                          Equal
000306        1        1                          Equal
000307        2        2                          Equal
000308        3        3                          Equal
000309        4        4                          Equal
000310        5        5                          Equal
000311        6        6                          Equal
000312        7        7                          Equal
000313        8        8                          Equal
000314        9        9                          Equal
000315       10       10                          Equal
000316       11       11                          Equal
000317       12       12                          Equal
000318       13       13                          Equal
000319       14       14                          Equal
000320       15       15                          Equal
000321       16       16                          Equal
000322       17       17                          Equal
000323       18       18                          Equal
000324       19       19                      Not Equal        2    Data Diffe
000325       20       20                      Not Equal        2    Data Diffe
000326       21       21                      Not Equal        2    Data Diffe
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
MA    a                                                                04/015
```

Figure 30-177 Differences are reported

9. The captured screens, where differences were found, are also presented at the end of this report.

The log display comparator report

Use the following steps to see another type of WTM report: the Log display comparator.

1. Return to the WTM reports panel and select option **9**, as shown in Figure 30-178.

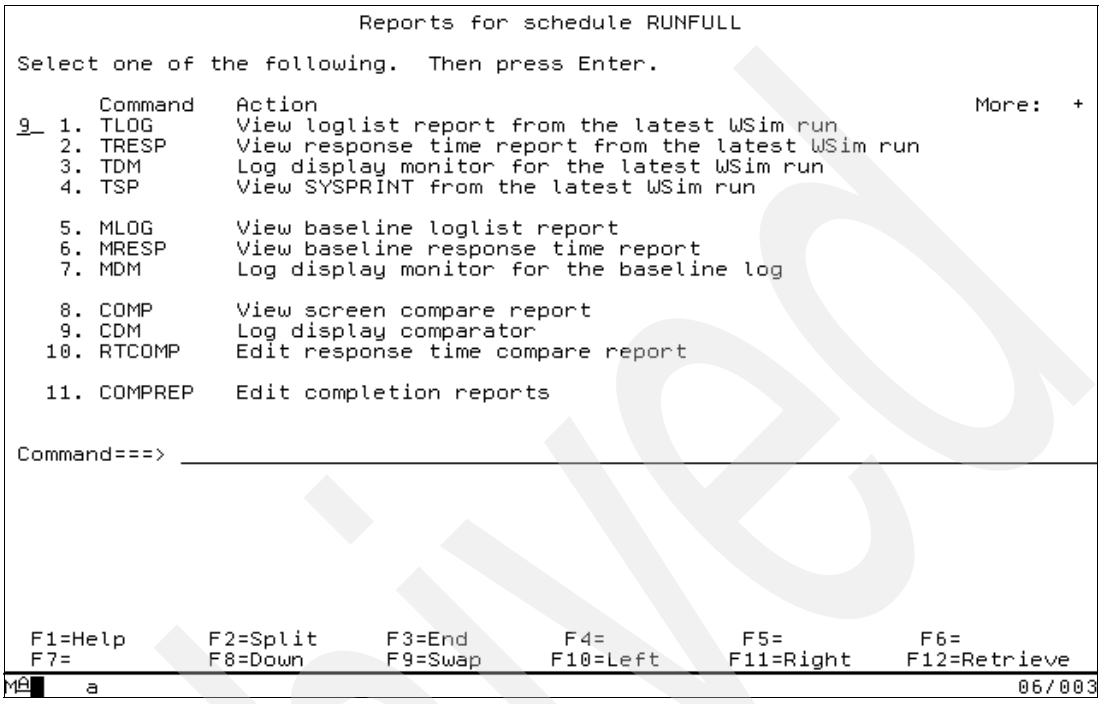


Figure 30-178 Log display comparator selected

2. On the Panel Display Options select to view the index of screen images, as shown in Figure 30-179.

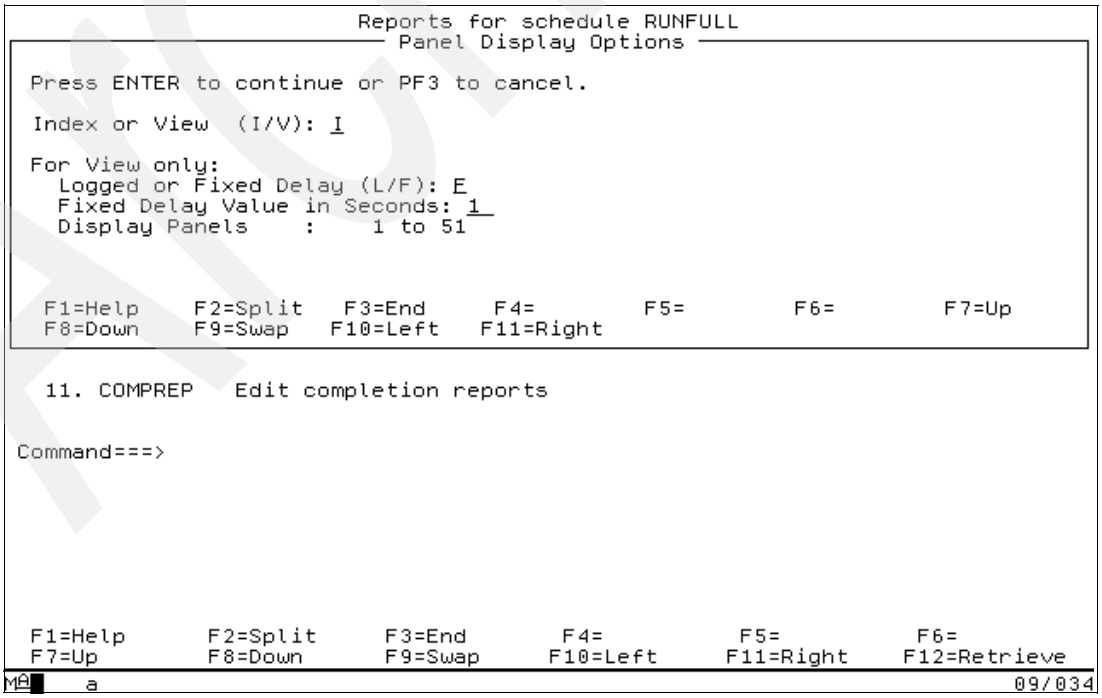


Figure 30-179 Report options

3. The next panel shows the index of the screens captured from the test run with the detected differences flagged (six in total). Enter the line command **v** in front of one of the screen images with differences and press Enter, as shown in Figure 30-180.

```
                          Resource Display Index          Row 1 to 24 of 51
Command ==>
Enter line command                          Number of screen differences:     6
Line commands: V to view panel, S to edit STL, D to delete.   Press PF3 to end.

  Timestamp AID    Testcase D Panel
_ 11491907          LOGON
_ 11492108          LOGON                                    Signon to C
_ 11492108  ENTER   LOGON                                    Signon to C
_ 11492308          LOGON
_ 11492308  ENTER   INITDB2    TDB2
_ 11492509          INITDB2                                Share Trading
_ 11492509  ENTER   TRADLOG                                Share Trading
_ 11492710          TRADLOG                                Share Trading
_ 11492710  ENTER   SELCOMP                                Share Trading
_ 11492910          SELCOMP                                Share Trading
_ 11492910  ENTER   BUY1                                   Share Trading
_ 11493111          BUY1                                   Share Trading
_ 11493111  ENTER   BUY1                                   Share Trading
_ 11493313          BUY1                                   Share Trading
_ 11493313  ENTER   SELL1                                  Share Trading
_ 11493513          SELL1                                  Share Trading
_ 11493513  ENTER   SELL1                                  Share Trading
_ 11493715          SELL1                                  Share Trading
_ 11493715  ENTER   RTQUOTE                                Share Trading
_ 11493916          RTQUOTE  * Data Difference Detected
_ 11493916  PF12    RTQUOTE  * Data Difference Detected
_ 11494117          RTQUOTE  * Data Difference Detected
_ 11494117  CLEAR   RTQUOTE
_ 11494317          LOGOFF
 F1=Help      F2=Split     F3=End       F4=          F5=          F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA    a                                                          07/002
```

Figure 30-180 Resource display index with differences reported

4. Select V and press Enter to see how WTM *flashes* the two screen images one over another to highlight the differences.

30.11 Display Monitor Facility

The Display Monitor Facility is a VTAM application program within WSim that can be used to:

- Display simulated 3270 screens.
- Display transmitted and received data flows for any simulated device.

It is activated whenever WSim simulation is running.

The Display Monitor Facility can be used to develop and debug scripts for display devices, to dynamically monitor tests when they are running, and to show interactions with host applications.

To use the Display Monitor Facility, you have to know the name of the Display Monitor VTAM APPL. The WSim installer should provide this name, which also should have been specified on the WTM Variables panel, as shown in Figure 30-181. To get there, enter the VARS command on the WTM main panel.

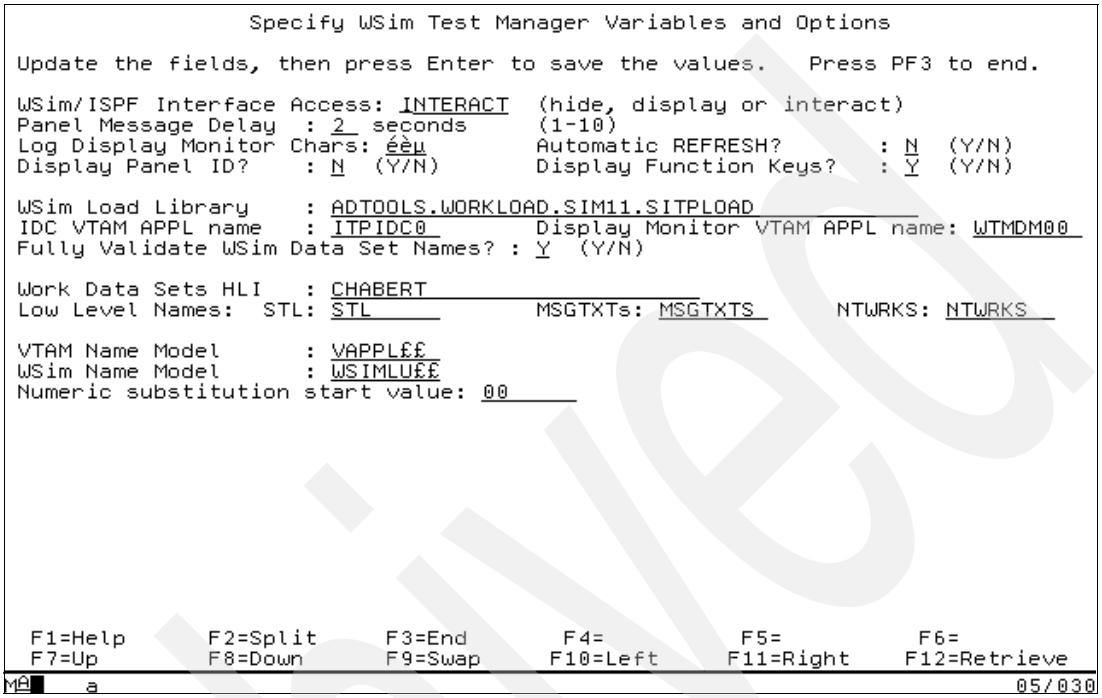


Figure 30-181 WTM variables panel

In our case, the Display Monitor VTAM APLL name is WTMDM00. This is the name that you have to log on to in order to use the facility.

Enter the RUN command from the WTM main panel or use option **4** to start working with WTM schedules.

Enter the line command **X** to start the simulation for the schedule TCTRAD. When it is started, have another terminal session open to connect to the Display Monitor VTAM APPL WTMDM00. The command shown in Figure 30-182 worked on the demonstration system we used. Consult with your system programmer for details in your specific environment.

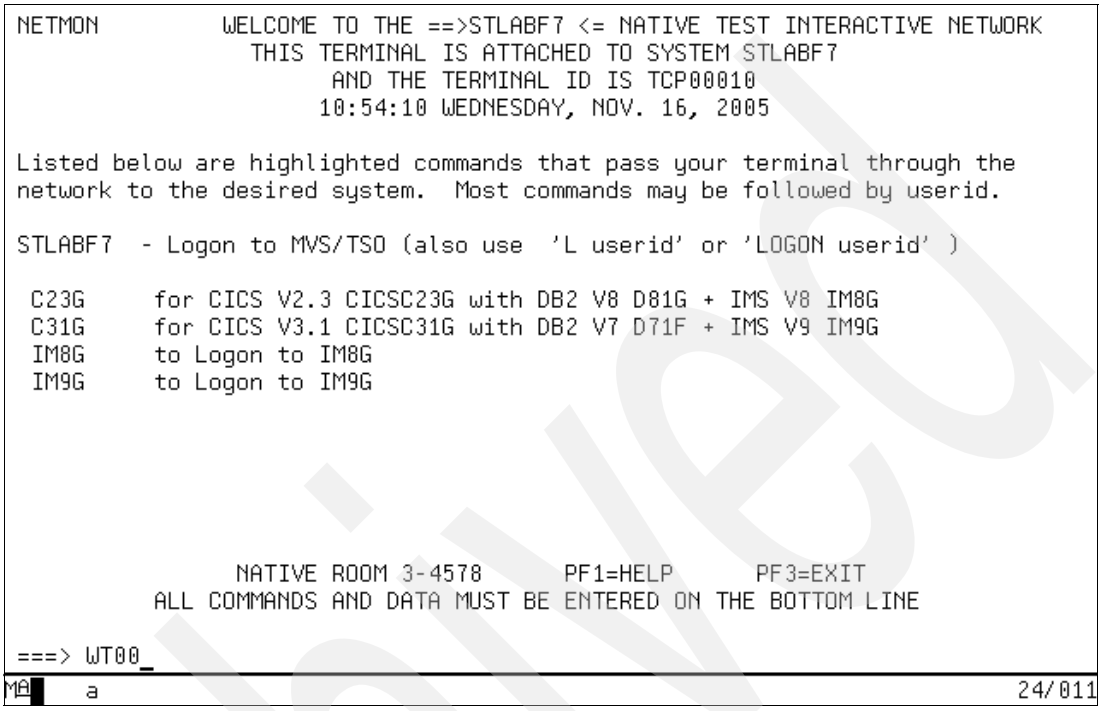


Figure 30-182 Log on to WTMDM00

As the simulation continues, the messages on the simulation console show the used network resource names, which were generated by WTM using the naming schema defined on the WTM variables panel (in our case it is WSIMLU##). You can also see the list of the resources for a schedule by entering the line command **r** on the WTM Process schedules panel. The schedule TSTTRAD1 is using the WSim name WSIMLU00.

On the Display Monitor Facility control panel shown in Figure 30-183, enter the name of the simulated resource (terminal) to be monitored (WSIMLU00) and press Enter.

```
            WSim Version 1 Release 1.0.1 Display Monitor Facility

Name   = WSIMLU00_              WSim name of simulated device or 3270 display
View   = SCREEN         DATA or SCREEN - show data stream or 3270 screen image

Screen image display only:
Update = XMITRECV       Monitoring display updated when:
                          MONITOR  - MONITOR statement is executed from script,
                          TIMER    - the specified time value expires, or
                          XMITRECV - data is transmitted/received by display.
Source = BLOCKS         Data stream sent to the monitoring display built from:
                          BLOCKS - WSim internal control blocks
                          DATA   - data transmitted/received by display.
Timer  = 10             1-600 Seconds when Update = TIMER
Aid    = ON             ON, OFF, or (row,column) location of AID display field

Data stream display only:
Lines  = 2              Maximum number of displayed data lines
Code   = EBCDIC         ASCII or EBCDIC - interpret data as ASCII or EBCDIC


ENTER    - Submits parameters to start monitoring of simulated display.
PA1/ATTN - Stops monitoring of simulated display.
PF3/PF15 - Ends Display Monitor Facility session.
MA    a                                                          03/019
```

Figure 30-183 Display Monitor Facility panel

The traffic on the simulated terminal appears on the Display Monitor Facility screen. We do not present these screens in this book, as they are just a replay of the captured screens.

Press PF3 to exit or use PA1 to interrupt and go back to the Display Monitor Facility control panel to change any options.

30.12 Load test

Load testing of applications requires running multiple simulated terminals concurrently and that each of the terminals is used by a different user. To perform load testing:

1. Enter the command **run** from the WTM main panel and then enter the command **add** on the next WTM panel to start working with the new schedule RUNLOAD, as shown in Figure 30-184.

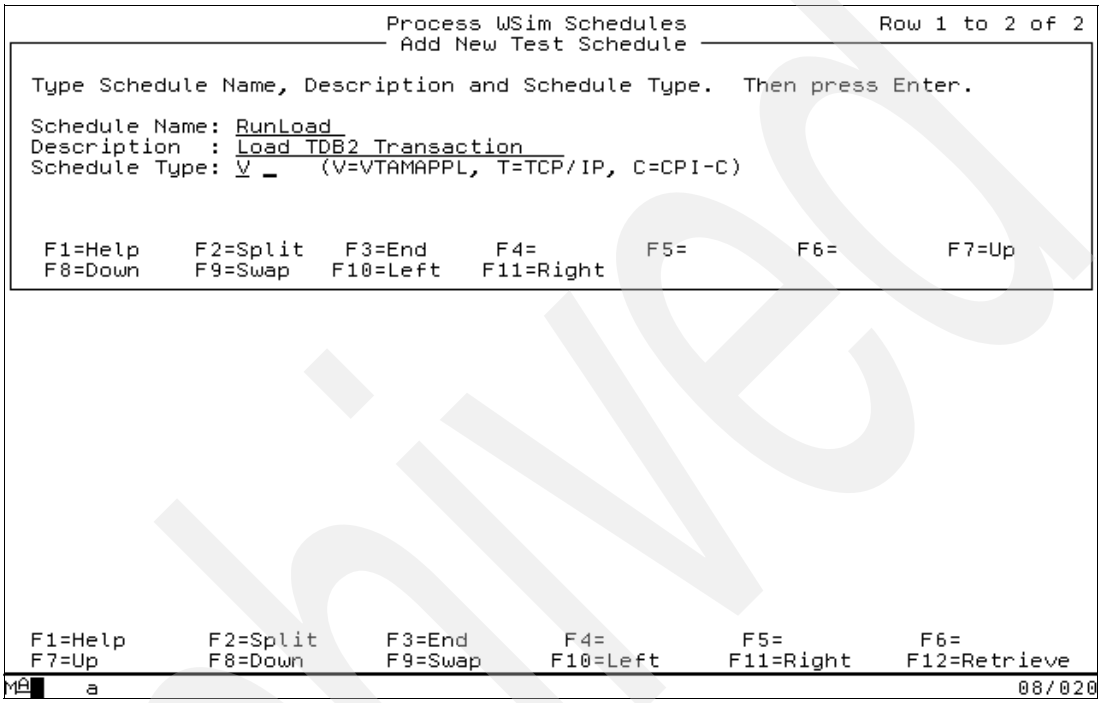


Figure 30-184 New schedule RunLoad

2. Press Enter. WTM displays a Milestones panel with information about the steps to be performed in order to create a new schedule, as shown in Figure 30-185.

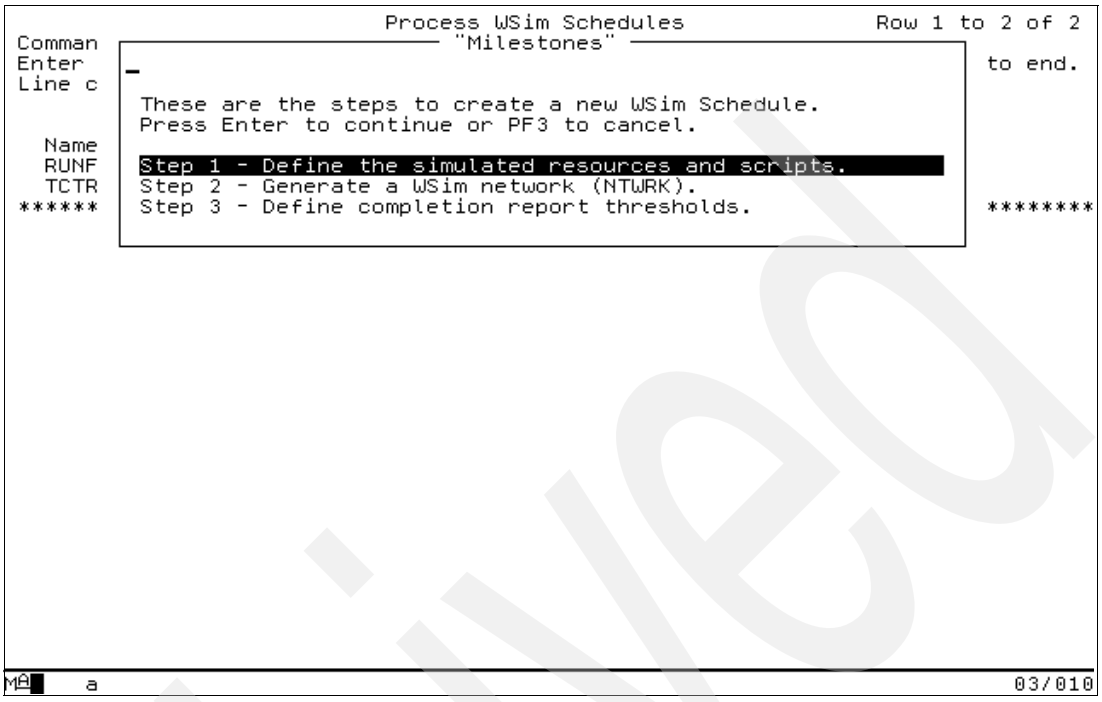


Figure 30-185 Milestones panel shows steps to be performed

3. The WTM Resource List panel for the schedule RUNLOAD is displayed, as shown in Figure 30-186, with the assigned VTAM and WSim names of the network resources. Enter the line command **s** to continue, and press Enter.

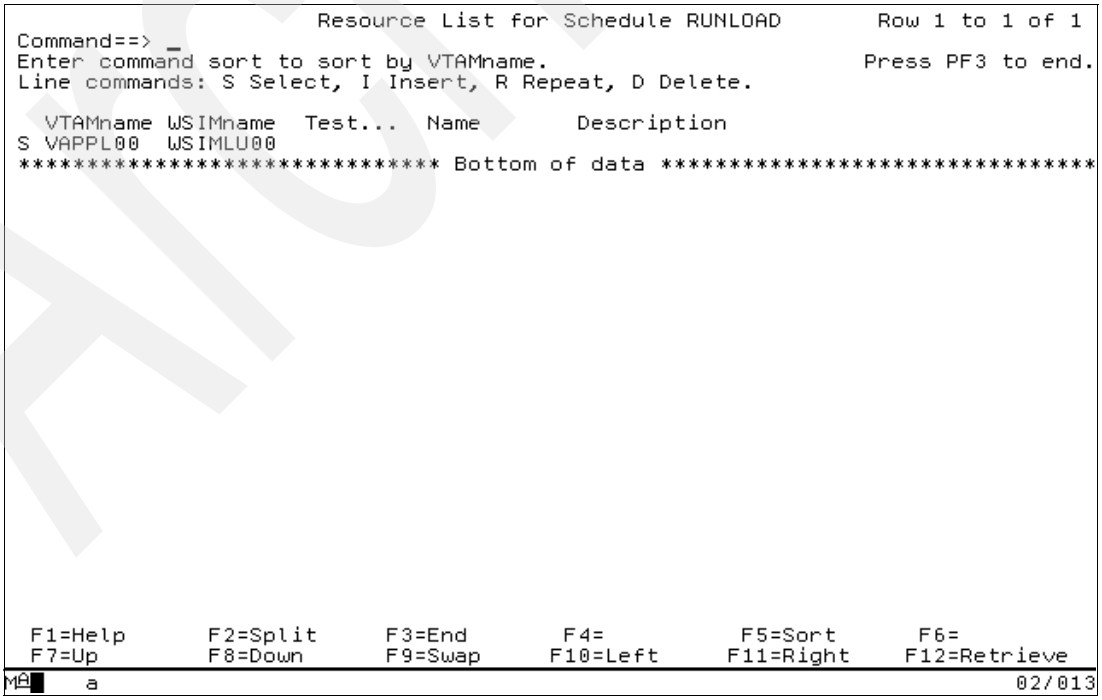


Figure 30-186 Simulated resources list for schedule Runload

4. In this simulation example, the test group TradFull has to be executed twice, as shown in Figure 30-187. You can schedule any combination of test items to be executed multiple times in any compatible sequence. Press PF3 to continue.

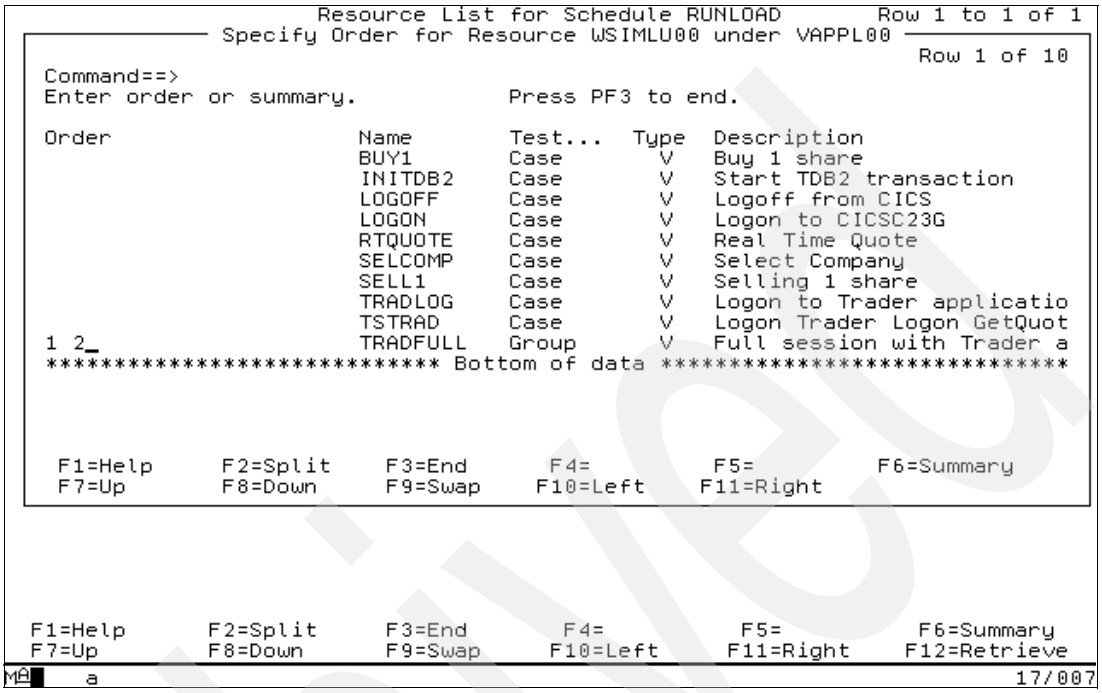


Figure 30-187 Testgroup TradFull to be executed twice

5. WTM returns to the Resources list panel and shows the test group TradFull being executed twice for this simulated terminal. To add an additional terminal use the line command **R** to repeat the terminal definition as shown in Figure 30-188, and press Enter.

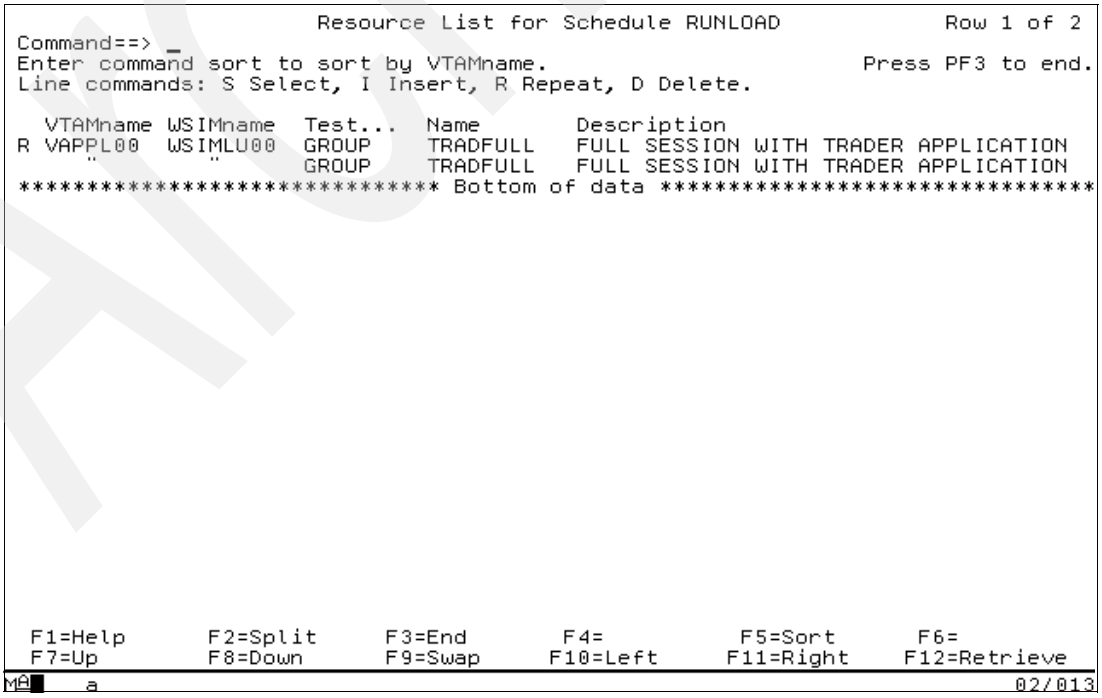


Figure 30-188 New simulated resource to be added

6. The number of the resources that can be simulated is limited. The maximum is defined by the product installer in the WTM VTAM application major node. For this simulation, leave the default value of one and press Enter, as shown in Figure 30-189.

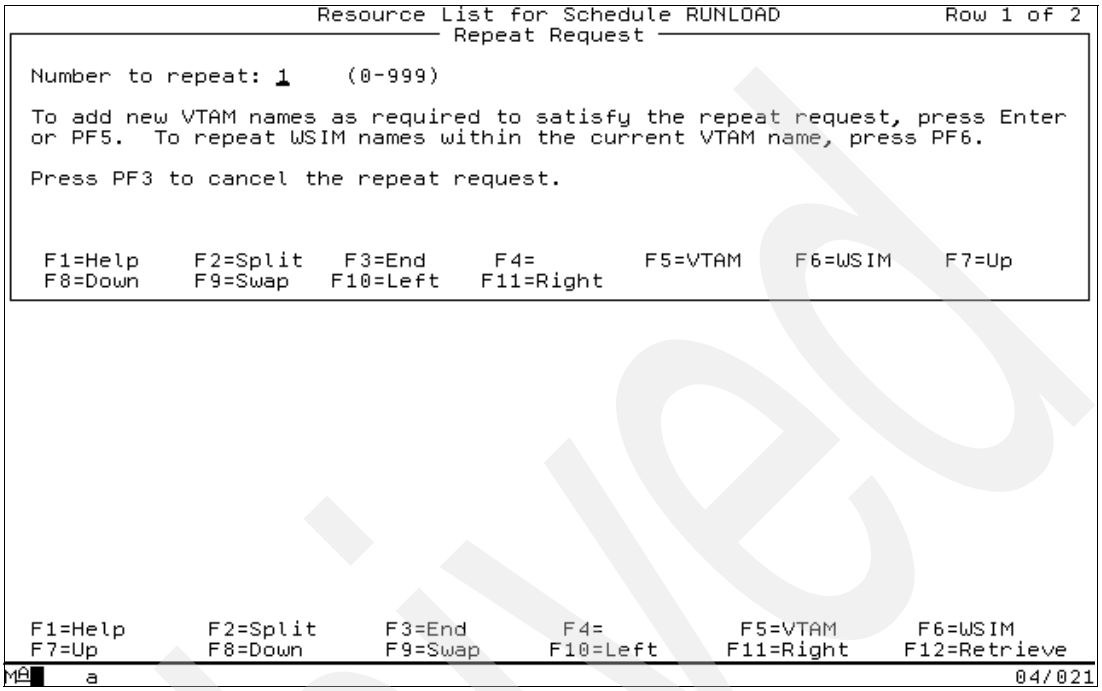


Figure 30-189 Adding new VTAM resource

7. The schedule RunLoad has two simulated network resources (terminals) defined now, as shown in Figure 30-190. These terminals run simultaneously.

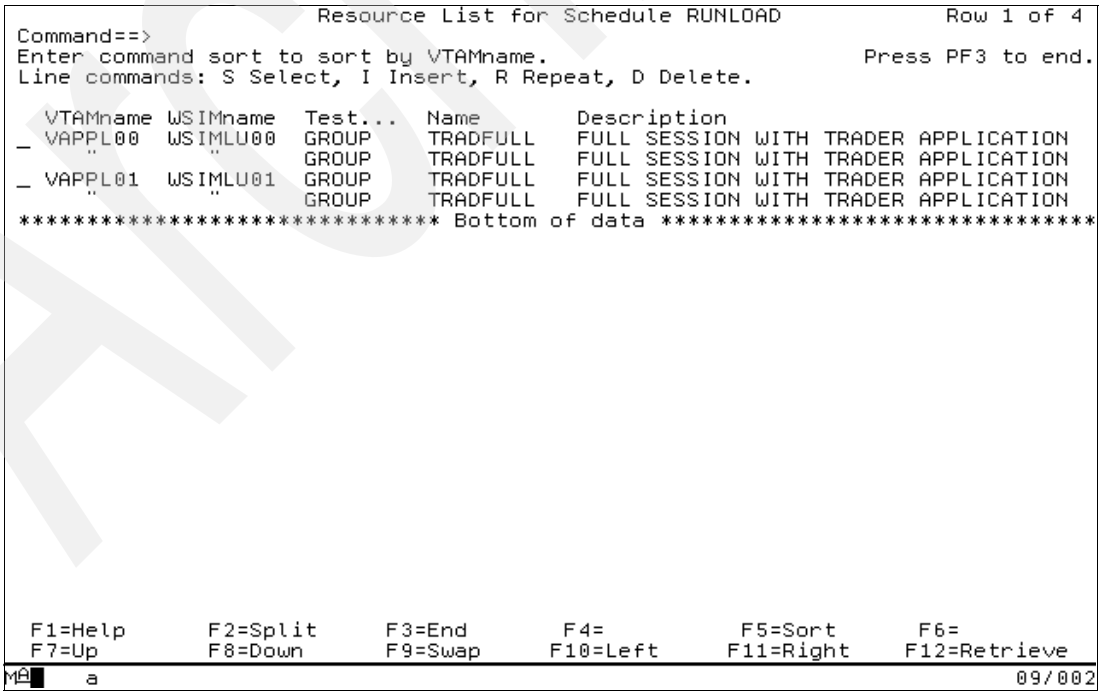


Figure 30-190 Two simulated resources (displays) defined for RunLoad

8. Press PF3. Since this is the first time, WTM proceeds to display the Completion Report thresholds panel, as shown in Figure 30-191.

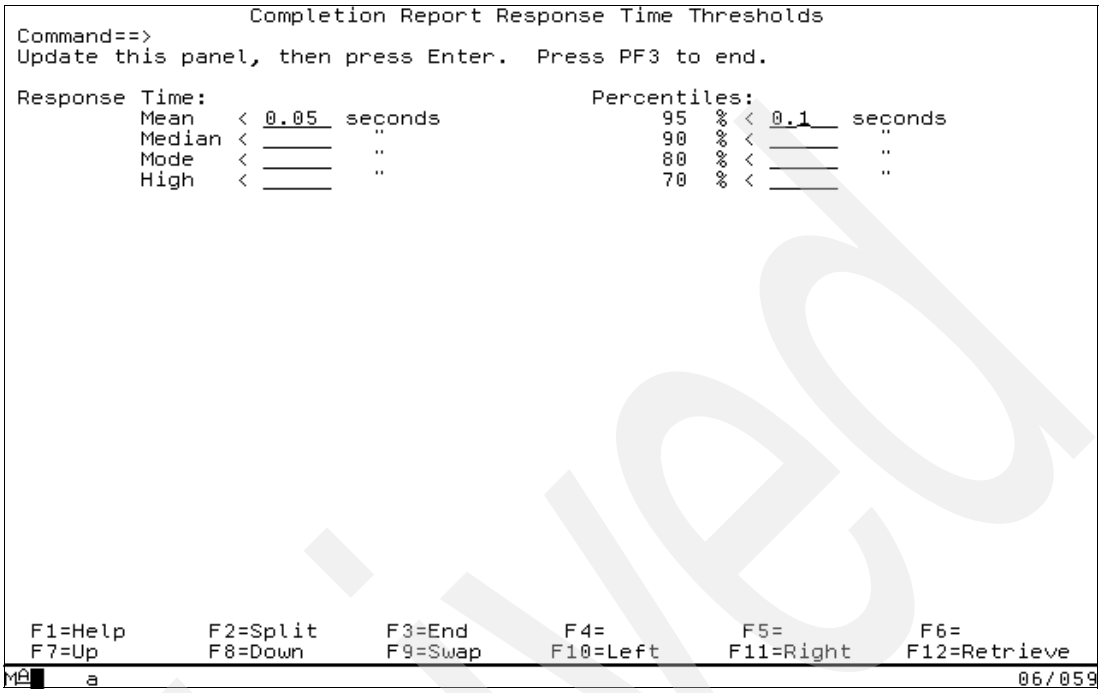


Figure 30-191 Time threshold defined

9. Change the threshold values if required, press Enter to save them, and press PF3 to return to the WTM Schedules panel. The new schedule RunFull is ready to be executed. To do some modifications to the script, enter the line command **W**, and press Enter, as shown in Figure 30-192.

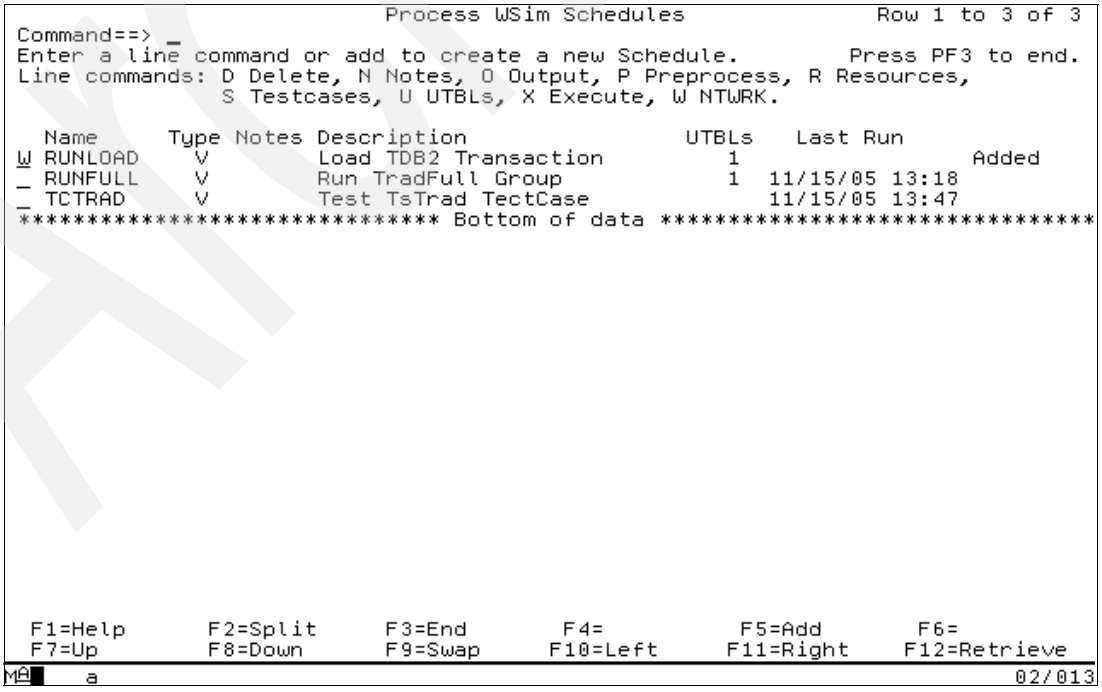


Figure 30-192 Work with network definition for RunLoad requested

10. The network definition for the schedule RUNLOAD opens for editing in an ISPF session. Scroll down to see the parameter UTI = 100 on line 19, as shown in Figure 30-193. This parameter defines the user time interval measured in 0.01 seconds. This is a delay between user messages (a *think time*). The value of 100 is equivalent to the *think time* of 1 second.

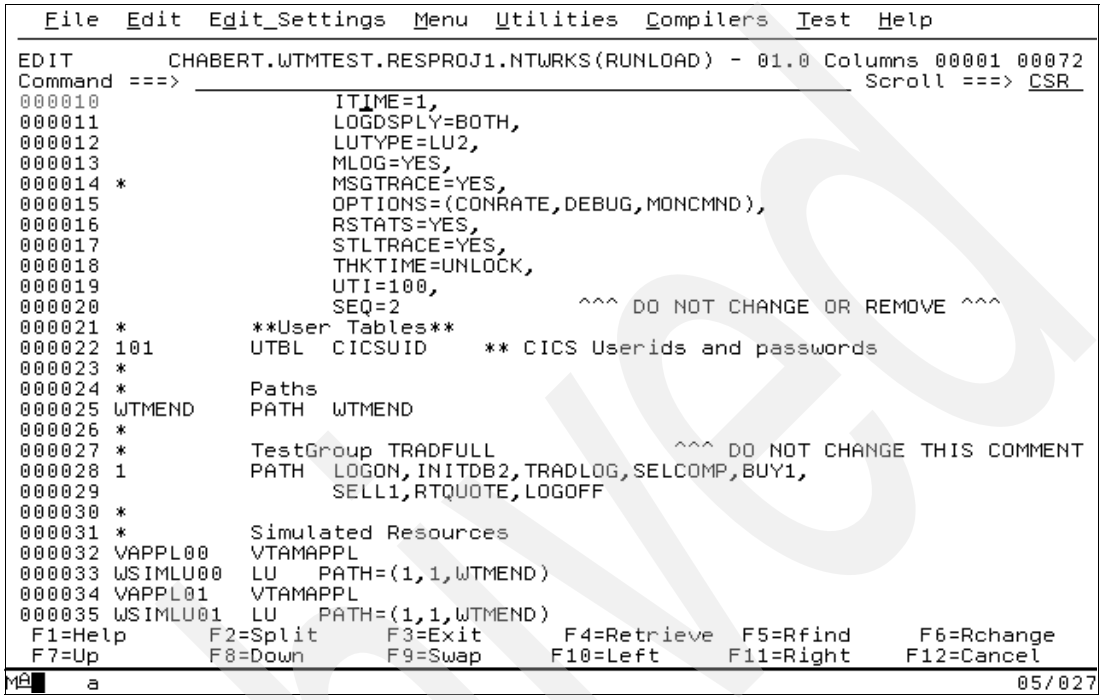


Figure 30-193 Network definition for RunLoad

11. If the value of the UTI parameter is changed, all the user messages are affected, and the intermessage interval is set to the new value but remains constant. We leave the value of the UTI to be equal to 100 (1 second), but define a random delay for each simulated interval, as shown in Figure 30-194.

    DELAY=R(n,m) results in a random intermessage delay in the range from n*UTI to m*UTI.

    The values entered result in the random delays from 5 to 50 seconds for the display WSIMLU00 and from 1 to 30 seconds for the display WSIMLU01. Press PF3 to exit the editing session and save the changes.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CHABERT.WTMTEST.RESPROJ1.NTWRKS(RUNLOAD) -          Invalid command
Command ===> _____________________________________________ Scroll ===> CSR
000012                 LUTYPE=LU2,
000013                 MLOG=YES,
000014 *               MSGTRACE=YES,
000015                 OPTIONS=(CONRATE,DEBUG,MONCMND),
000016                 RSTATS=YES,
000017                 STLTRACE=YES,
000018                 THKTIME=UNLOCK,
000019                 UTI=100,
000020                 SEQ=2              ^^^ DO NOT CHANGE OR REMOVE ^^^
000021 *          **User Tables**
000022 101        UTBL  CICSUID    ** CICS Userids and passwords
000023 *
000024 *          Paths
000025 WTMEND     PATH  WTMEND
000026 *
000027 *          TestGroup TRADFULL              ^^^ DO NOT CHANGE THIS COMMENT
000028 1          PATH  LOGON,INITDB2,TRADLOG,SELCOMP,BUY1,
000029                  SELL1,RTQUOTE,LOGOFF
000030 *
000031 *          Simulated Resources
000032 VAPPL00    VTAMAPPL
000033 WSIMLU00   LU   PATH=(1,1,WTMEND),delay=r(5,15)
000034 VAPPL01    VTAMAPPL
000035 WSIMLU01   LU   PATH=(1,1,WTMEND),delay=r(1,20)
****** ***************************** Bottom of Data ****************************

 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
MA    a                                                                04/015
```

Figure 30-194 Random delays for simulated resources added

12. It is a good idea to ensure that manual changes to a network definition did not introduce any errors, so enter the line command P to preprocess the schedule RunLoad. WTM processes the network definition statements, the STL programs, and the message generation decks (MSGTXTs) and report problems, if any, as shown in Figure 30-195.

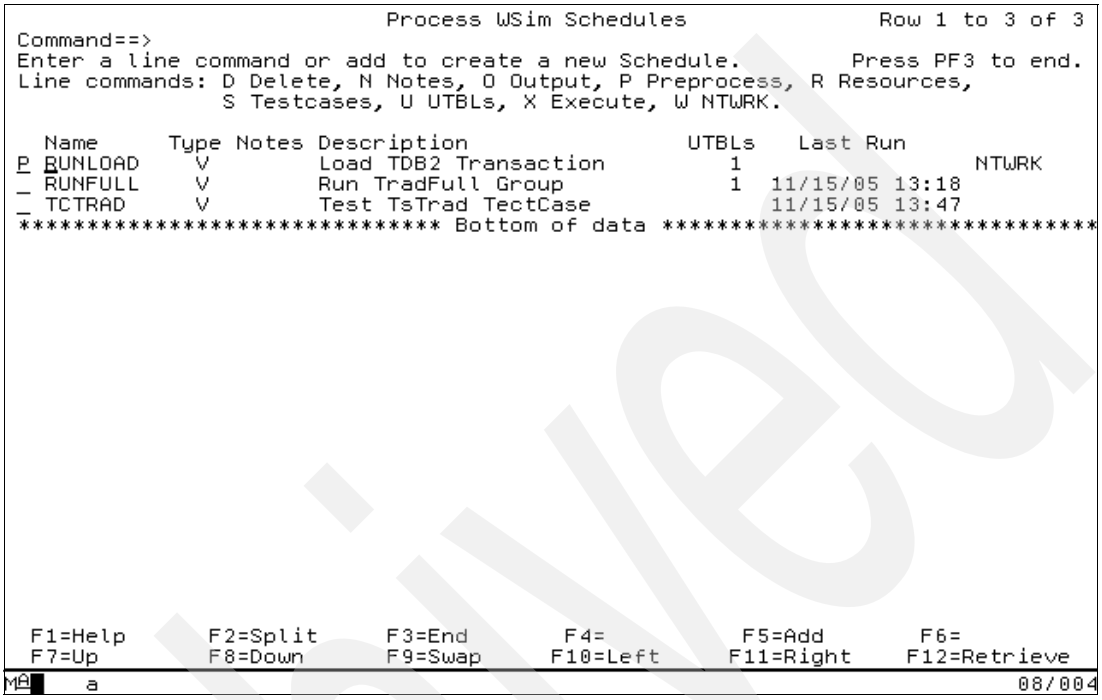


Figure 30-195 Preprocessing (no simulation run) for RunLoad requested

13. As shown in Figure 30-196, WTM did not find any coding problems, so the preprocess is successfully completed. You can browse the output by pressing PF6.

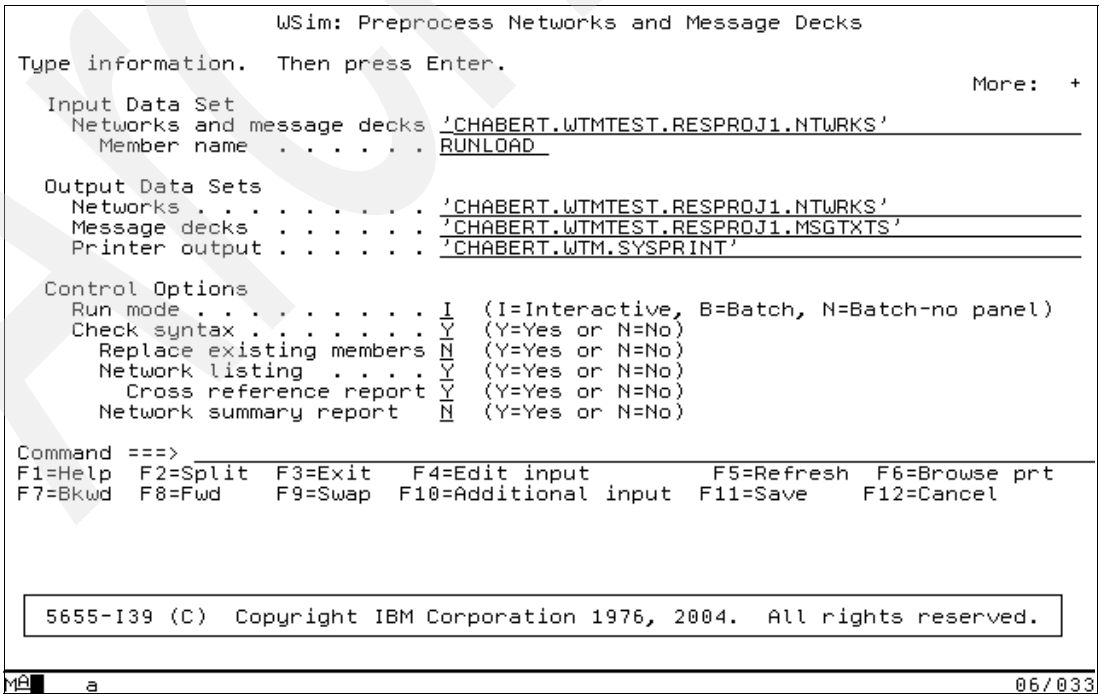


Figure 30-196 Processing for RunLoad is successful

14. Press PF3. On the next panel enter the line command **X** and press Enter to start the simulation, as shown in Figure 30-197.

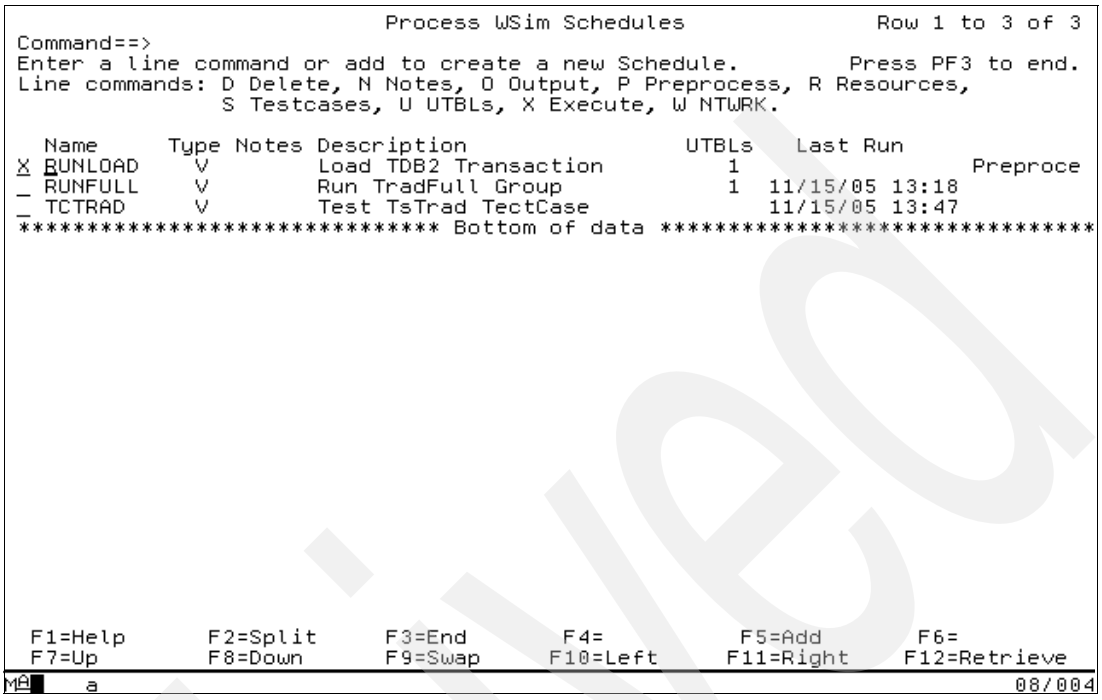


Figure 30-197 Simulation run for RunLoad requested

15. As usual, WTM presents an opportunity to change the names for the log data sets and make a choice as to whether to create the optional completion report, as shown in Figure 30-198. Make any required changes and press Enter.

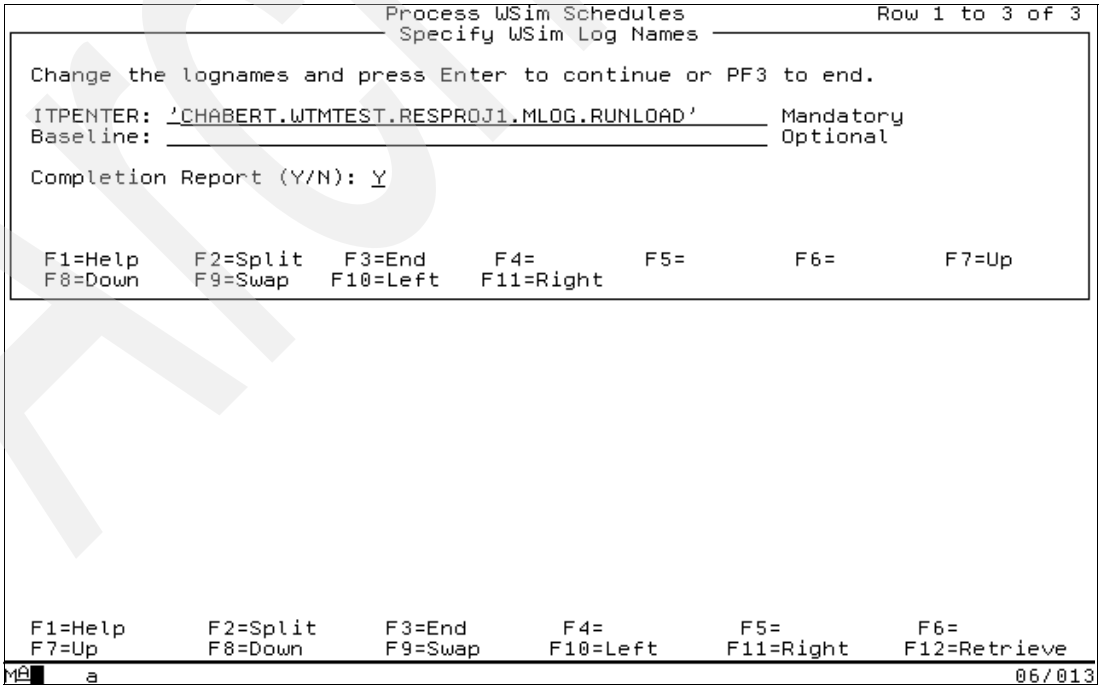


Figure 30-198 Log data set for RunLoad simulation run defined

16. WTM starts the simulation RunLoad. The WSim run messages are presented in two parts, shown in Figure 30-199.

```
ITP016I Workload Simulator (WSim) Version 1 Release 1.0.1, Feb. 10 2005 06:50
ITP003I WSim INITIALIZATION COMPLETE
ITP200I DISPLAY MONITOR FACILITY ACTIVE USING APPL WTMDM00
ITP029I INITIALIZATION COMPLETE FOR NETWORK RUNLOAD
ITP006I NETWORK RUNLOAD STARTED
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: LOGON Finished
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: LOGON Finished
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: INITDB2 Finished
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: TRADLOG Finished
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: INITDB2 Finished
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: SELCOMP Finished
ITP077I NETWORK RUNLOAD MESSAGE RATES 22 RECEIVED, 12 SENT
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: TRADLOG Finished
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: BUY1 Finished
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: SELCOMP Finished
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: SELL1 Finished
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: BUY1 Finished
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: RTQUOTE Finished
ITP077I NETWORK RUNLOAD MESSAGE RATES 10 RECEIVED, 10 SENT
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: LOGOFF Finished
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: SELL1 Finished
ITP137I RUNLOAD  WSIMLU00-00001 - Attempt to read beyond end of CICSUID table
ITP137I RUNLOAD  WSIMLU00-00001 - WTMEND: 1 out of 2 finished.
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: RTQUOTE Finished
ITP077I NETWORK RUNLOAD MESSAGE RATES 17 RECEIVED, 10 SENT
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: LOGOFF Finished
ITP137I RUNLOAD  WSIMLU01-00001 - Attempt to read beyond end of CICSUID table
ITP137I RUNLOAD  WSIMLU01-00001 - WTMEND: 2 out of 2 finished.
ITP137I RUNLOAD  WSIMLU01-00001 - WTMEND: All scripts run.  Issuing ZEND.
ITP002I RUNLOAD  WSIMLU01-00001 - ZEND
ITP201I DISPLAY MONITOR FACILITY IS CLOSED DOWN
*** _
MA    a                                                          32/006
```

Figure 30-199 RunLoad simulation run messages

17. The simulated terminals WSIMLU00 and WSIMLU01 were running concurrently. The message `Attempt to read beyond end of CICSUID table` is caused by the fact that the UTBL CICSUID access was set to be the single sequential, as shown in Figure 30-134 on page 1033.

    The optional completion report for this simulation is shown in Figure 30-200.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
EDIT       CHABERT.WTMTEST.RESPROJ1.CREP(R14) - 01.00       Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ********************************
000001 Report for RUNLOAD - Load TDB2 Transaction.
000002 Run completed at 14:54:42 on 15 Nov 2005.
000003 This run successfully met the response time targets:
000004   95%_Percentile actual (0.00)  target(0.1)
000005   Mean actual (0.00)  target(0.05)
000006 Loglist Extract:
000007 RUN  TIME 14.45.41,  NOVEMBER 15, 2005  VERSION  1  RELEASE  1.0.1
000008 14460967 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: LOGON Finished
000009 14461767 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: LOGON Finished
000010 14461867 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: INITDB2 Finished
000011 14462968 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: TRADLOG Finished
000012 14463768 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: INITDB2 Finished
000013 14463969 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: SELCOMP Finished
000014 14464169 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: TRADLOG Finished
000015 14465372 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: BUY1 Finished
000016 14465570 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: SELCOMP Finished
000017 14471475 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: SELL1 Finished
000018 14472673 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: BUY1 Finished
000019 14473477 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: RTQUOTE Finished
000020 14475878 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: LOGOFF Finished
000021 14480275 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: SELL1 Finished
000022 14480978 ITP137I RUNLOAD  WSIMLU00-00001 - Attempt to read beyond end of
000023 14483576 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: RTQUOTE Finished
000024 14484777 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: LOGOFF Finished
000025 14485677 ITP137I RUNLOAD  WSIMLU01-00001 - Attempt to read beyond end of
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
MA    a                                                              04/015
```

Figure 30-200 Completion report for RunLoad simulation

18. Press PF3 to return to the WTM Schedules panel and enter the line command **o** to work with output reports, as shown in Figure 30-201.

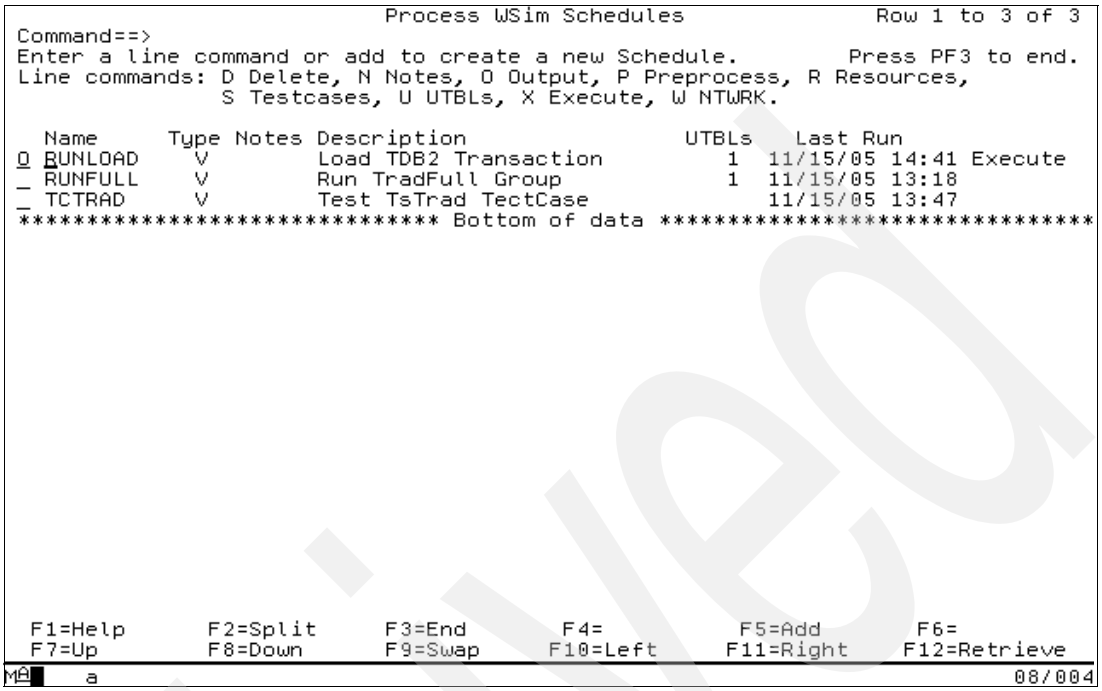


Figure 30-201 Work with RunLoad simulation run output requested

19. On the next panel, Reports for schedule RunLoad, select option **2** to view the response time report, as shown in Figure 30-202.

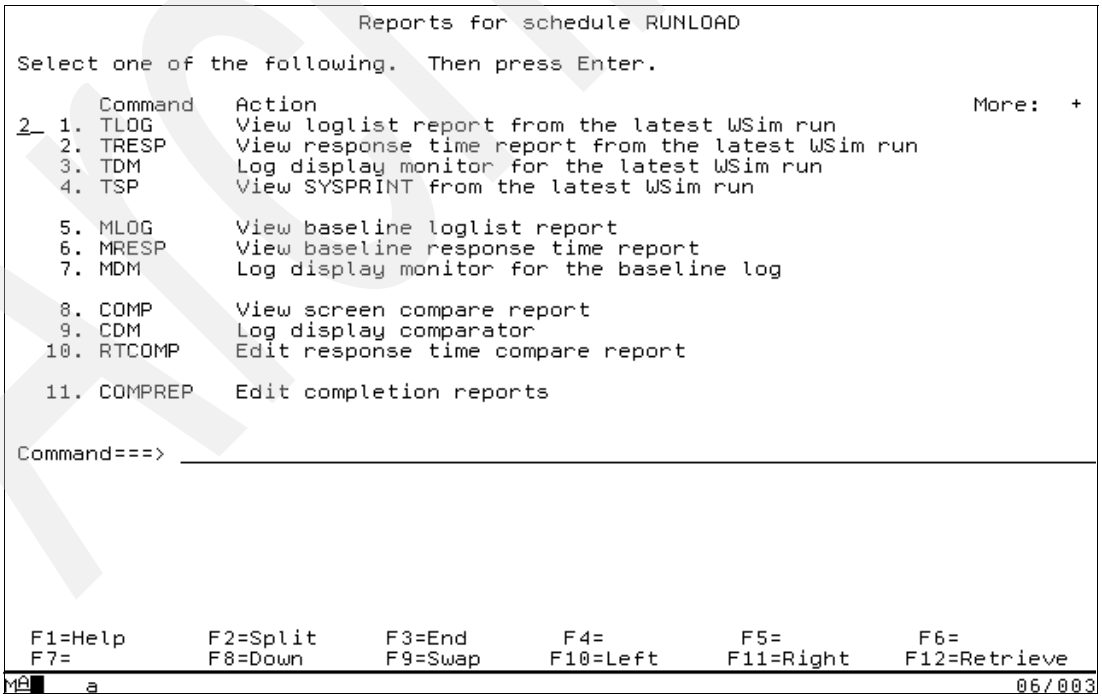


Figure 30-202 Response time report for RunLoad selected

20.A fragment of the WSim time response report is shown in Figure 30-203.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CHABERT.WTM.SYSPRINT                          Columns 00002 00073
Command ===> _                                            Scroll ===> CSR
000115 WSim RESPONSE TIME REPORT
000116 RUN  TIME 14.45.41,  NOVEMBER 15, 2005  VERSION  1  RELEASE  1.0.1
000117 ------------------------------------------------------------------------
000118 SUMMARY  REPORT    NETWORK  ALL NETWORKS      PROCESS  SYSTEM      TIME
000119                                               EXIT                 START
000120                                               TERMTYPE             END T
000121 ------------------------------------------------------------------------
000122 RESPONSE TIME    COUNT   RESPONSE TIME    COUNT   RESPONSE TIME    COUNT
000123          0.00       29            0.10       1
000124 MEAN   RESPONSE              0.00    MESSAGES SENT            32    NUMBE
000125 MEDIAN RESPONSE              0.10       AVERAGE LENGTH        24      PE
000126 MODE   RESPONSE              0.00       PER MINUTE            10    RESPO
000127 LOW    RESPONSE              0.00    MESSAGES RECEIVED        36    VARIA
000128 HIGH   RESPONSE              0.10       AVERAGE LENGTH       311    95 PE
000129 AVERAGE QUEUE TIME           0.00       PER MINUTE            11
000130 PERCENTILE    RESPONSE TIME         AVERAGE
000131      10                   0.00           0.00
000132      20                   0.00           0.00
000133      30                   0.00           0.00
000134      40                   0.00           0.00
000135      50                   0.00           0.00
000136      60                   0.00           0.00
000137      70                   0.00           0.00
000138      80                   0.00           0.00
000139      90                   0.00           0.00
000140      95                   0.00           0.00
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
MA    a                                                            04/015
```

Figure 30-203 Response time report fragment

30.13 Using SNA trace to create test case

In this section we briefly discuss what should be done to create a test case using option 2 on the WTM Add New Test case panel, shown in Figure 30-9 on page 959.

The Generalized Trace Facility (GTF) is a service aid that can be used to record and diagnose system and program problems. GTF is a part of the MVS system product. Using GTF, a user can record a variety of system and program events on all of the processors in the installation.

The events that GTF traces are specified as options in a PARMLIB member, as shown in Example 30-2.

Details of GTF operation, which include storage required, where output goes, and recovery for GTF, are defined in a cataloged procedure in SYS1.PROCLIB. GTF can trace system and program events both above and below 16 megabytes. For each event it traces, GTF produces trace records as its output. Writing trace data into data sets impacts system performance.

When using GTF to create SNA traces, the following options are required:

RNIO Requests recording of all Virtual Telecommunications Access Method (VTAM) network activity

USRP Requests recording of all data that the GTRACE macro passes to GTF

The option USRP requires a list of specific event identifiers (EIDs), which can represent users, program products, or IBM subsystem and component events. Here is a list of assigned EIDs related to VTAM:

FE2 TSO/VTAM TGET/TPUT trace
FE3 VTAM reserved
FF1 VTAM buffer contents trace (USER)
FF0 VTAM SMS (buffer use) trace
FE1 VTAM internal table
FEF VTAM buffer contents trace

As a result, the following must be present in the PARMLIB member to specify what data is be recorded for the WSim VTAM trace, as shown in Example 30-2.

Example 30-2 SYS1.PARMLIB(GTFWSIM) member

```
TRACE=RNIO,USRP
USR=(FE2,FE3,FF1,FF0,FE1,FEF)
END
```

Example 30-3 represents a fragment of a PARMLIB proc member that can be used to start the GTF traces with parameters defined in the GTFWSIM member, shown in Example 30-3.

Example 30-3 SYS1.PROCLIB(GTFPROC) member fragment

```
//GTFPROC PROC MEMBER=GTFWSIM
//        EXEC PGM=AHLGTF,PARM='MODE=EXT,DEBUG=NO,TIME=YES'
//        TIME=1440,REGION=6M
//SYSLIB  DD DSNAME=SYS1.PARMLIB(&PARMLIB),DISP=SHR
//IEFRDER DD DSNAME=xxxxxx,DISP=(NEW,CATLG),
// ...
```

The parameter MODE=EXT forces GTF to direct the output to a trace data set defined by GTFOUTXX or IEFRDER DD statements.

The parameter DEBUG=NO specifies whether GTF attempts recovery after encountering an error. If DEBUG=YES, GTF does not attempt any recovery. Instead, GTF issues an error message and end after encountering any error so that the contents of the trace table immediately prior to the error remain intact. If DEBUG=NO, which is the default, GTF does the following operations:

- For errors in GTF processing, GTF continues processing after doing one or more of the following tasks:
  - Flagging the trace record or trace record field associated with the error
  - Issuing a message to the console to notify the operator that an error occurred
  - Suppressing the error or function in which the error occurred
- For errors that do not occur in GTF itself, GTF ends abnormally. If GTF stops processing, that does not cause any other task to also stop.

The parameter TIME=YES specifies that every GTF trace record has a time stamp, as well as the block time stamp associated with every block of data. The time stamp is the eight-byte time of day (TOD) clock value at the local time in which GTF puts the record into the trace table. GTF does not accept TIME=NO. All output records have time stamps.

We assume that the VTAM startup procedure name is NET and that the user has all required authorizations.

To start the GTF trace, on the SDSF Primary Option Menu issue the LOG command. On the next screen, issue the following:

```
/F NET,TRACE,TYPE=BUF,ID=vtamappl,AMOUNT=FULL
```

This informs the VTAM (NET application) that the user wants to collect information. FULL trace is mandatory for the trace output to be used by the Workload Simulator, for the subsystem with the APPLID VTAMAPPL.

The next command is:

```
/S GTFPROC.USER
```

It starts the GTF trace using the proc presented in Example 30-3 on page 1076 with parameters as defined in Example 30-2 on page 1076. USER is a name to manage this specific trace.

The users can now log on to application VTAMAPPL and run transactions and so forth, and log off from the application VTAMAPPL. It is very important that the full process be traced. If the traced terminals are connected before the trace started, the Workload Simulator cannot use this trace.

To stop the GTF trace, issue the command:

```
/P USER
```

When adding test cases using option 2 on the WTM Add New Test case panel, you must specify which GTF trace data set is to be used and which WSim log data set is produced.

WTM displasy all real network resources (displays) used during the trace, so you must make a selection as to which one is going to be used for the current test case.

31

Basics of scripting for Workload Simulator

In this chapter we discuss the basics of scripting for Workload Simulator (WSim).

First, we review the scripts that were generated by Workload Simulator Test Manager during the activities described in the previous chapter. Next we review statements used for network definitions and basic concepts of coding message generation decks in Structured Translator Language (STL). Last, we present operator commands that can be used to communicate with WSim.

31.1 WTM-generated scripts review

Every WSim script has two components:

- Network definition statements describing the configuration of the network being simulated
- Message generation decks that send and receive messages

In this section we review some of the scripts that were created by WTM to illustrate what was being done behind the curtain. The network definition files, created by WTM when the simulations were being prepared (described in the previous chapter) are presented in Figure 31-1.

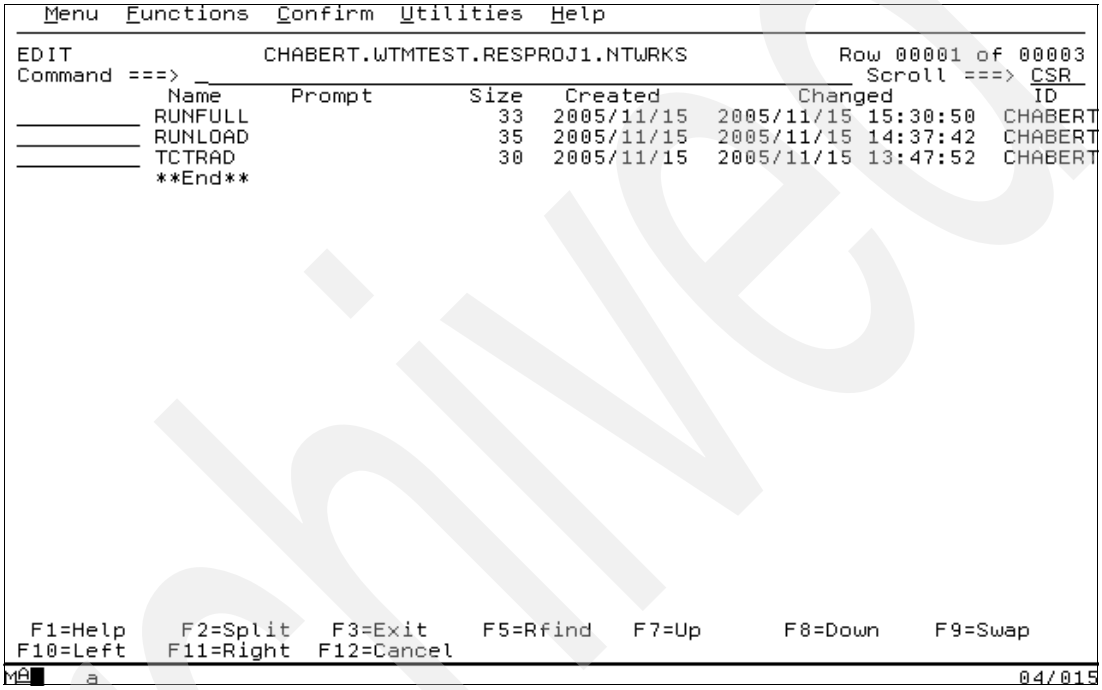


Figure 31-1 List of network definitions generated by WTM

The network definition for the simulation RunFul is presented in Figure 31-2 and Figure 31-3. Definitions and significance of the entries are presented in the following figures.

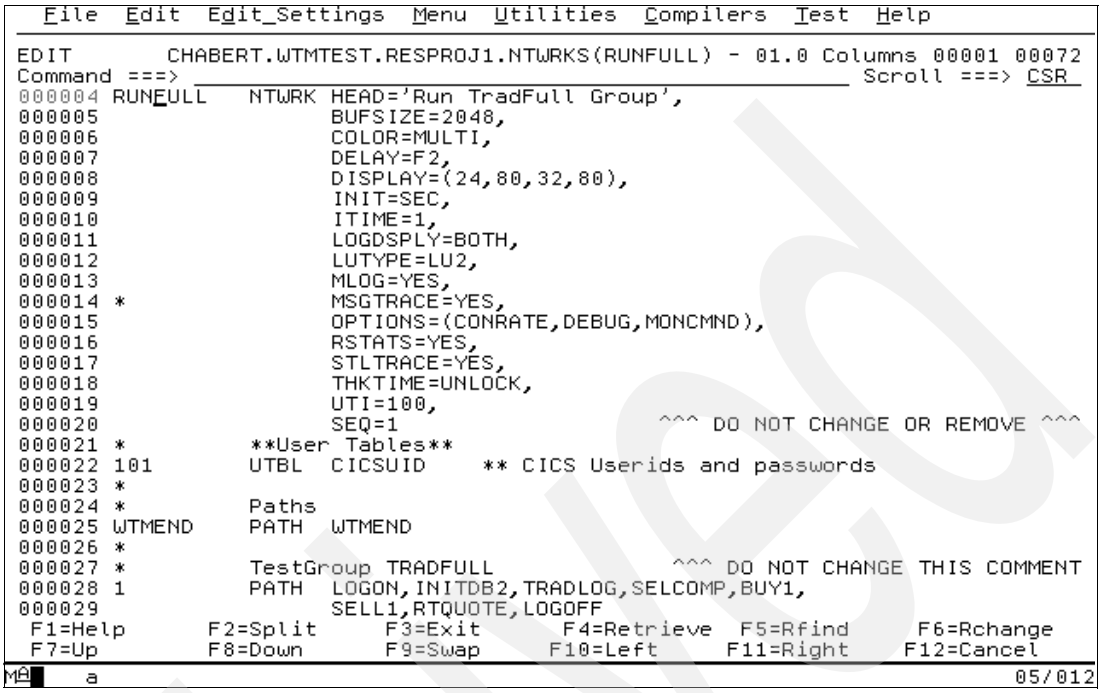


Figure 31-2 Network definition RunFull (1 of 2)

The NTWRK statement is always in the first position in the network definition. This network is named RUNFULL. The parameters include these:

HEAD A comment to be shown on reports.

COLOR Multiple color support.

DELAY Delay time, in this example set to two UTIs.

DISPLAY Primary and alternate screen sizes.

INIT The secondary LU (terminal) initiates the session.

ITIME Time (in minutes) between interval reports.

LOGDSPLY Write display buffer to log both before and after message generation.

LUTYPE 3270 type terminal.

OPTIONS Options for this network are:

- **CONRATE** Print interval reports at the WSim console.
- **DEBUG** Write trace data for TCP and CPI-C.
- **MONCMND** Show console commands initiated in a MSGTXT on the WSim console.

RSTATS Response time statistics to be accumulated for LUs

STLTRACE STL trace records to be written to the log.

THKTIME When *think time* delay interval begins (in this case when the keyboard is unlocked).

UTI Network-level user time interval in 0.01 seconds.

The UTBL statement specifies that the user table CICSUID is to be used in at least one of the corresponding STL programs. It references the member CICSUID in the partitioned data set (PDS) CHABERT.WTMTEST.RESPROJ1.MSGTXTS. The name of 101 was assigned by WSim to this UTBL.

The first PATH statement, named WTMEND, defines the MSGTXT WTMEND that was automatically generated and inserted by WTM to issue the **zend** WSim console command to end the simulation run (Figure 31-3).

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CHABERT.WTMTEST.RESPROJ1.NTWRKS(RUNFULL) - 01.0 Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
000023 * _
000024 *         Paths
000025 WTMEND    PATH  WTMEND
000026 *
000027 *         TestGroup TRADFULL              ^^^ DO NOT CHANGE THIS COMMENT
000028 1         PATH  LOGON,INITDB2,TRADLOG,SELCOMP,BUY1,
000029                 SELL1,RTQUOTE,LOGOFF
000030 *
000031 *         Simulated Resources
000032 VAPPL00   VTAMAPPL
000033 WSIMLU00  LU   PATH=(1,1,WTMEND)
****** **************************** Bottom of Data ****************************
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
MA    a                                                              05/011
```

Figure 31-3 Network definition RunFull (2 of 2)

Summary:

- The second PATH statement, named 1, defines the list of MSGTXTs and the order in which they are executed.
- The VTMAPPL statement must precede any LU statements and defines that the following LUs simulate VTAM resources.
- The LU statement defines a single network resource and the script program path to be executed by it, in this case the path 1 first and then the path WTMEND.

The set of STL programs, which includes the STL programs for each of the test cases created by WTM, is shown in Figure 31-4.

```
  Menu  Functions  Confirm  Utilities  Help
 ------------------------------------------------------------------------------
EDIT              CHABERT.WTMTEST.RESPROJ1.STL               Row 00001 of 00014
Command ===> _____________________________________________ Scroll ===> CSR
          Name     Prompt        Size   Created          Changed          ID
_________ BUY1                    44   2005/11/14  2005/11/14 18:38:43  CHABERT
_________ CICSUID                  9   2005/11/15  2005/11/15 11:31:22  CHABERT
_________ CICSUIDX
_________ CPICCON                103   2005/11/11  2005/11/11 16:54:12  CHABERT
_________ CPICVARA                49   2005/11/11  2005/11/11 16:54:12  CHABERT
_________ INITDB2                 40   2005/11/14  2005/11/14 16:17:43  CHABERT
_________ LOGOFF                  41   2005/11/14  2005/11/14 18:50:23  CHABERT
_________ LOGON                   58   2005/11/14  2005/11/15 11:16:56  CHABERT
_________ RTQUOTE                 40   2005/11/14  2005/11/14 18:43:54  CHABERT
_________ SELCOMP                 35   2005/11/14  2005/11/14 18:25:33  CHABERT
_________ SELL1                   44   2005/11/14  2005/11/14 18:32:38  CHABERT
_________ TRADLOG                 41   2005/11/14  2005/11/14 18:19:26  CHABERT
_________ TSTRAD                 159   2005/11/14  2005/11/14 13:42:51  CHABERT
_________ WTMVARS                543   2005/11/11  2005/11/15 11:13:16  CHABERT
          **End**
 F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
F10=Left   F11=Right  F12=Cancel
MA    a                                                              04/015
```

Figure 31-4 List of STL programs

We continue with the review of the WTM-generated STL programs using the STL program for the test case LOGON as an example, as shown in Figure 31-5.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
EDIT       CHABERT.WTMTEST.RESPROJ1.STL(LOGON) - 01.03     Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ********************************
000001 àprogram=LOGONT
000002 àinclude wtmvars
000003 LOGON: msgtxt
000004 /*-------------------------------------------------------------------------*/
000005 /* ITPIDC: DISPLAY=TSOCON   APPLICATION=CICSC23G 15:52:12.45 11/14/05*/
000006 /*   ----------- DISPLAY CHARACTERISTICS AND FEATURES ------------   */
000007 /*   ALTCSET=APL                APLCSID=(963,310)                    */
000008 /*   BASECSID=(695,1147)        CCSIZE=(8,12)     COLOR=MULTI        */
000009 /*   DBCS=NO                                                         */
000010 /*   DISPLAY=(24,80,32,80)      DLOGMOD=LSX32703  EXTFUN=YES         */
000011 /*   FLDOUTLN=NO                FLDVALID=NO       HIGHLITE=YES       */
000012 /*   MAXNOPTN=0                 PS=NONE           UOM=INCH           */
000013 /*-------------------------------------------------------------------------*/
000014 /* ITPLSGEN: SCRIPT GENERATION PARAMETERS        15:53:00.58 11/14/05*/
000015 /*   INPUT    CHABERT.WTMUSER.IDCLOG(LOGON)                        */
000016 /*   OUTPUT   CHABERT.WTMTEST.RESPROJ1.STL                         */
000017 /*   MSGTXT   LOGON                                                */
000018 /*   NODELAY                                                       */
000019 /*   GENERATE CHANGED                                              */
000020 /*   LU       IDCSLU-1                                             */
000021 /*   STL      TRACE=LOGONT                                         */
000022 /*   NOVERIFY                                                      */
000023 /*-------------------------------------------------------------------------*/
000024
000025 /*------------------------------------------------- 15521245 00001 */
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
MA    a                                                              04/015
```

Figure 31-5 STL program for test case Logon (1 of 3)

Summary:

- The statement @PROGRAM specifies the start of the STL program and defines its name, in this case LOGONT.
- The statement @INCLUDE retrieves a member from the SYSLIB concatenation as program statements. The member WTMVARS is a standard include member for WTM.
- The statement MSGTXT defines the name of the MSGTXT member to be generated from this STL program, in this case LOGON.
- The statement ONIN, as shown in Figure 31-6, is an asynchronous statement.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CHABERT.WTMTEST.RESPROJ1.STL(LOGON) - 01.03     Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
000024
000025 /*------------------------------------------------- 15521245 00001 */
000026 onin0001: onin substr(ru,1,1) = 'F5'x,
000027                 then found = on
000028 found = off
000029 initself('CICSC23G','LSX32703')
000030 do while found = off              /* wait for onin0001 data received */
000031  wait until onin
000032 end
000033 deact onin0001
000034
000035 /* 15:52:12.47 ITP1507I SESSION STARTED WITH APPLICATION CICSC23G */
000036
000037 /*------------------------------------------------- 15522001 00001 */
000038 WTM_panel_ID = 'PNL00001'
000039 log 'WTM_panel_ID' WTM_panel_ID
000040 cursor(10,26)
000041 ereof
000042 charset 'field'
000043 /* WTM has replaced the following line with a user table: */
000044 /* type 'CHABERT' */
000045 call CICSUIDX     /* Access CICSUID usertable */          /* WTM */
000046 type CICSUID_USERID                                       /* WTM */
000047 cursor(11,26)
000048 /* WTM has replaced the following line with a user table: */
000049 /* upnd = 'ACDE436701F74B6F900C81093F0B97DD267CFCB536'x */
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
MA    a                                                          04/015
```

Figure 31-6 STL program for test case LOGON (2 of 3)

Summary:

- The value of the variable FOUND is set to ON when the first character of the variable RU is equal to the hexadecimal 'F5'. The variable RU is a special variable that contains the SNA request/response unit portion of incoming or outgoing data.
- The value of the variable FOUND is first set to OFF.
- The statement INITSELF starts a session with the VTAM application with APPLID CICSC22F from this LU. User data, which equals LSX32702, is also sent to that VTAM application.
- The statement DO WHILE loops executing the statement WAIT UNTIL ONIN until the session gets initialized and a response is received.
- The statement DEACT ONIN0001 deactivates the asynchronous ONIN statement labeled ONIN0001.

The statement LOG writes a record, containing the name that WTM assigned to the panel, to the log data set, as shown in Figure 31-7.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CHABERT.WTMTEST.RESPROJ1.STL(LOGON) - 01.03     Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
000038 WTM_panel_ID = 'PNL00001'
000039 log 'WTM_panel_ID' WTM_panel_ID
000040 cursor(10,26)
000041 ereof
000042 charset 'field'
000043 /* WTM has replaced the following line with a user table: */
000044 /* type 'CHABERT' */
000045 call CICSUIDX     /* Access CICSUID usertable */          /* WTM */
000046 type CICSUID_USERID                                       /* WTM */
000047 cursor(11,26)
000048 /* WTM has replaced the following line with a user table: */
000049 /* upnd = 'ACDE436701F74B6F900C81093F0B97DD267CFCB536'x */
000050 upnd = x2c(CICSUID_PASSWORD)                              /* WTM */
000051 userexit('ITPUMNDX',upnd)
000052 cursor(12,26)
000053 transmit using enter
000054
000055 /* 15:52:43.58 ITP1508I SESSION ENDED WITH APPLICATION CICSC23G */
000056
000057 say 'WTM:' msgtxtid() 'Finished'
000058 endtxt
****** **************************** Bottom of Data ****************************



 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
MA    a                                                                05/018
```

Figure 31-7 STL program for test case LOGON (3 of 3)

Summary:

- The cursor is moved to the position specified by row and column on the screen (10,26).
- The statement EREOF erases from the cursor current position to the end of the field.
- The next statement defines that that character set is determined by field attributes (this is the default).
- The CALL statement executes the MSGTXT, which was generated by WTM to access the UTBL CICSUID.
- The statement TYPE places data into the screen image. In this case, it is the data from the column CICSUID_USERID in the UTBL CICSUID.
- These steps are repeated for the password.
- TRANSMIT the data and simulate pressing Enter.
- The statement SAY writes a line to the WSim console.
- This is the end of the STL program generated by WTM for the test case LOGON.

31.2 Network definition statements

Network definition statements specify the following information:

- Type of the simulated resources in the network
- Attributes of the simulated resources in the network
- Connections between the simulated resources and the system under test
- Special information about delays, logic tests, the order in which message generation decks are used, logging or tracing of the messages, and so forth

Network definition statements consist of the name (optional except on the NTWRK statement), the statement, and one or more operands (optional), as shown in Example 31-1.

Example 31-1 NTWRK statement

```
RESNET1 NTWRK REPORT=RATE,HEAD=' RESNET1 RATE REPORT '
```

The network named RESNET1 is defined, the interval report include only the network totals and message rates, and the heading of the report is `RESNET1 RATE REPORT`.

NTWRK is always the first statement used to define a network. It names the network and specifies characteristics that apply to the network as a whole. It also can specify operands that establish defaults for lower-level statements. All other statements in the network definition follow the NTWRK statement in a prescribed order. The statements from the general simulation statements group immediately follow the NTWRK statement.

Different statements are used depending on what type of a network is being simulated. For example, when simulating LUs accessing a VTAM application, the VTAMAPPL and LU statements must be used; when simulating CPI-C TPs, APPCLU and TP statements; when simulating TCP-IP clients, TCP/IP and DEV statements.

Not all statements are mandatory and some could be coded more than once, but all the statements in each and every group, including the optional ones, should follow the prescribed order.

VTAMAPPL statement

Use the VTAMAPPL statement to define a VTAM application program.

Use either the name field or the APPLID operand to specify the symbolic name of the application. This name must match the corresponding entry in VTAM's configuration tables (VTAMLST), created using an APPL definition statement in VTAM.

Use the PASSWD operand to specify the password associated with the symbolic name of the VTAM application. The value specified must match the value of the PRTCT operand on the APPL statement in VTAM.

LU statement

Use the LU statement to define one or more LU half-sessions and specify the type of half-session to be simulated. WSim simulates each LU half-session as a separate entity (single display, terminal, device) for message generation and message logging. Operands on the LU statement can be used to define half-session characteristics required to generate and interpret data streams associated with corresponding products, for example, 3270 devices.

At least one LU statement must follow a VTAMAPPL statement, as shown in Example 31-2.

Example 31-2 VTAMAPPL and LU statements

```
RESVTAM1 NTWRK INIT=SEC
WSIMLU VTAMAPPL
USERRES1 LU LUTYPE=LU2,RESOURCE=TSO
```

APPCLU statement

Use the APPCLU statement to define CPI-C LU.

Use either the name field or the APPLID operand to specify the symbolic name of the LU. This name must match the corresponding entry in VTAM's configuration tables (VTAMLST), which was created using an APPL definition statement in VTAM, and this corresponding entry must specify APPC=YES.

Use the PASSWD operand to specify the password associated with the symbolic name of the VTAM application. The value specified must match the value of the PRTCT operand on the APPL statement in VTAM.

TP statement

Use the TP statement to define one or more transaction programs and to specify whether this TP is a client or a server. WSim simulates each TP as a separate entity for message generation and message logging.

At least one TP statement must follow an APPCLU statement, as shown in Example 31-3.

Example 31-3 APPCLU and TP statements

```
RESCPIC1 NTWRK
LU1 APPCLU
TP1 TP TPTYPE=CLIENT
```

TCPIP statement

Use the TCPIP statement to define a connection to the IBM TCP/IP for MVS product on the local host. Use the TCPNAME operand to specify the name of the TCP/IP virtual machine or address space.

DEV statement

Use the DEV statement to define a TCP/IP client to be simulated.

Use the TYPE operand to specify the type of client. Multiple clients of different types can be specified for a single TCP/IP statement. Use the SERVADDR operand to specify a server to which the client can connect. Use the PORT operand to specify a port to be used for connection.

At least one DEV statement must follow a TCPIP statement, as shown in Example 31-4.

Example 31-4 TCPIP and DEV statements

```
RESTCPIP NTWRK
RESCONN TCPIP
RES1 DEV TYPE=TN3270,SERvaDDR=9.255.255.255,PORT=23
RES2 DEV TYPE=FTP,SERvaDDR=9.255.255.254
```

General order of network definition statements

The order of valid network definition statements is listed in Table 31-1.

Table 31-1 Order of valid network definition statements

Statement	Can be followed by	Comments
NTWRK	NTWRKLOG, MSGDISK, RATE, UTBL, SIDEINFO, RN, UDIST, INCLUDE, IF, PATH	Required and must be the first statement of a network
NTWRKLOG	MSGDISK, RATE, UTBL, SIDEINFO, RN, UDIST, INCLUDE, IF, PATH	Optional
MSGDISK	RATE, UTBL, SIDEINFO, RN, UDIST, INCLUDE, IF, PATH	Optional
RATE	RATE, UTBL, SIDEINFO, RN, UDIST, INCLUDE, IF, PATH	Required if using rate table delays
UTBL	UTBL, SIDEINFO, RN, UDIST, INCLUDE, IF, PATH	Optional
SIDEINFO	SIDEENT	Optional
SIDEENT	SIDEENT, SIDEEND	At least one is required after SIDEINFO
SIDEEND	RN, UDIST, INCLUDE, IF, PATH	Required after the last SIDEENT
RN	RN, UDIST, INCLUDE, IF, PATH	Optional
UDIST	UDIST, INCLUDE, IF, PATH	Optional
INCLUDE	INCLUDE, IF, PATH	Optional, mostly used to define required but not referenced message generation decks
IF	INCLUDE, IF, PATH	Optional
PATH	PATH, DIST, UTI, FILE, APPCLU, VTAMAPPL, TCPIP	At least one is required
DIST	DIST, UTI, FILE, APPCLU, VTAMAPPL, TCPIP	Optional
UTI	UTI, FILE, APPCLU, VTAMAPPL, TCPIP	Optional
FILE	FILE, APPCLU, VTAMAPPL, TCPIP	Optional
APPCLU	TP	Optional
TP	TP, APPCLU, VTAMAPPL, TCPIP, FE	At least one is required after APPCLU
VTAMAPPL	LU	Optional
LU	LU, VTAMAPPL, TCPIP, FE	At least one is required after VTAMAPPL
TCPIP	DEV	Optional
DEV	DEV, TCPIP, FE	At least one is required after TCPIP
FE	FE	Optional

31.3 Coding STL programs

STL is a high-level structured programming language that can be used to create message generation decks and define terminals and devices to be simulated by WSim. STL utilizes constants, variables, expressions, and structured control statements.

An STL program is usually divided into one or more procedures. The STL Translator translates STL programs into message generation decks, each message generation deck corresponding to one STL procedure. Network definitions can be included into STL programs. The STL Translator invokes the preprocessor to validate and store the network definition statements.

STL is similar but not equivalent to REXX. STL uses many of the same commands and functions as REXX. However, STL is not the run-time scripting language; therefore, WSim must process or translate STL into MSGTXT format prior to test execution.

Variables and constants

In STL, you can use variables and constants as data items in your programs. A variable contains data that is used by a program in a certain way, but whose value can vary. In a program, each variable has a unique symbolic name.

Constants are values that do not change in the course of program execution. Constants can be used to initialize variables, to test the contents of variables, and to generate messages. They can also make up part of STL expressions. Names can be assigned to most constants. STL substitutes the actual constant value for the constant name.

Variables are classified as one of three data types:

- Integer
- String
- Bit

Integer variables can take any positive integer value from 0 to 2147483647. When a network is first initialized, the value of an integer variable is 0. Integer variables translate into counters in the scripting language. An integer variable's value wraps if incremented beyond 2147483647 or decreased below 0. That is, if a variable has a value of 2147483647 and 1 is added to it, the result is 0. If a variable has a value of 2 and 3 is subtracted from it, the result is 2147483647.

String variables can contain only characters. They must be from 0 to 32767 characters long. A character can be any 1-byte value, from '00'X to 'FF'x. When a network is first initialized, the value of a string variable is '' (the null string, which is represented by a pair of single or double quotation marks). String variables translate into save areas in the scripting language.

Bit variables represent binary data items. STL bit variables can have the value of ON or OFF. When a network is initialized, all bit variables have a value of OFF. Bit variables translate into switches in the scripting language.

Special variables are available to be used by your STL program:

BUFFER or **SCREEN** — For nondisplay terminals, the device buffer; for display terminals, the screen image. Use the SUBSTR and ROWCOL functions to query a screen area, for example, x = substr(screen,rowcol(10,20),6).

DATA or **TH** — Incoming or outgoing data, including SNA headers if present.

RH	The SNA request/response header (RH) portion of incoming or outgoing data plus the SNA request/response unit (RU).
RU	The SNA Request/response unit (RU) portion of incoming or outgoing data.

Constants are classified as:

- Integer
- String
- Hexadecimal string
- Bit data

Once a type is associated with a variable or named constant, you cannot change the type in the course of a program.

Integer constants are positive decimal integers from 0 to 2147483647.

String constants are any set of characters. A string constant must be enclosed in a pair of single or double quotation marks (the string delimiter character). If a string constant contains the string delimiter character (a single or double quotation mark), that character must be entered twice so that it can be recognized. You probably want to use single quotation marks as your delimiter character for strings containing double quotation marks, and double quotation marks as a delimiter for strings containing single quotation marks. You do *not* have to use the same delimiter character throughout your program; you can use whichever is most appropriate for each string you are enclosing. The string constants ‘’ and “” have a length of zero and are called the null string.

Hexadecimal string constants are specified by enclosing pairs of hexadecimal digits in string delimiters followed by the character x or X. Each pair of hexadecimal digits represents a single character in the string.

Bit constants can take the value ON or OFF. These constants can be assigned to bit variables or used to test the current setting of a bit variable.

Functions

STL provides a number of built-in functions that can be used to access and manipulate data. All functions return a value, which can be an integer, string, or bit value. You can use functions as expressions or parts of expressions wherever variables can be used. STL function names are reserved words. You cannot use them as names or labels.

A function consists of the name of an STL function, followed by a left parenthesis, any arguments for the function, and a right parenthesis. The function arguments consist of values you are supplying to the function. Even if the function does not use arguments, you must code the parentheses. If you specify more than one argument, you must separate them with commas.

You can use the SUBSTR function (substring function) to reference a portion (or a substring) of a string expression.

The INDEX function returns an integer value that gives the position of a target string in a source string. If the target string is not found in the source string, the function returns a value of 0.

A number of functions perform translations between data types, for example:

B2X() and **X2B()**	From binary string to hexadecimal string and the opposite
C2D() and **D2C()**	From hexadecimal string to its decimal (integer) value and the opposite
C2X() and **X2C()**	From character (EBCDIC) string to hexadecimal string and the opposite
CHAR()	From integer value to its EBCDIC character representation

Statements

STL has two types of statements:

- Synchronous
- Asynchronous

Synchronous statements are executed in logical order, as programmed, as shown in Example 31-5.

Example 31-5 Synchronous statement

```
if substr(screen,1,4) = 'WSIM' then do
cursor(1,1)
type 'Workload Simulator'
end
transmit using enter
```

Asynchronous statements are executed when WSim detects a specified condition when a message is sent or received, as shown in Example 31-6.

Example 31-6 Asynchronous statement

```
onin01: onin substr(ru,1,1)='F5'X then,
found = on
```

The following asynchronous STL statements allow you to test messages received and transmitted:

- ONIN

  Defines a condition that is to be tested when data is received by a simulated terminal
- ONOUT

  Defines a condition that is to be tested when data is transmitted by a simulated terminal
- ON SIGNALED

  Defines an action to be taken when the specified event is signaled (by means of a SIGNAL or QSIGNAL command)
- WAIT UNTIL and QUIESCE UNTIL

  Interrupts STL execution and defines a condition that, when met, allows STL program execution to resume (can also be coded on the TRANSMIT statement)
- IF

  Coded in a NTWRK statement, it checks a condition at a network level when data is sent or received

The DEACT statement deactivates an asynchronous condition.

Several statements are available for working with the screen images, simulating the operator pressing various keys. Some of these statements are presented in Table 31-2.

Table 31-2 STL statements for work with screen images

Statement	Action
TYPE expression	Type the expression at the current cursor position.
CURSOR(*n.m*)	Move the cursor to row *n* and column *m*.
CURSOR("UP")	Move the cursor up one row.
CURSOR("DOWN", 10)	Move the cursor down 10 rows.
TAB	Tab to the right (move cursor to next input field on screen).
BTAB	Tab to the left (back up one input field).
HOME	Home key (move cursor to beginning of first input field on screen).
RESET	Reset key.
INSERT	Insert key.
DELETE	Delete beginning with the character at the current cursor position.
EREOF	Erase to end of field.

STL provides several statements to transmit messages from simulated terminals, receive messages, and take actions based upon messages sent or received.

Using the TRANSMIT statement is the most common method of sending messages. (The TRANSMIT statement cannot be used in CPI-C transaction program simulations. Use the CMSEND statement instead.)

The most typically used form of the TRANSMIT statement is shown in Example 31-7.

Example 31-7 Common form of TRANSMIT statement

```
TRANSMIT USING key
```

In this example, *key* can be ENTER (default if omitted), CLEAR, PFnn, PAn, SYSREQ, and some others.

The TRANSMIT statement does not wait for a response. Execution of the program continues immediately, unless the user optionally coded a WAIT statement with TRANSMIT, as shown in Example 31-8.

Example 31-8 TRANSMIT and WAIT statements

```
TRANSMIT USING ENTER AND WAIT UNTIL ONIN
```

The WAIT statement causes the STL program to wait until a message is received. The STL program waits and then continues execution after a message is received.

The common forms of the WAIT statement are presented in Table 31-3.

Table 31-3 Common forms of WAIT statement

WAIT statement	Action
WAIT UNTIL ONIN	Wait until a message is received.
WAIT UNTIL ONOUT	Wait until a message is sent (by an asynchronous statement).
WAIT UNTIL POSTED(*event name*)	Wait until the named event has been posted.
WAIT UNTIL SIGNALED(*event name*)	Wait until the named event has been signaled.

STL provides mechanisms to control the flow of programs by using a subset of statements called structured flow-of-control statements. These statements enable WSim to move through a program in a nonlinear fashion.

Structured flow-of-control statements give an ability to have a procedure call another procedure, passing control to the second procedure. They also allow taking a specific action depending upon conditions that occur.

There are four types of control statements:

- CALL
- IF/THEN/ELSE
- SELECT
- DO

The CALL statement shifts control of program execution from one procedure to another procedure. The IF/THEN/ELSE statement and the SELECT statement group execute statements selectively depending on conditions that exist. The DO statement group enables you to group statements logically and in some cases to execute statements repetitively.

31.4 WSim operator commands

You can issue WSim operator commands to:

- Initialize a simulated network.
- Start a simulation run.
- Query the status of a simulated network device.
- End the simulation run.

The WSim operator commands can be entered when running WSim in the TSO foreground session and when running WSim as a batch job.

A few examples of WSim operator commands are shown in Table 31-4.

Table 31-4 Samples of WSim operator commands

WSim operator command	Result
I testnet	Initialize the network named testnet.
I testnet,S	Initialize the network named testnet and start the network activity.
S	Start activity for all simulated resources in all initialized networks.
S testnet	Start activity for all simulated resources in the network named testnet.
Q	Display the status of all initialized networks.
Q lu	Display the status of the simulated resource named lu.
W lu	Display response time statistics (RSTATS) for the simulated resource named lu.
P	Stop activity for all simulated resources in all networks.
P resourcename	Stop activity for the resource named resourcename.
ZEND	Causes orderly shutdown of the WSim and writes the message log buffers to the log data set.

Part 9

Scenarios

In this part of the book, we cover various scenarios using the Problem Determination and Deployment Tools, such as:

- Introduction to scenarios
- Guided tour of the Trader application
- Scenario 1: Invalid VSAM data generating an abend
- Scenario 2: Using Debug Tool
- Determining if there is an error in the DB2 data or program logic
- Using Workload Simulator to create a TRADER workload
- Using Application Performance Analyzer with the WSim workloads

32

Introduction to the scenarios

These scenarios were designed to highlight features of the Problem Determination and Deployment Tools in a brief, but effective manner.

In this chapter we cover these topics:

- An overview of the scenarios
- How to install the application software
- The system configuration
- How to validate the installation

32.1 Overview

The scenarios presented in this book are based on the Trader stock trading application, which was written specifically to demonstrate the features of the Application Development Tools. The users of this application might be investors checking their holdings or buying and selling shares of stock. The application takes two forms:

- CICS transactions
- Batch jobs

Each form can access data on VSAM, DB2, and IMS.

Note: This application does not reflect real-world securities processing. It is merely designed to demonstrate the features of the Application Development Tools.

The next chapter presents a more detailed look at the Trader application. In the other chapters in this part, we create scenarios based on the Trader application. In each scenario but the last two, we deliberately introduce errors into the application to allow us to demonstrate the functionality of the tools. We then describe, in detail, the steps that you take to isolate the error and to correct the problem.

In the scenarios described in the following chapters in this part, all the applications are running on a single partition.

32.1.1 Overview of the programs

The Trader application is used to maintain a stock portfolio held by an individual. This application enables you to:

- Obtain quotes (in batch mode, you list portfolios and their values).
- Buy more shares of a company's stock.
- Sell currently held shares of a company's stock.

The Trader application uses two data sources:

- Company data
- Customer data

The company source contains the stock name and the past week's quotes. The customer source contains a record for each customer and company that he or she owns, including the number of shares held.

There can be two VSAM files and two DB2 tables in the MYTRADD DB2 plan and two IMS databases (in batch mode only).

In the CICS application, the transaction input is taken directly from an online user's interactions. In the batch application, the user's input is replaced with a sequential file that contains several records representing the day's transactions.

Overview of the CICS program

Figure 32-1 shows the processing that occurs in the CICS application.

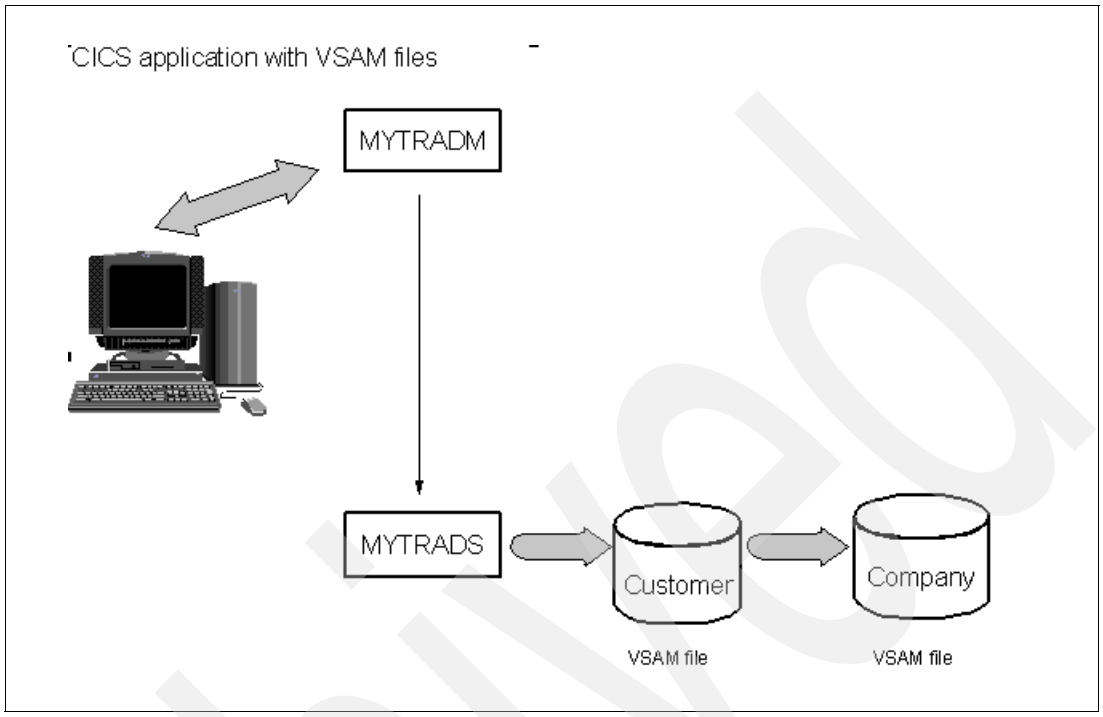


Figure 32-1 Trader Application: Single user transaction with CICS

Note: When you invoke this application, you can use any user name and password. But if you want to see the status from previous trading, use the same user name each time.

Overview of the batch program

Figure 32-2 shows the processing that occurs in the batch application.

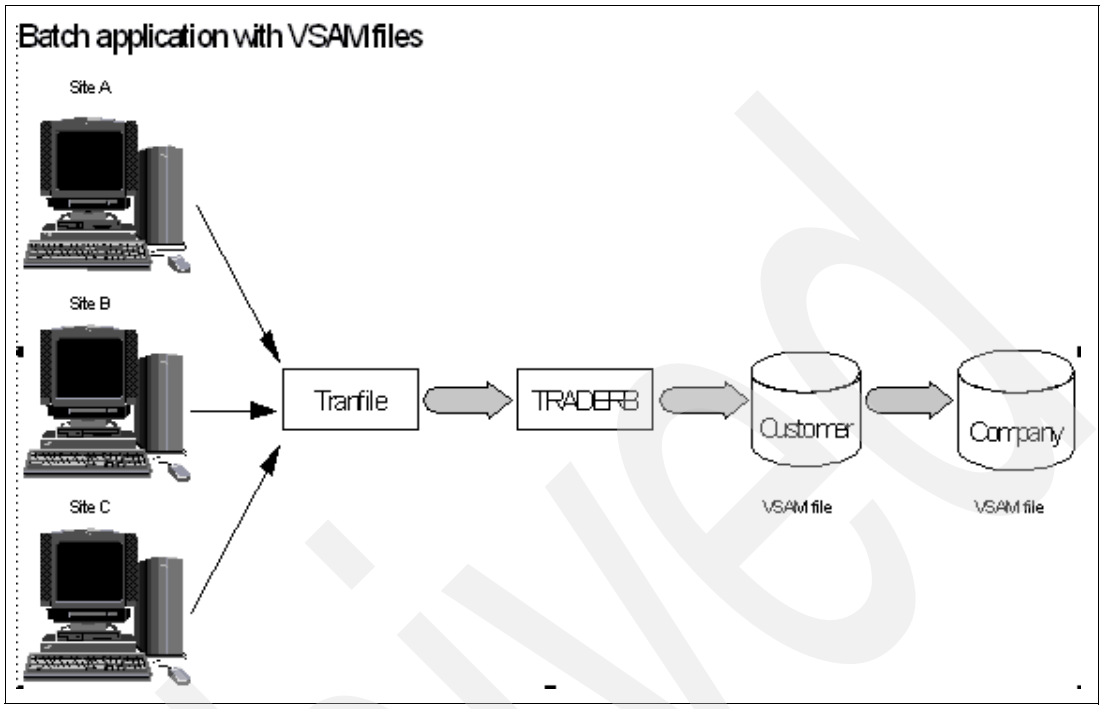


Figure 32-2 Trader application: multiple remote site transactions with batch

Note: You should always list the holdings of a user name to determine the number of shares in a portfolio before you begin to trade with it.

32.1.2 Software prerequisites

The application programs listed in Table 32-1 were created for this book and are installed on our system. These application programs were designed to demonstrate the functionality of the Application Development Tools.

Table 32-1 Application programs used in Trader application

Application program	Subsystem	Purpose
MYTRADMV MYTRAD	CICS	Retrieve customer information from VSAM files.
TRADERB	Batch	Process customer transaction from sequential and VSAMfiles.
MYTRADMD MYTRADD	CICS and DB2	Retrieve information from DB2 tables.
TRADERD	Batch and DB2	Process customer information from sequential file and DB2 tables.
TRADERI	Batch and IMS	Process customer information from sequential file and IMS databases.

32.2 Install the application software

In this section we describe how to install the application software that you use to run the different forms of the Trader application. You can install and use only the parts relevant to your system configuration.

If you intend to follow the examples on your own, you must have the system software. Refer to 32.3.1, “About the software prerequisites” on page 1105. We assume that you have access to a similar configuration.

32.2.1 Installing the demo files

You must install the application software that you intend to use to run the different forms of the Trader application.

Locating the Web material

The Web material associated with this book is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

`ftp://www.redbooks.ibm.com/redbooks/SG247192`

Alternatively, you can go to the IBM Redbooks Web site at:

`www.ibm.com/redbooks`

Select the **Additional materials** and open the directory that corresponds with the Redbooks form number, SG24-7372.

Using the Web material

The additional Web material that accompanies this book includes the following file:

File name	Description
SG247372.zip	Zipped code samples

System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space	4 MB for the downloaded zip file and unpacked files
Operating system	Windows 2000/XP
Processor	Pentium®
Memory	128 MB

How to use the Web material

Create a subdirectory (folder) on your workstation and unzip the contents of the Web material zip file into this folder.

The extracted files are all in binary format. They are the output of the TSO TRANSMIT command.

Use your mainframe file transfer protocol to upload the binary files. You must use the following attributes: FB, LRECL=80, BLKSIZE=3120.

After each file is uploaded, issue the following command from the TSO READY prompt:

```
RECEIVE INDA(xxxx)
```

In this command, *xxxx* is the name of the file.

If you issue the command against the *source* file, you receive the messages shown in Example 32-1.

Example 32-1 Receive INDA(xxxx) messages

```
INMR901I Dataset DAVINR1.REDBOOK.PDPAK.SOURCE from DAVINR1 on NODENAME
INMR906A Enter restore parameters or 'DELETE' or 'END' +
```

You can to reply as shown in Example 32-2.

Example 32-2 Receive INDA(xxxx) reply

```
indsn(pdtools.redbook.source)
```

The default high-level qualifier assigned to the file is be your TSO user ID, named *userid*.

Note: You can delete the zipped file and the temporary folder after you finish uploading all of the files.

32.2.2 Customization

Edit the JCL data set and change all instances of TSOUSERID to your TSO user ID. You can use the File Manager Find/Change utility to perform this step.

You must edit some members of the *userid*.PDTOOLS.REDBOOK.JCL data set to validate the following required information:

- For DB2:
  - DB2 load library
  - DB2 run-time library
  - DB2 DBRM library
- For IMS:
  - IMS reslib
  - IMS maclib
  - IMS genlib, genliba, and genlibb
- For COBOL:
  - COBOL compiler load library
- For Language Environment
  - LE run-time library
- For CICS:
  - CICS system load library
  - CICS user load library
  - CICS macro/DSECT library
  - CICS CSD data set name
- For Debug Tool:
  - Debug Tool system load library

The modifications must be made in these members of *userid*.PDTOOLS.REDBOOK.JCL:

BIND	Bind plan
CICSDB2C	CICS DB2 COBOL
CICSIMSC	CICS IMS COBOL
COBBATCH	Batch Trader compilation and link-edit
COBBDB2	Batch Trader DB2 compilation and link-edit
COBBIMS	Batch Trader IMS compilation and link-edit
COBCICS	Trader CICS and VSAM compilation and link-edit
COBPROC	Called by COBCICS
COBPROCB	Called by COBBATCH
DATA	Trader DB2 tables load
DB2COBBA	Called by COBBDB2
DB2CXCOB	Called by CICSDB2C
DBIMS	IMS databases generation
DEFPDPAK	Load Trader definition to CICS
DEFVSAM1	Trader VSAM files definition and load
DFHMAPS	Called by GENMAP
GENMAP	CICS mapset generation
GRANT	GRANT DB2 plan to public
IMSGO	Called by TRADERI
PDPAK	CICS definition for all versions of Trader
TABLES	DB2 Tables definition
TRADERB	Batch execution of Trader for VSAM data
TRADERBS	Batch execution of Trader for VSAM data for SEQ input
TRADERD	Batch execution of Trader for DB2
TRADERI	Batch execution of Trader for IMS

In addition, if you have not already done so, you must change the string, TSOUSERID, to your own TSO user ID.

You also must get your DB2 subsystem name. The following jobs use it:

- BIND
- DATA
- GRANT
- TABLES

32.2.3 Setting up the applications

The starting point for the scenarios is an established stock trading application.

Generate the data

Perform the following steps if you want to set up this application at your site:

1. Generate the NEWTRAD mapset using the GENMAP job.
2. Define the two VSAM data sets (COMPFILE and CUSTFILE) with the DEFVSAM1 job.

   This loads the VSAM files with sample data. If you wish, you can use the File Manager copy utility to perform this step.
3. Define the DB2 tables, CUSTOMER_DETAILS and COMPANY_DETAILS, with *userid*.PDTOOLS.REDBOOK.JCL(TABLES).
4. Populate these DB2 tables with *userid*.PDTOOLS.REDBOOK.JCL(DATA) or using File Manager.

Generate the executables

Perform the following steps to generate the executables:

1. Compile all of the COBOL programs with the appropriate batch compile job, as shown in Table 32-2.

   Note: The batch job *userid*.PDTOOLS.REDBOOK.JCL(CICSDB2C) allows you to preprocess, compile, and link-edit the program MYTRADD.

   The batch job *userid*.PDTOOLS.REDBOOK.JCL(COBCICS) contains the JCL to compile a COBOL/CICS program, which can be used to compile MTRADMD. You have to change the program name in the symbolic parameter MEM and in the NAME card of the link-edit step.

Table 32-2 COBOL application programs with compile job name

Program	Member	Proc
MYTRADMV	COBCICS	COBPROC
MYTRADS	COBCICS	COBPROC
TRADERB	COBBATCH	COBPROCB
MYTRADD	CICSDB2C	DB2CXCOB
MYTRADMD	COBCICS	COBPROC
TRADERD	COBBDB2	DB2COBBA
MYTRADMI	COBCICS	COBPROC

   Note: Make certain that you validate the names of all the product libraries before you submit these batch jobs. For some batch jobs, you might be required to pre-allocate your output data sets.

   To compile your COBOL, DB2, or CICS applications, we recommend that you use the DB2 (DB2 V7 or later) or CICS (CICS-TS V2.3 or later) integrated precompiler or translator for two reasons:

   – Debugging becomes much more comfortable, because you are not presented with the inserted resolution of your EXEC SQL or EXEC CICS statements.
   – If you use the external precompiler or translator, you must save the precompiler/translator output (which becomes the input to the compiler) to a permanent data set. This is not required if you compile with the integrated precompiler/translator, since your source is directly input to the compilers.

2. Create the DB2 plan, MYTRADD, with *userid*.PDTOOLS.REDBOOK.JCL(BIND).
3. Grant execution access to this plan with *userid*.PDTOOLS.REDBOOK.JCL(GRANT).
4. Define all the necessary application resources to CICS:
   – The MYTRADxx programs from step 5
   – The mapset NEWTRAD from step 1
   – The transactions MYTD and TDB2
   – The two VSAM files from step 2

   These resource definitions are contained in *userid*.PDTOOLS.REDBOOK.JCL(PDPAK).

   Review this file for changes that are relevant to your site's standards.

5. To add these definitions to the DFHCSD, the CICS definitions list, use *userid*.PDTOOLS.REDBOOK.JCL(DEFPDPAK).

   Install the defined resources.

   Check that everything is here with transaction CEDA under CICS to display the contents of the PDPAK group, as shown in Example 32-3.

Example 32-3 Result of CEDA DIS GROUP(PDPAK)

```
DB2ENTRY    DBMYTDB2
            DBMYTDB3
FILE        COMPFILE
            CUSTFILE
MAPSET      NEWTRAD
PROGRAM     MYTRADMV
            MYTRADS
            MYTRADMD
            MYTRADMX
            MYTRADD
            MYTRADD3
            MYTRADMI
            MYTRADI
            MYTRADMY
            MYTRADI3
TRANSACTION TIM3
            TIMS
            MYTD
            TDB2
            TDB3
```

32.3 About the system configuration

You can follow along with each of the scenarios even if you do not install the application programs. However, if you do want to run the applications, you also must have the appropriate system configuration.

This section briefly reviews the software that was installed on our system, and what you must run the applications on yours.

32.3.1 About the software prerequisites

The Trader application and the scenarios have been tested with the following software installed and configured:

- z/OS V1R7
- CICS Transaction Server V2.3
- DB2 Universal Database V8.1
- IMS V8

The Application Development Tools for z/OS and OS/390:

- IBM Debug Tool Version 7 Release 1
- IBM Fault Analyzer for z/OS Version 7 Release 1
- IBM File Manager for z/OS Version 7 Release 1
- IBM Application Performance Analyzer Version 7 Release 1
- IBM Workload Simulator for z/OS and OS/390 Version 1 Release 1

It is possible that other levels of these software components might work, but the applications were tested with the levels listed here.

32.3.2 About the CICS configuration

A summary of the steps required to set up the CICS configuration follows:

- The CICS resource definitions are specified in *userid*.PDTOOLS.REDBOOK.JCL(PDPAK).
- An entry for the PDPAK group was added to the site's CICS definitions list, DFHCSD.
- After all of the batch compile jobs completed, the load modules for the CICS COBOL application programs MYTRADMV, MYTRADMD, MYTRADMI, MYTRADS, MYTRADD, and MYTRADI were in our CICS user load library, which is in the CICS DFHRPL.

32.3.3 About the DB2 configuration

A summary of the steps required to set up the DB2 configuration follows:

- The batch job *userid*.PDTOOLS.REDBOOK.JCL(BIND) creates a plan used by the CICS application.
- The plan name is MYTRADD, which is specified in the DB2 entry of the CICS definitions.
- The batch job *userid*.PDTOOLS.REDBOOK.JCL(GRANT) grants execution access to the plan MYTRADD.
- The batch job *userid*.PDTOOLS.REDBOOK.JCL(TABLES) defines the tables CUSTOMER_DETAILS and COMPANY_DETAILS.
- The batch job *userid*.PDTOOLS.REDBOOK.JCL(DATA) populates these tables.

32.3.4 Validating the installation

Follow the instructions in Guided Tour of the Trader Application to validate the installation. It presents the CICS/VSAM and batch/VSAM aspects of the application.

33

Guided tour of the Trader application

After you have established all of the components of the applications in your environment, you must validate the installation.

The Trader application is used to maintain a stock portfolio held by an individual. This application enables you to:

- Obtain quotes.
- Buy more shares of a company's stock.
- Sell currently held shares of a company's stock.

Note: This example was designed to demonstrate the capabilities of the Problem Determination and Deployment Tools. Therefore, a minimal amount of code was developed. This application does *not* represent real-world securities processing.

33.1 Getting started

Before you can start the applications, the subsystems must be started. The subsystems required for Trader application are CICS, DB2, and IMS.

33.2 Trader application in CICS

In this section we present the CICS/VSAM application. The panels and process are also valid for DB2 data.

33.2.1 Logging on to the application

Perform the following steps to start the Trader application:

1. Verify that you have installed this application correctly.
2. Log on to your demonstration CICS region.
3. Enter the transaction `MYTD`.
4. Enter the user name `ERIC` and the password `xxxx`, as shown in Figure 33-1.

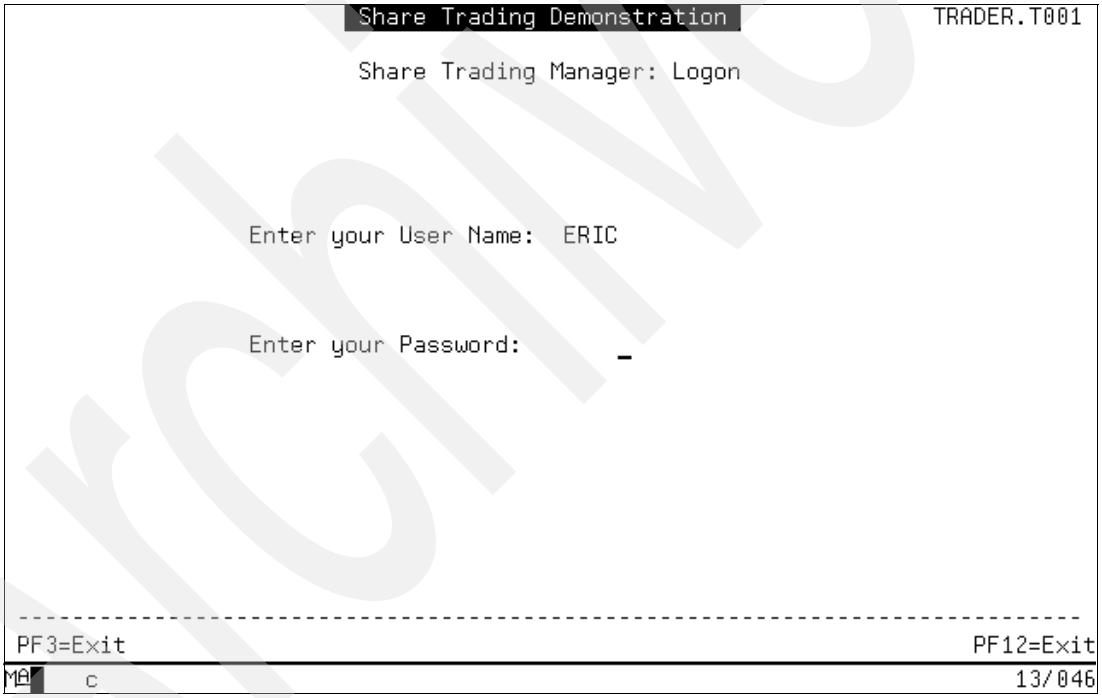


Figure 33-1 CICS Trader logon panel

Note: In the Trader application, navigation keys are displayed at the bottom of each screen. PF3 is used to go back to the previous screen (except on the Logon screen) and PF12 is used to terminate the application.

33.2.2 Running the Trader application

After the logon, the Company Selection panel is displayed, as shown Figure 33-2. Select a company you want to trade.

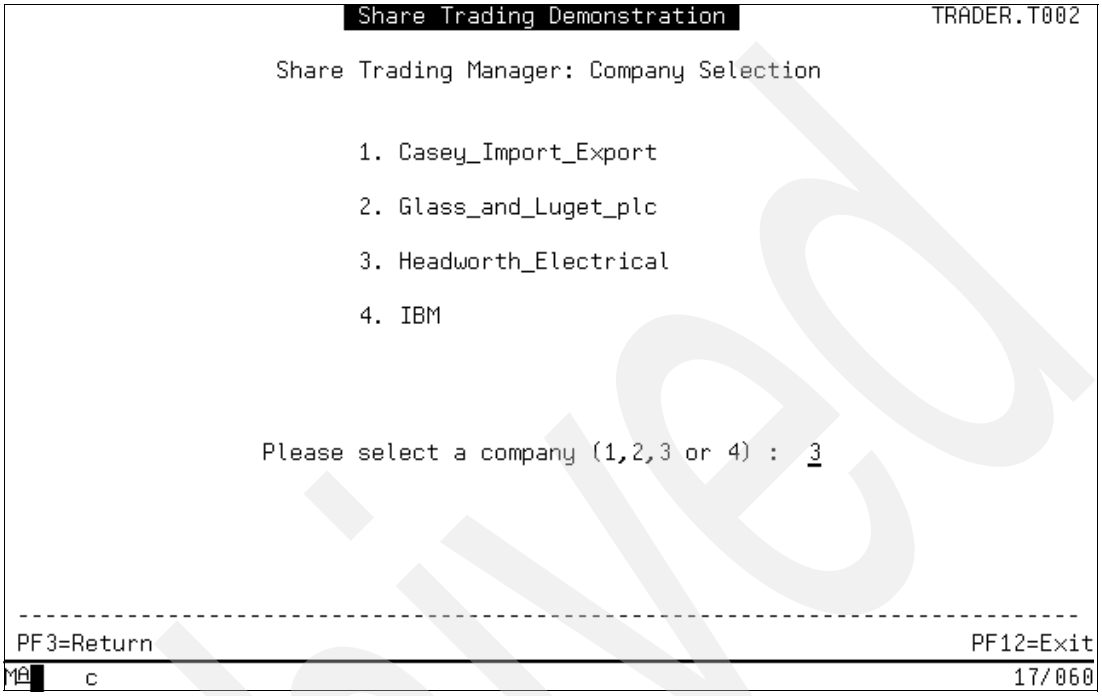


Figure 33-2 CICS Trader list of companies

The Options panel is displayed, as shown in Figure 33-3.

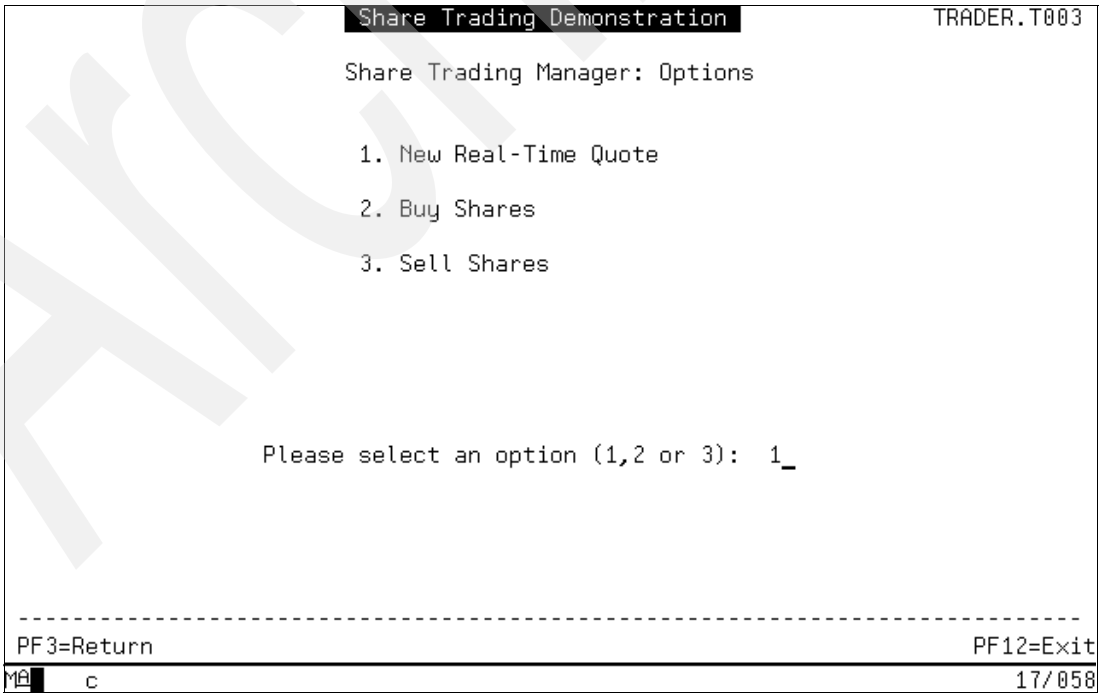


Figure 33-3 CICS Trader actions available for this company

On this panel, you select the trading option you want to perform:

- Obtain real-time quotes for a company.
- Buy additional shares of the company.
- Sell existing shares of the company.

Continue by selecting an option.

33.2.3 Obtaining real-time quotes

To obtain real-time quotes, request a list of real-time quotes for the selected company by selecting option 1. Share prices from the prior week are displayed including net present value, as shown in Figure 33-4.

```
                        Share Trading Demonstration                TRADER.T004

                    Share Trading Manager: Real-Time Quote

      User Name:     ERIC

      Company Name:  Headworth_Electrical

      Share Values:                    Commission Cost:
         NOW:        00124.00             for Selling:        011
         1 week ago: 00141.00             for Buying:         012
         6 days ago: 00138.00
         5 days ago: 00137.00
         4 days ago: 00133.00
         3 days ago: 00133.00
         2 days ago: 00133.00        Number of Shares Held: 0005
         1 day ago:  00131.00        Value of Shares Held:  000000620.00



Request Completed OK
--------------------------------------------------------------------------------
PF3=Return                                                          PF12=Exit
MA  c                                                                  01/001
```

Figure 33-4 CICS Trader quotes

This screen displays the price of the company's shares over the past seven days, the number of shares held, and the value of those shares based on the current day's price. Data is retrieved from:

- The company's share price is read from the VSAM file COMPFILE (*userid*.PDTOOLS.REDBOOK.COMPFILE).
- The details of the user's portfolio (such as the number of shares held) are read from the VSAM file CUSTFILE (*userid*.PDTOOLS.REDBOOK.CUSTFILE).

Press PF3 to return to the Options panel.

33.2.4 Buying shares

Use the following steps to buy shares:

1. Request to buy shares of the selected company by selecting option **2**.
2. Enter the number of shares to purchase, as shown in Figure 33-5.

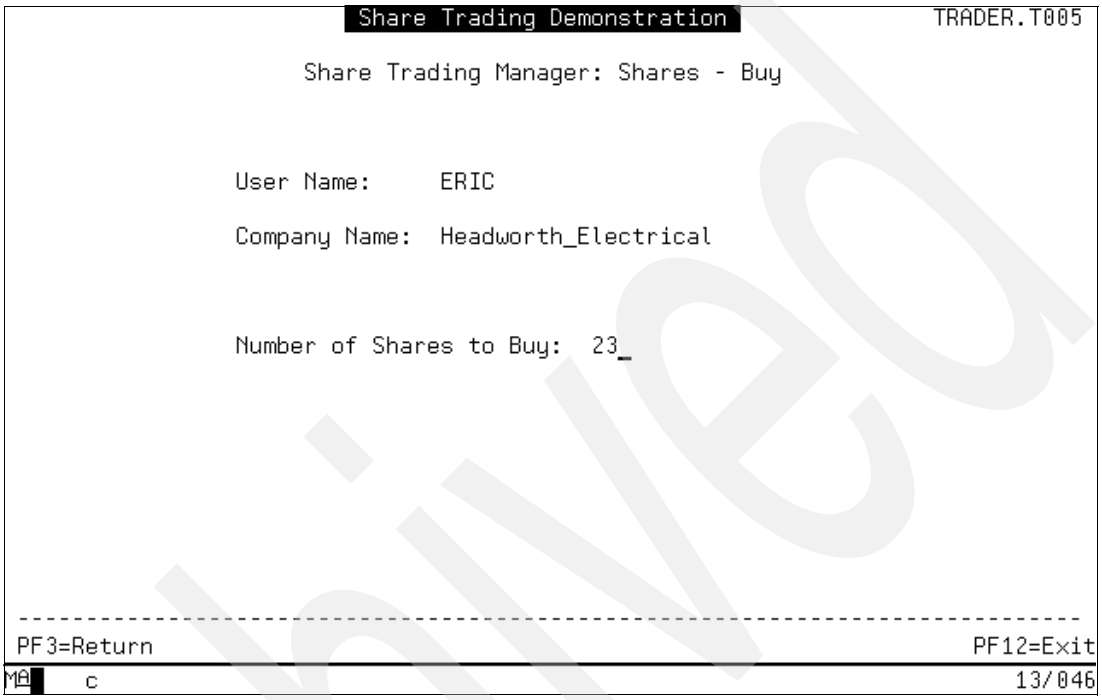


Figure 33-5 CICS Trader buy shares

3. A confirmation message is issued.

   The Options panel is re-displayed with a message in the bottom left-hand corner indicating the status of the transaction:

   ```
   Request Completed OK
   ```

   If the process is successful, the value of the number of shares held is updated in the CUSTFILE.

33.2.5 Selling shares

Use the following steps to sell shares:

1. Request to sell shares of the selected company by selecting option **3**.
2. Enter the number of shares to sell, as shown in Figure 33-6.

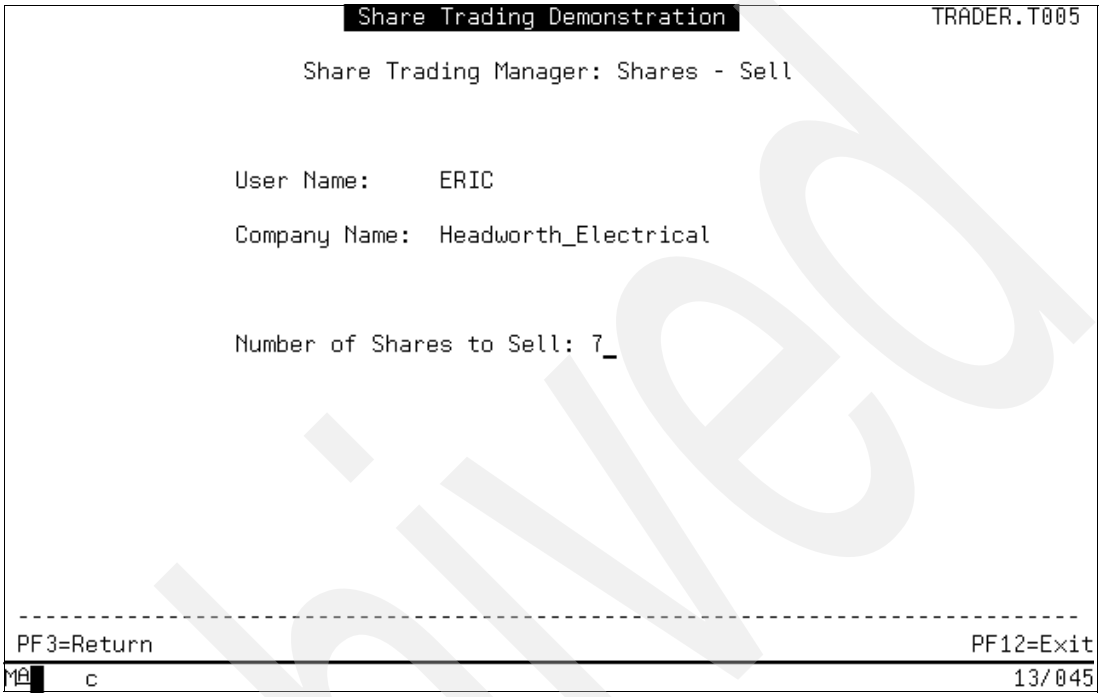


Figure 33-6 CICS Trader sell shares

3. A confirmation message is issued.

   The Options panel is re-displayed with a message in the bottom left-hand corner indicating the status of the transaction:

   ```
   Request Completed OK
   ```

   If the process is successful, the value of the number of shares held is updated in the CUSTFILE.

33.3 Running the Trader application in batch

Use the following steps to run the Trader application in batch:

1. Verify that you have installed this application correctly.
2. Create a transaction file that contains sample records to buy, sell, and list shares in one company.

You can use the one in *userid*.PDTOOLS.REDBOOK.SAMPLES(TRANFILE).

33.3.1 Running the batch job

To run the Trader application, submit the batch job TRADER, in *userid*.PDTOOLS.REDBOOK.JCL (Example 33-1).

Example 33-1 TRADER batch job

```
//TRADERB  JOB 1,RACFUSER,TIME=1440,NOTIFY=&SYSUID,REGION=4M,
//             CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//*****************************************************************
//GO       EXEC PGM=TRADERB,PARM=RPTOPTS(YES)
//STEPLIB  DD DISP=SHR,DSN=CHABERT.BOOK2005.LOAD
//SYSPRINT DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//COMPFILE DD DISP=SHR,DSN=CHABERT.BOOK2005.COMPFILE
//CUSTFILE DD DISP=SHR,DSN=CHABERT.BOOK2005.CUSTFILE
//TRANSACT DD DISP=SHR,DSN=CHABERT.BOOK2005.TRANFILE
//REPOUT   DD SYSOUT=*
//TRANREP  DD SYSOUT=*
//*
```

This job invokes the program TRADERB, which reads the transaction file. The contents of the file dictate the actions of the program, as reported in Example 33-2.

Example 33-2 TRANFILE sample

```
12345MARYLENE                .Glass_and_Luget_plc BUY_SELL       00781
12345ALBANE                  .Veck_Transport      BUY_SELL       23691
12345CHARLENE                .Headworth_ElectricalBUY_SELL       01501
12345RACHEL                  .IBM                 BUY_SELL       02501
12345MINNA                   .Casey_Import_Export BUY_SELL       00231
12345SALLA                   .ShareSelect         BUY_SELL       00061
12345EMMY                    .Glass_and_Luget_plc BUY_SELL       00782
12345JOUSO                   .Veck_Transport      BUY_SELL       23692
12345NIISKU                  .Headworth_ElectricalBUY_SELL       01502
12345ERIC                    .IBM                 BUY_SELL       02502
12345LEO                     .Casey_Import_Export BUY_SELL       00232
12345GARRY                   .ShareSelect         BUY_SELL       00062
12345PETER                   .Glass_and_Luget_plc BUY_SELL       00781
12345KLAS                    .Veck_Transport      BUY_SELL       22691
12345BJORN                   .Headworth_ElectricalBUY_SELL       01501
12345KLEO                    .IBM                 BUY_SELL       02501
12345MARYLENE                                     SHARE_VALUE
12345ALBANE                                       SHARE_VALUE
12345CHARLENE                                     SHARE_VALUE
12345RACHEL                                       SHARE_VALUE
12345MINNA                                        SHARE_VALUE
12345SALLA                                        SHARE_VALUE
12345EMMY                                         SHARE_VALUE
12345JOUSO                                        SHARE_VALUE
12345NIISKU                                       SHARE_VALUE
12345ERIC                                         SHARE_VALUE
12345LEO                                          SHARE_VALUE
12345GARRY                                        SHARE_VALUE
12345PETER                                        SHARE_VALUE
12345KLAS                                         SHARE_VALUE
12345BJORN                                        SHARE_VALUE
12345KLEO                                         SHARE_VALUE
```

Each record is validated against the company file. If a BUY or a SELL request is found, the appropriate program logic is invoked. The customer file is updated as a result.

33.3.2 Checking the result

All reports are written to the JES spool.

The share trading activity from this batch job is reported, as shown in Example 33-3.

Example 33-3 REPOUT

```
CUSTOMER : ALBANE                                              11/17/2005
-------------------------------------------------------------------------
     COMPANY                  SHARES    SHARE            TOTAL
                              HELD      VALUE            COST
-------------------------------------------------------------------------
     Casey_Import_Export         0        79.00             .00
     Glass_and_Luget_plc         0        19.00             .00
     Headworth_Electrical        0       124.00             .00
     IBM                         0       163.00             .00
     ShareSelect                 0       119.00             .00
     Veck_Transport           7107        36.00      255,852.00
CUSTOMER : CHARLENE                                            11/17/2005
-------------------------------------------------------------------------
     COMPANY                  SHARES    SHARE            TOTAL
                              HELD      VALUE            COST
-------------------------------------------------------------------------
     Casey_Import_Export         0        79.00             .00
     Glass_and_Luget_plc         0        19.00             .00
     Headworth_Electrical      450       124.00       55,800.00
     IBM                         0       163.00             .00
     ShareSelect                 0       119.00             .00
     Veck_Transport              0        36.00             .00
CUSTOMER : RACHEL                                              11/17/2005
-------------------------------------------------------------------------
     COMPANY                  SHARES    SHARE            TOTAL
                              HELD      VALUE            COST
-------------------------------------------------------------------------
     Casey_Import_Export         0        79.00             .00
```

Example 33-4 TRANREP

```
-------------------------------------------------------------------------------
CUSTOMER            COMPANY                QTY  REQ-TYP  STATUS
-------------------------------------------------------------------------------
MARYLENE            Glass_and_Luget_plc     78  BUY      PROCESSED SUCCESFULLY
ALBANE              Veck_Transport        2369  BUY      PROCESSED SUCCESFULLY
CHARLENE            Headworth_Electrical   150  BUY      PROCESSED SUCCESFULLY
RACHEL              IBM                    250  BUY      PROCESSED SUCCESFULLY
MINNA               Casey_Import_Export     23  BUY      PROCESSED SUCCESFULLY
SALLA               ShareSelect              6  BUY      PROCESSED SUCCESFULLY
PETER               Glass_and_Luget_plc     78  BUY      PROCESSED SUCCESFULLY
KLAS                Veck_Transport        2369  BUY      PROCESSED SUCCESFULLY
BJORN               Headworth_Electrical   150  BUY      PROCESSED SUCCESFULLY
KLEO                IBM                    250  BUY      PROCESSED SUCCESFULLY
```

This completes the batch/VSAM application. The results are also valid for DB2 and IMS.

34

Scenario 1: Invalid VSAM data generating an abend

In this chapter we describe the application components that exist in the CICS environment on our system and show how they are set up.

We force the application to abend, describe the steps required to identify the cause of an abend in the application using Fault Analyzer and Debug Tool, and explain how to manipulate the data to correct the problem using File Manager.

34.1 Setting up the components

Two types of components must be established for this scenario:

- CICS components
- Program products
  - Debug Tool
  - Fault Analyzer
  - File Manager

34.1.1 CICS components

The components used by the Trader application are listed in Table 34-1.

Table 34-1 CICS components of the Trader application for the VSAM scenario

Component	Details	Remarks
Programs	MYTRADMV MYTRADS	CICS COBOL programs
Tran ID	MYTD	Transaction associated with the program MYTRADMV
Mapset	NEWTRAD	BMS mapset containing all the maps used by the application
Files	CHABERT.C23.TRADER.CUSTFILE CHABERT.C23.TRADER.COMPFILE	VSAM files used by the application
Copybooks	CUSTFILE COMPFILE	File definition for CUSTFILE and COMPFILE

34.1.2 Program products

To use the Problem Determination Tools with this scenario, make sure that you have the following output or supporting files for each product.

Debug Tool

Ensure that Debug Tool is correctly installed in your CICS region.

Fault Analyzer

Ensure that Fault Analyzer is correctly installed in your CICS region. As an example, the transaction CFA returns the panel shown in Figure 34-1.

```
                    Fault Analyzer Control Transaction

Options: I=Install U=Uninstall
                   Current Status/Error Message
_ XPCABND          Installed
_ XDUREQ           Uninstalled
_ LE Exit          Installed
_ SDUMP Screening  Uninstalled

         Current     HWM  Setting
Active      0000    0001     0001
Waiting     0000    0000     0020




IDITRACE status   OFF

PF1=Help PF3=Exit PF5=Clear FND Area PF9=IVP PF11=IDITRACE ENTER=Update
```

Figure 34-1 CFA transaction

You must have a compiler listing or side file for the programs MYTRADMV and MYTRADS.

If you are not using the supplied JCL to compile these programs, make sure that you specify the following compiler options:

```
LIST,SOURCE,XREF,MAP
```

File Manager

You must have the following copybook that contains the record structure of the VSAM files CHABERT.C23.TRADER.CUSTFILE and CHABERT.C23.TRADER.COMPFILE: Copybooks CUSTFILE and COMPFILE in CHABERT.TRADER.COPYLIB.

Make sure that you run the DEFVSAM1 batch job to load the VSAM files.

34.2 Tracking an abend in the application

To demonstrate the capabilities of the Problem Determination Tools, this section describes how to force the application to abend and then step through the process of fixing it. Debug Tool is used to follow the execution of the application. Fault Analyzer is used to identify the cause of the abend. File Manager is used to correct the error in a VSAM application file.

- Access the Trader application by selecting:
  - MYTD transaction
  - Userid = TSDEMO
  - Password = itso
  - 4 = IBM
  - 1 = New Real-time quote

In this example, you attempt to obtain the real-time quotes of IBM for the customer, TSDEMO. After you select option **1** on the Options menu of the CICS Trader application and press Enter, the application abends, with a CICS-issued transaction abend message (Figure 34-2).

```
                    Share Trading Demonstration                TRADER.T003

                    Share Trading Manager: Options

                     1. New Real-Time Quote

                     2. Buy Shares

                     3. Sell Shares



                Please select an option (1,2 or 3):  1



DFHAC2206 18:20:52 CICSC31F Transaction MYTD failed with abend 4094. Updates
 to local recoverable resources backed out.
```

Figure 34-2 Abend 4094 (ASRA) in MYTD

34.2.1 Viewing the abend in Fault Analyzer

As we can see in Figure 34-3, Fault ID F00368 contains the abending program MYTRADS with the abend code ASRA. Make an Interactive reanalysis by entering an **I** in front of the Fault ID and pressing Enter.

```
  File  Options  View  Services  Help
IBM Fault Analyzer - Fault Entry List                    Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR

Fault History File or View  : 'IDI.HIST.CICSC31F'

   Fault_ID Job/Tran Program  User_ID  Abend  Date       Time     Dups  Sys/Jo
I  F00368 MYTD     MYTRADS   SIMCOCK  ASRA   2006/11/07 18:04:56    1 CICSC3
_  F00367 OSPH     OSTETPH1  ZHONG    ASRA   2006/11/07 11:25:52      CICSC3
_  F00366 CETR     DFHMCE    SIMCOCK  ATNI   2006/11/07 07:19:51      CICSC3
_  F00365 OSC1     DFHZARQ   ZHONG    ATCV   2006/11/06 16:58:08      CICSC3
_  F00364 CFA      IDIXFA    SIMCOCK  ASRA   2006/11/02 21:01:38      CICSC3
_  F00363 OSPH     OSTETPH1  ZHONG    ATND   2006/11/02 14:53:49      CICSC3
_  F00362 CFA      IDIXFA    SIMCOCK  ASRA   2006/11/01 23:23:28      CICSC3
_  F00361 CFA      IDIXFA    SIMCOCK  FLT1   2006/11/01 21:26:20      CICSC3
_  F00360 PSC1     PSTESTC1  TIMOTHY  4038   2006/10/27 11:22:37      CICSC3
_  F00359 PSC1     PSTESTC1  TIMOTHY  AICA   2006/10/27 11:20:36    1 CICSC3
_  F00358 PSC1     PSTESTC1  TMROSS   AEYH   2006/10/27 10:19:12    1 CICSC3
_  F00357 MNAP     PSMAIN    TIMOTHY  PMN1   2006/10/27 10:17:16      CICSC3
_  F00356 PSPM     n/a       TIMOTHY  4038   2006/10/26 18:03:42    1 CICSC3
_  F00355 PSPM     PSTESTPM  TIMOTHY  4038   2006/10/26 17:39:13      CICSC3
_  F00354 PSC1     PSTESTC1  TIMOTHY  ASRA   2006/10/26 15:47:55      CICSC3
_  F00353 PSC1     n/a       TIMOTHY  4038   2006/10/26 15:20:23      CICSC3
_  F00352 PSC1     PSTESTC1  TIMOTHY  4038   2006/10/26 14:14:16    1 CICSC3
_  F00351 PSC1     n/a       TIMOTHY  4038   2006/10/26 13:08:57    1 CICSC3
_  F00350 PSC1     PSTESTC1  TIMOTHY  4038   2006/10/26 12:58:03      CICSC3
_  F00349 PSC1     PSTESTC1  TIMOTHY  ASRA   2006/10/26 12:41:01      CICSC3
_  F00348 PSC1     n/a       TIMOTHY  4038   2006/10/26 12:19:17      CICSC3

 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 34-3 Interactive reanalysis

After the Interactive reanalysis has finished, the Interactive Reanalysis Report is displayed, as shown in Figure 34-4.

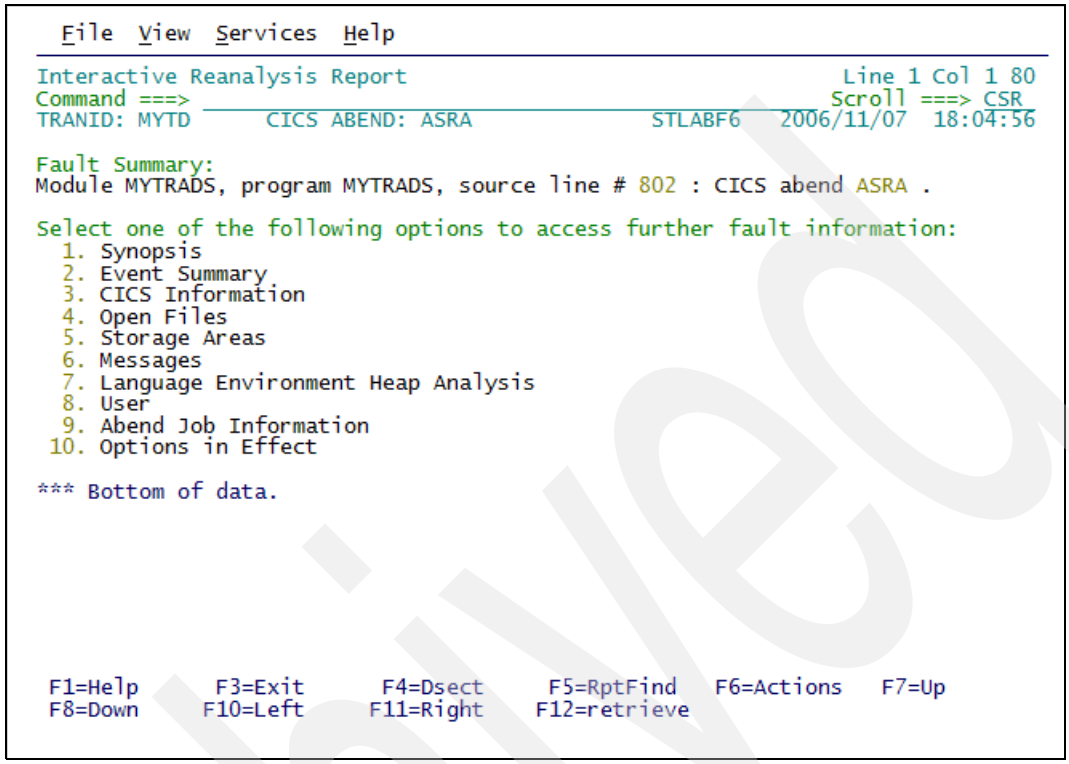


Figure 34-4 Interactive reanalysis report

Here you can see that the abend occurred at line 802. You use this information when you involve Debug Tool. To look for more details, choose option **1** to see the Synopsis, as shown in Figure 34-5.

```
  File  View  Services  Help
Synopsis                                                    Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2006/11/07  18:04:56

A CICS abend ASRA occurred in module MYTRADS program MYTRADS at offset
X'1DC6'.

A program-interruption code 0007 (Data Exception) is associated with this
abend and indicates that:

  A decimal digit or sign was invalid.

The cause of the failure was program MYTRADS in module MYTRADS.  The COBOL
source code that immediately preceded the failure was:

  Source
  Line #
  000802             MULTIPLY DECIMAL-SHARE-VALUE BY DEC-NO-SHARES
  000803               GIVING DECIMAL-SHARE-VALUE

The COBOL source code for data fields involved in the failure:

  Source
  Line #
  000115           03 DEC-NO-SHARES             PIC S9(7) COMP-3.
  000220              07 DECIMAL-SHARE-VALUE PIC 9(11)V99.

Data field values at time of abend:

  DEC-NO-SHARES       = X'F0F2F0F0'  *** Cause of error ***
  DECIMAL-SHARE-VALUE = 163.00

 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 34-5 Interactive reanalysis synopsis

The synopsis shows:

- The source line where the abend occurred
- The variables involved and their values at abend

You can see that DEC-NO-SHARES contains invalid data. To obtain more information about the variable, place the cursor on line 000115, as shown in Figure 34-6, and press Enter.

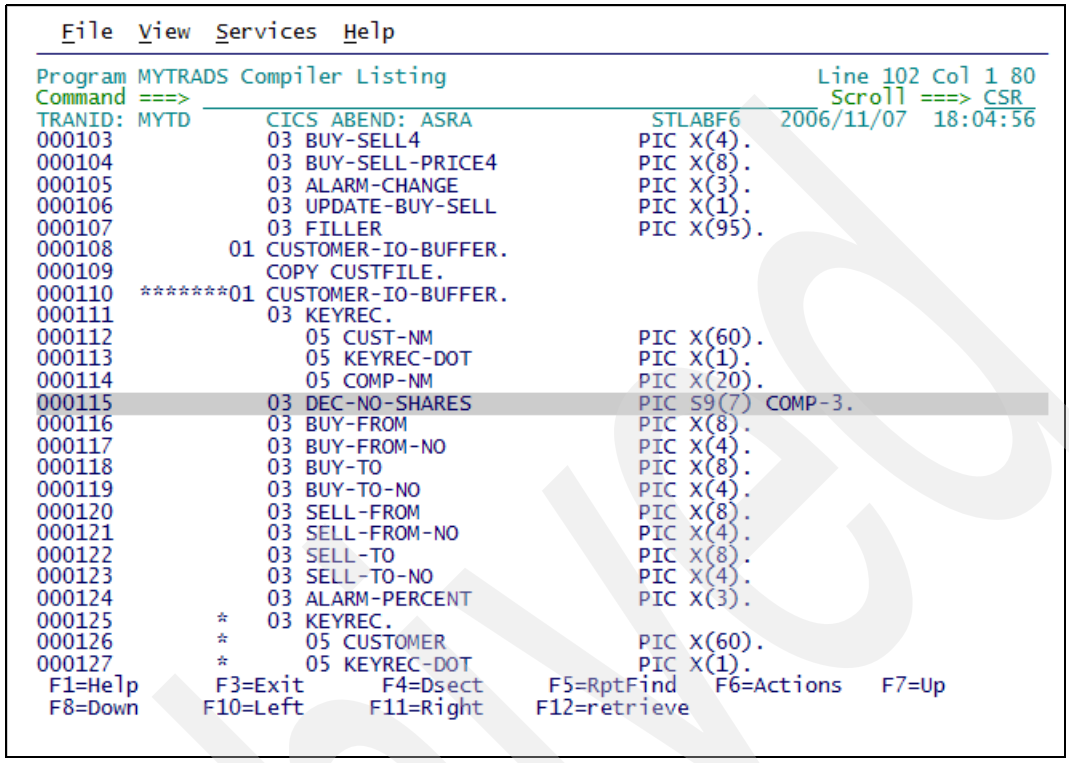


Figure 34-6 Complier listing

The resulting message is:

```
DEC-NO-SHARES is a part of CUSTOMER-IO-BUFFER.
```

So now you have the following questions:

- Is the bad data coming from the file that has been read?

  or

- Is it the program that put the wrong data in the field?

We discuss how to answer these questions in 34.2.2, “Using Debug Tool to analyze the problem” on page 1128 to analyze the problem. But before that, let us find the actual record in the dump.

How to find the record in the dump

From the Interactive Analysis Report, select option **2** (Event Summary) and the panel shown in Figure 34-7 is displayed.

```
  File  View  Services  Help
Event Summary                                                Line 1 Col 1 80
Command ===> _____________________________________________  Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2006/11/07  18:04:56

Event           Fail  Module   Program  EP
#  Type         Point Name     Name     Name     Event Location (*)      Loaded
 1 Call               DFHAPLI  DFHAPLI1 n/a      P+2FC2                  IBM.TS
 2 Call               CEEPLPKA n/a      CEECRINI E+AF6                   CEE.SC
 3 Call               CEEPLPKA n/a      CEECRINV E+302                   CEE.SC
 4 Link               MYTRADMV MYTRADMV MYTRADMV L#1253 P+24D6           CHABER
 5 Call               DFHAPLI  DFHAPLI1 n/a      P+2FC2                  IBM.TS
 6 Call               CEEPLPKA n/a      CEECRINI E+AF6                   CEE.SC
 7 Call               CEEPLPKA n/a      CEECRINV E+302                   CEE.SC
 8 Abend ASRA  ***** MYTRADS  MYTRADS  MYTRADS  L#802 P+1DC6             CHABER


(*) One or more of the following abbreviations might appear in the "Event
    Location" column:

    F#n  Source file number (refer to detailed event information for file
         identification)
    L#n  Source file line number
    S#n  Listing file statement number (refer to detailed event information
         for file identification)
    M+x  Offset from start of load module
    P+x  Offset from start of program
    E+x  Offset from start of entry point

 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 34-7 Event Summary

Event number **8** is your point of failure. Select it and press Enter to view the panel shown in Figure 34-8. Press PF8 to scroll through the event details as shown in Figure 34-9 to Figure 34-12.

```
  File  View  Services  Help
Event 8 of 8: Abend ASRA *** Point of Failure ***                Top of data
Command ===> _____________________________________________ Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2006/11/07  18:04:56

Previous Event Details


CICS Abend Code . . . . . : ASRA
Program-Interruption Code . : 0007 (Data Exception)
  A decimal digit or sign was invalid.

COBOL Source Code:
  Source
  Line #
  000802             MULTIPLY DECIMAL-SHARE-VALUE BY DEC-NO-SHARES
  000803               GIVING DECIMAL-SHARE-VALUE

Data Field Declarations:
  Source
  Line #
  000115           03 DEC-NO-SHARES            PIC S9(7) COMP-3.
  000220                07 DECIMAL-SHARE-VALUE PIC 9(11)V99.

Data Field Values:
  DEC-NO-SHARES       = X'F0F2F0F0'  *** Cause of error ***
  DECIMAL-SHARE-VALUE = 163.00

The SYSDEBUG file used for the above was found in
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 34-8 Point of Failure (1 of 5)

```
  File  View  Services  Help
Event 8 of 8: Abend ASRA *** Point of Failure ***            Line 26 Col 1 80
Command ===> ____________________________________________    Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2006/11/07  18:04:56
CHABERT.FBI.TEST.SIDEFILE(MYTRADS).

Load Module Name. . . . . : CHABERT.FBI.TEST.LOAD(MYTRADS)
  At Address. . . . . : 153348B0
  Load Module Length. . . . : X'3C80'
  Link-Edit Date and Time . : 2004/11/26  11:09:01

Program and Entry Point Name: MYTRADS
  At Address. . . . . : 153348D8 (Module MYTRADS offset X'28')
  Program Length. . . . . : X'300E'
  Program Language. . . . . : COBOL (Compiled using IBM Enterprise COBOL for
                              z/OS and OS/390 V3 R4 M0 on 2004/11/26 at
                              11:08:56)

Machine Instruction . . . . : FCA3D2C08459  MP    704(11,R13),1113(4,R8)
  At Address. . . . . : 1533669E (Program MYTRADS offset X'1DC6')
  AMODE . . . . . : 31
  Failing Operand . . . . . : Second operand
  First Operand Address . . : 1433AD90 (249668208 bytes of storage
                              addressable)
  First Operand Length. . . : 11
  First Operand Storage . . : 00000000 00000000 16300F  *..........*
  Second Operand Address. . : 1435B381 (Module MYTRADS program MYTRADS
                              WORKING-STORAGE SECTION BLW=0000 + X'459',
                              symbol DEC-NO-SHARES, source line # 115 -
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 34-9 Point of Failure (2 of 5)

```
  File  View  Services  Help
Event 8 of 8: Abend ASRA *** Point of Failure ***            Line 51 Col 1 80
Command ===> ____________________________________________    Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2006/11/07  18:04:56
                              249535615 bytes of storage addressable)
  Second Operand Length . . : 4
  Second Operand Storage. . : F0F2F0F0   *0200*

Program Status Word (PSW) . : 078D1000 953366A4

General Purpose Registers:
  R0: 1435ADFC (249537028 bytes of storage addressable)
  R1: 1433AD90 (249668208 bytes of storage addressable)
  R2: 153365EC (Module MYTRADS program MYTRADS + X'1D14', source line # 764 )
  R3: 0000001D (2019 bytes of storage addressable)
  R4: 1435AF6B (Module MYTRADS program MYTRADS WORKING-STORAGE SECTION
               BLW=0000 + X'43')
  R5: 40000000 (Storage invalid)
  R6: 143296B8 (Module MYTRADS program MYTRADS LINKAGE SECTION BLL=0002 +
               X'0', symbol DFHCOMMAREA, source line # 315 )
  R7: 001400D0 (Module MYTRADS program MYTRADS LINKAGE SECTION BLL=0001 +
               X'0', symbol DFHEIBLK, source line # 283 )
  R8: 1435AF28 (Module MYTRADS program MYTRADS WORKING-STORAGE SECTION
               BLW=0000 + X'0', symbol WRITEQ-WORDS, source line # 38 )
  R9: 1435ACC0 (249537344 bytes of storage addressable)
  R10: 15334A10 (Module MYTRADS program MYTRADS + X'138')
  R11: 15336396 (Module MYTRADS program MYTRADS + X'1ABE', source line # 676 )
  R12: 153349D4 (Module MYTRADS program MYTRADS + X'FC')
  R13: 1433AAD0 (249668912 bytes of storage addressable)
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=retrieve
```

Figure 34-10 Point of Failure (3 of 5)

```
  File  View  Services  Help
Event 8 of 8: Abend ASRA *** Point of Failure ***              Line 76 Col 1 80
Command ===> ____________________________________________________ Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2006/11/07  18:04:56
  R14: 95336B16 (Module MYTRADS program MYTRADS + X'223E', source line # 899 )
  R15: 00000000 (2048 bytes of storage addressable)

Associated Control Blocks

User EXEC Interface Block (EIB) at Address 001400D0 :
  Task Start Time . . . . . : 18:04:56              (EIBTIME - HH:MM:SS)
  Task Start Date . . . . . : 2006/11/07            (EIBDATE - YYYY/MM/DD)
  Transaction ID. . . . . : MYTD                    (EIBTRNID)
  Task Number . . . . . : 514                       (EIBTASKN)
  Terminal ID . . . . . : 0017                      (EIBTRMID)
  Cursor Position . . . . . : 0539                  (EIBCPOSN)
  Communication Area Length : 372                   (EIBCALEN)
  Attention ID. . . . . : ENTER                     (EIBAID)
  Last CICS Command . . . . : WRITEQ TS             (EIBFN)
  RESP Condition. . . . . : NORMAL                  (EIBRESP)
  RESP Condition Reason . . : 00000000              (EIBRESP2)
  Data Set ID . . . . . : COMPFILE                  (EIBDS)
  Request ID. . . . . : n/a                         (EIBREQID)
  TS Queue Name . . . . . : CEBR0000                (EIBRSRCE)
  Syncpoint Required. . . . : No                    (EIBSYNC)
  Facility Free Required. . : No                    (EIBFREE)
  Continue Receiving Data . : No                    (EIBRECV)
  Attach Header Data in RU. : No                    (EIBATT)
  RU Indicates End-of-Chain : No                    (EIBEOC)
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 34-11 Point of Failure (4 of 5)

```
  File  View  Services  Help
Event 8 of 8: Abend ASRA *** Point of Failure ***              Line 92 Col 1 80
Command ===> ____________________________________________________ Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2006/11/07  18:04:56
  RESP Condition Reason . . : 00000000              (EIBRESP2)
  Data Set ID . . . . . : COMPFILE                  (EIBDS)
  Request ID. . . . . : n/a                         (EIBREQID)
  TS Queue Name . . . . . : CEBR0000                (EIBRSRCE)
  Syncpoint Required. . . . : No                    (EIBSYNC)
  Facility Free Required. . : No                    (EIBFREE)
  Continue Receiving Data . : No                    (EIBRECV)
  Attach Header Data in RU. : No                    (EIBATT)
  RU Indicates End-of-Chain : No                    (EIBEOC)
  User Data Contains FMH. . : No                    (EIBFMH)
  Data is Complete. . . . . : No                    (EIBCOMPL)
  Signal Received . . . . . : No                    (EIBSIG)
  CONFIRM Request Received. : No                    (EIBCONF)
  Error Code Received . . . : No                    (EIBERRCD)
  SYNCPOINT ROLLBACK Req'd. : No                    (EIBSYNRB)
  No Data Sent. . . . . : No                        (EIBNODAT)
  Rollback. . . . . : No                            (EIBRLDBK)

Associated Messages

CEE3207S The system detected a data exception (System Completion Code=0C7).

Associated Storage Areas

*** Bottom of data.
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 34-12 Point of Failure (5 of 5)

Select **Associated Storage Areas**, where you can see the working-storage section with its values, as shown in Figure 34-13.

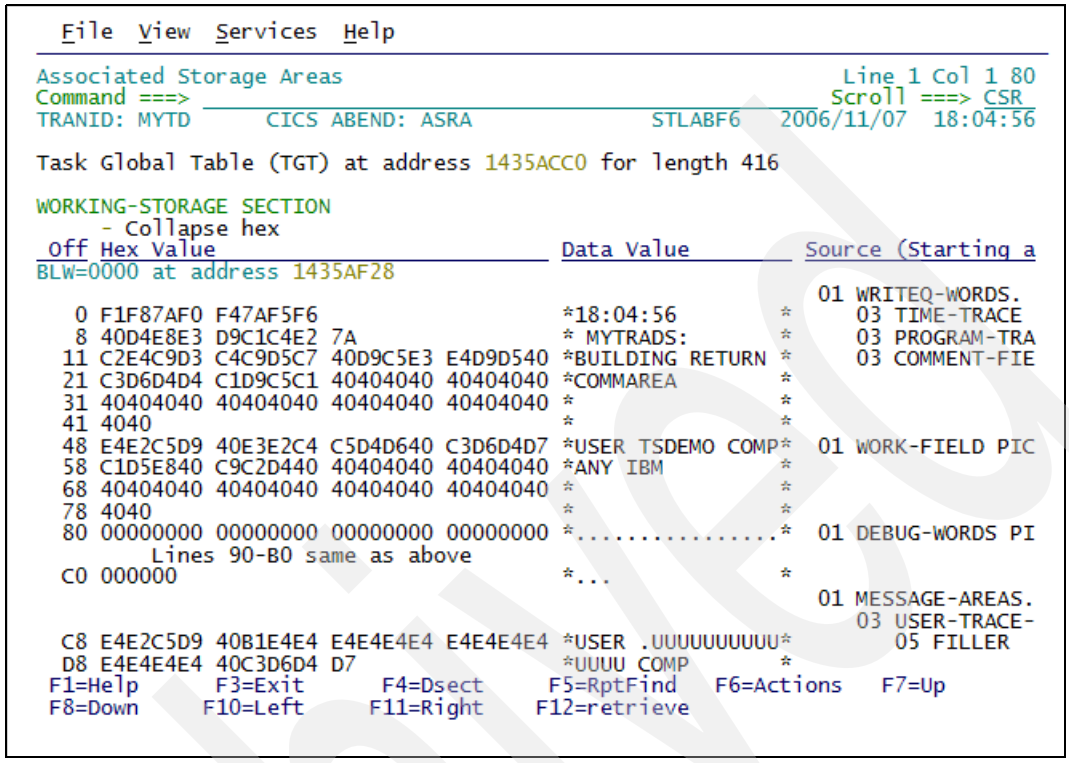


Figure 34-13 Associated storage areas

Scroll down using PF8 or issue the **find** command to get to the area where the current record is located in the working-storage section, as shown in Figure 34-13.

```
  File  View  Services  Help
Associated Storage Areas                                    Line 101 Col 3 82
Command ===>                                                 Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2006/11/07  18:04:56
8D 40404040 40404040                    *                *      03 BUY-SELL-PRIC
95 40404040                             *                *      03 BUY-SELL4
99 40404040 40404040                    *                *      03 BUY-SELL-PRIC
A1 404040                               *                *      03 ALARM-CHANGE
A4 40                                   *                *      03 UPDATE-BUY-SE
A5 40404040 40404040 40404040 40404040 *                 *      03 FILLER
       Lines 3B5-3E5 same as above
F5 40404040 40404040 40404040 404040    *                *
                                                            01 CUSTOMER-IO-BUFF
                                                            03 KEYREC.
08 E3E2C4C5 D4D64040 40404040 40404040 *TSDEMO           *      05 CUST-NM
18 40404040 40404040 40404040 40404040 *                 *
28 40404040 40404040 40404040 40404040 *                 *
38 40404040 40404040 40404040          *                 *
44 4B                                  *.                *      05 KEYREC-DOT
45 C9C2D440 40404040 40404040 40404040 *IBM              *      05 COMP-NM
55 40404040                            *                 *
59 F0F2F0F0                            *0200             *      03 DEC-NO-SHARES
5D 40404040 40404040                   *                 *      03 BUY-FROM
65 40404040                            *                 *      03 BUY-FROM-NO
69 F0F0F1F1 F04BF0F0                   *00110.00         *      03 BUY-TO
71 F0F1F0F0                            *0100             *      03 BUY-TO-NO
75 F0F0F1F8 F04BF0F0                   *00180.00         *      03 SELL-FROM
7D F0F2F0F0                            *0200             *      03 SELL-FROM-NO
81 F0F0F1F7 F04BF0F0                   *00170.00         *      03 SELL-TO
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down      F10=Left     F11=Right    F12=retrieve
```

Figure 34-14 Associated storage areas - CUSTOMER-IO-BUFFER (EBCDIC)

As you can see, the record key is:

- CUST-NM = TSDEMO
- KEYREC-DOT =.
- COMP-NM = IBM

You also can see the field in error DEC-NO-SHARES (highlighted for illustration purposes), has a value of character 0200 i.e. HEX F0F2F0F0. This is an invalid value for a COMP-3 field. This is the reason for the abend.

What we know about the abend so far

Here is what we know:

- The abending program line number is 806.
- Fields involved in abend are DEC-NO-SHARES and DECIMAL-SHARE-VALUE.
- The field in error DEC-NO-SHARES with a value of F0F2F0F0.
- The record that contains the field in error.

Here is what we do *not* know:

- Is the bad data coming from the file that has been read?

  or

- Is it the program that puts the wrong data in the field?

34.2.2 Using Debug Tool to analyze the problem

The first step for debugging the abending program is to define it to Debug Tool using the CICS transaction CADP. As shown in Figure 34-15, define the program MYTRADS and the user ID SIMCOCK to Debug Tool, so it is just when the user ID SIMCOCK runs the program MYTRADS that Debug Tool is involved. All other parameters on the screen are set at default by Debug Tool. After entering the values, press PF2 to update the debug profile.

```
CADP      -    CICS Application Debugging Profile Manager   -     CICSC31F

Create Compiled Debugging Profile ==> ANS       for SIMCOCK


CICS Resources To Debug (use * to specify generic values e.g. *, A*, AB*, etc.)
Transaction      ==> *                                  Applid   ==> CICSC31F
Program          ==> MYTRADS                            Userid   ==> SIMCOCK
Compile Unit     ==> *                                  Termid   ==> *
                                                        Netname  ==> *


Debug Tool Language Environment Options
Test Level       ==> All                                          (All,Error,None)
Command File     ==>
Prompt Level     ==> PROMPT
Preference File  ==>

Other Language Environment Options
==>
==>
==>
==>

Enter=Create PF1=Help 2=Save options as defaults 3=Exit 10=Replace 12=Return
```

Figure 34-15 Debug Tool CICS control: Primary menu

Access the Trader application as detailed in 34.2, “Tracking an abend in the application” on page 1117:

Debug Tool is now activated for program MYTRADS, as shown in Figure 34-16.

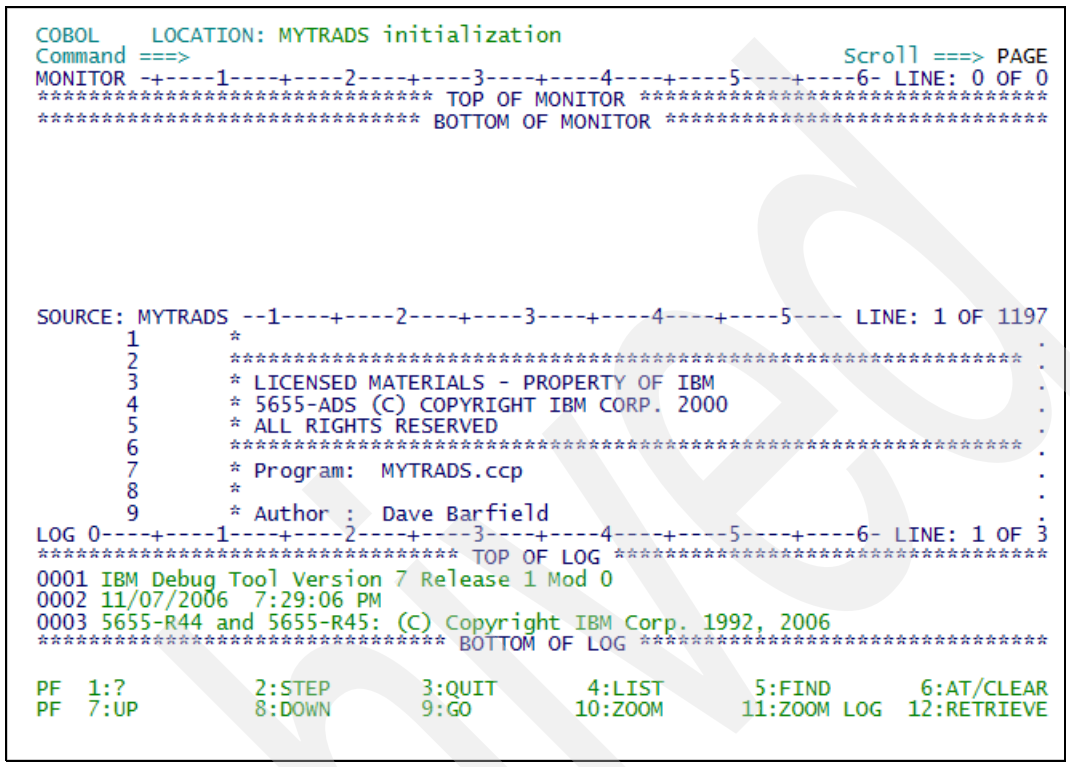


Figure 34-16 DEBUG Tool activation

What you want to see is the value of the variable DEC-NO-SHARES:

- Before the record has been read
- After the record is has been read

To do this, define a pause at 543, as shown in Figure 34-17. This is the pause before reading the record.

```
COBOL    LOCATION: MYTRADS initialization
Command ===>                                                  Scroll ===> PAGE
SOURCE: MYTRADS --1----+----2----+----3----+----4----+----5-- LINE: 541 OF 1197
     541        ************************************************************ .
     542         READ-CUSTFILE SECTION.                                      .
     543             MOVE 'READING RECORD FROM CUSTOMER FILE' TO COMMENT-FIELD .
     544             PERFORM WRITEQ-TS                                       .
     545        * Build record key                                           .
     546             MOVE USERID TO CUST-NM  OF CUSTOMER-IO-BUFFER           .
     547             MOVE '.' TO KEYREC-DOT                                  .
     548             MOVE COMPANY-NAME TO COMP-NM OF CUSTOMER-IO-BUFFER      .
     549        *EXEC CICS READ                                              .
     550        *          FILE('CUSTFILE')                                  .
     551        *          INTO(CUSTOMER-IO-BUFFER)                          .
     552        *          LENGTH(LENGTH OF CUSTOMER-IO-BUFFER)              .
     553        *          RIDFLD(KEYREC)                                    .
     554        *          NOHANDLE                                          .
     555        *          END-EXEC                                          .
     556             Move LENGTH OF CUSTOMER-IO-BUFFER to dfhb0020           .
     557             Call 'DFHEI1' using by content x'0602f0002700008000f0f0f4 .
     558        -    '404040' by content 'CUSTFILE' by reference             .
     559             CUSTOMER-IO-BUFFER by reference dfhb0020 by reference KEY .
     560             end-call                                                .
     561                                                                     .
     562                                                                     .
     563             MOVE 'READ' TO CICS-FUNCTION                            .
     564             PERFORM TRACE-CICS-ERROR                                .
     565             EVALUATE EIBRESP                                        .
     566        * DFHRESP(NORMAL) = 0                      INSERTED BY TRANSL .
     567                  WHEN 0                                             .
PF  1:?       2:STEP      3:QUIT      4:LIST      5:FIND      6:AT/CLEAR
PF  7:UP      8:DOWN      9:GO       10:ZOOM     11:ZOOM LOG  12:RETRIEVE
```

Figure 34-17 Setting break on 543

To list the value of DEC-NO-SHARES in the monitor window, give the command:

MONITOR LIST DEC-NO-SHARES

The results are shown in Figure 34-18

```
COBOL    LOCATION: MYTRADS initialization
Command ===>                                                Scroll ===> PAGE
MONITOR -+----1----+----2----+----3----+----4----+----5----+----6- LINE: 1 OF 2
******************************** TOP OF MONITOR ********************************
                                  ----+----1----+----2----+----3----+----4----
0001  1 DEC-NO-SHARES
0002    The environment is not yet fully initialized.
****************************** BOTTOM OF MONITOR ******************************




SOURCE: MYTRADS --1----+----2----+----3----+----4----+----5--- LINE: 55 OF 1197
      55              05 FILLER  PIC X(25) VALUE 'RRRRRR INVALID           '. .
      56           03 SUB-FUNCTION-NOT-FOUND-MSG.                           .
      57              05 FILLER  PIC X(25) VALUE 'FUNCTION BUY/SELL CALLED '. .
      58              05 FILLER  PIC X(25) VALUE 'WITH AN INVALID SUBCODE  '. .
      59           03 OVERFLOW-MSG.                                         .
      60              05 FILLER  PIC X(25) VALUE 'OVERFLOW WHEN CALCULATING'. .
      61              05 FILLER  PIC X(25) VALUE ' SHARE VALUE             '. .
      62           03 TOO-MANY-SHARES-MSG.                                  .
      63              05 FILLER  PIC X(25) VALUE 'CUSTOMER TRIED TO SELL MO'. .
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6- LINE: 1 OF 6
0001 IBM Debug Tool Version 7 Release 1 Mod 0
0002 11/07/2006  7:29:06 PM
0003 5655-R44 and 5655-R45: (C) Copyright IBM Corp. 1992, 2006
0004  AT 543 ;
0005  MONITOR
0006    LIST DEC-NO-SHARES ;
PF  1:?         2:STEP      3:QUIT      4:LIST      5:FIND      6:AT/CLEAR
PF  7:UP        8:DOWN      9:GO       10:ZOOM     11:ZOOM LOG  12:RETRIEVE
```

Figure 34-18 Monitor of DEC-NO-SHARES

Press PF9 to GO. This instructs Debug Tool to execute the program, which is now executed to line 543 where we set the pause. This is shown in Figure 34-19. The value of DEC-NO-SHARES is shown in the monitor window. As you can see, the value is X'00000000'. This is the value before the record has been read.

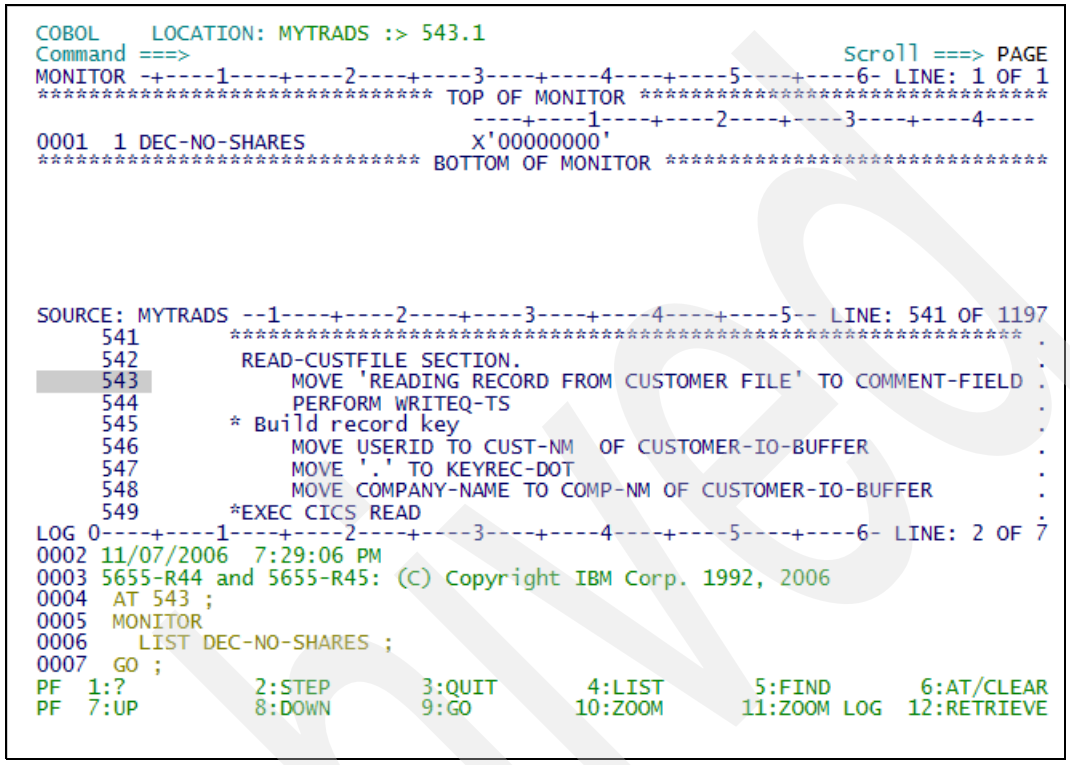


Figure 34-19 DEC-NO-SHARES value at line 543

Set the second pause to get the value of DEC-NO-SHARES after the record has been read. This is done by giving the command AT 563, as shown in Figure 34-20.

```
COBOL    LOCATION: MYTRADS :> 543.1
Command ===>                                                Scroll ===> PAGE
MONITOR -+----1----+----2----+----3----+----4----+----5----+----6- LINE: 1 OF 1
******************************** TOP OF MONITOR ********************************
                                  ----+----1----+----2----+----3----+----4----
0001  1 DEC-NO-SHARES               X'00000000'
****************************** BOTTOM OF MONITOR ********************************




SOURCE: MYTRADS --1----+----2----+----3----+----4----+----5-- LINE: 559 OF 1197
     559            CUSTOMER-IO-BUFFER by reference dfhb0020 by reference KEY .
     560            end-call                                                  .
     561                                                                      .
     562                                                                      .
     563            MOVE 'READ' TO CICS-FUNCTION                              .
     564            PERFORM TRACE-CICS-ERROR                                  .
     565            EVALUATE EIBRESP                                          .
     566         * DFHRESP(NORMAL) = 0                        INSERTED BY TRANSL .
     567              WHEN 0                                                  .
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6- LINE: 3 OF 8
0003 5655-R44 and 5655-R45: (C) Copyright IBM Corp. 1992, 2006
0004  AT 543 ;
0005  MONITOR
0006    LIST DEC-NO-SHARES ;
0007  GO ;
0008  AT 563 ;
PF  1:?         2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP        8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 34-20 Pause at line 563

Press PF9 to GO again. The program executes to this line and you can see the value of DEC-NO-SHARES after the record has been read, as shown in Figure 34-21. The value is X'F0F2F0F0', which is the value you saw in the dump.

In the next section we describe how to use File Manager to fix the incorrect value.

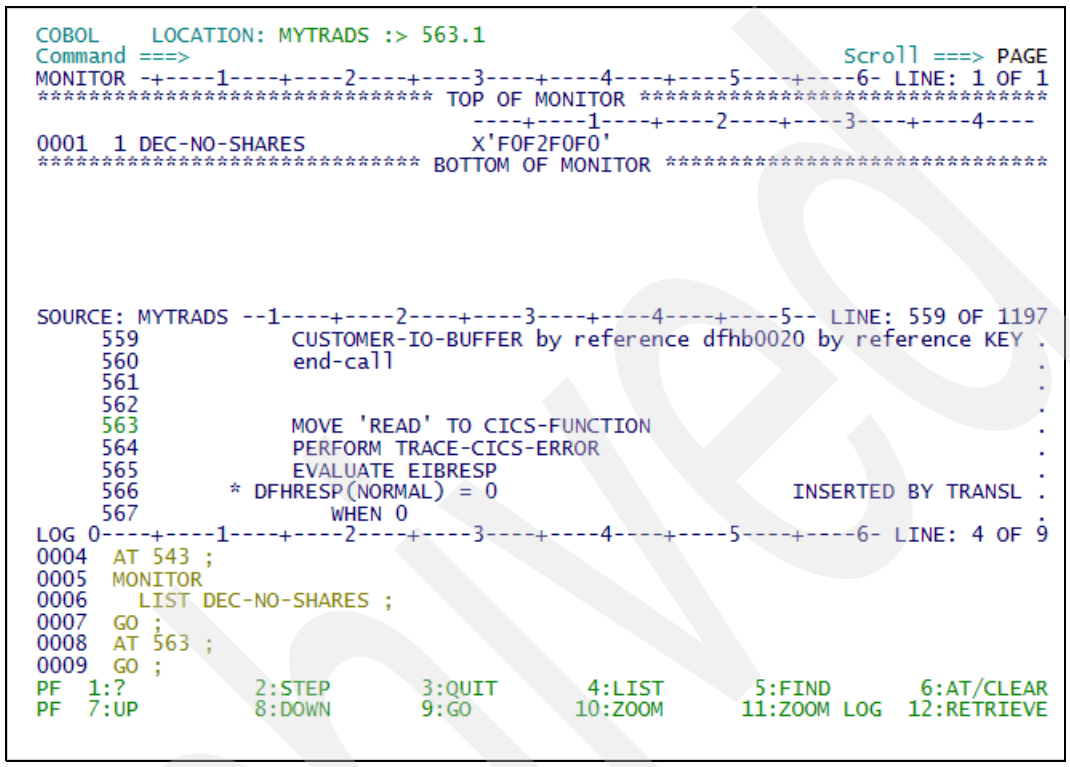


Figure 34-21 DEC-NO-SHARES value at line 563

34.2.3 Using File Manager to correct a problem with data

File Manager is used to correct the invalid data in the application file.

Note: You must close the CUSTFILE in the CICS region before you attempt to edit it. If you do not, File Manager displays the following error message when you edit the file:

```
VSAM OPEN RC X'08', Error Code X'A8
```

Call the CEMT transaction CEMT Set File (CUSTFILE) CLO ENA and press Enter.

Access File Manager in your ISPF session. Select option **2** (Edit) and enter the VSAM data set name, copybook data set name, and starting position TSDEMO, as shown in Figure 34-22. Press Enter.

```
 Process   Options   Help
--------------------------------------------------------------------------------
File Manager                    Edit Entry Panel
Command ===> _____________________________________________________________________

Input Partitioned, Sequential or VSAM Data Set, or HFS file:
   Data set/path name 'CHABERT.C23G.TRADER.CUSTFILE'                    +
   Member . . . . . _________     Blank or pattern for member list
   Volume serial  . . ______      If not cataloged
   Start position . . TSDEMO                          +
   Record limit . . . _________   Record sampling  _
   Inplace edit . . . _           Prevent inserts and deletes
Copybook or Template:
   Data set name  . . 'CHABERT.TRADER.COPYLIB'
   Member . . . . . CUSTFILE__    Blank or pattern for member list
Processing Options:
 Copybook/template   Start position type   Enter "/" to select option
 1  1. Above         _  1. Key             _  Edit template  _ Type (1,2,S)
    2. Previous         2. RBA             _  Include only selected records
    3. None             3. Record number   _  Binary mode, reclen _____
    4. Create dynamic                      _  Create audit trail

 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forwar
 F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 34-22 File Manager - Edit Entry Panel

The panel shown in Figure 34-23 appears, displaying records containing TSDEMO as the value of the CUST-NM.

```
Process   Options   Help
--------------------------------------------------------------------------------
dit             CHABERT.C23G.TRADER.CUSTFILE                    Rec 0 of 4
ommand ===> _____________________________________________________ Scroll PAG
    Refresh on save N__                          Type KSDS      Format TAB
     CUST-NM                                                    KEYREC-DO
     Ä3                                                         Ä4
     AN 1:60                                                    AN 61:1
     <---+----1----+----2----+----3----+----4----+----5----+----> -
00000 ****  Key used to set start point  ****
00001 TSDEMO                                                    .
00002 TSDEMO                                                    .
00003 TSDEMO                                                    .
00004 TSDEMO                                                    .
00005 ****  End of data  ****

F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 34-23 File Manager: CUSTFILE (edit 1)

Press PF11 to scroll to the right, finding the invalid data. File Manager displays the invalid data as a string of highlighted asterisks, as shown in Figure 34-24.

```
 Process   Options   Help
--------------------------------------------------------------------------
Edit            CHABERT.C23G.TRADER.CUSTFILE                  Rec 0 of 4
Command ===> _____________________________________________________ Scroll P
     Refresh on save N                                 Type KSDS     Format T
       COMP-NM              DEC-NO-SHARES BUY-FROM BUY-FROM-NO BUY-TO   BUY
       Ã5                          Ã6 Ã7       Ã8           Ã9       Ã10
       AN 62:20                 PD 82:4 AN 86:8  AN 94:4     AN 98:8  AN
       <---+----1----+---->     <---+--> <---+--> <-->        <---+--> <--
000000 ****  Key used to set start point  ****
000001 Casey_Import_Export      ******** 00080.00 0100         00050.00 015
000002 IBM                      ********                       00110.00 010
000003 ShareSelect              ********
000004 SportSelect              ********
000005 ****  End of data  ****




 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChang
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 34-24 File Manager: CUSTFILE (edit 2)

Place the cursor on the record containing the invalid data (remember that the record had the key TSDEMO.IBM) and press PF2 to display the record in SNGL format, as shown in Figure 34-25. You could also change the record in Figure 34-25. The last few steps are to show the action being performed in SNGL mode.

```
 Process   Options   Help
--------------------------------------------------------------------------
Edit            CHABERT.C23G.TRADER.CUSTFILE                  Rec 1 of 4
Command ===> _____________________________________________________ Scroll P
                                          Type KSDS     Zoom Format S
                                               Top Line is 6    of 16
Current 01: FILEMANAGER-DEFAULT-01
Field                          Data
  3 COMP-NM                    Casey_Import_Export
  2 DEC-NO-SHARES              ********
  2 BUY-FROM                   00080.00
  2 BUY-FROM-NO                0100
  2 BUY-TO                     00050.00
  2 BUY-TO-NO                  0150
  2 SELL-FROM                  00098.00
  2 SELL-FROM-NO               0200
  2 SELL-TO                    00045.00
  2 SELL-TO-NO                 0100
  2 ALARM-PERCENT              016
***  End of record  ***


 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChang
 F7=Up        F8=Down      F9=Swap     F12=Cancel
```

Figure 34-25 File Manager: CUSTFILE (edit 3)

Type the correct value in the field DEC-NO-SHARES and save the record, as shown in Figure 34-26.

```
 Process   Options   Help
 ------------------------------------------------------------------------
Edit            CHABERT.C23G.TRADER.CUSTFILE                  Rec 1 of 4
Command ===> _____________________________________________________ Scroll P
                                                Type KSDS     Zoom Format S
                                                     Top Line is 6    of 16
Current 01: FILEMANAGER-DEFAULT-01
Field                           Data
  3 COMP-NM                     Casey_Import_Export
 2 DEC-NO-SHARES                    1232
 2 BUY-FROM                     00080.00
 2 BUY-FROM-NO                  0100
 2 BUY-TO                       00050.00
 2 BUY-TO-NO                    0150
 2 SELL-FROM                    00098.00
 2 SELL-FROM-NO                 0200
 2 SELL-TO                      00045.00
 2 SELL-TO-NO                   0100
 2 ALARM-PERCENT                016
***  End of record  ***


 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChang
 F7=Up        F8=Down      F9=Swap     F12=Cancel
```

Figure 34-26 File Manager: CUSTFILE (edit 4)

Note: Once any data in the record is changed, all of the fields associated with the record are highlighted.

34.2.4 Running the application after the fix

You have finished correcting the invalid data in the CUSTFILE. Now access the Trader application in CICS and obtain a real-time quote for IBM for customer TSDEMO.

34.3 Summary of scenario 1

In this chapter we described the various components that make up the CICS environment on our system and how they are set up. We reviewed the processing performed by the CICS Trader application. We detailed a process that:

- Used Debug Tool to follow the execution of the application
- Used Fault Analyzer to identify the cause of an abend in the application
- Described File Manager's capability to identify and correct the data that caused the problem

35

Scenario 2: Using Debug Tool

In this chapter we describe the application components that exist in the batch environment in our system and show how they are set up.

We explain the processing that is performed in the batch Trader application.

The application is forced to produce incorrect output, and then we describe the steps required to identify the logic error in the application using Debug Tool in batch mode. We also define how to step through the program to isolate and correct the problem using Debug Tool in foreground mode.

35.1 Setting up the components

Two types of components have to be established for this scenario:

- Batch components
- Program products
  - Debug Tool

35.1.1 Batch components

Components used by the Trader application are listed in Figure 35-1.

Table 35-1 Components of the Trader application for the batch scenario

Component	Details	Remarks
Program	TRADERB	Batch COBOL program
JCL	TRADER	JCL to run the batch application
Files	GRACINE.TRADER.CUSTFILE GRACINE.TRADER.COMPFILE GRACINE.TRADER.TRANFILE	VSAM files and sequential transaction file used by the application
Copybooks	CUSTFILE COMPFILE TRANFILE	File definition for Customer file, company file and transaction file.

35.1.2 Program products

To use the Problem Determination Tools with this scenario, make sure you have the following output or supporting files for the following product:

- **Debug Tool**

You must have a compiler listing or side file for the program TRADERB.

If you are not using the supplied JCL to compile these programs, make sure you specify the following compiler options:

```
LIST, XREF,MAP,RENT,TEST
```

If you prefer to use a side file instead of a compiler listing, include the SEPARATE suboption of the TEST compiler option. Remember that the side file required by Debug Tool is different from the one required by Fault Analyzer. Make sure that you run the DEFVSAM1 batch job to load the VSAM files.

35.2 Walkthrough of the batch Trader application

The batch Trader application is used to maintain stock portfolios held by individuals. You execute a batch job that processes a day's worth of trading activity, which:

- Lists portfolios and their value
- Buys shares of a company's stock
- Sells shares of a company's stock

Note: This example was designed to demonstrate the capabilities of the Problem Determination Tools. Therefore, a minimal amount of code was developed. This application does *not* represent real-world securities processing.

35.2.1 Trader batch job

The JCL to run the batch Trader application is shown in Figure 35-1.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
------------------------------------------------------------------------------
EDIT       GRACINE.TRADER.JCL(TRADERD1) - 01.08            Columns 00001 00072
Command ===> _                                              Scroll ===> PAGE
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000001 //TRADERD1 JOB ,CLASS=A,NOTIFY=&SYSUID,MSGCLASS=H,MSGLEVEL=(1,1),
000002 //          REGION=0M
000003 //GO        EXEC PGM=TRADERB,
000004 //          PARM='/TEST(,INSPIN,,)'
000005 //STEPLIB   DD DISP=SHR,DSN=GRACINE.TRADER.LOAD
000006 //          DD DISP=SHR,DSN=ADTOOLS.DT610.SEQAMOD
000007 //SYSPRINT DD SYSOUT=*
000008 //SYSABEND DD SYSOUT=*
000009 //COMPFILE DD DISP=SHR,DSN=GRACINE.TRADER.COMPFILE
000010 //CUSTFILE DD DISP=SHR,DSN=GRACINE.TRADER.CUSTFILE
000011 //TRANSACT DD DISP=SHR,DSN=GRACINE.TRADER.SAMPLES(TRANFILE)
000012 //INSPREF  DD DISP=SHR,DSN=GRACINE.TRADER.INSPREF(PREF01)
000013 //REPOUT DD SYSOUT=*
000014 //TRANREP DD SYSOUT=*
000015 //INSPIN DD DISP=SHR,DSN=GRACINE.TRADER.CMDS(CMD01)
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 35-1 JCL to run the batch Trader application

The job invokes program TRADERB, which reads a member of a PDS file with DDname of TRANSACT, to obtain the day's transactions. The program processes each of the records in this file. At the same time, the program reads the company file (COMPFILE) and reads and updates the customer file (CUSTFILE).

After the program processes the input file, it generates two output reports: REPOUT, which contains a list of all customer portfolios, and TRANREP, which contains a detailed list of the transaction activity and processing status.

The transaction file

The transaction file is an 80-byte, sequential file that is input to the Trader application as shown in Figure 35-2. It can contain three types of requests:

- List shares
- Buy shares
- Sell shares

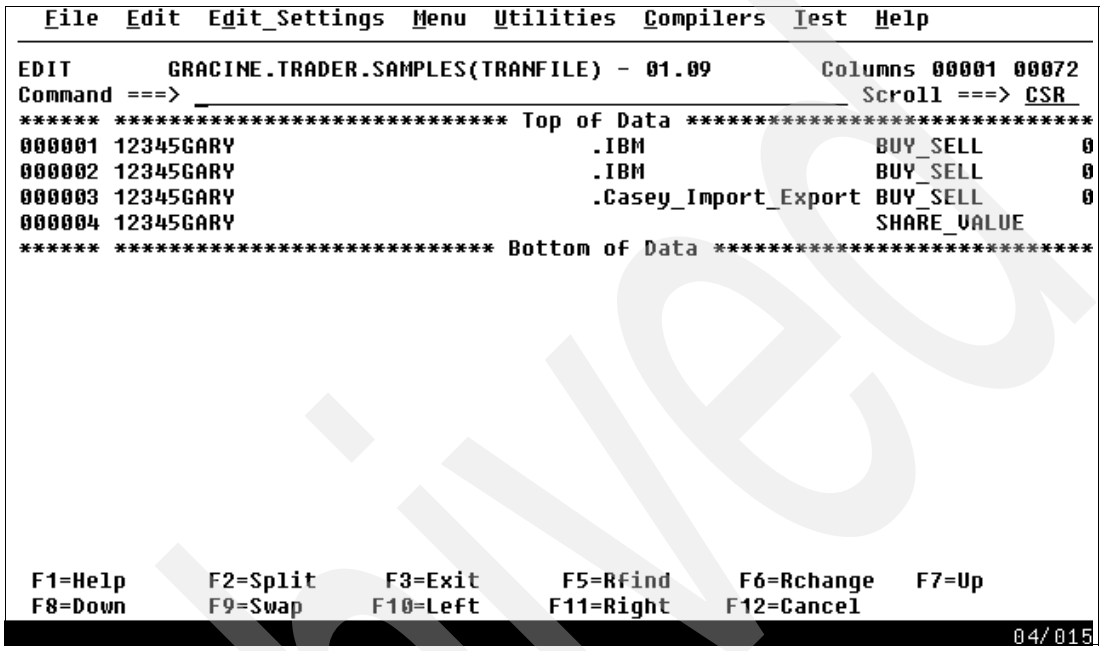


Figure 35-2 Batch trader application transaction file (1 of 2)

Figure 35-3 shows the second part of the screen display; you can press the F11 key to move it to the right of the screen.

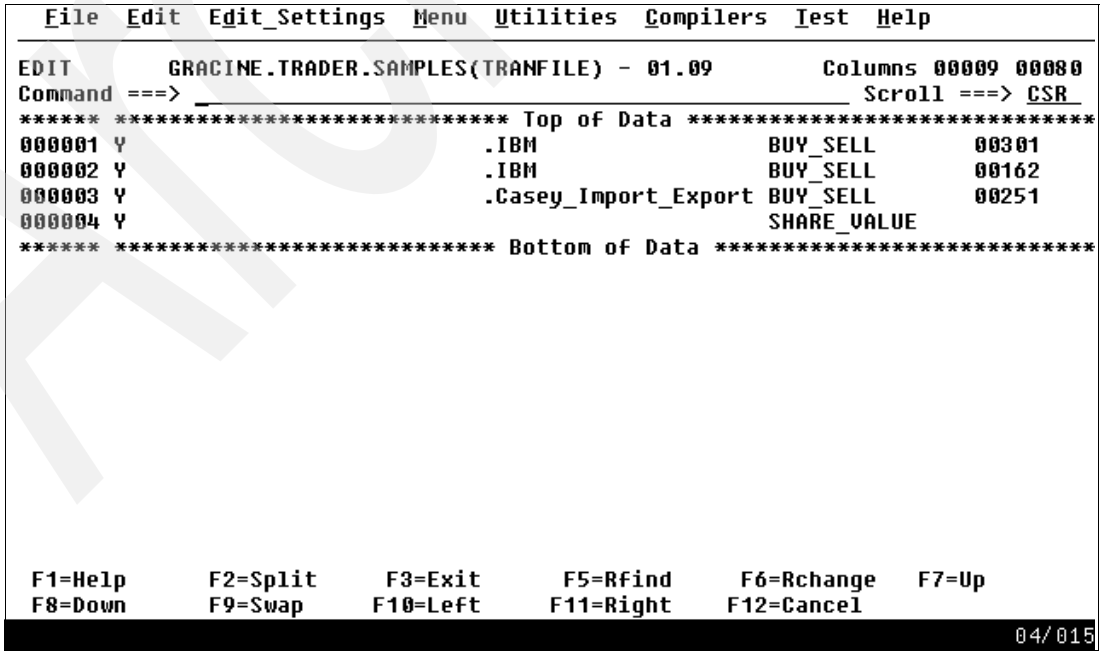


Figure 35-3 Batch trader application transaction file (2 of 2)

The record layout for the transaction file is listed in Figure 35-2.

Table 35-2 Transaction file record layout

Column	Description	Field Name
1-5	Account Number	TR-ACCOUNT-NUMBER
6-35	Customer Name	TR-CUSTOMER-NAME
36	Dot	FILLER
37-51	Company Name	TR-COMPANY-NAME
52-71	Request Type	TR-REQUEST-TYPE
71-75	Number of shares (buy or sell)	TR-NO-OF-SHARES
76	Transaction Type (buy or sell)	TR-SUBTYPE
77-80	Blank	FILLER

After the program TRADERB reads each record, it examines the TR-REQUEST-TYPE field to determine the type of processing to perform.

35.2.2 Listing shares

If the field TR-REQUEST-TYPE contains the value SHARE_VALUE, the program uses the value in the TR-CUSTOMER-NAME field to print a report that lists the shares held by that customer in each company the customer owns, as shown in Figure 35-4.

```
  Display  Filter  View  Print  Options  Help
 -------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY TRADERD1 JOB04776  DSID   103 LINE 1       COLUMNS 02- 81
 COMMAND INPUT ===> ■                                         SCROLL ===> CSR
CUSTOMER : GARY                                              11/18/2005
-----------------------------------------------------------------
    COMPANY                SHARES    SHARE            TOTAL
                           HELD      VALUE            COST
-----------------------------------------------------------------
    Casey_Import_Export     504       79.00        39,816.00
    IBM                     762      163.00       124,206.00
-------------------------------------------------------------------------------
CUSTOMER                        COMPANY               QTY  REQ-TYP  STATUS
-------------------------------------------------------------------------------
GARY                            IBM                    30  BUY      PROCESSE
GARY                            IBM                    16  SELL     PROCESSE
GARY                            Casey_Import_Export    25  BUY      PROCESSE
****************************** BOTTOM OF DATA ********************************



  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=IFIND     F6=BOOK
  F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 35-4 Batch trader application list shares report

35.2.3 Buying and selling shares

If field TR-REQUEST-TYPE contains the value BUY_SELL, and field TR-SUB-TYPE contains a value of 1, the program processes a request to buy the number of shares in TR-NO-OF-SHARES.

If field TR-REQUEST-TYPE contains the value BUY_SELL, and field TR-SUB-TYPE contains a value of 2, the program processes a request to sell the number of shares in TR-NO-OF-SHARES.

After the process completes successfully, the program updates the Customer file, GRACINE.TRADER.CUSTFILE.

The program also produces a transaction report, as shown in Figure 35-5. This report lists the transaction file input request and the status of the processing. The STATUS column in the report lists how the request was processed. If the processing is successful, the message PROCESSED is printed, otherwise the message *ERROR* is printed.

```
CUSTOMER                          COMPANY              QTY  REQ-TYP  STATUS
GARY                              IBM                   30  BUY      PROCESS
GARY                              IBM                   16  SELL     PROCESS
GARY                              Casey_Import_Export   25  BUY      PROCESS
******************************** BOTTOM OF DATA ********************************
```

Figure 35-5 Batch Trader application Transaction report listing BUY / SELL shares (1 of 2)

The second part of the screen display is shown in Figure 35-6; press the F11 key to scroll to the right of the screen.

```
  Display  Filter  View  Print  Options  Help
SDSF OUTPUT DISPLAY TRADERD1 JOB04776  DSID   103 LINE 1       COLUMNS 55- 134
COMMAND INPUT ===> ■                                        SCROLL ===> CSR
       11/18/2005
-----------
    TOTAL
    COST
-----------
 39,816.00
124,206.00
---------------------------
    QTY  REQ-TYP  STATUS
---------------------------
     30  BUY      PROCESSED SUCCESFULLY
     16  SELL     PROCESSED SUCCESFULLY
     25  BUY      PROCESSED SUCCESFULLY
******************************** BOTTOM OF DATA ********************************
```

Figure 35-6 Batch Trader application transaction report listing BUY / SELL shares (2 of 2)

To demonstrate the capabilities of the Problem Determination Tools, next we describe how to force the application to encounter an error. Consider that your business user, Joe, tells you about a problem with the output contained in one of the reports. You step through the process of fixing it, and use Debug Tool (in batch and foreground mode) to first identify and then to isolate a problem in the application program logic.

In this example, you have a Transaction file that contains the day's trading activity for the customer, RB_DEMO:

- Buy 30 shares of IBM.
- Sell 16 shares of IBM.
- Buy 25 shares of Casey Import/Export.
- List the shares held by Gary.

This activity is represented by the records shown in Figure 35-7.

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
 BROWSE    GRACINE.TRADER.SAMPLES(TRANFILE) - 01.09   Line 00000000 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
********************************* Top of Data ********************************
12345GARY                          .IBM                 BUY_SELL       00301
12345GARY                          .IBM                 BUY_SELL       00162
12345GARY                          .Casey_Import_Export BUY_SELL       00251
12345GARY                                               SHARE_VALUE
******************************** Bottom of Data ******************************
```

Figure 35-7 Transaction record for batch scenario

You submit the batch job, TRADER. The TRADERB application program reads the input from the Transaction file and processes the requests. The results of the transaction processing is printed as a report, as shown in Figure 35-8.

```
-------------------------------------------------------------------------------
CUSTOMER                          COMPANY               QTY  REQ-TYP  STATUS
-------------------------------------------------------------------------------
GARY                              IBM                    30  BUY      PROCESSE
GARY                              IBM                    16  SELL     PROCESSE
GARY                              Casey_Import_Export    25  BUY      PROCESSE
****************************** BOTTOM OF DATA ********************************
```

Figure 35-8 TRANREP report from batch job TRADERB (1 of 2)

Use the PF11 key to scroll to the right of the screen for more information as shown in Figure 35-9.

```
---------------------------
    QTY  REQ-TYP  STATUS
---------------------------
     30  BUY      PROCESSED SUCCESFULLY
     16  SELL     PROCESSED SUCCESFULLY
t    25  BUY      PROCESSED SUCCESFULLY
****************************** BOTTOM OF DATA ********************************
```

Figure 35-9 TRANREP report from batch job TRADERB (2 of 2)

Your business user, Joe, who reviews these reports on a daily basis, tells you there is an error. He shows you the report from September 30th. It only lists the shares held by the customer Gary in the company Casey_Import_Export, which does not reconcile with the account.

You check the Transaction Report in Figure 35-8 and sure enough, it shows that the buy/sell requests for IBM and Casey_Import_Export were processed successfully. To make sure, you access the CICS Trader application to review Gary's account. The shares for both of these companies are listed.

You can see there is a problem printing all of the shares held by a customer. You know from experience that it has something to do with the program logic, because the buy requests have been processed successfully, and two new records have been written to the customer file for Gary.

You decide to investigate further and use Debug Tool.

35.2.4 Using Debug Tool in batch mode to try to find the error

You are going to use Debug Tool to show you the flow of the program so that you can find out where the program is experiencing the problem. You want to list the paragraphs that are performed when the job executes. To do this, create a commands file for Debug Tool commands, and instruct Debug Tool to use this file at the start of the debug session.

Setting up the commands file

For this example, create a commands file. This can be any fixed-block, 80-byte sequential file, or a member of a partitioned data set (PDS). Figure 35-10 contains the commands to list the paragraphs that are performed when the program executes.

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
 BROWSE    GRACINE.TRADER.CMDS(CMD01) - 01.09         Line 00000000 Col 001 080
 Command ===> ____________________________________________ Scroll ===> CSR
 ******************************** Top of Data ********************************
 AT GLOBAL LABEL PERFORM;
       LIST LINES %LINE;
       GO;
       END-PERFORM;
       GO;
       QUIT;
 ******************************** Bottom of Data *****************************









  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 35-10 Debug Tool commands to list paragraph names

This routine requests a listing of the line number and name of each paragraph (label) in the program.

Running Debug Tool in batch mode

You can also create a batch job to invoke Debug Tool to debug your program. The fastest way to do this is to modify the Trader batch job. Then use the JCL shown in Figure 35-11.

Include the TEST runtime option and point to your Commands file. The output from the Commands file is directed to the JES spool (although it could also go to a sequential file).

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
 BROWSE    SE65273.DEBUG.TESTDT.JCL(TRADJCL1) - 01.00 Line 00000000 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
******************************** Top of Data ********************************
//TRADERDB JOB ,CLASS=A,NOTIFY=&SYSUID,MSGCLASS=H,MSGLEVEL=(1,1),
//         REGION=0M
// EXEC PGM=TRADERB,
// PARM='/TEST(,INSPIN,,)'
//COMPFILE DD DSN=GRACINE.TRADER.COMPFILE,DISP=(SHR,,)
//CUSTFILE DD DSN=GRACINE.TRADER.CUSTFILE,DISP=(SHR,,)
//INSPIN DD DSN=GRACINE.TRADER.CMDS(CMD01),DISP=(SHR,,)
//INSPLOG DD SYSOUT=*
//INSPREF DD DSN=GRACINE.TRADER.INSPREF(PREF01),DISP=(SHR,,)
//REPOUT DD SYSOUT=*
//STEPLIB DD DSN=CHABERT.BOOK2005.LOAD,DISP=(SHR,,)
// DD DSN=ADTOOLS.DT610.SEQAMOD,DISP=(SHR,,)
//SYSABEND DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//TRANREP DD SYSOUT=*
//TRANSACT DD DSN=GRACINE.TRADER.SAMPLES(TRANFILE),DISP=(SHR,,)
****************************** Bottom of Data ******************************
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 35-11 Batch job to run Debug Tool for quick problem identification

Make the following changes to the JCL:

- Add a comma after the program name.
- Include a parameter that specifies overrides to runtime options, and include TEST and your commands file.
- Add the load library for Debug Tool to the STEPLIB concatenation (if it is in LINKLIST).
- Add the DD statement INSPIN and use your commands file.
- Add the DD statement INSPLOG and use the JES spool for the log file.

Submit this job. After the batch job completes, review the output of the log file.

Contents of the log file

Debug Tool lists each of the line numbers and paragraph names in the log file, as shown in Figure 35-12 and Figure 35-13.

```
  Display  Filter  View  Print  Options  Help
--------------------------------------------------------------------------------
SDSF OUTPUT DISPLAY TRADERDB JOB04988  DSID   101 LINE 5       COLUMNS 02- 81
COMMAND INPUT ===> _                                          SCROLL ===> CSR
     *          not exist, or is not accessible.
     * *** Commands file commands follow ***
       AT GLOBAL LABEL
         PERFORM
           LIST LINES %LINE ;
           GO ;
         END-PERFORM ;
       GO ;
     *      326       MAINLINE SECTION.
     *
     *      348       MAINLINE-EXIT.
     *
     *      380       SETUP-FILES-EXIT.
     *
     *      412       CLOSEDOWN-FILES-EXIT.
     *
     *      446       BUY-SELL SECTION.
     *
 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=IFIND     F6=BOOK
 F7=UP        F8=DOWN      F9=SWAP      F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 35-12 Log file listing paragraphs performed during program execution (Part 1 of 2)

Scroll down using the PF8 key to display the rest of the log file as shown in Figure 35-13.

```
  Display  Filter  View  Print  Options  Help
--------------------------------------------------------------------------------
SDSF OUTPUT DISPLAY TRADERDB JOB04988  DSID   101 LINE 23      COLUMNS 02- 81
COMMAND INPUT ===> _                                          SCROLL ===> CSR
     *      553       VALIDATE-COMPANY-EXISTS SECTION.
     *
     *      650       READ-COMPFILE SECTION.
     *
     *      669       READ-COMPFILE-EXIT.
     *
     *      557       VALIDATE-COMPANY-EXISTS-EXIT.
     *
     *      466       BUY-SELL-BUY-FUNCTION SECTION.
     *
     *      576       GET-SHARE-VALUE-EXIT.
     *
     *      601       READ-CUSTFILE-EXIT.
     *
     *      495       CALCULATE-SHARES-BOUGHT SECTION.
     *
     *      509       CALCULATE-SHARES-BOUGHT-EXIT.
     *
 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=IFIND     F6=BOOK
 F7=UP        F8=DOWN      F9=SWAP      F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 35-13 Log file listing paragraphs performed during program execution (Part 2 of 2)

Review the program's processing along with the log file

Review what TRADERB is designed to do to try and isolate the problem. Recall that the customer file has one record for every company in which the customer holds shares. When a transaction to list shares is processed, the program starts to read the customer file. It reads the records one at a time and prints the details, until the record of a different customer is read.

Review the Transaction file and see the two transactions. You realize that it does not matter if Gary had no shares in IBM and Casey_Import_Export before the Trader batch job executed, because two records were written to the customer file when the program processed these records. One was for IBM and another was for Casey_Import_Export.

Recognize that when TRADERB processes the record in the transaction file to list the shares held by Gary, the paragraph READ-CUSTFILE-NEXT should be executed at least four times (one read past the current Customer record).

Look carefully at the Log file, which shows that READ-CUSTFILE-NEXT is only executed twice. This shows you there is a problem with the logic in the section of the program that reads the Customer file.

35.2.5 Using Debug Tool in DTSU to pinpoint the solution

Here we discuss a solution using Debug Tool.

Setting up a DTSU session

For this example, run the TRADERB application from DTSU.

Start your DTSU session using the following steps:

1. Select **DTU** from ISPF option (Debug Tool Utilities) or enter the command **`exec 'your_hlq.seqaexec(eqastart)'`** from ISPF option 6.
2. Select option **2** (Manage and Use Debug Tool Setup Files).
3. In Figure 35-14, enter a member name, like TRADERB. (This implies that you have used DTSU before, and your control file was allocated before). Do not insert a slash just before initialize New setup file for DB2 (/), since this is not a DB2 application.

```
------------------- Debug Tool Foreground - Edit Setup File ------------------
Command ===> ____________________________________________________________

Setup File Library:
   Project . . . SE65273
   Group . . . . TEST     . . . ________ . . . ________ . . . ________
   Type  . . . . DTSF
   Member  . . . TRADERB          (Blank or pattern for member selection list)
                                  (or existing or new member name)

Other Data Set Name:
   Data Set Name . . . ____________________________________________________
   Volume Serial . . . ______     (If not cataloged)

_ Initialize New setup file for DB2 (/)




 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 35-14 DTSU edit setup, file 1

4. We now copy an existing JCL as shown in Figure 35-15. We enter the COPY command at the command line and press Enter.

```
EDIT - Edit Setup File 'SE65273.TEST.DTSF(TRADERB)'              Row 1 to 1 of 1
Command ===> COPY_____________________________________________ Scroll ===> PAGE

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name _________
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs / OS/VS COBOL - Run-time Options  / Program Arguments

_ Enter / to modify parameters ____________________________________________________
_____________________________________________________________________________

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ****************** Top of Data ********************* _____
******************************** Bottom of data ********************************



 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 35-15 COPY command

5. On the next panel, as shown in Figure 35-16, we make the selection of which JCL member to use. We press Enter to get a selection list.

```
------------- Debug Tool Foreground - Copy from Setup File or JCL ------------
Command ===> ____________________________________________________________

Select data to copy into 'SE65273.TEST.DTSF(TRADERB)'

Setup File or JCL Library:
   Project . . . GRACINE
   Group . . . . TRADER  . . . ________ . . . ________ . . . ________
   Type  . . . . JCL
   Member  . . . _________       (Blank or pattern for member selection list)
                                 (or existing or new member name)

Other Data Set Name:
   Data Set Name . . . ____________________________________________________
   Volume Serial . . . ______    (If not cataloged)


Note: When you copy from another setup file the entire contents are copied.
      When copying from JCL you can select the information you want to copy.



 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 35-16 Copy panel

After pressing Enter, a selection panel displays as shown in Figure 35-17. As shown, we select member TRADERD1.

```
  Menu  Functions  Confirm  Utilities  Help
 ------------------------------------------------------------------------------
 LIBRARY          GRACINE.TRADER.JCL                        Row 00052 of 00065
 Command ===> ____________________________________________ Scroll ===> PAGE
    Name     Prompt          Size    Created          Changed             ID
 _ IDISAMP1                   51   2001/06/11   2002/10/17 14:55:40    DAVINR1
 _ IDISAMP2                   20   2001/06/11   2002/10/17 14:55:51    DAVINR1
 _ IMSCOBBA                   77   2002/10/03   2003/11/19 15:25:00    ERIC
 _ IMSCXCOB                   99   2002/10/07   2004/09/01 12:16:06    ERIC
 _ IMSGO                      51   2002/10/03   2002/10/17 10:11:51    DAVINR1
 _ PDTCOPY                    27   2001/07/18   2001/07/31 21:58:25    DAVIN7
 _ RUNSTAT                    15   2004/10/20   2004/10/20 15:39:41    CHABERT
 _ TABLES                     48   1999/08/09   2004/11/04 16:32:35    CHABERT
 _ TRADERB                    13   2001/06/21   2005/11/18 14:05:09    SE65273
 _ TRADERBS                   20   2001/06/21   2004/11/22 19:02:00    CHABER2
 _ TRADERD                    23   2002/09/25   2004/11/22 19:02:05    CHABER2
 S TRADERD1 _                 19   2004/11/26   2005/11/18 15:17:54    SE65273
 _ TRADERI                    17   2002/10/07   2004/11/22 19:02:09    CHABER2
 _ TRANFILE                    3   2004/11/21   2004/11/26 10:04:33    GRACINE
   **End**


  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 35-17 Selection

As shown in Figure 35-18, the JCL statement has been parsed, and by entering **S*** (select all) we select all JCL statements to use.

```
 ---------------- Debug Tool Foreground - Copy from JCL Datas Row 1 to 14 of 16
 Command ===> S*_____________________________________________ Scroll ===> PAGE

 Enter S* on the command line or on a Sel line to select all JCL statements.
 Enter S on a Sel line to select that JCL statement.
 Enter RESET to deselect all JCL statements.

 Sel   JCL Image
 ____ //TRADERD1 JOB
 ____ //* from 'GRACINE.TRADER.JCL (TRADERD1)'
 ____ //GO EXEC PGM=TRADERB,
 ____ //  PARM='/TEST(,INSPIN,,)'
 ____ //STEPLIB  DD  DISP=SHR,DSN=GRACINE.TRADER.LOAD
 ____ //         DD  DISP=SHR,DSN=ADTOOLS.DT610.SEQAMOD
 ____ //SYSPRINT DD  SYSOUT=*
 ____ //SYSABEND DD  SYSOUT=*
 ____ //COMPFILE DD  DISP=SHR,DSN=GRACINE.TRADER.COMPFILE
 ____ //CUSTFILE DD  DISP=SHR,DSN=GRACINE.TRADER.CUSTFILE
 ____ //TRANSACT DD  DISP=SHR,DSN=GRACINE.TRADER.SAMPLES(TRANFILE)
 ____ //INSPREF  DD  DISP=SHR,DSN=GRACINE.TRADER.INSPREF(PREF01)
 ____ //REPOUT   DD  SYSOUT=*
 ____ //TRANREP  DD  SYSOUT=*
  F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
 F12=Cancel
```

Figure 35-18 Select all

All JCL statements are now selected, as shown in Figure 35-19.

```
---------------- Debug Tool Foreground - Copy from JCL Datas Row 1 to 14 of 16
Command ===> _____________________________________________ Scroll ===> PAGE

Enter S* on the command line or on a Sel line to select all JCL statements.
Enter S on a Sel line to select that JCL statement.
Enter RESET to deselect all JCL statements.

Sel   JCL Image
S___ //TRADERD1 JOB
____ //* from 'GRACINE.TRADER.JCL (TRADERD1)'
S___ //GO EXEC PGM=TRADERB,
____ //  PARM='/TEST(,INSPIN,,)'
S___ //STEPLIB  DD  DISP=SHR,DSN=GRACINE.TRADER.LOAD
S___ //         DD  DISP=SHR,DSN=ADTOOLS.DT610.SEQAMOD
S___ //SYSPRINT DD  SYSOUT=*
S___ //SYSABEND DD  SYSOUT=*
S___ //COMPFILE DD  DISP=SHR,DSN=GRACINE.TRADER.COMPFILE
S___ //CUSTFILE DD  DISP=SHR,DSN=GRACINE.TRADER.CUSTFILE
S___ //TRANSACT DD  DISP=SHR,DSN=GRACINE.TRADER.SAMPLES(TRANFILE)
S___ //INSPREF  DD  DISP=SHR,DSN=GRACINE.TRADER.INSPREF(PREF01)
S___ //REPOUT   DD  SYSOUT=*
S___ //TRANREP  DD  SYSOUT=*
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 35-19 Selection finished

We now set up the test parameters, which we do by entering a **/** in front of the text, as shown in Figure 35-20.

```
EDIT - Edit Setup File 'SE65273.TEST.DTSF(TRADERB)'          Row 1 to 5 of 13
Command ===> _____________________________________________ Scroll ===> PAGE

Modify information and use the  Run command to execute,
       or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name TRADERB__
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs / OS/VS COBOL - Run-time Options  / Program Arguments

/ Enter / to modify parameters '/TEST(,INSPIN,,)'_____________________________

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ****************** Top of Data ********************  _____
___ COMPFILE 1   _ 'GRACINE.TRADER.COMPFILE'                          SHR
___ CUSTFILE 1   _ 'GRACINE.TRADER.CUSTFILE'                          SHR
___ INSPIN   1   _ 'GRACINE.TRADER.CMDS(CMD01)'                       SHR
___ INSPLOG  1   _ SYSOUT=*                                           _____
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 35-20 Enter a / in front of the text

After entering a **/** and pressing Enter, the Modify Parameter String panel displays as shown in Figure 35-21. As we are debugging in full-screen mode, we enter a **/** as shown.

```
--------------- Debug Tool Foreground - Modify Parameter String --------------
Command ===> _____________________________________________________________________
                                                                        More:     +
Modify Test options, other run-time options, and Program arguments

Select Test Options:

Test Option     ==> TEST            Test/Notest
Test Level      ==> ALL             All/Error/None
Commands File                       *, DDname, or Data Set Name
                ==> INSPIN
Prompt Level    ==> PROMPT          Prompt/NoPrompt
Preference File                     *, DDname, or Data Set Name
                ==> INSPPREF

Select (/) a session type and provide parameters:

 / Full-screen mode
   Terminal LU     ==> _____________ blank or MFI VTAM Terminal LU

 _ Remote debug mode
   Connection type ==> ________      SINGLE/MULTIPLE socket
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 35-21 Defining debug in Full Screen mode

After pressing PF3, the previous panel is displayed as shown in Figure 35-22 with parameter string update as shown. We now enter the command **RUN** on the command line and start the debugging.

```
EDIT - Edit Setup File 'SE65273.TEST.DTSF(TRADERB)'           Row 1 to 5 of 13
Command ===> RUN_____________________________________________ Scroll ===> PAGE

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs / OS/VS COBOL - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,'INSPIN',PROMPT,'MFI:INSPPREF')  '

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ _______ ___ _ ***************** Top of Data ********************* _____
___ COMPFILE 1   _ 'GRACINE.TRADER.COMPFILE'                          SHR
___ CUSTFILE 1   _ 'GRACINE.TRADER.CUSTFILE'                          SHR
___ INSPIN   1   _ 'GRACINE.TRADER.CMDS(CMD01)'                       SHR
___ INSPLOG  1   _ SYSOUT=*                                           _____
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 35-22 Start debug

Using MFI for the scenario

Once Debug Tool is started with MFI, the panel in Figure 35-23 is displayed.

```
COBOL    LOCATION: TRADERB :> 332.1
Command ===>                                                  Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF (
****************************** TOP OF MONITOR ******************************
****************************** BOTTOM OF MONITOR ***************************




SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 324 OF 812
     324        PROCEDURE DIVISION.                                          .
     325       ***************************************************************.
     326        MAINLINE SECTION.                                            .
     327                                                                     .
     328            MOVE FUNCTION CURRENT-DATE (1:8) TO WS-CURRENT-DATE      .
     329       *    ACCEPT    WS-CURRENT-DATE    FROM DATE                   .
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 20
****************************** TOP OF LOG **********************************
0001 IBM Debug Tool Version 6 Release 1 Mod 0
0002 11/21/2005 12:37:52 PM
0003 5655-P14 and 5655-P15: (C) Copyright IBM Corp. 1992, 2005
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 35-23 Debug Tool with MFI at start of program TRADERB

For now, the MFI is being used to determine that the problem occurs when reading the customer file; a decision is made to set a breakpoint at the START command when issued on the Customer file.

Find the first occurrence of the string START-CUSTFILE and set the breakpoint at START-CUSTFILE by typing the command FIND and in quotes “START-CUSTFILE” on the command line as shown in Figure 35-24.

```
COBOL    LOCATION: TRADERB initialization
Command ===> FIND "START-CUSTFILE"                            Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF 0
****************************** TOP OF MONITOR ******************************
****************************** BOTTOM OF MONITOR ***************************




SOURCE: TRADERB --1----+----2----+----3----+----4----+----5----+ LINE: 1 OF 812
     1        ***************************************************************.
     2        * LICENSED MATERIALS - PROPERTY OF IBM                         .
     3        * 5655-ADS (C) COPYRIGHT IBM CORP. 2004                        .
     4        * ALL RIGHTS RESERVED                                          .
     5        ***************************************************************.
     6        * PROGRAM:  TRADERB                                            .
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6- LINE: 1 OF 3
****************************** TOP OF LOG **********************************
0001 IBM Debug Tool Version 6 Release 1 Mod 0
0002 11/21/2005 12:52:08 PM
0003 5655-P14 and 5655-P15: (C) Copyright IBM Corp. 1992, 2005
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 35-24 The FIND command

In the source window, the cursor now is positioned at START-CUSTFILE as shown in Figure 35-25.

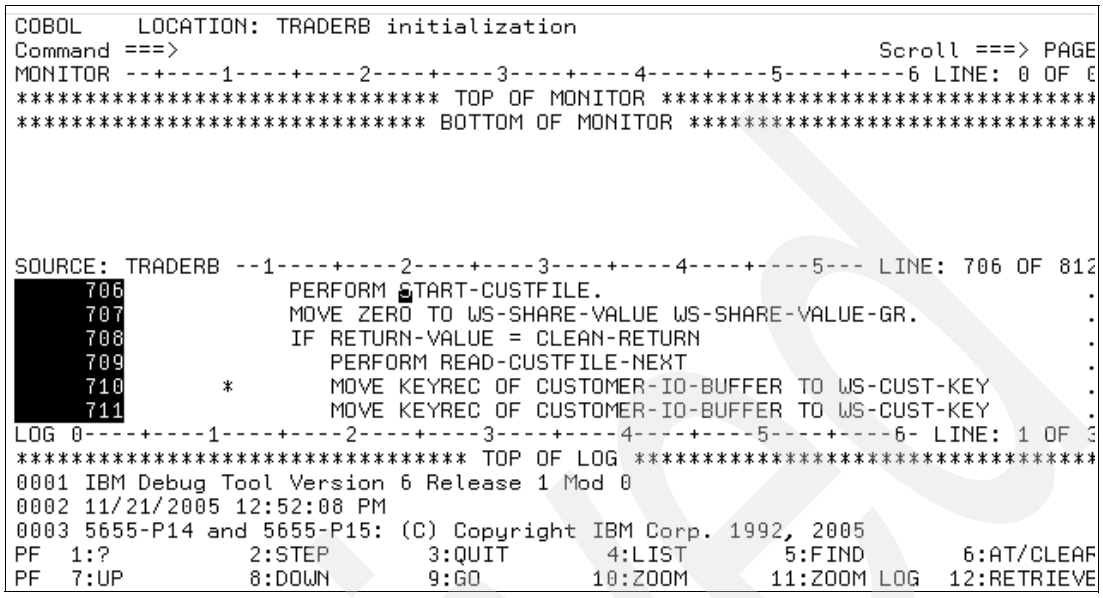


Figure 35-25 Positioned

As we can see, this is line 706, so we set a break point on that line by giving the command: `AT 706.` We press Enter and the panel in Figure 35-26 in displayed.

Review the code within the current paragraph and notice the following code:

- Line 709 contains the instruction, PERFORM READ-CUSTFILE-NEXT.
- There is a MOVE statement that uses two variables, KEYREC and WS-CUST-KEY.

We want to see what happens to the values in these fields (KEYREC and WS-CUST-KEY) when the customer file is read.

Enter the following command on the command line:

MONITOR LIST (KEYREC, WS-CUST-KEY);

```
COBOL    LOCATION: TRADERB initialization
Command ===> AT 706                                          Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF 0
******************************** TOP OF MONITOR ********************************
******************************** BOTTOM OF MONITOR *****************************




SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 706 OF 812
     706             PERFORM START-CUSTFILE.                                  .
     707             MOVE ZERO TO WS-SHARE-VALUE WS-SHARE-VALUE-GR.           .
     708             IF RETURN-VALUE = CLEAN-RETURN                           .
     709                PERFORM READ-CUSTFILE-NEXT                            .
     710       *        MOVE KEYREC OF CUSTOMER-IO-BUFFER TO WS-CUST-KEY       .
     711                MOVE KEYREC OF CUSTOMER-IO-BUFFER TO WS-CUST-KEY       .
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6- LINE: 1 OF 3
******************************** TOP OF LOG ************************************
0001 IBM Debug Tool Version 6 Release 1 Mod 0
0002 11/21/2005 12:52:08 PM
0003 5655-P14 and 5655-P15: (C) Copyright IBM Corp. 1992, 2005
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 35-26 To set breakpoint at line 706

After setting the breakpoint and given the MONITOR command we press PF9 (go) to let the program execute to next breakpoint.

As shown in Figure 35-27, the program has stopped execution at line 706 and the value for the variables is shown in the monitor window.

```
COBOL    LOCATION: TRADERB :> 706.1
Command ===>                                                 Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 8
0001   1 02 TRADERB:>KEYREC
0002        03 TRADERB:>CUST-NM     'GARY
0003                               '
0004        03 TRADERB:>KEYREC-DOT     '.'
0005        03 TRADERB:>COMP-NM     'Casey_Import_Export '
0006     WS-CUST-KEY   X'00000000000000000000000000000000000000000000000000000000
0007     00000000000000000000000000000000000000000000000000000000000000000000
SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 706 OF 812
     706             PERFORM START-CUSTFILE.                                  .
     707             MOVE ZERO TO WS-SHARE-VALUE WS-SHARE-VALUE-GR.           .
     708             IF RETURN-VALUE = CLEAN-RETURN                           .
     709                PERFORM READ-CUSTFILE-NEXT                            .
     710       *        MOVE KEYREC OF CUSTOMER-IO-BUFFER TO WS-CUST-KEY       .
     711                MOVE KEYREC OF CUSTOMER-IO-BUFFER TO WS-CUST-KEY       .
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6 LINE: 8 OF 11
0008 breakpoint.
0009 The current location is TRADERB ::> TRADERB :> 706.1.
0010  MONITOR
0011    LIST ( KEYREC, WS-CUST-KEY ) ;
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 35-27 Breakpoint reached, values displayed

Press PF2 to step through the program one statement at a time.

As shown in Figure 35-28, the program successfully executed the PERFORM START-CUSTFILE statement, and the value in the KEYREC field is Gary.

Press PF2 until you reach line 719. While you do, pay attention to the value of the variables in the Monitor window.

As you can see in Figure 35-28, line 719, PERFORM CALCULATE-SHARE-VALUE is performed until the values in the variables KEYREC and WS-CUST-KEY are not equal.

At this point, you can see the values of both the variables are still equal. The value of the field CUST-NM of KEYREC is Gary and the value of the field, COMP-NM of KEYREC, is Casey_Import_Export. Control is transferred to the CALCULATE-SHARE-VALUE paragraph and the record details are printed.

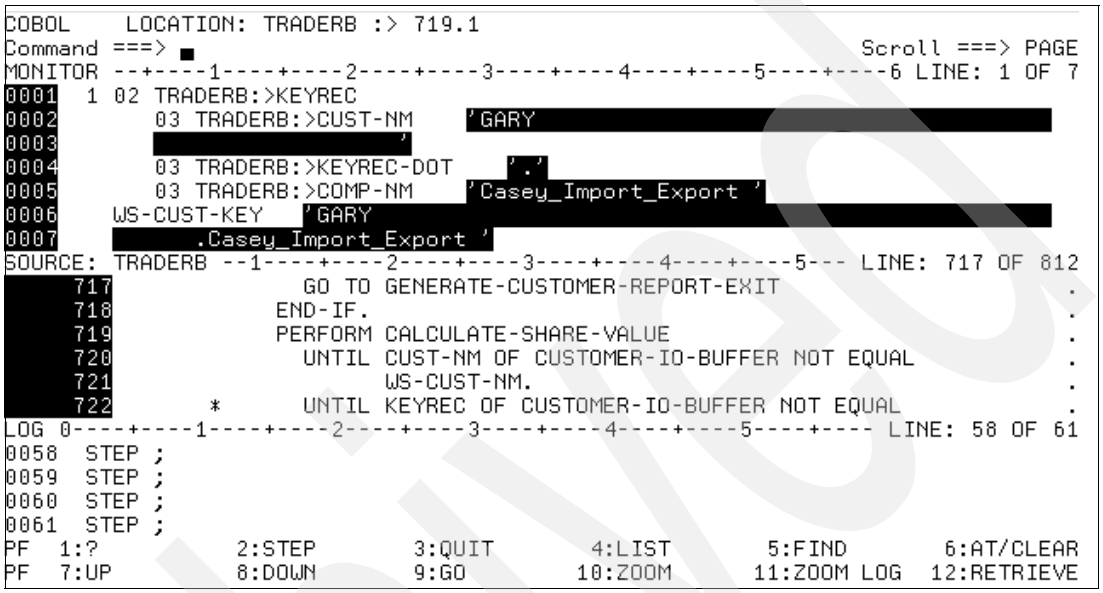


Figure 35-28 Line 719, PERFORM CALCULATE-SHARE-VALUE

Continue to press PF2 until the next READ statement.

Check the values of these variables after the READ statement.

The values in the variables are different, as shown in Figure 35-29, and the READ process for customer Gary is terminated.

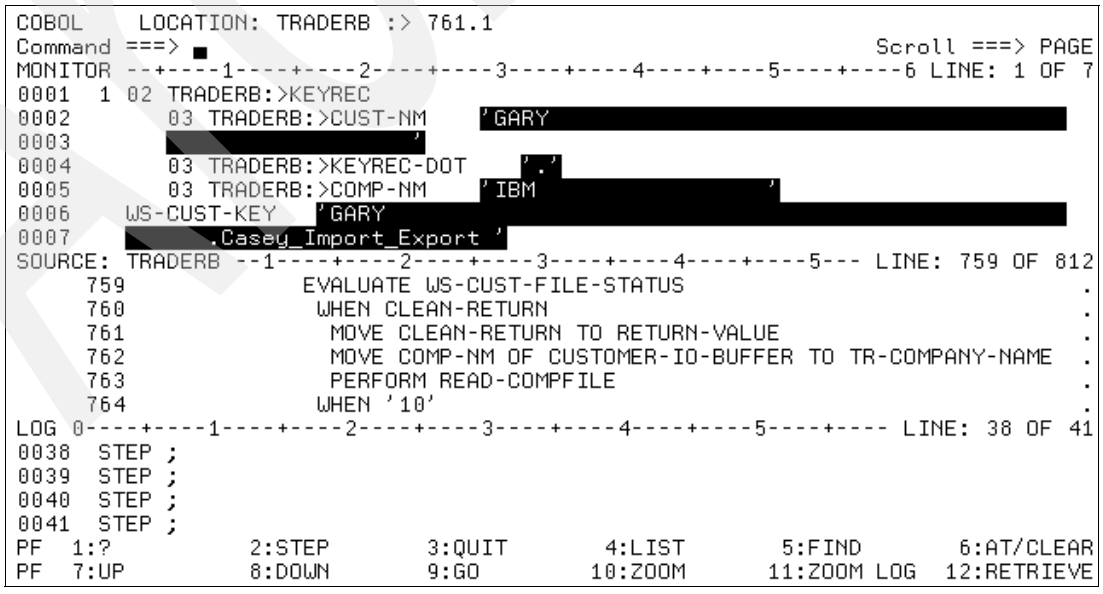


Figure 35-29 Variables are different

You can see that the record is for GARY because the field CUST-NM of KEYREC has that value.

But the key value, KEYREC, is different from WS-CUST-KEY because the field COMP-NM of KEYREC has a new value, IBM, and the variable WS-CUST-KEY still has the old value.

Because these values are different, control is not transferred to the CALCULATE-SHARE-VALUE paragraph

Enter the `QUIT ABEND` command on the command line to end the session.

You found that saving the value of the previously read key value of the Customer record and checking it with the key value immediately after the next read is causing the problem.

Because the customer has one record for every company in which he holds shares, the program logic must be changed to check only the CUST-NM of KEYREC. Saving the CUST-NM field of KEYREC and checking it just after a READ NEXT should solve the problem.

Here are changes you make to the program (the corrected “after lines” were commented out):

Before (line 710)

```
MOVE KEYREC OF CUSTOMER-IO-BUFFER TO WS-CUST-KEY
```

After (line 711)

```
MOVE CUST-NM OF CUSTOMER-IO-BUFFER TO WS-CUST-NM
```

Before (line 721, 722)

```
PERFORM CALCULATE-SHARE-VALUE
   UNTIL KEYREC OF CUSTOMER-IO-BUFFER NOT EQUAL
         WS-CUST-KEY.
```

After (line 719, 720)

```
PERFORM CALCULATE-SHARE-VALUE
   UNTIL CUST-NM OF CUSTOMER-IO-BUFFER NOT EQUAL
         WS-CUST-NM.
```

Before (line 736)

```
MOVE CUST-NM OF CUSTOMER-IO-BUFFER TO WS-CUST-NM
```

After (line 735)

```
MOVE KEYREC OF CUSTOMER-IO-BUFFER TO WS-CUST-KEY
```

35.2.6 Executing the batch application after the fix

Recompile the program after you finish correcting the program logic in TRADERB.

Submit the batch job, TRADERB.

35.2.7 Summary of scenario 2

In this chapter we described the various components that make up the batch environment in our system and how they are set up.

We reviewed these topics:

1. The processing performed by the batch Trader application
2. A process that used Debug Tool in batch mode to identify a possible problem in the application
3. Debug Tool's capability in foreground mode to pinpoint an error to allow it to be corrected

36

Is the error in DB2 data or program logic?

In this chapter we describe the application components that exist in the CICS and DB2 environments on our system and explain how they are set up.

Our application is forced to encounter an error. We describe the steps required to identify the cause of the problem in the application using Debug Tool, and show how to manipulate the data to correct the problem using File Manager DB2.

36.1 Setting up the components

Two types of components must be established for this scenario:

- CICS and DB2 components
- Program products:
  - Debug Tool
  - File Manager DB2

36.1.1 CICS and DB2 components

The components used by the Trader application are listed in Table 36-1.

Table 36-1 CICS component of the Trader Application for the DB2 scenario

Component	Details	Remarks
Programs	MYTRADMD MYTRADD	CICS DB2 COBOL programs
Tran ID	TDB2	CICS transaction associated with the program
Mapset	NEWTRAD	BMS mapset containing all the maps used by the application
Tables	CUSTOMER_DETAILS COMPANHY_DETAILS	DB2 tables used by the application

36.1.2 Program products

To use the Problem Determination Tools with this scenario, make sure that you have the following output or supporting files for each product.

Debug Tool

You must have a compiler listing or side file for the programs MYTRADMD and MYTRADD.

If you are not using the supplied JCL to compile these programs, make sure that you specify the following compiler options:

```
LIST,XREF,RENT,TEST
```

If you prefer to use a side file instead of a compiler listing, include the SEPARATE sub-option of the TEST compiler option. Recall that the side file required by Debug Tool is the same as the one required by Fault Analyzer.

File Manager

You should use the templates for the DB2 tables CUSTOMER_DETAILS and COMPANY_DETAILS dynamically created at first edit of these tables.

Make sure that you run the TABLES batch job to create the DB2 tables, and then run the DATA batch job to load the DB2 tables.

36.2 Tracking a problem in the application

To demonstrate the capabilities of the Problem Determination Tools, we show you how to force the application to encounter an error and step you through the process of fixing it. Debug Tool is used to follow the execution of the application and identify and isolate the problem. File Manager is used to correct an error in a DB2 database.

36.2.1 Recreating the error

In this example, you invoke the Trader application and perform the following actions for the customer, RB_DEMO, using IBM as the company in which shares are traded:

1. Launch the Trader application (TDB2 transaction).
2. Select IBM as the company you want to trade.
3. Select option **1** (New Real-Time Quote) on the Options panel to display the number of shares available. This amount might differ from one run of the transaction to another depending on the operations done on the data. The result could be something like that shown in Figure 36-1.

```
                        Share Trading Demonstration                TRADER.T004

                    Share Trading Manager: Real-Time Quote

      User Name:     RB_DEMO

      Company Name:  IBM

      Share Values:                      Commission Cost:
         NOW:            109.00             for Selling:        010
         1 week ago:     111.00             for Buying:         005
         6 days ago:     112.00
         5 days ago:     111.00
         4 days ago:     109.00
         3 days ago:     107.00
         2 days ago:     107.00          Number of Shares Held: 0075
         1 day ago:      106.00          Value of Shares Held:        8175.0



Request Completed OK
--------------------------------------------------------------------------------
PF3=Return                                                          PF12=Exit
```

Figure 36-1 Number of shares available

4. Press PF3 Return and select option **3** (Sell Shares) on the Options panel.

5. On the Shares - Sell panel, shown in Figure 36-2, sell more shares (100) than the number held (75).

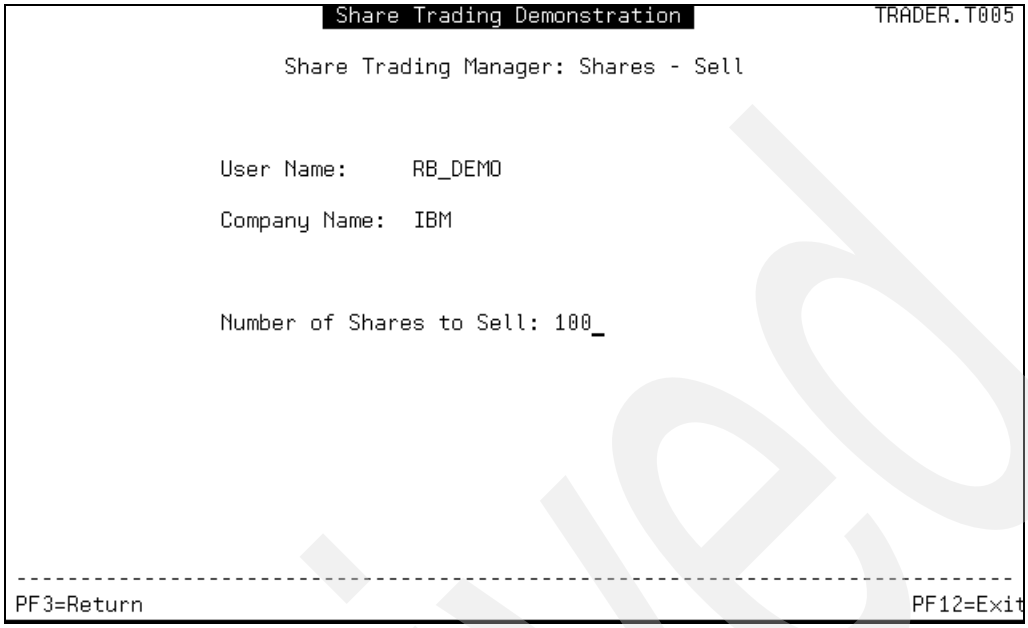


Figure 36-2 Recreating a problem in the Trader application

The Options panel is re-displayed with the message:

```
Request Completed OK
```

6. Select option **1** (New Real-Time Quote) on the Options panel to obtain real-time quotes and a listing of the shares held, as is shown in Figure 36-3.

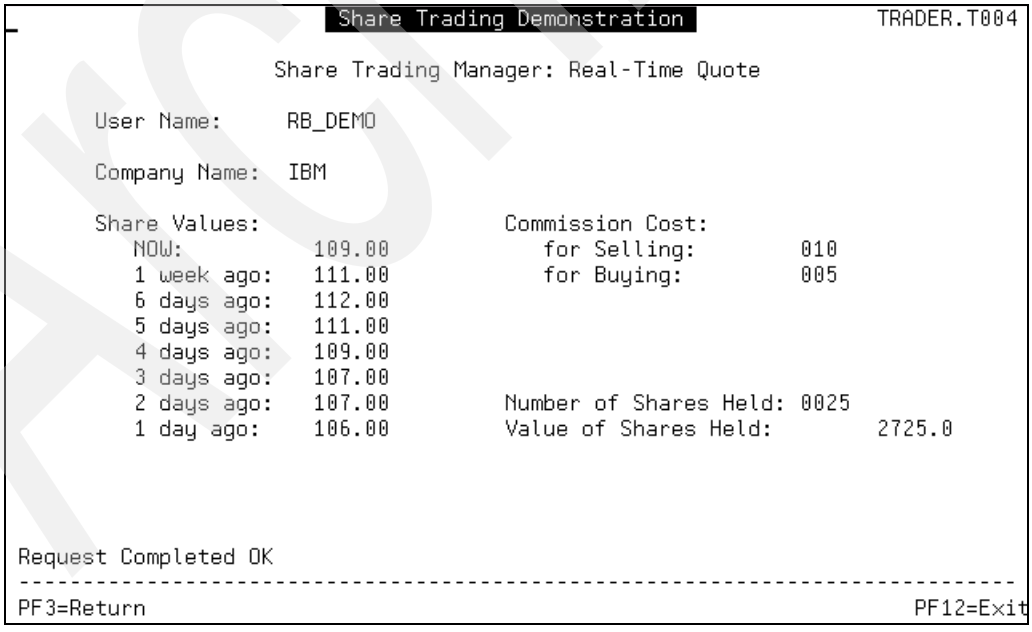


Figure 36-3 Recreating a problem in the Trader application

Note: The share details would be listed. The Number of Shares Held field would have a value of 25 and it should be -25 (75 - 100 = -25).

7. Press PF3 to return to the Options panel.
8. Select option **2** (Buy Shares) and buy 25 shares.

   The Options panel is redisplayed with the message:

   ```
   Request Completed OK.
   ```

9. Once again, select option **1** (New Real-Time Quote) and review the number of shares held. Figure 36-4 shows this value as 0 (-25 + 25 = 0).

```
  Display  Filter  View  Print  Options  Help
 -------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY TRADERD1 JOB04776  DSID   103 LINE 1       COLUMNS 01- 80
 COMMAND INPUT ===> ■                                         SCROLL ===> CSR
1CUSTOMER : GARY                                             11/18/2005
0----------------------------------------------------------------
      COMPANY                   SHARES    SHARE            TOTAL
                                HELD      VALUE            COST
 ----------------------------------------------------------------
      Casey_Import_Export        504        79.00         39,816.00
      IBM                        762       163.00        124,206.00
```

Figure 36-4 Recreating a problem in the Trader application (part 3

Clearly, there is a serious problem with this series of transactions.

We believe that the problem is with the data in the table, CUSTOMER_DETAILS, or in the program that reads the table.

We decide to look first at the specific customer record in the database to see if that helps us understand more about the problem.

Tip: Sell five more shares before going to the next step in the analysis process.

36.2.2 Viewing the data in File Manager/DB2

The following steps allow you to view data:

1. To determine which DB2 was used, from CICS we issue the command:

   ```
   CEMT I DB2CONNECTION
   ```

The panel shown in Figure 36-5 is returned.

```
I DB2CONN
STATUS:  RESULTS - OVERTYPE TO MODIFY
  Accountrec( None )              Planexitname( DSNCUEXT )
  Authid( TIMOTHY  )              Priority( High )
  Authtype(            )          Purgecyclem( 00 )
  Comauthid(           )          Purgecycles( 30 )
  Comauthtype( Cuserid )          Resyncmember(                )
  Comthreadlim( 0001 )            Signid( CICSC31F )
  Comthreads(0000)                Security(                )
  Connecterror( Sqlcode )         Standbymode( Reconnect )
  Connectst( Connected )          Statsqueue( CDB2 )
  Db2groupid(      )              Tcblimit( 0012 )
  Db2id( D81H )                   Tcbs(0000)
  Db2release(0810)                Threaderror(N906d)
  Drollback(Rollback)             Threadlimit( 0003 )
  Msgqueue1( CDB2 )               Threads(0000)
  Msgqueue2(      )               Threadwait( Twait )
  Msgqueue3(      )
  Nontermrel( Release )
  Plan(           )

                                                SYSID=C31F APPLID=CICSC31F
 RESPONSE: NORMAL                          TIME:  13.49.52  DATE: 12.14.06
PF 1 HELP       3 END       5 VAR       7 SBH 8 SFH 9 MSG 10 SB 11 SF
```

Figure 36-5 CICS DB2Connection

2. Access File Manager/DB2 in your ISPF session.

   If your system contains only one active DB2 subsystem, File Manager/DB2 automatically connects to that subsystem. However, since we are working in an environment that contains more than one active DB2 subsystem, we must select a DB2 subsystem before File Manager/DB2 can connect to it.

   Type over the ID of the DB2 subsystem currently shown in the DB2 SSID field with the ID of the active DB2 subsystem you want and press Enter, or delete the contents of this field and select from the list displayed when you press Enter. The File Manager/DB2 - DB2 Subsystem Selection menu is displayed, as shown in Figure 36-6.

```
 Process   Help
--------------------------------------------------------------------------
FM/DB2 (D81H)              DB2 Subsystem Selection              Row 1 of 16
Command ===> _____________________________________________________ Scroll PAGE

Sel SSID Status    Description                                    Prefix
--------------------------------------------------------------------------
    DB1E ACTIVE    DB2 VERSION 7 LPAR F6                          -DB1E
    D72F ACTIVE    DB2 VERSION 7 LPAR F6                          -D72F
s   D81H ACTIVE    DB2 VERSION 8 LPAR F6                          -D81H
    D82G ACTIVE                                                   -D82G
    D91F ACTIVE    DB2 VERSION 9 LPAR F6                          -D91F
    DB0E GROUP
    D60F INACTIVE
    D70F GROUP     DB2 V7 DATA SHARING GROUP
    D80G GROUP
    D80H GROUP
    D90F GROUP
    D80G INACTIVE
    D70F INACTIVE  DB2 V7 DATA SHARING GROUP
    D61F INACTIVE                                                 -D61F
    D71F INACTIVE                                                 -D71F
F1=Help      F2=Split     F3=Exit      F5=Refresh   F6=Show      F7=Up
F8=Down      F9=Swap     F12=Cancel
```

Figure 36-6 File Manager/DB2 - data sharing group

3. Press Enter, select option **1** (Browse), and press Enter. The DB2 Browse panel shown in Figure 36-7 is displayed.

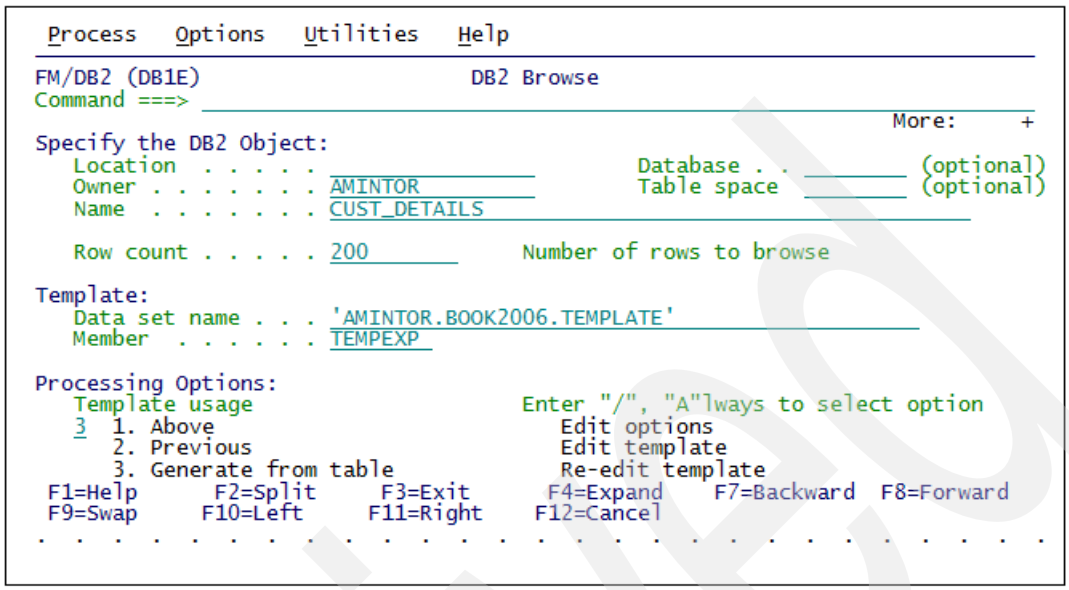


Figure 36-7 File Manager/DB2 - Browse panel

4. Specify the following information:
   - The table owner user ID used at installation time.
   - The table name.
   - Select 3 (Generate) from table in the Processing Options field.

   The Table Browse panel is displayed in Figure 36-8.

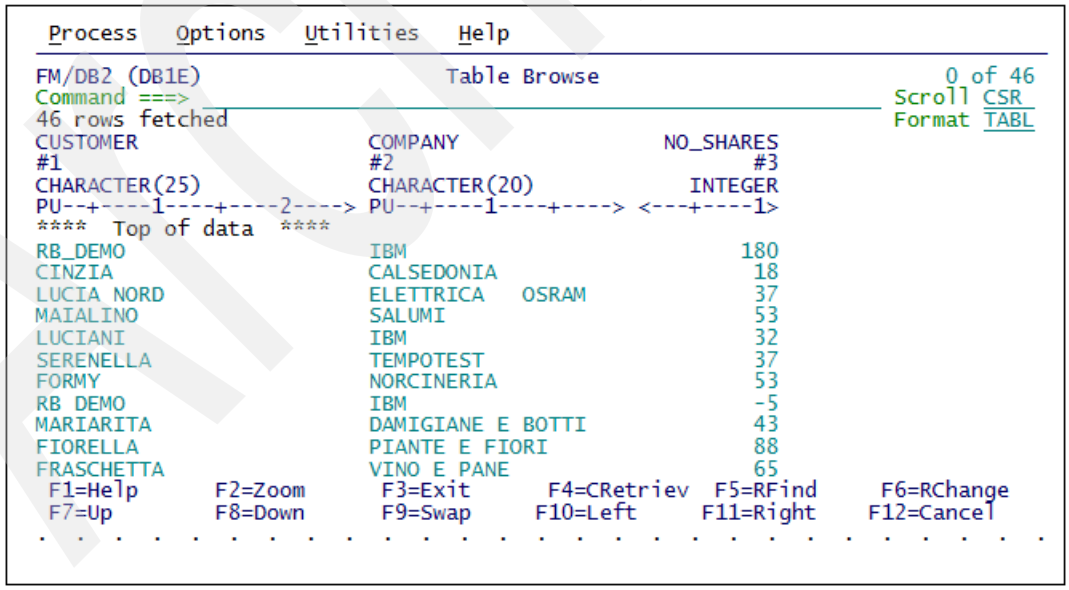


Figure 36-8 File Manager-DB2 - Table Browse panel

5. The record containing the string RB_DEMO is displayed as the first (and only) page in the panel. Locate the record that has a value of IBM in the COMPANY column.

   You can see in Figure 36-9 that the value in the NO_SHARES column is -5. This is incorrect data in the application. No negative value is allowed.

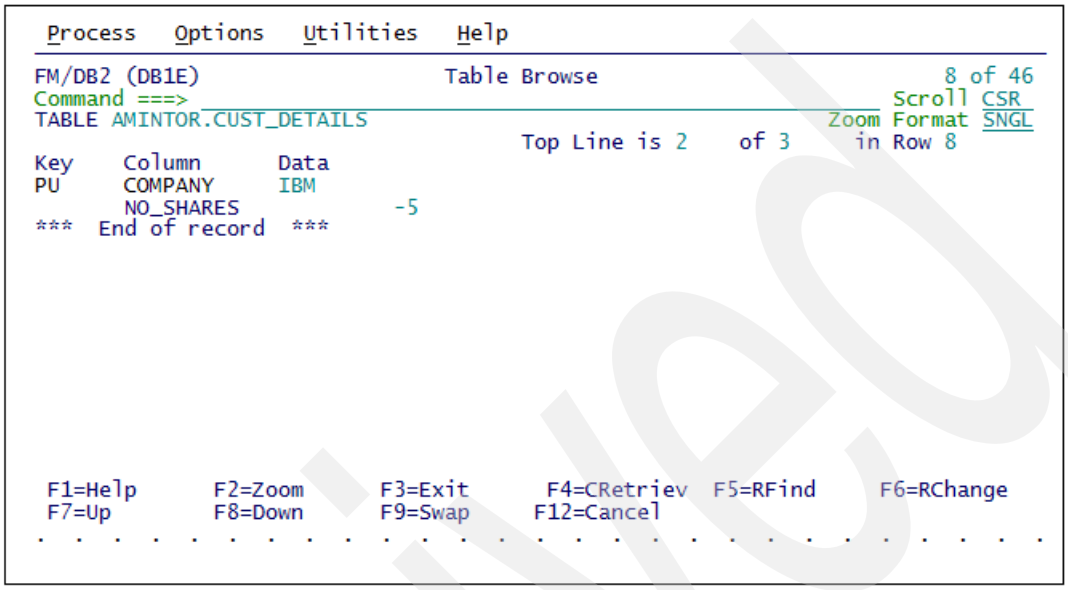


Figure 36-9 File Manager-DB2 - Table Browse record

At this point, we believe that the problem is due to faulty logic in the program that updates the CUST_DETAILS table.

We can review the compiler listing to get an overview of the program and to see where the table is processed.

We decide to debug the program with Debug Tool.

36.2.3 Using Debug Tool to identify the logic problem

Set up and use a debug session for the TDB2 transaction in your CICS region by performing the following steps:

1. Enter transaction ID CADP to update your profile, as shown in Figure 36-10.

```
CADP      -    CICS Application Debugging Profile Manager   -     CICSC23G

Create LE Debugging Profile ==> LEO       for SE65273  activated by SE65273
                                                       to 0012

CICS Resources To Debug (use * to specify generic values e.g. *, A*, AB*, etc.)
Transaction      ==> *                              Applid   ==> CICSC23G
Program          ==> MYTRADMX                       Userid   ==> SE65273
Compile Unit     ==> *                              Termid   ==> 0012
                                                    Netname  ==> TCP00012

Debug Tool Language Environment Options
Test Level       ==> All                                  (All,Error,None)
Command File     ==>
Prompt Level     ==> PROMPT
Preference File  ==>

Other Language Environment Options
==>
==>
==>
==>

Enter=Create PF1=Help 2=Save options as defaults 3=Exit 10=Replace 12=Return
```

Figure 36-10 CICS - CADP transaction profile

2. Press PF10 to replace.
3. PF3 to exit from this panel.
4. Enter the transaction ID `TDB2`.

   The debug session is started, as shown in Figure 36-11.

```
 COBOL    LOCATION: MYTRADMD ENTRY
 Command ===>                                                Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 2
******************************** TOP OF MONITOR ********************************
0001  1 EQA1252I ********** AUTOMONITOR **********
0002    EQA1945I There are no variables in the statement to display.
****************************** BOTTOM OF MONITOR ******************************
SOURCE: MYTRADMD -1----+----2----+----3----+----4----+----5---- LINE: 7 OF 1587
       7          PROGRAM-ID. MYTRADMD.                                        .
       8                                                                       .
       9          ENVIRONMENT DIVISION.                                        .
      10                                                                       .
      11          DATA DIVISION.                                               .
      12          WORKING-STORAGE SECTION.                                     .
      13                                                                       .
      14          01  TASK-DATA.                                               .
      15              02 DATA1                      PIC S9(8) COMP.           .
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 92 OF 94
0092 EQA1239I The program is currently entering block MYTRADMD ::> MYTRADMD.
0093  SET PF24 "RETRIEVE" = IMMEDIATE RETRIEVE ;
0094 EQA1239I The program is currently entering block MYTRADMD ::> MYTRADMD.
****************************** BOTTOM OF LOG ******************************
 PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
 PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      19/015
```

Figure 36-11 Debug session starting for program MYTRADMD

5. Issue the following commands on the command line to stop the program's execution when the program MYTRADD is invoked:

```
AT APPEARANCE MYTRADD
AT ENTRY MYTRADMD::>MYTRAD;
```

> **Note:** You can also use the one-line syntax:
>
> ```
> AT APPEARANCE MYTRADD perform AT ENTRY MYTRADMD::>MYTRAD; end-perform;
> ```

6. Run the program using PF12 (note that PF9 is the default key).
7. Press PF12 repeatedly and enter the appropriate values until the Shares - Buy screen is displayed.
8. Select option **2** (Buy Shares) and press Enter.
9. In the Shares - Buy screen, enter 5 in the Number of Shares to Buy field, and press Enter.
10. Press PF12 to continue program execution.

    The program stops when the program MYTRADD is invoked.
11. Issue the following command to monitor the value of the NO-SHARES field (the host variable for the column NO_SHARES in the CUSTOMER_DETAILS table):

```
MONITOR LIST NO-SHARES;
```

> **Note:** Two other variables are interesting and can also be monitored:
>
> ```
> MONITOR LIST NO-OF-SHARES-DEC
> MONITOR LIST %HEX (SHARES-OVERFLOW)
> ```
>
> This last one must keep a value of X'00'.

The value of these variables is displayed in the Monitor window, as shown in Figure 36-12.

```
 COBOL    LOCATION: MYTRADD :> 691.1
 Command ===> _                                              Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 3
********************************* TOP OF MONITOR *********************************
0001  1 NO-SHARES  +0000000000
0002  2 NO-OF-SHARES-DEC  0000
0003  3 %HEX ( SHARES-OVERFLOW )   X'00'
SOURCE: MYTRADD --1----+----2----+----3----+----4----+----5-- LINE: 691 OF 1432
     691           Call 'DFHEI1' using by content x'0e08000007000010000f0f0f6
     692         -    '404040' end-call
     693
     694              .
     695          MAINLINE-EXIT.
     696              EXIT.
     697         ***************************************************************
     698          BUY-SELL SECTION.
     699              EVALUATE UPDATE-BUY-SELL
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 110 OF 113
0110  MONITOR
0111    LIST NO-OF-SHARES-DEC ;
0112  MONITOR
0113    LIST %HEX ( SHARES-OVERFLOW ) ;
 PF 13:PK 1-12   14:CLR KEEP  15:QUIT      16:LIST CSR  17:FIND      18:AT/CLEAR
 PF 19:TOP       20:BOTTOM    21:GO 1      22:ZOOM WIN  23:ZOOM LOG  24:RETRIEVE
                                                                      02/015
```

Figure 36-12 Monitoring the values

12. Press PF9 (the default key is PF2) to step through the program one line at a time. As you do, keep monitoring the value of NO-SHARES in the Monitor window.

    You see that the value in NO-SHARES is -5, as shown in Figure 36-13, after the record in the CUSTOMER_DETAILS table is read in the READ-CUSTOMER-TABLE paragraph.

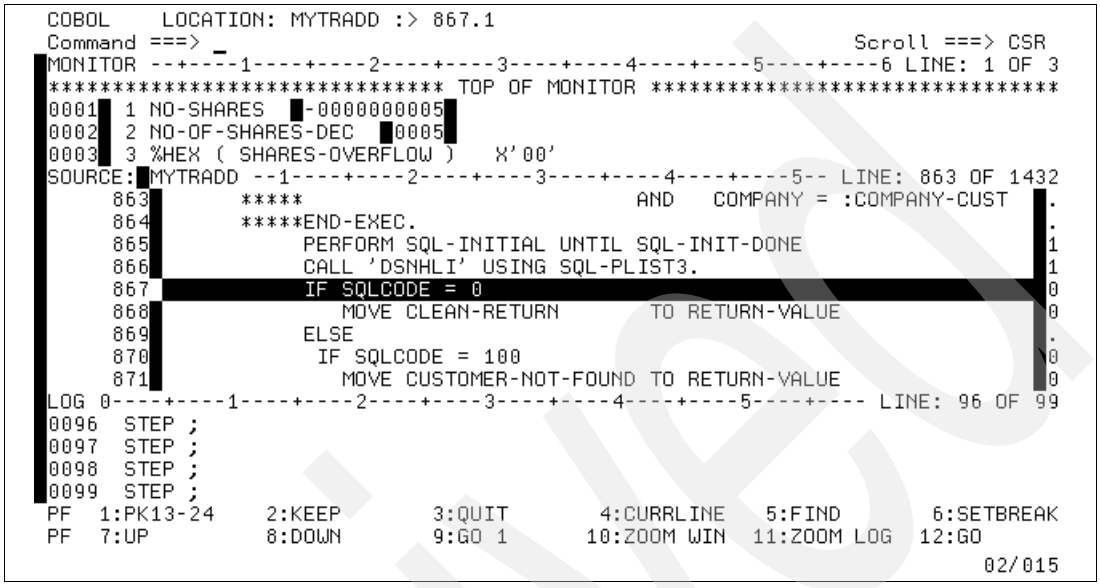


Figure 36-13 Monitoring the value in NO-SHARES after table read

13. Press PF9 to check the program flow before the program updates the customer table.

    The number of shares to be bought is added to the existing value in NO-SHARES in the CALCULATE-SHARES-BOUGHT paragraph. This is done before the table is updated in the UPDATE-CUSTOMER-TABLE paragraph, as shown in Figure 36-14.

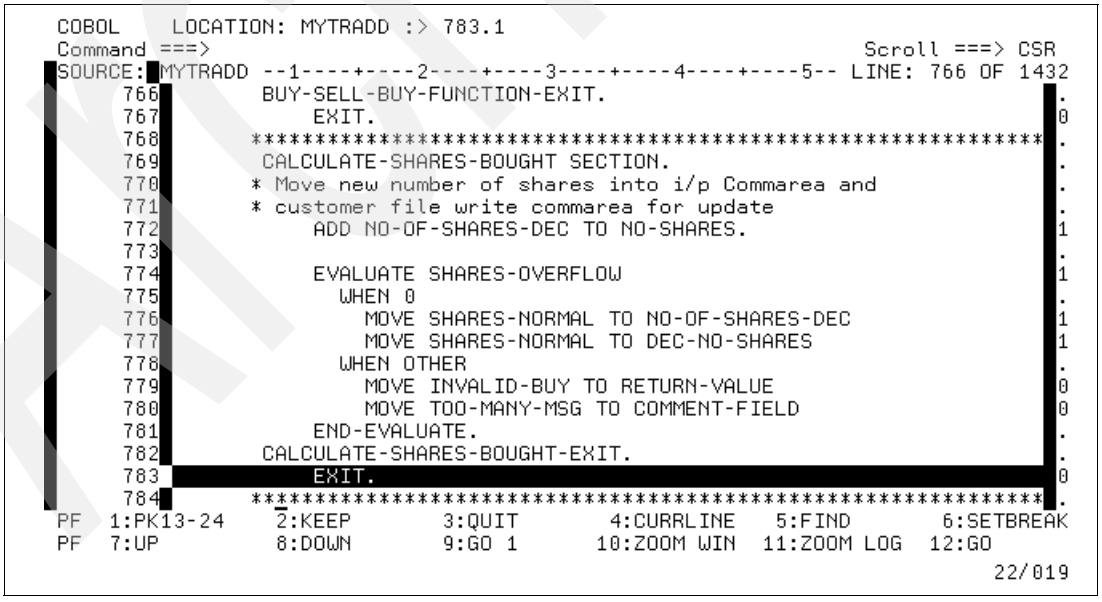


Figure 36-14 CALCULATE -SHARES-BOUGHT section

The value of NO-SHARES is now 0, as shown in Figure 36-15.

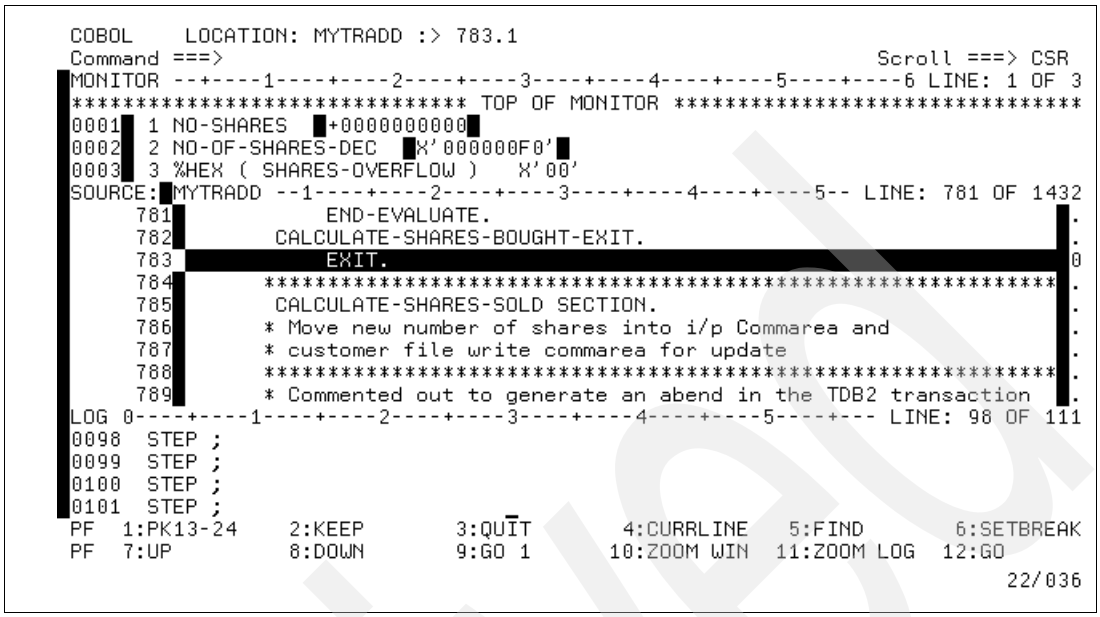


Figure 36-15 Monitoring the value in NO-SHARES now in error

Conclusion 1: The buy process actually *zeros* the value. Therefore, the display shows zero number of shares.

We continue the debugging session to review the sell processing portion of the program.

14. Press PF12. The Options panel is displayed.
15. Select option **3** (Sell Shares) and press Enter.

    Enter 5 in the Number of Shares to Sell field.

16. Press PF9 to step through the program one line at a time. Continue to watch the value of NO-SHARES in the Monitor window.

    You can see that the value of NO-SHARES after the READ-CUSTOMER-TABLE paragraph is executed is 0, as shown in Figure 36-16.

```
 COBOL    LOCATION: MYTRADD :> 814.1
 Command ===>                                                  Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 3
****************************** TOP OF MONITOR *******************************
0001  1 NO-SHARES  +0000000000
0002  2 NO-OF-SHARES-DEC  0005
0003  3 %HEX ( SHARES-OVERFLOW )   X'00'
SOURCE:MYTRADD --1----+----2----+----3----+----4----+----5-- LINE: 812 OF 1432
     812        * Check whether we have any shares to sell                 .
     813             PERFORM READ-CUSTOMER-TABLE THRU READ-CUSTOMER-EXIT.  1
     814           EVALUATE RETURN-VALUE                                   0
     815             WHEN CLEAN-RETURN                                     .
     816                    PERFORM CALCULATE-SHARES-SOLD                  0
     817                    PERFORM UPDATE-CUSTOMER-TABLE                  0
     818                    PERFORM BUILD-RESP-COMMAREA                    0
     819        *         END-IF                                           .
     820             WHEN CUSTOMER-NOT-FOUND                               .
LOG 0----+----1----+----2----+----3----+----4----+----5----+--- LINE: 96 OF 103
0096  STEP ;
0097  STEP ;
0098  STEP ;
0099  STEP ;
 PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
 PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      22/017
```

Figure 36-16 Monitoring the value in NO-SHARES after table read

17. Press PF9 to continue executing the program.

    You can see that the value of NO-SHARES is a negative value (-5), as shown in Figure 36-17, after the SUBSTRACT statement in the CALCULATE-SHARES-SOLD section.

```
 COBOL    LOCATION: MYTRADD :> 798.1
 Command ===>                                                  Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 3
****************************** TOP OF MONITOR *******************************
0001  1 NO-SHARES  -0000000005
0002  2 NO-OF-SHARES-DEC  0005
0003  3 %HEX ( SHARES-OVERFLOW )   X'00'
SOURCE:MYTRADD --1----+----2----+----3----+----4----+----5-- LINE: 794 OF 1432
     794        *         MOVE TOO-MANY-SHARES-MSG TO COMMENT-FIELD        .
     795        *    ELSE                                                  .
     796                SUBTRACT NO-OF-SHARES-DEC FROM NO-SHARES          1
     797                GIVING NO-SHARES.                                 .
     798                MOVE NO-SHARES TO NO-OF-SHARES-DEC.               0
     799        *    END-IF.                                               .
     800        ************************************************************.
     801        * End of commented out                                     .
     802        ************************************************************.
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 103 OF 106
0103    LIST %HEX ( SHARES-OVERFLOW ) ;
0104  STEP ;
0105  STEP ;
0106  STEP ;
 PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
 PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      22/017
```

Figure 36-17 Monitoring the value in NO-SHARES after calculation

Conclusion 2: It is clear that the problem is program logic. There is no validation of the number of shares held by a customer before the sell is processed.

Let us figure out how to correct this problem. We must add a validation routine to the program that encapsulates the following logic: If the shares held by the customer are less than the number of shares to be sold, then the transaction is not performed and a warning message is issued. This stops negative values from appearing in the database.

Example 36-1 shows the updated code in the CALCULATE-SHARES-SOLD paragraph.

Example 36-1 Coding changes in MYTRADD to correct the error

```
IF NO-OF-SHARES-DEC IS GREATER THAN NO-SHARES
 THEN
   MOVE INvaLID-SALE TO RETURN-vaLUE
   MOVE TOO-MANY-SHARES-MSG TO COMMENT-FIELD
ELSE
   SUBTRACT NO-OF-SHARES-DEC FROM NO-SHARES
   GIVING NO-SHARES
   MOVE NO-SHARES TO NO-OF-SHARES-DEC
END-IF.
```

Attention: You might encounter a bad return when buying shares. Monitor the variable SHARES-OVERFLOW and make sure its value is x'00', as shown in Figure 36-18. To change its value, enter the command:

```
MOVE X'00' TO SHARES-OVERFLOW
```

```
 COBOL    LOCATION: MYTRADD :> 868.1
 Command ===>                                                Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 3
****************************** TOP OF MONITOR ********************************
0001  1 NO-SHARES  +0000000005
0002  2 NO-OF-SHARES-DEC  0010
0003  3 %HEX ( SHARES-OVERFLOW )   X'00'
SOURCE:MYTRADD --1----+----2----+----3----+----4----+----5-- LINE: 864 OF 1432
     864      _ *****END-EXEC.                                                .
     865           PERFORM SQL-INITIAL UNTIL SQL-INIT-DONE                    1
     866           CALL 'DSNHLI' USING SQL-PLIST3.                            1
     867           IF SQLCODE = 0                                             1
     868              MOVE CLEAN-RETURN       TO RETURN-VALUE                 0
     869           ELSE                                                       .
     870            IF SQLCODE = 100                                          0
     871              MOVE CUSTOMER-NOT-FOUND TO RETURN-VALUE                 0
     872            ELSE                                                      .
LOG 0----+----1----+----2----+----3----+----4----+----5----+--- LINE: 97 OF 100
0097  STEP ;
0098  STEP ;
0099  STEP ;
0100  STEP ;
 PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
 PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      09/015
```

Figure 36-18 Check the SHARES-OVERFLOW variable

Remember, to enable the corrected programs for CICS, first you have to remove the old ones. Follow the following instructions.

TSO

Run the setup jobs again to generate new copies of these programs. Remember that you also have to rebind the DB2 DBRM.

CICS

Make these copies available. Enter the command:

```
cemt set program(MYTRAD*) new
```

Tip: To ease the work, create a second set of programs, plan, transaction, and DB2 entry, and present either the erroneous program or the correct one.

36.2.4 Using File Manager/DB2 to correct the data

We decide to use File Manager/DB2 to correct the invalid data in the NO_SHARES column in the CUSTOMER_DETAILS table to rectify the problem in the database. Use these steps to do this:

1. Access File Manager/DB2 in your ISPF session.
2. Select option **2** (Edit) and press Enter.
3. The DB2 Edit panel is displayed, as shown in Figure 36-19.

```
 Process   Options   Utilities   Help
------------------------------------------------------------------------------
FM/DB2 (DB1E)                        DB2 Edit
Command ===> ____________________________________________________________
                                                                More:     +
Specify the DB2 Object:
   Location  . . . . . _____________       Database . . ________ (optional)
   Owner . . . . . . . AMINTOR          Table space  ________ (optional)
   Name  . . . . . . . CUST_DETAILS

   Row count . . . . . 200          Number of rows to edit

Template:
   Data set name . . . 'AMINTOR.BOOK2006.TEMPLATE'
   Member  . . . . . . TEMPEXP

Processing Options:
   Template usage                    Enter "/", "A"lways to select option
   3  1. Above                          Edit options
      2. Previous                       Edit template
      3. Generate from table            Re-edit template
 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap     F10=Left     F11=Right    F12=Cancel
. . . . . .
```

Figure 36-19 File Manager/DB2 - table edit entry panel

4. Specify the following information:
   - The table owner: This is the user ID used at installation time.
   - The table name.
   - Select 3 in the Processing Options field.

The Table Edit panel is displayed, as shown in Figure 36-20.

```
 Process   Options   Utilities   Help
FM/DB2 (DB1E)                    Table Edit                          0 of 46
Command ===>                                                      Scroll CSR
46 rows fetched                                                   Format TABL
       CUSTOMER                  COMPANY                NO_SHARES
       #1                        #2                            #3
       CHARACTER(25)             CHARACTER(20)            INTEGER
       PU--+----1----+----2----> PU--+----1----+----> <---+----1>
000000 ****  Top of data  ****
000001 RB_DEMO                   IBM                          180
000002 CINZIA                    CALSEDONIA                    18
000003 LUCIA NORD                ELETTRICA   OSRAM              37
000004 MAIALINO                  SALUMI                        53
000005 LUCIANI                   IBM                           32
000006 SERENELLA                 TEMPOTEST                     37
000007 FORMY                     NORCINERIA                    53
000008 RB DEMO                   IBM                           -5
000009 MARIARITA                 DAMIGIANE E BOTTI             43
000010 FIORELLA                  PIANTE E FIORI                88
000011 FRASCHETTA                VINO E PANE                   65
 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 36-20 File Manager/DB2 - Table Edit panel

5. Change the value in the NO_SHARES column to 0 for customer RB_DEMO's holdings in IBM.
6. Enter `SAVE` on the command line.

   The change is saved and the message `Commit issued` is displayed, as shown in Figure 36-21.

```
 Process   Options   Utilities   Help
FM/DB2 (DB1E)                    Table Edit                    Commit issued
Command ===>                                                      Scroll CSR
TABLE AMINTOR.CUST_DETAILS                                        Format TABL
       CUSTOMER                  COMPANY                NO_SHARES
       #1                        #2                            #3
       CHARACTER(25)             CHARACTER(20)            INTEGER
       PU--+----1----+----2----> PU--+----1----+----> <---+----1>
000000 ****  Top of data  ****
000001 RB_DEMO                   IBM                          180
000002 CINZIA                    CALSEDONIA                    18
000003 LUCIA NORD                ELETTRICA   OSRAM              37
000004 MAIALINO                  SALUMI                        53
000005 LUCIANI                   IBM                           32
000006 SERENELLA                 TEMPOTEST                     37
000007 FORMY                     NORCINERIA                    53
000008 RB DEMO                   IBM                            0
000009 MARIARITA                 DAMIGIANE E BOTTI             43
000010 FIORELLA                  PIANTE E FIORI                88
000011 FRASCHETTA                VINO E PANE                   65
 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 36-21 File Manager/DB2 - Table Edit entry panel with corrected data saved

7. You can check the value you entered by switching to hexadecimal presentation. Enter the command `HEX ON` and press Enter. The data is displayed as shown in Figure 36-22.

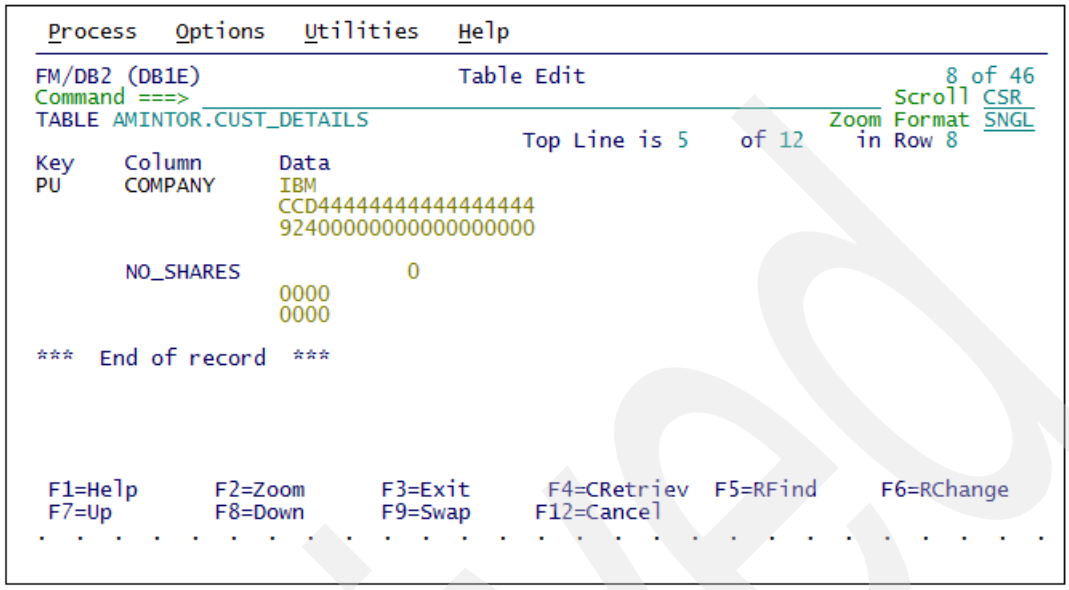


Figure 36-22 Hexadecimal edition

36.3 Summary of scenario 3

In this chapter we described the various components that make up the CICS and DB2 environments in our system and explained how they are set up.

We reviewed these topics:

- The processing performed by the CICS DB2 Trader application
- A process that used Debug Tool, running under CICS, to identify a problem with the logic in the application
- File Manager/DB2 capability to correct the data that resulted from the problem

37

Using WSim to create workloads

In this chapter we describe how to create workloads to be monitored with Application Performance Analyzer. We create several workloads:

MYTD	CICS VSAM using MYTD
TDB2	CICS DB2 using TDB2
TRADERB	Batch VSAM using TRADERB
TRADERD	Batch DB2 using TRADERD
TRADERI	Batch IMS using TRADERI
VSAM	CICS VSAM with batch VSAM using both MYTD and TRADERB
DB2	CICS DB2 and batch DB2 using both TDB2 and TRADERD
BATCH	All batch using TRADERB, TRADERD, and TRADERI
ALLTRAD	All batch (TRADERB, TRADERD, and TRADERI) and all CICS (MYTD and TDB2)

37.1 Elementary testcase

To make things as flexible as possible, we create several small testcases. These testcases are pieces of a puzzle that we can group together to create a workload.

37.1.1 Elementary testcases for CICS

CICSON	CICS: Log on.
CICSOFF	CICS: Log off.

37.1.2 Elementary testcases for the MYTD transaction

MYTDON	MYTD: Start and log on.
MYTDCOM	MYTD: Select a company.
MYTDBUY	MYTD: Buy shares.
MYTDSEL	MYTD: Sell shares.
MYTDQUO	MYTD: Quotation.
MYTDOFF	MYTD: Log off.

37.1.3 Elementary testcases for the TDB2 transaction

TDB2ON	TDB2: Start and log on.
TDB2COM	TDB2: Select a company.
TDB2BUY	TDB2: Buy shares.
TDB2OFF	TDB2: Log off.
TDB2SEL	TDB2: Sell shares.
TDB2QUO	TDB2: Quotation.

37.1.4 Elementary testcases for batch

TSOON	Log on.
TRADERB	Submit.
TRADERD	Submit.
TRADERI	Submit.
TSOOFF	Log off.

37.2 Workload definition

We now have to combine several testcases to create our workloads, as defined below.

37.2.1 MYTD: CICS VSAM using MYTD

MYTD workload is a WSim GROUP that contains the following WSim testcases:

- CICSON: CICS: Log on.
- MYTDON: MYTD: Start and log on.
- MYTDCOM: MYTD: Select a company.
- MYTDBUY: MYTD: Buy shares.
- MYTDSEL: MYTD: Sell shares.
- MYTDQUO: MYTD: Quotation.
- MYTDOFF: MYTD: Log off.
- CICSOFF: CICS: Log off.

37.2.2 TDB2: CICS DB2 using TDB2

TDB2 workload is a WSim GROUP that contains the following WSim testcases:

- CICSON: CICS: Log on.
- TDB2ON: TDB2: Start and log on.
- TDB2COM: TDB2: Select a company.
- TDB2BUY: TDB2: Buy shares.
- TDB2SEL: TDB2: Sell shares.
- TDB2QUO: TDB2: Quotation.
- TDB2OFF: TDB2: Log off.
- CICSOFF: CICS: Log off.

37.2.3 TRADERB: batch VSAM using TRADERB

TRADERB workload is a WSim GROUP that contains the following Wsim testcases:

- TSOON: Log on.
- TRADERB: Submit.
- TSOOFF: Log off.

37.2.4 TRADERD: batch DB2 using TRADERD

TRADERD workload is a WSim GROUP that contains the following WSim testcases:

- TSOON: Log on.
- TRADERD: Submit.
- TSOOFF: Log off.

37.2.5 TRADERI: Batch IMS using TRADERI

TRADERD workload is a WSim GROUP that contains the following WSim testcases:

- TSOON: Log on.
- TRADERI: Submit.
- TSOOFF: Log off.

37.2.6 VSAM: CICS VSAM with batch VSAM using both MYTD and TRADERB

VSAM workload is a WSim CYCLE that contains the following WSim GROUP:

- MYTD: CICS VSAM using MYTD.
- TRADERB: batch VSAM using TRADER

37.2.7 DB2: CICS DB2 and batch DB2 using both TDB2 and TRADERD

DB2 workload is a WSim CYCLE that contains the following WSim GROUP:

- TDB2: CICS VSAM using MYTD
- TRADERD: batch VSAM using TRADERB

37.2.8 BATCH: All batch using TRADERB, TRADERD, and TRADERI

BATCH workload is a WSim GROUP that contains the following WSim testcases:

- TSOON: Log on.
- TRADERB: Submit.
- TRADERD: Submit.
- TRADERI: Submit.
- TSOOFF: Log off.

37.2.9 ALLTRAD: all batch (TRADERB, TRADERD, and TRADERI) and all CICS (MYTD and TDB2)

ALLTRAD workload is a WSim CYCLE that contains the following WSim GROUP:

- TDB2: CICS DB2 using TDB2
- MYTD: CICS VSAM using MYTD
- BATCH: all batch using TRADERB, TRADERD, and TRADERI

Note: We do not describe all of the definitions that have to be done to create all the workloads, but only one.

37.3 WSim definitions required for the WSim Cycle named DB2

We go through all steps required to create the DB2 workload:

1. PROJECT: Definition.
2. Test CASE: Definition.
3. Test CASE: Interactive Data Capture.
4. Test CASE: Change the IDC log data set.
5. Test GROUP: Definition.
6. Test CYLCE: Definition.
7. User Table: Creation.
8. Random number generation.
9. Set a delay before transmission.
10. Create a network and schedule its associated workload.

37.3.1 Project: Definition

From the main WSim Test Manager pane, select option **P**, as shown in Figure 37-1, to add a new project.

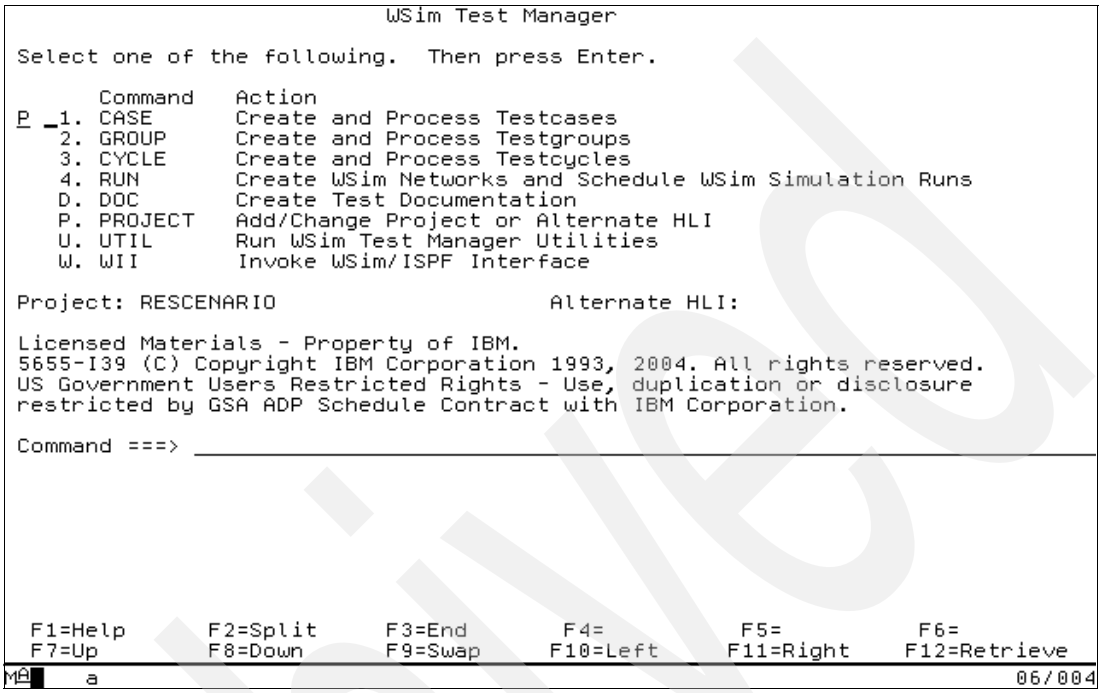


Figure 37-1 WSim Test Manager: main panel (project selection)

Type **add** on the command line, as shown in Figure 37-2, and press Enter.

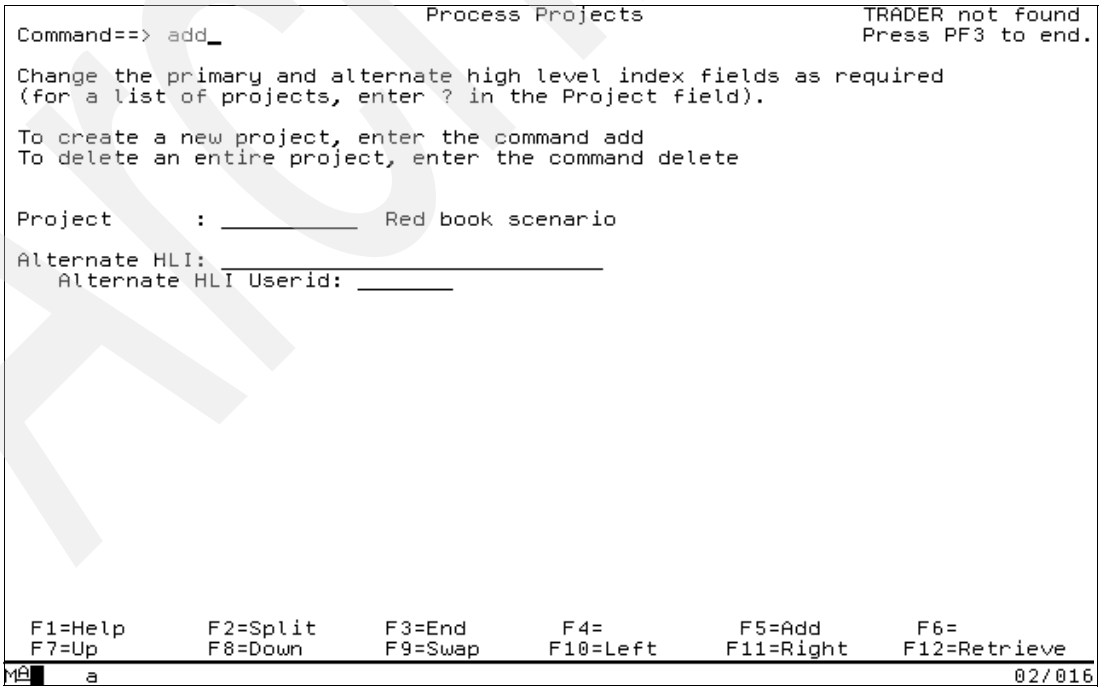


Figure 37-2 Add a new project

We give the project name, description, and high level qualifier, as shown in Figure 37-3, and we press Enter. Workload Simulator creates files to manage our project.

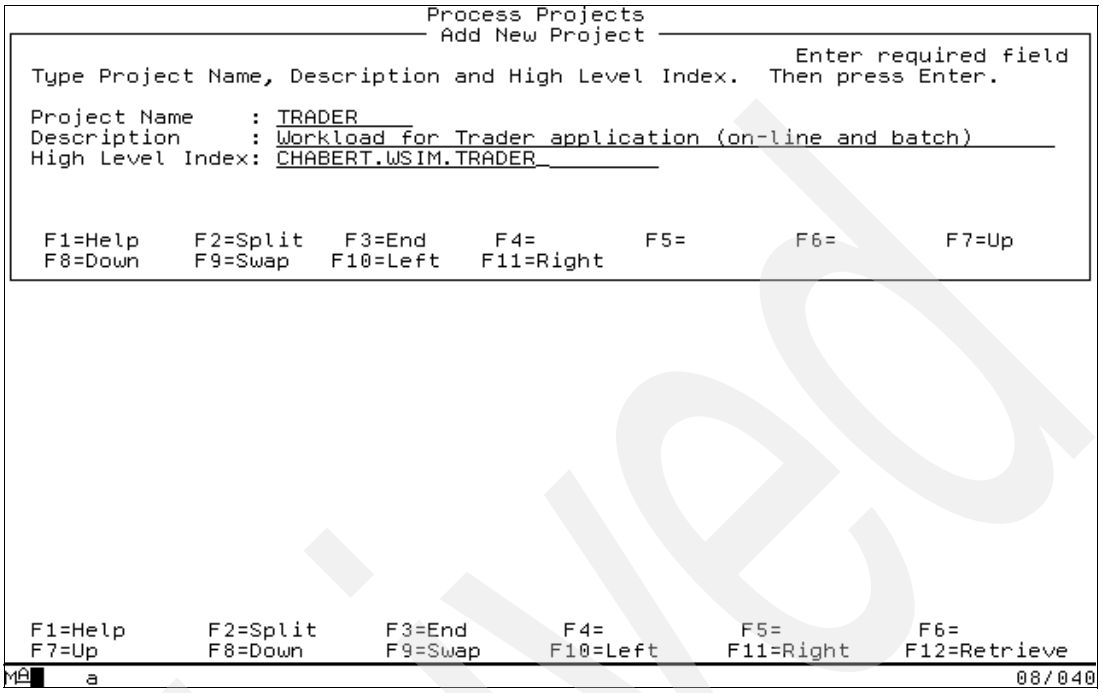


Figure 37-3 Project definition parameters

37.3.2 Test Case: Definition

When WSim has defined the project, we select from the main WSim Test Manager panel option **1** to create our first testcase, as shown in Figure 37-4.

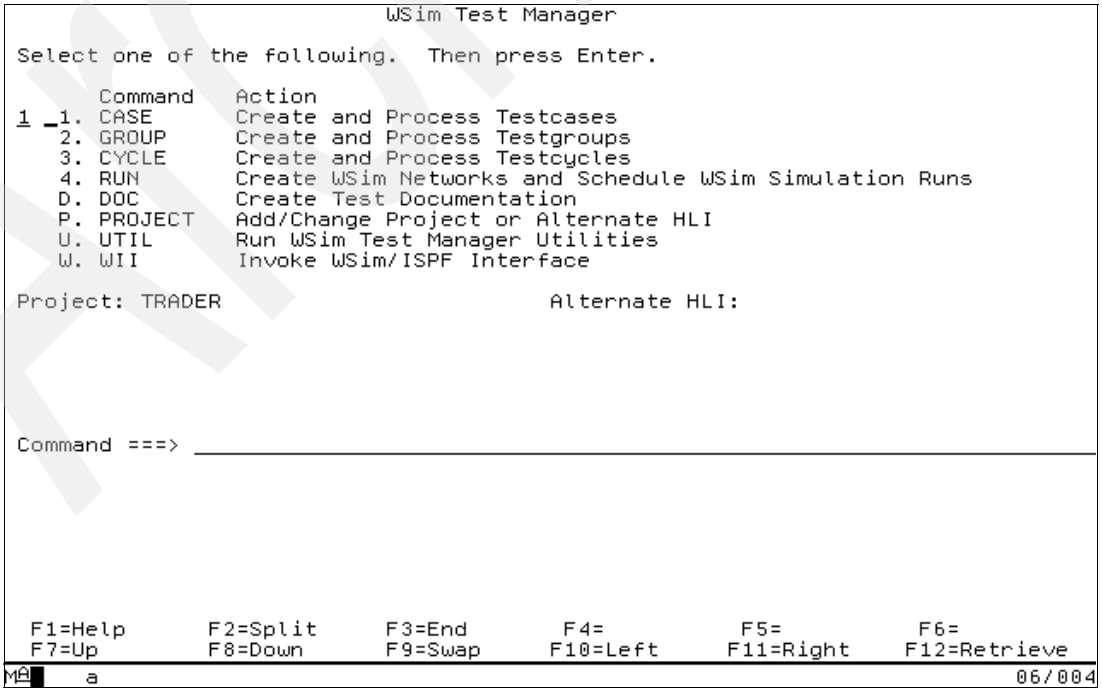


Figure 37-4 WSim Test Manager: main panel (testcase selection)

Type the **add** command line to create a new test case, as shown in Figure 37-5.

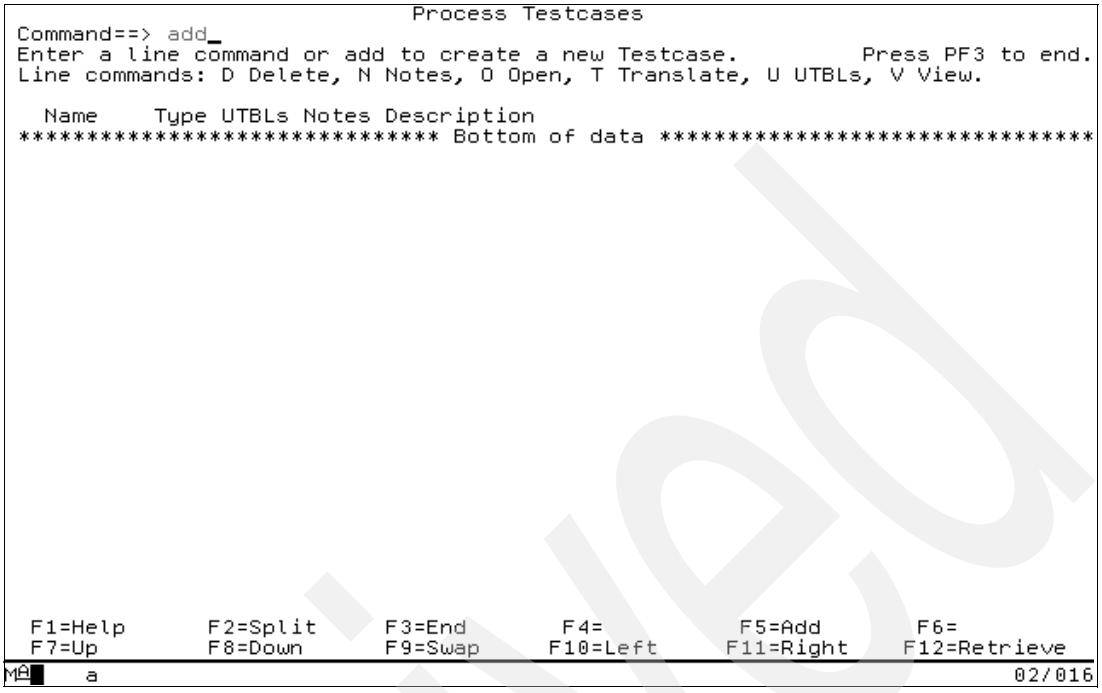


Figure 37-5 Add a new test case

We give the testcase name and description, and we use the Interactive Data Capture to create the script, as shown in Figure 37-6.

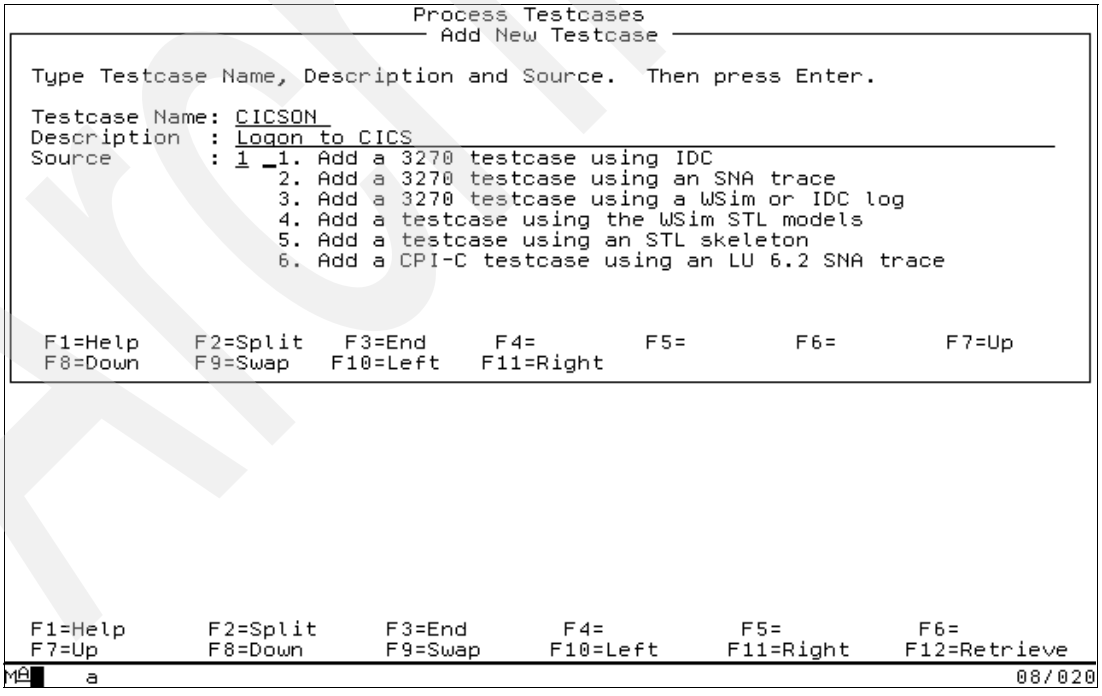


Figure 37-6 Testcase definition parameters

37.3.3 Test Case: Interactive Data Capture

We now start to record the first step of our first scenario by selecting option **1**, as shown in Figure 37-7.

```
IDCMAIN          WSim Interactive Data Capture (IDC) Utility

Select one of the following, then press Enter.

1 1. Start a session with a host application and capture data

  2. Generate an STL program from captured data
  3. Generate a message generation deck from captured data

  4. End the IDC utility program









WSim Version 1 Release 1.0.1  Program Number 5655-I39

Licensed Materials - Property of IBM
5655-I39 (C) Copyright IBM Corporation 1976, 2004.  All Rights Reserved.
US Government Users Restricted Rights - Use, duplication or disclosure
restricted by GSA ADP Schedule Contract with IBM Corporation.
F1=Help  F3=Exit  F12=Cancel
MA    a                                                          05/004
```

Figure 37-7 WSim IDC: start data capture

First we must log on to CICS. Therefore, we specify its application name, as shown in Figure 37-8.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . CICSC23G
  Logon mode name . . . . . . . . . . LSX32703 (Optional)
  Logon user data . . . . . . . . . . ____________________ (Optional)


IDC log data set name . . . . . . . . CHABERT.WTMUSER.IDCLOG(CICSON)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . Y        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1      (PAn, PFnn, CLEAR, or ATTN)









F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
MA    a                                                          06/040
```

Figure 37-8 WSim IDC: application name specification

Our first test case is named CICSON, because we just have to sign on, as shown in Figure 37-9.

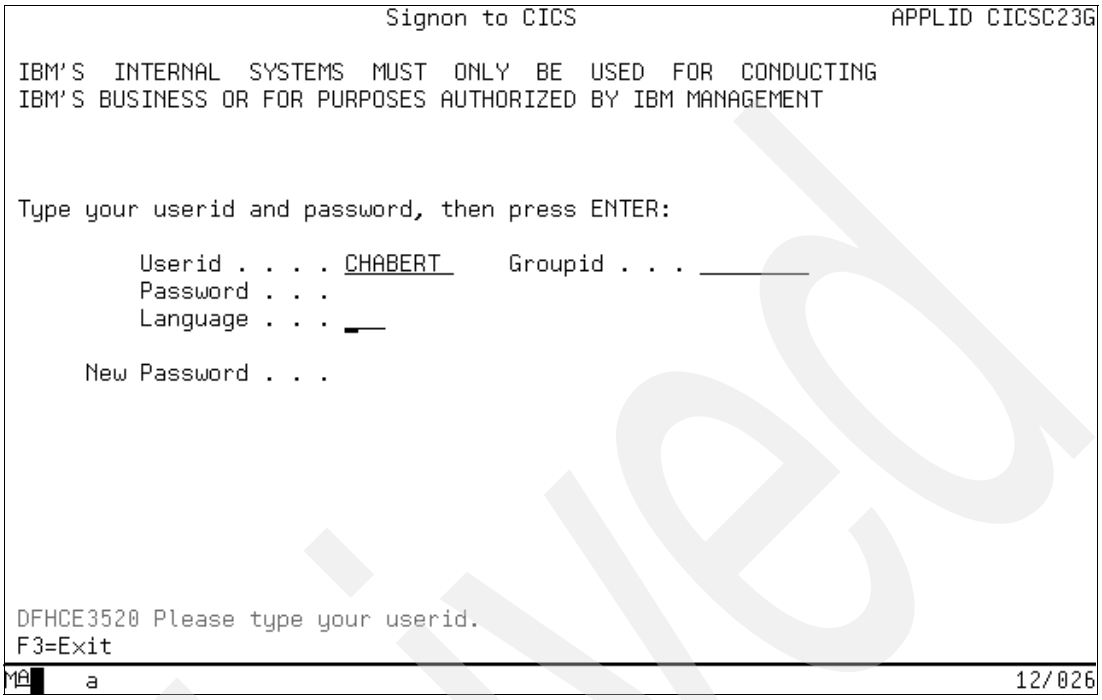


Figure 37-9 CICS sign-on

When our sign-on is complete, we exit from the Interactive Data Capture by using the IDC escape key defined in Figure 37-8 on page 1186, as shown here in Figure 37-10.

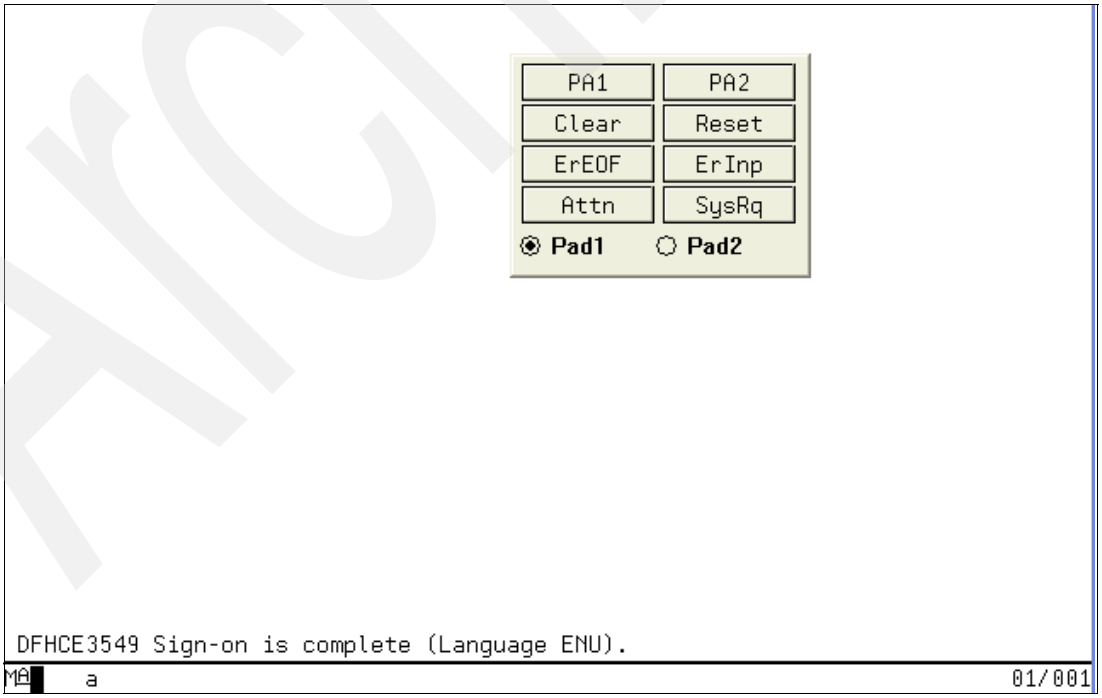


Figure 37-10 Escape key to exit from testcase CICSON

37.3.4 Test CASE: Change the IDC log data set

We do not have to stop the Interactive Data Capture, but just to change the log file used by WSim to record data to be used later to create the associated STL statements. Select option **6** from the WSim IDC: Escape Actions screen, as shown in Figure 37-11.

```
IDCESCA              WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

6 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : ON
Current IDC log data set: CHABERT.WTMUSER.IDCLOG(CICSON)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
MA    a                                                        06/004
```

Figure 37-11 WSim IDC: Change IDC log data sets

The member name we give for the new log data set is the name of the next testcase TDB2ON (to log on to TDB2). When we press Enter, WSim closes the previous log and opens the new one, as shown in Figure 37-12.

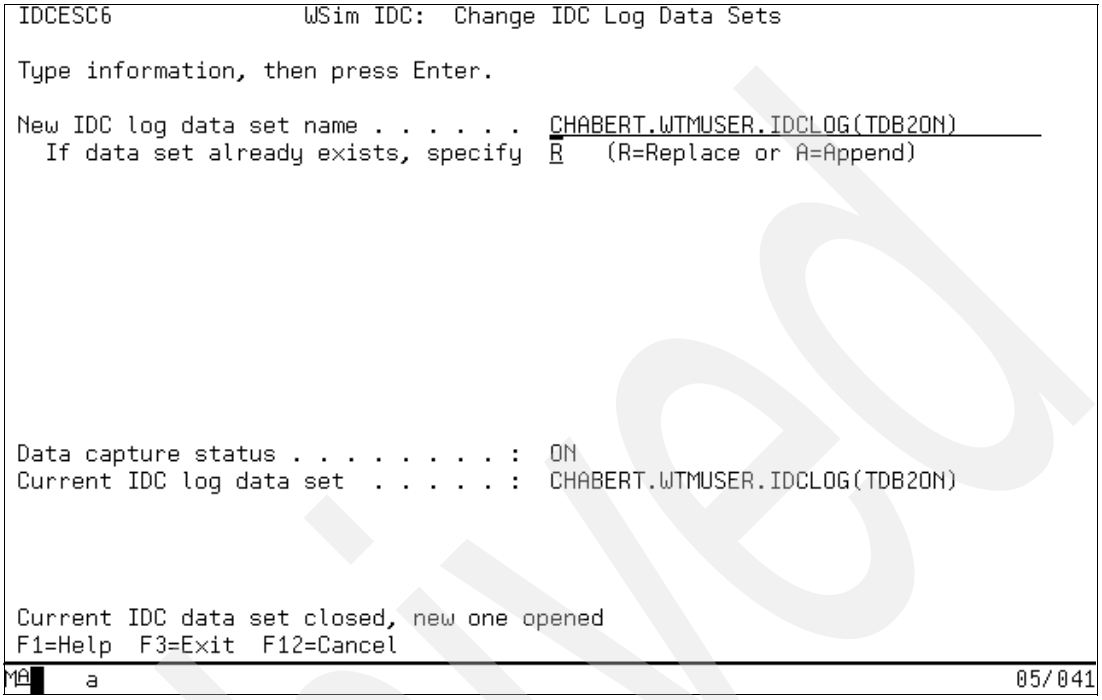


Figure 37-12 WSim IDC: new log data set for TDB2ON

When we press the PF3 key, we return to the CICS screen. We enter the TDB2 transaction and we log on to the Trader application. Our second testcase, TDB2ON, is now completed by selecting the WSim escape key, as shown in Figure 37-13, to stop it.

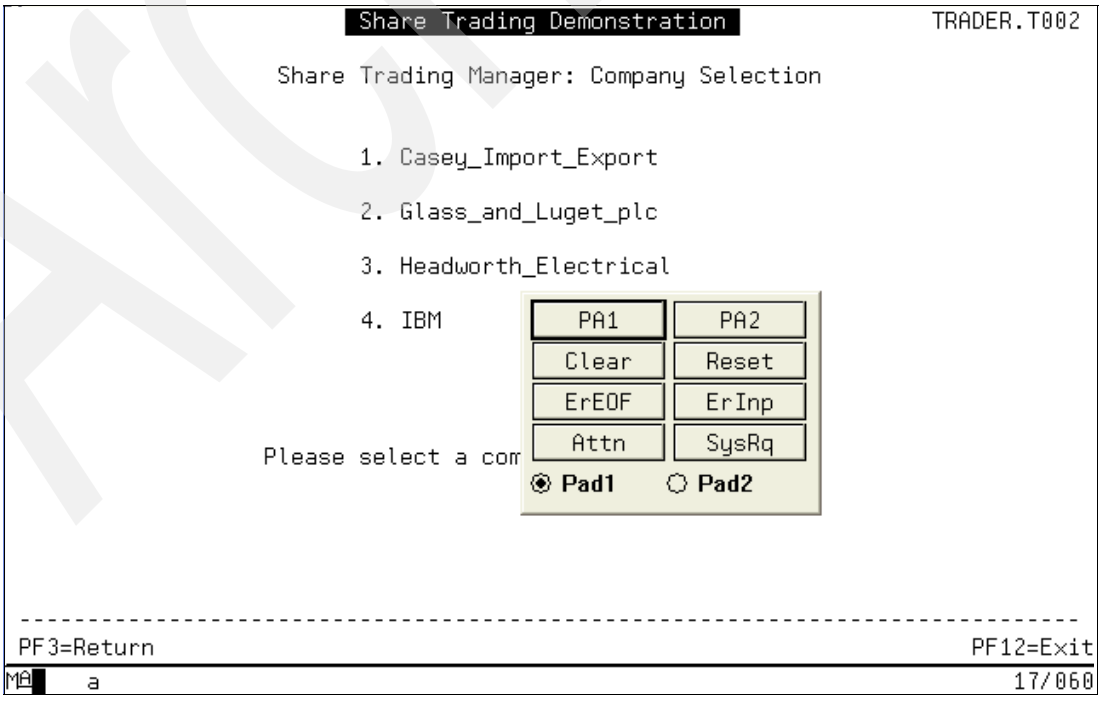


Figure 37-13 Escape key to exit from testcase TDB2ON

We do not have to stop the full IDC process, but we just must change the IDC log data set, as shown in Figure 37-14.

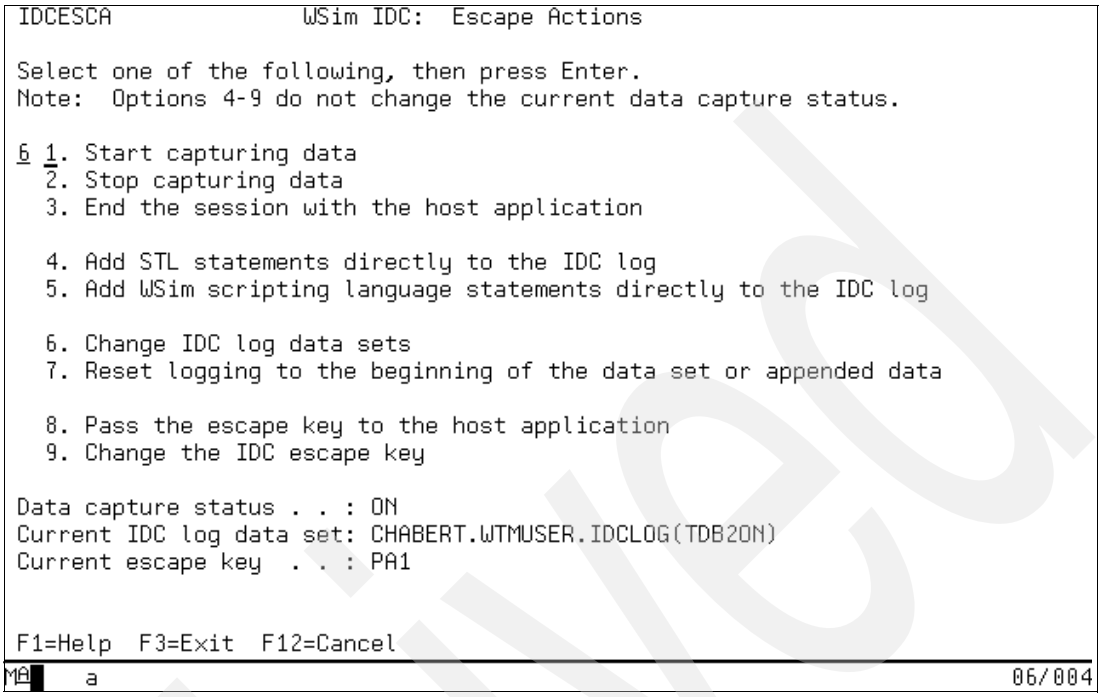


Figure 37-14 WSim IDC: Change IDC log data sets

The new member name TDB2COM is the name of the next testcase (to select the company name). When we press Enter, WSim closes the previous log and opens the new one, as shown in Figure 37-15.

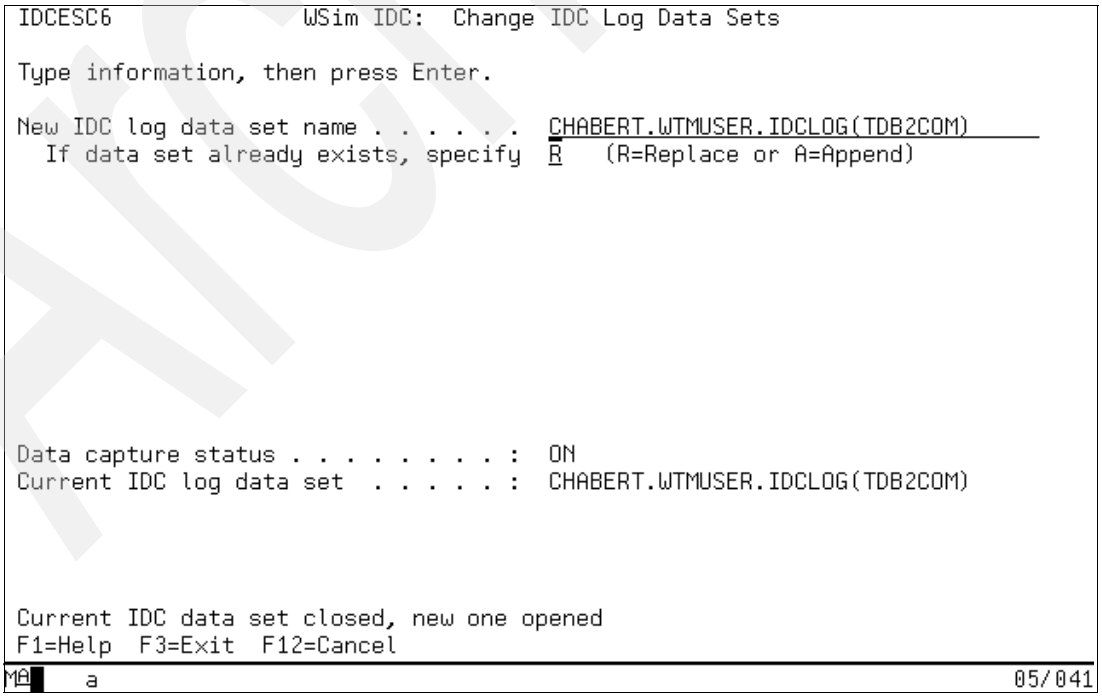


Figure 37-15 WSim IDC: new log data set for TDB2COM

When we press the PF3 key, we return to the previous CICS screen. We select the company. Our third testcase ends here. It is why we select the WSim escape key, as shown in Figure 37-16.

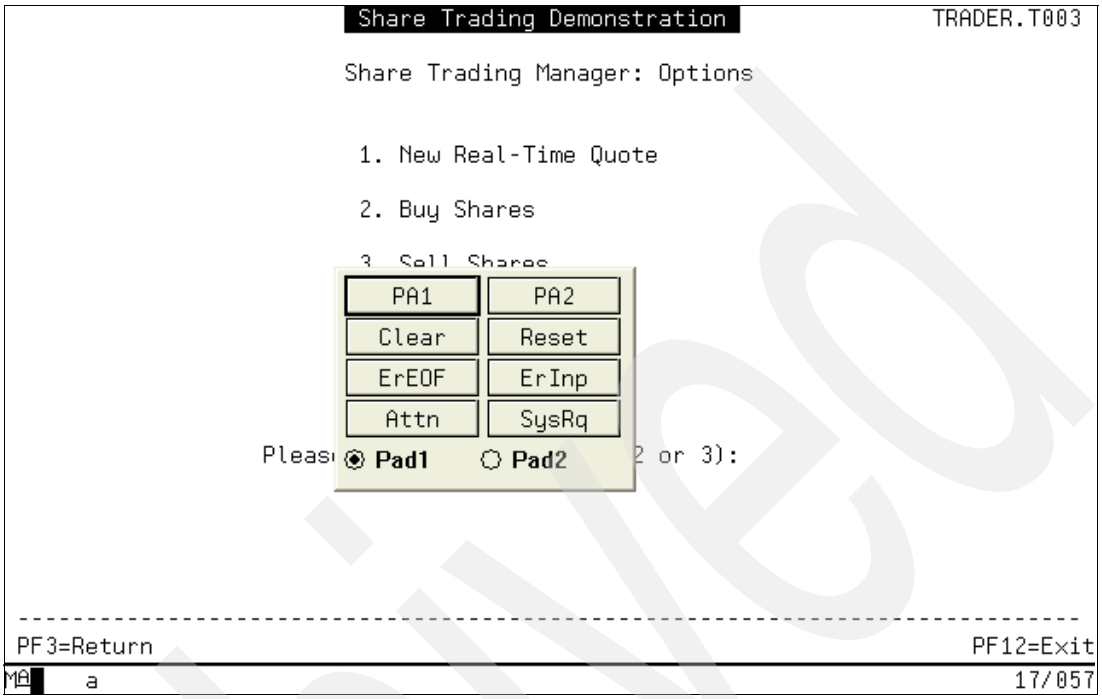


Figure 37-16 Escape key to exit from testcase TDB2COM

One more time we do not want to stop the Interactive Data Capture. We have to continue the same process:

1. Change the Interactive Data Capture log data set, as shown in Figure 37-14 on page 1190.
2. Start the Interactive Data Capture for the next testcase with a new log data set, as shown in Figure 37-15 on page 1190.
3. Run the testcase to record it.
4. Use the escape key PA1 when we reach the end of the current testcase, as shown in Figure 37-16.

Do this until all CICS test cases have been recorded. For the testcase CICSOFF we enter the CESF LOGOFF transaction. We exit from both CICS and the Interactive Data Capture, as shown in Figure 37-17.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . CICSC23G
  Logon mode name . . . . . . . . . . LSX32703 (Optional)
  Logon user data . . . . . . . . . . ____________________ (Optional)


IDC log data set name . . . . . . . . CHABERT.WTMUSER.IDCLOG(CICSOFF)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . Y        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1      (PAn, PFnn, CLEAR, or ATTN)




ITP1508I SESSION ENDED WITH APPLICATION CICSC23G
F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
MA    a                                                        06/040
```

Figure 37-17 WSim IDC: end session with host application

WSim Test Manager completes, with the first log data set that has been created, the creation of our first test case. To create another testcase we enter the **add** command line and we press Enter, as shown in Figure 37-18.

```
                              Process Testcases                 Row 1 of 1
Command==> add_
Enter a line command or add to create a new Testcase.         Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
_ CICSON    V            Logon to CICS                              Added
******************************* Bottom of data ********************************









 F1=Help      F2=Split     F3=End       F4=          F5=Add       F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA    a                                                          02/016
```

Figure 37-18 Create a new test case

We give the testcase name and description and select option **3** to specify that we have the Interactive Data Capture log data set required for its creation, as shown Figure 37-19.

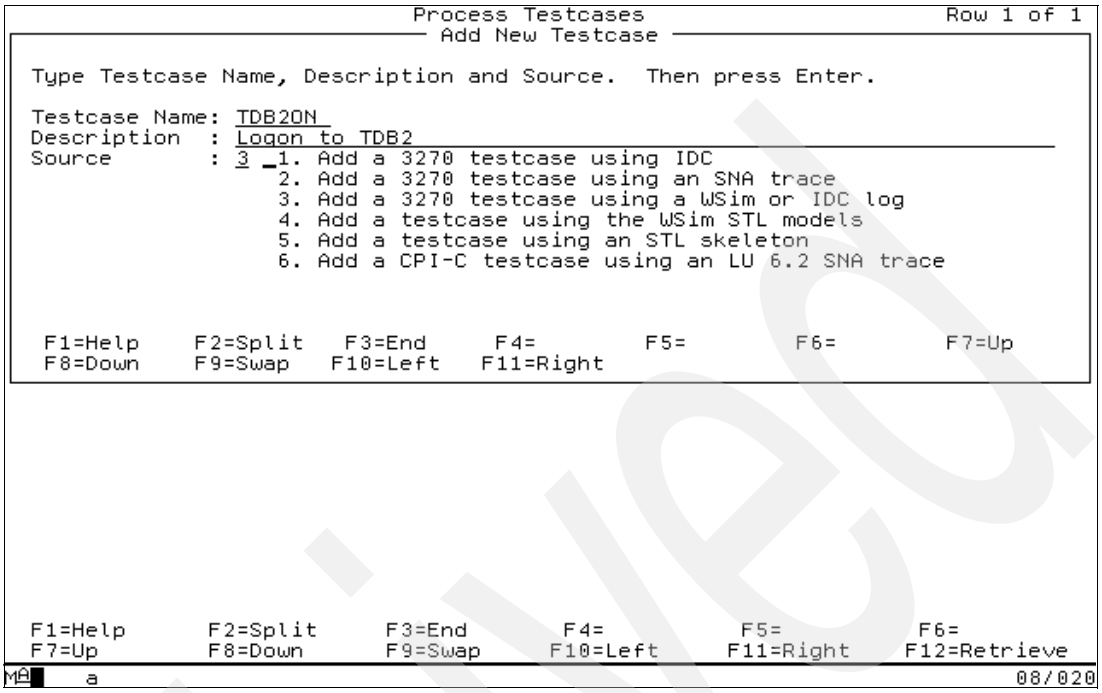


Figure 37-19 Test case creation using an existing IDC log

WSim Test Manager displays a pop-up window where we specify the Wsim log data set that has to be used, as shown in Figure 37-20.

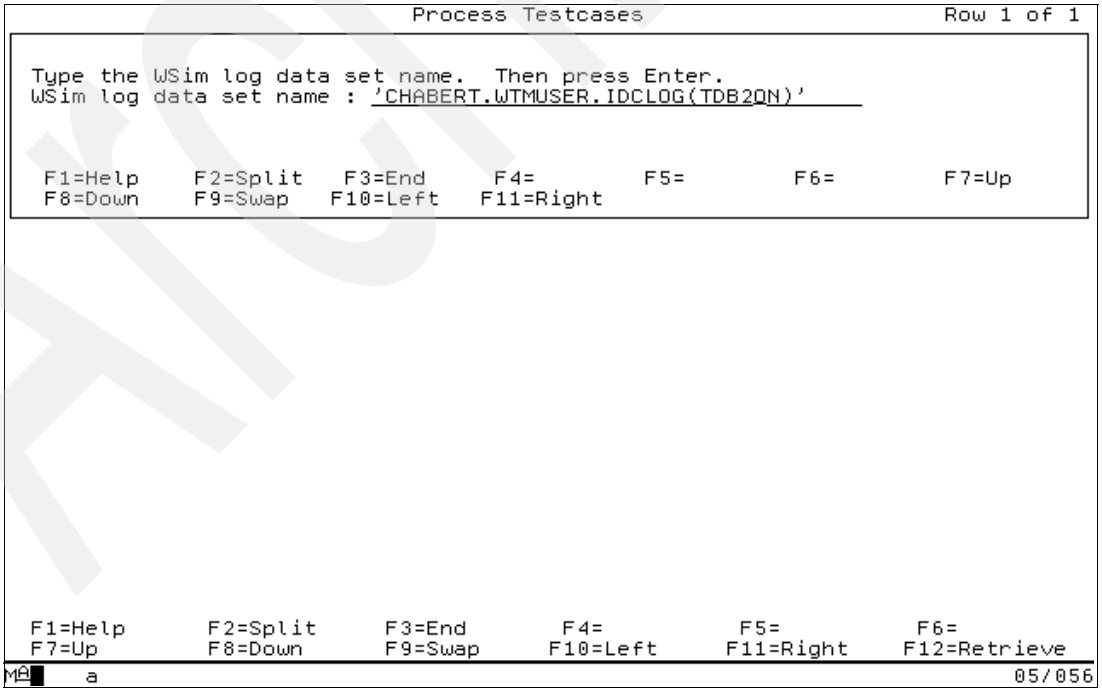


Figure 37-20 WSim log data set to be used

When we enter the PF3 key, WSim uses this log data set to create the new STL program.

We have to do the same process for all of the other CICS testcases that have been recorded into a WSim log data set:

1. Add command line to create a new test case, as in Figure 37-18 on page 1192.
2. Specify the testcase name and description, as in Figure 37-19 on page 1193.
3. Specify that an existing log data set is to be used, as in Figure 37-20 on page 1193.

To have our full workload, we now have to create the TSO workload. We use the same method as for the CICS workload:

1. Start an Interactive Data Capture, but with TSO instead of CICS, as shown in Figure 37-21.
2. Create several log data sets, one for each of the test cases below:
   - TSOON
   - TRADERB
   - TRADERD
   - TRADERI
   - TSOOFF
3. Generate from each its associated STL program.

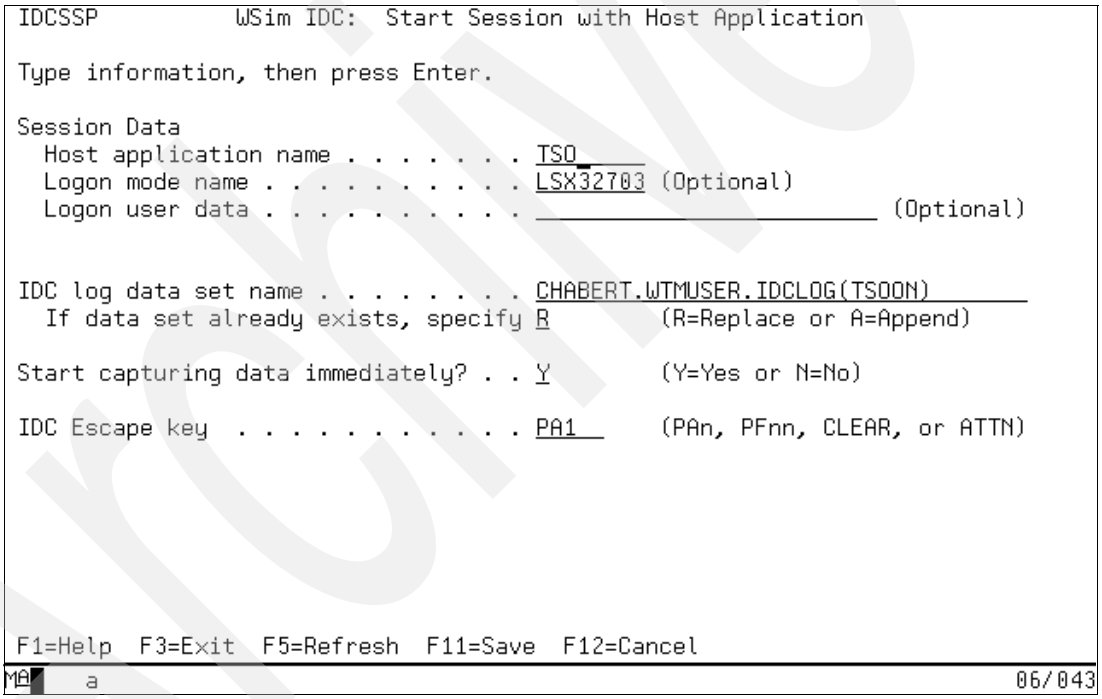


Figure 37-21 WSim IDC: host application name for TSO

Figure 37-22 shows the list of all our test cases.

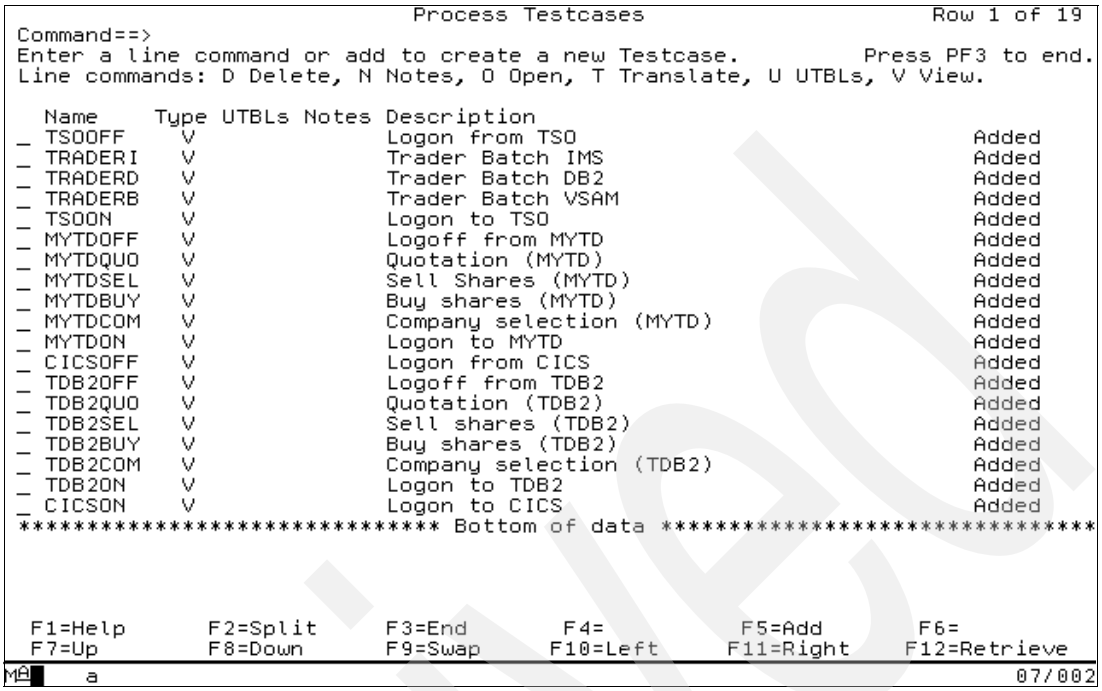


Figure 37-22 List of test cases

The test cases are pieces used to create a workload. We have to group several of them to create a real workload. The first step is to create a group.

37.3.5 Test GROUP: Definition

A testgroup is an ordered list of testcases items. Testgroups encourage modular testcases for us. A logon testcase is included in several testgroups. As a result, if the logon process changes, only the logon (CICSON or TSOON) testcase must be changed, even if it is used in a number of different testgroups.

To create a new testgroup, enter the **ADD** command in the command line at the top of the Process Testgroups panel, or press PF, as shown in Figure 37-23.

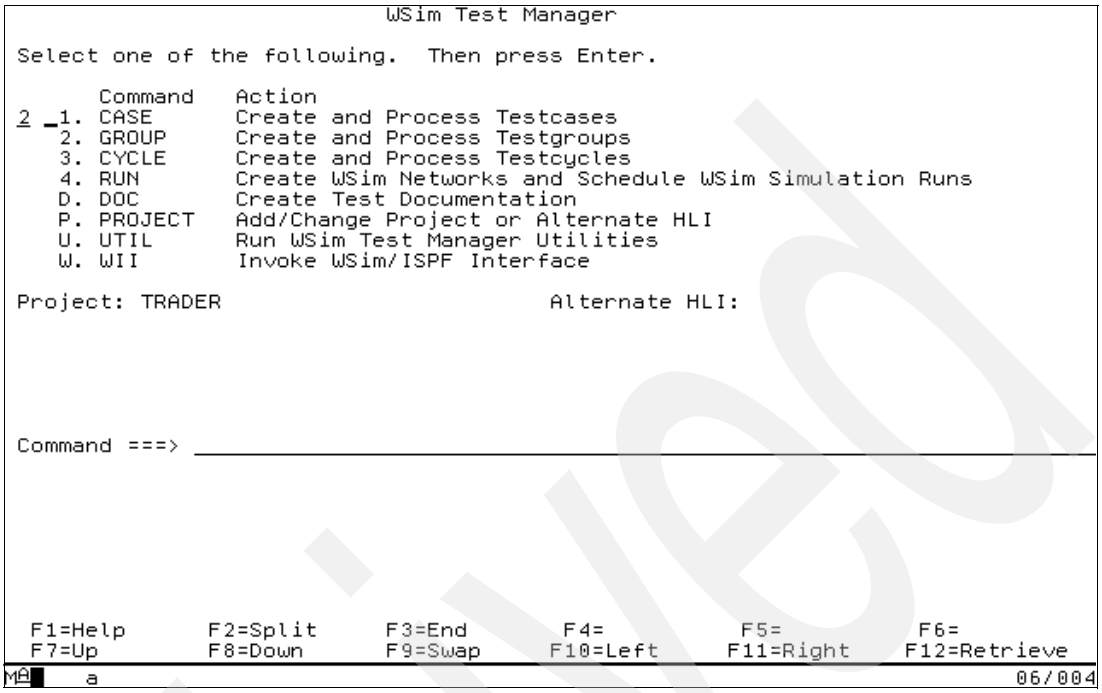


Figure 37-23 Adding a testgroup

WSim Test Manager displays a screen where we have to specify the testgroup name, an optional description, and the type of testgroup (V for VTAMAPPL, T for TCP/IP, or C for CPI-C), as shown in Figure 37-24.

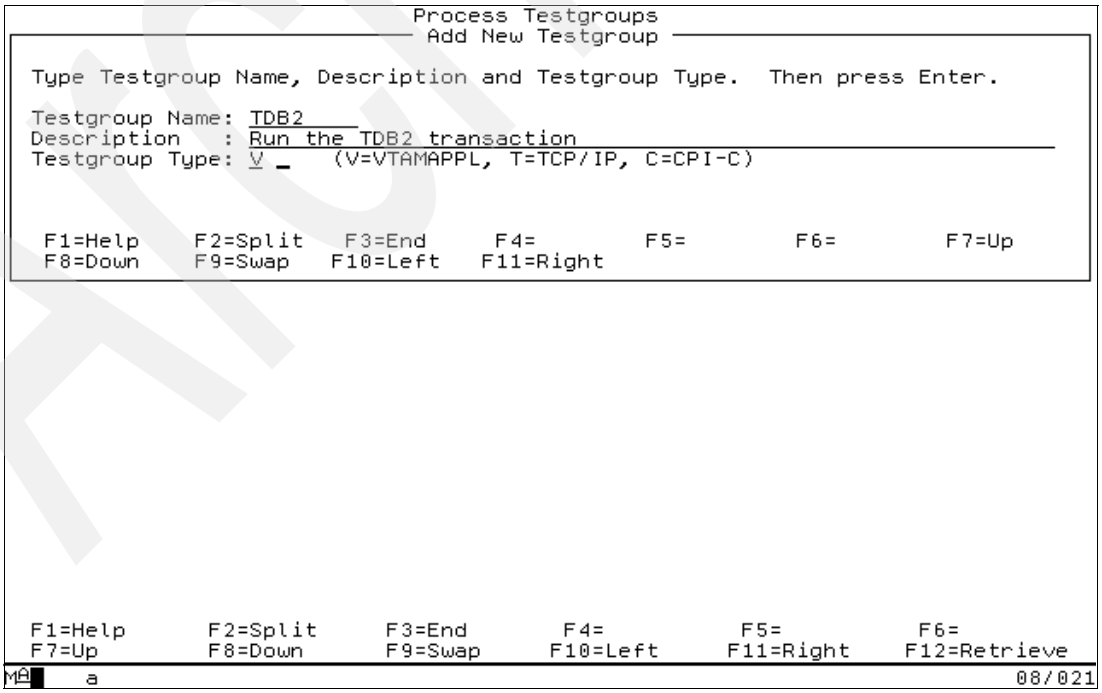


Figure 37-24 Testgroup parameters

A selection list is displayed containing all possible test items that can be referenced by this type of testgroup. A panel displays the test items, which are sorted in alphabetical order. Select the test items by specifying the reference order, as shown in Figure 37-25.

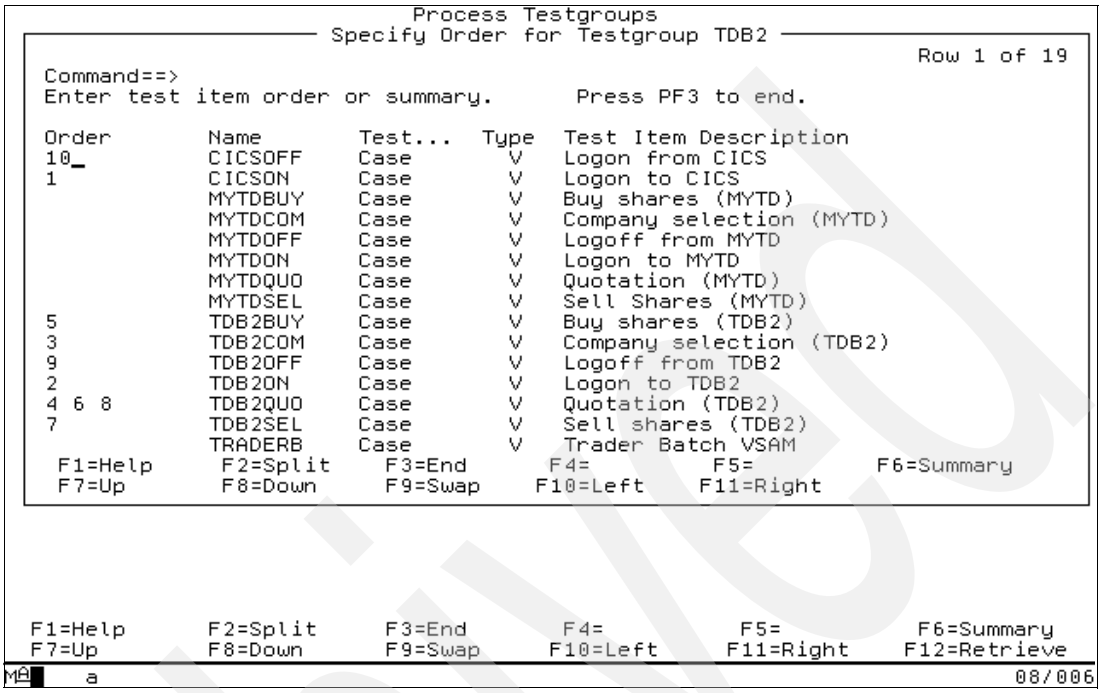


Figure 37-25 Testgoup: Testcases selection

Press PF3 to return to the Process Testgroups panel, as shown in Figure 37-26.

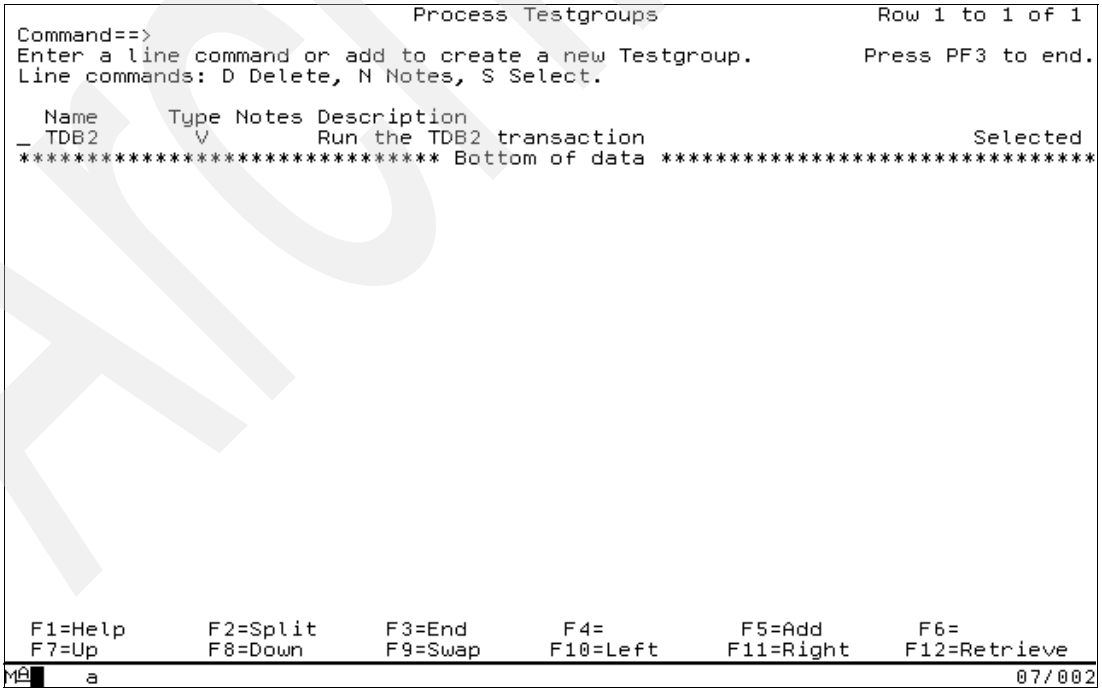


Figure 37-26 Process Testgroups panel

Do the same process for all of our testgroups. The full list is shown in Figure 37-27.

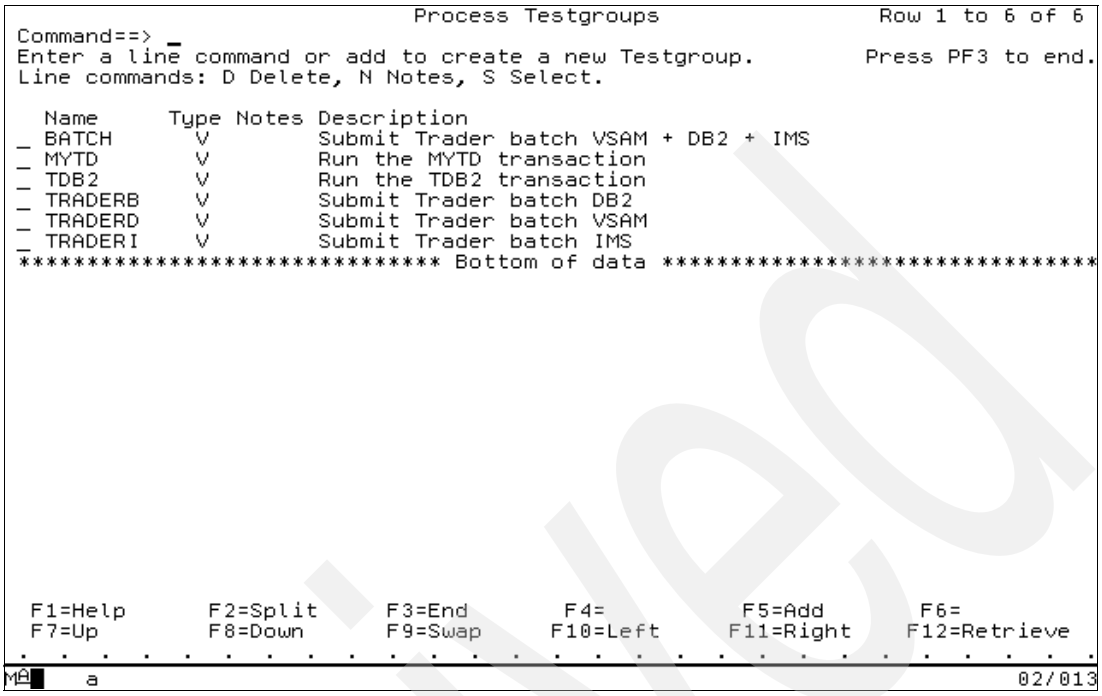


Figure 37-27 List of our testgroups

We now group several testcases or testgroups to a testcycle to create a more complex scenario.

37.3.6 Test CYLCE: Definition

A testcycle is an ordered list of test items (cases and groups). A testcycle is another level of organization for test items. A testcycle can result in a very complex set of test items, and is a simple and powerful way of handling large numbers of modular scripts.

From the WSim Test Manager main menu select option **3** or enter the `CYCLE` command, as shown in Figure 37-28.

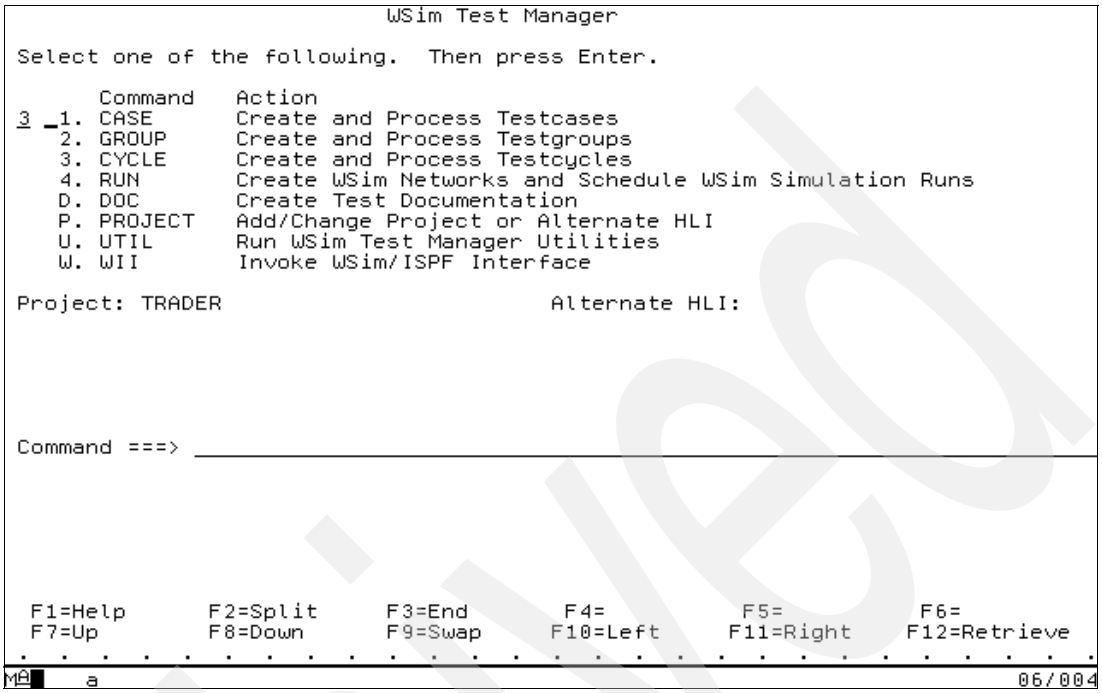


Figure 37-28 Create a testcycle

Wsim displays the Add New Testcycle panel, as shown in Figure 37-29.

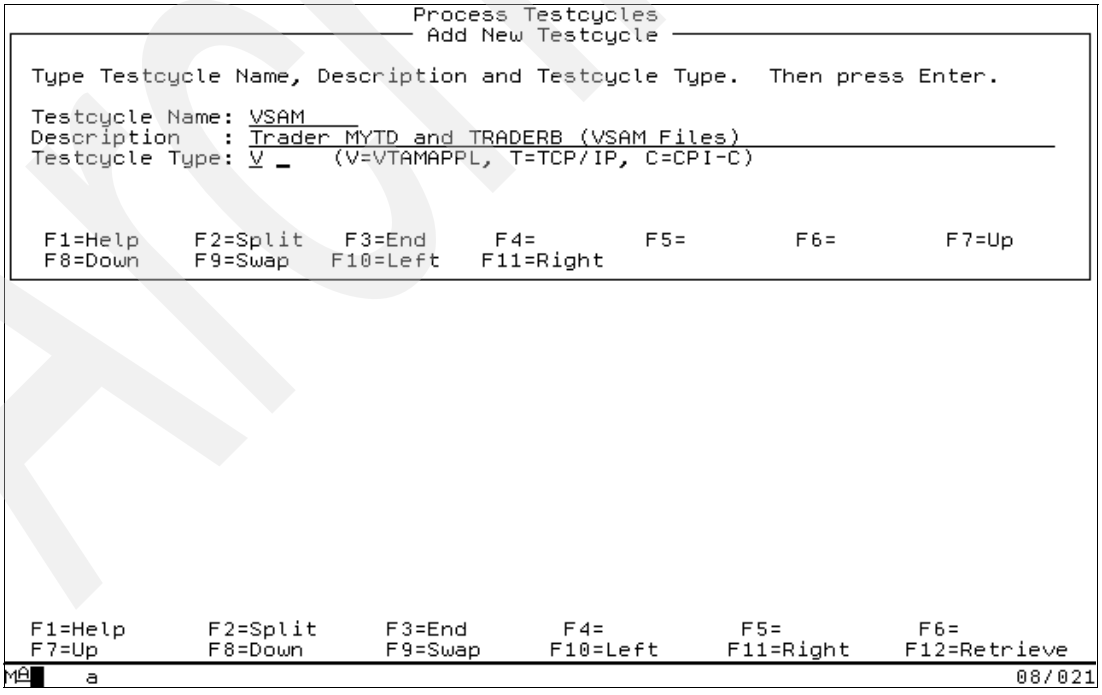


Figure 37-29 Add New Testcycle panel

We specify a testcycle name, an optional description, and the type of testcycle (V for VTAMPPL, T for TCP/IP, or C for CPI-C).

Then a list of test items (cases and groups) that can be referenced by the testcycle is displayed by WSim Test Manager, as shown in Figure 37-30.

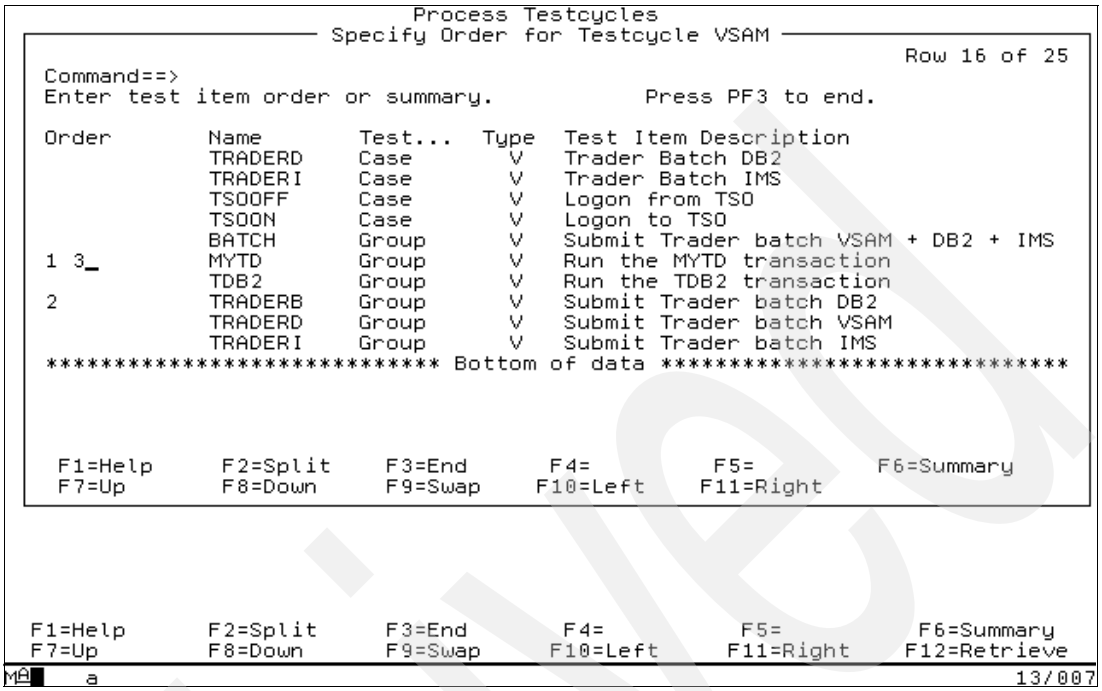


Figure 37-30 List of test items for testcycle

The test items are sorted in alphabetical order based on test item type. Select the test items by specifying the reference order. Press PF3 to return to the Process Testcycles panel.

When all our test cycles are defined, the following list (Figure 37-31) should be displayed.

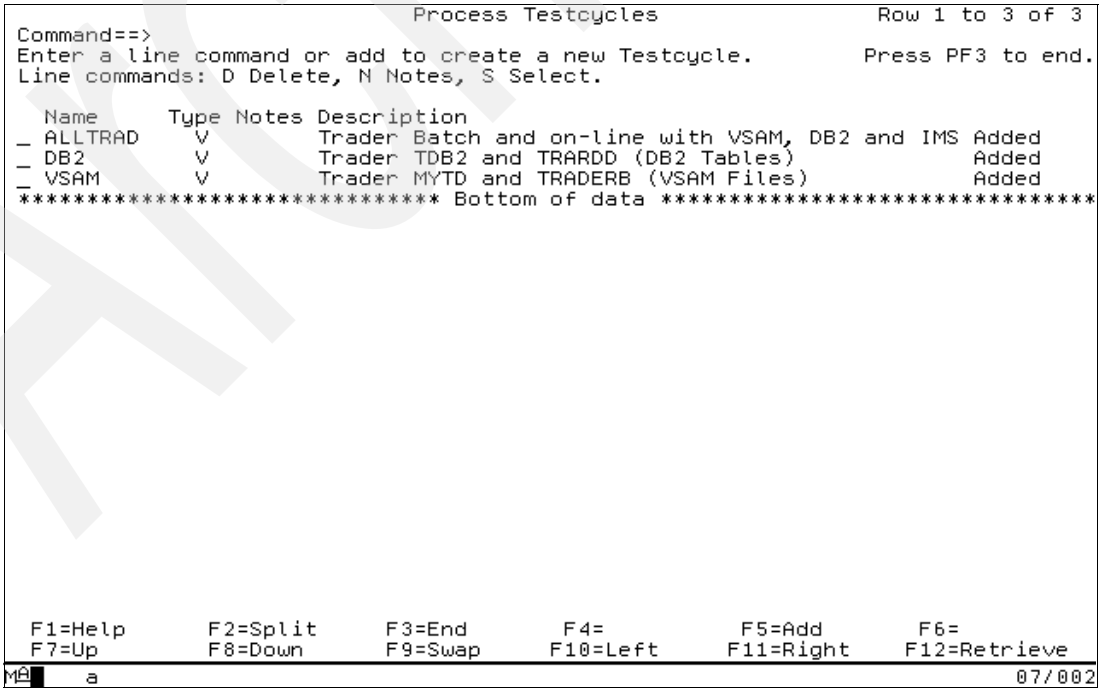


Figure 37-31 List of testcycles

We now must create user tables to:

- Select a CICS user ID/password.
- Select the company name used by Trader.

37.3.7 User Table: Creation

A User Data Table (UTBL) is defined in WSim as a list of string constants. The WSim Test Manager organizes each UTBL into fields, so that each line in a UTBL has one or more fields (for example, a user ID and a password).

WSim Test Manager can automatically generate WSim user data tables and the STL source code to use them. This function can be invoked by editing the STL source in the WSim Test Manager, as shown in Figure 37-32.

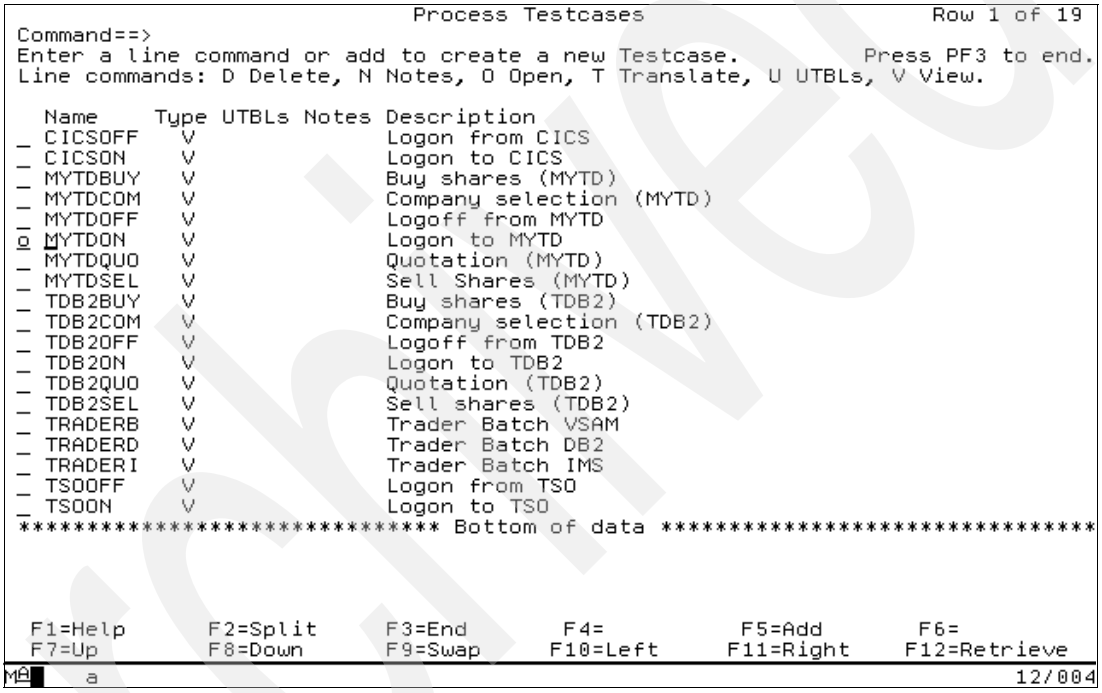


Figure 37-32 Open a testcase

To edit the STL source of MYTDON, type the UTBL command, place the cursor on the line containing the user ID to be replaced, and press Enter, as shown in Figure 37-33.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CHABERT.WSIM.TRADER.STL(MYTDON) - 01.01          Columns 00001 00072
Command ===> utbl                                            Scroll ===> CSR
000024
000025 /* 14:24:31.82 ITP1507I SESSION STARTED WITH APPLICATION CICSC23G */
000026
000027 /*-------------------------------------------------- 14244828 00001 */
000028 WTM_panel_ID = 'PNL00001'
000029 log 'WTM_panel_ID' WTM_panel_ID
000030 cursor(24,48)
000031 ereof
000032 cursor(1,1)
000033 charset 'field'
000034 type 'MYTD'
000035 transmit using enter
000036
000037 /*-------------------------------------------------- 14245414 00003 */
000038 WTM_panel_ID = 'PNL00002'
000039 log 'WTM_panel_ID' WTM_panel_ID
000040 cursor(9,42)
000041 ereof
000042 charset 'field'
000043 type 'eric'
000044 cursor(13,42)
000045 ereof
000046 upnd = '03897FC101AF1F87541AD8C827093B0180661A09AA'x
000047 userexit('ITPUMNDX',upnd)
000048 transmit using enter
000049
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
MA    a                                                          24/002
```

Figure 37-33 Create user table for CICS user ID - password

When the first pop-up window is displayed, choose **1** because a completely new UTBL is required, and the panel shown in Figure 37-34 is displayed.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
─ ┌──────────────── Create WSim User Table (UTBL) ─────────────────┐
E │                                                  Already in Use │ 0072
C │ Type the values for the following fields.  Then press Enter.    │ CSR
0 │                                                                 │
0 │ User Table Name         : CICSUSR                               │ /
0 │ Table Description       : User and password to logon to CICS    │
0 │                                                                 │ 1 */
0 │ Field Name              : CICSUSER                              │
0 │ Field Description       : CICS Userid                           │
0 │ Field Delimiter         : °__                                   │
0 │                                                                 │
0 │ Access Type (1,2 or 3) : 2  1.  Random                          │
0 │                             2.  Single Sequential               │
0 │                             3.  Single Sequential (repeated)    │
0 │                                                                 │
0 │                                                                 │
0 │  F1=Help    F2=Split   F3=End     F4=        F5=        F6=     │ 3 */
0 │  F7=Up      F8=Down    F9=Swap    F10=Left   F11=Right          │
0 └─────────────────────────────────────────────────────────────────┘
000040 cursor(9,42)
000041 ereof
000042 charset 'field'
000043 type 'eric'
000044 cursor(13,42)
000045 ereof
000046 upnd = '03897FC101AF1F87541AD8C827093B0180661A09AA'x
000047 userexit('ITPUMNDX',upnd)
000048 transmit using enter
000049
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
MA    a                                                          07/031
```

Figure 37-34 User table definition

A user table name and field name must be entered. The table and field descriptions are optional.

There are three access types:

- *Random* sets up STL code to randomly access the UTBL. This is useful for using a random spread of the whole UTBL.
- *Single Sequential* sets up STL code to access the UTBL in strict sequential order. When the end of the UTBL has been reached, no further accesses are allowed. This is useful for user IDs and passwords where there is a limited number of entries in the UTBL.
- *Single Sequential Repeated* sets up STL code to access the UTBL in sequential order and resets the UTBL pointer to the top of the UTBL when the bottom of the UTBL is reached. This is useful when accessing a limited number of UTBL rows but reusing the UTBL is desired.

The CICSUSER field is now defined. We must add a new field CICSPSWD to the existing user table. Edit the STL source, type the `UTBL` command, place the cursor on the line containing the encrypted password to be replaced, and press Enter. When the pop-up window is displayed, choose option **2**, as shown in Figure 37-35.

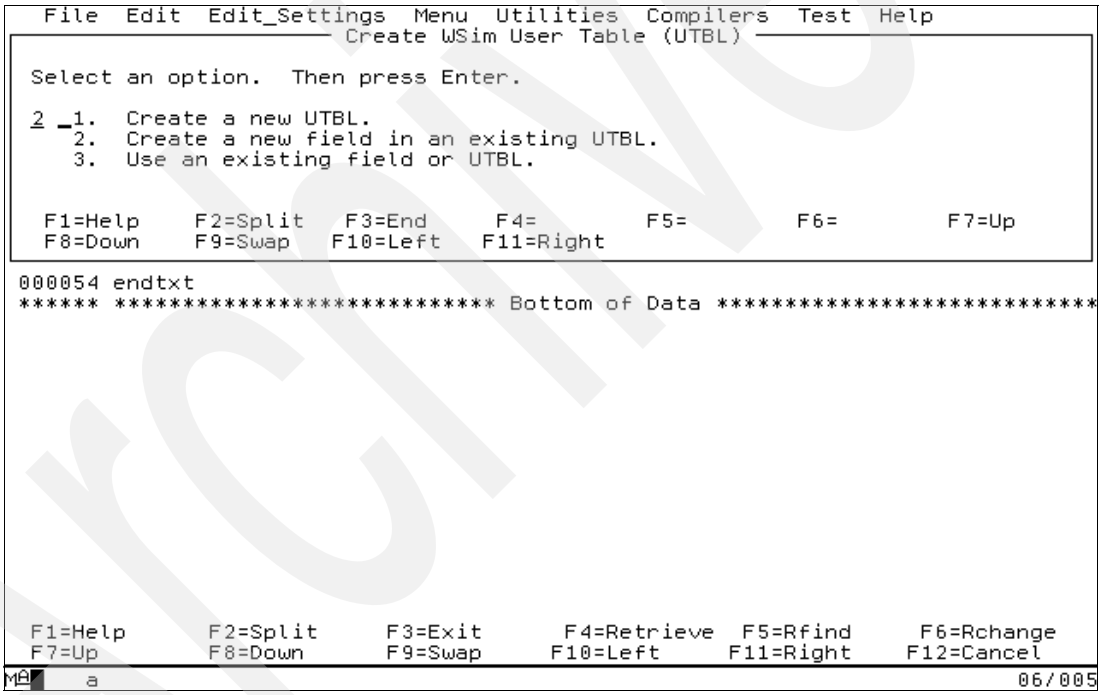


Figure 37-35 Add new field to an existing data table

We now must do the same for the TDB2ON test case, but because we use the same user ID/password as for MYTDON, we reuse the existing data table. Edit the STL source, type the `UTBL` command, place the cursor on the line containing the encrypted password to be replaced, and press Enter. When the pop-up window is displayed, choose option **3**, as shown in Figure 37-36.

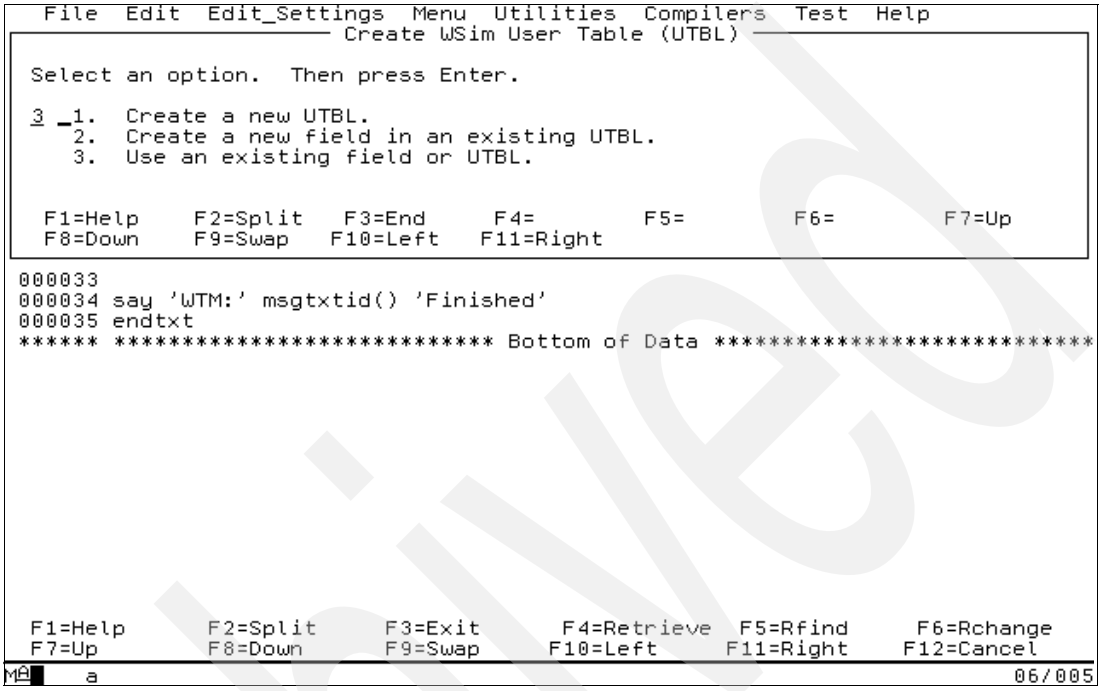


Figure 37-36 Use an existing field on UTBL

We now edit the user table COMPANY to add the company number, as shown in Figure 37-37.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CHABERT.WSIM.TRADER.STL(COMPANY) - 01.01        Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 /* This user table was created by WSim/TM to be accessed */
000002 /* by the Random method. */
000003 /* Table Description: Company list */
000004 /* WTM_FIELD: COMPANY ° Company name */
000005 COMPANY: msgutbl
000006 '1°'
000007 '2°'
000008 '3°'
000009 '4'
000010 endutbl
****** **************************** Bottom of Data ****************************
                                      -
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
MA    a                                                              19/042
```

Figure 37-37 Company user table

We also edit the user table CICSUSR to define the other user ID and password, as shown in Figure 37-38.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CHABERT.WSIM.TRADER.STL(CICSUSR) - 01.03        Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 /* This user table was created by WSim/TM to be accessed */
000002 /* by the Single Sequential method. */
000003 /* Table Description: User and password to logon to CICS */
000004 /* WTM_FIELD: CICSUSER ° CICS Userid */
000005 /* WTM_FIELD: CICSPSWD ° User Password */
000006 CICSUSR: msgutbl
000007 'eric°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000008 'albane°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000009 'charlene°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000010 'rachel°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000011 'marylene°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000012 'minna°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000013 'salla°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000014 'emmy°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000015 'jouso°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000016 'niisku°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000017 'klas°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000018 'bjorn°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000019 'leo°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000020 'kleo°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000021 'juho°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000022 'garry°03897FC101AF1F87541AD8C827093B0180661A09AA°'
000023 endutbl
****** **************************** Bottom of Data ****************************
 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
MA    a                                                              04/015
```

Figure 37-38 CICSUSR user table

37.3.8 Random number generation

We also must have a random number of shares to buy or to sell when executing test cases MYTDBUY, MYTDSEL, TDB2BUY, or TDB2SEL.

Edit each of these STL sources, and replace the share number with the RNUM command, as shown in Figure 37-39.

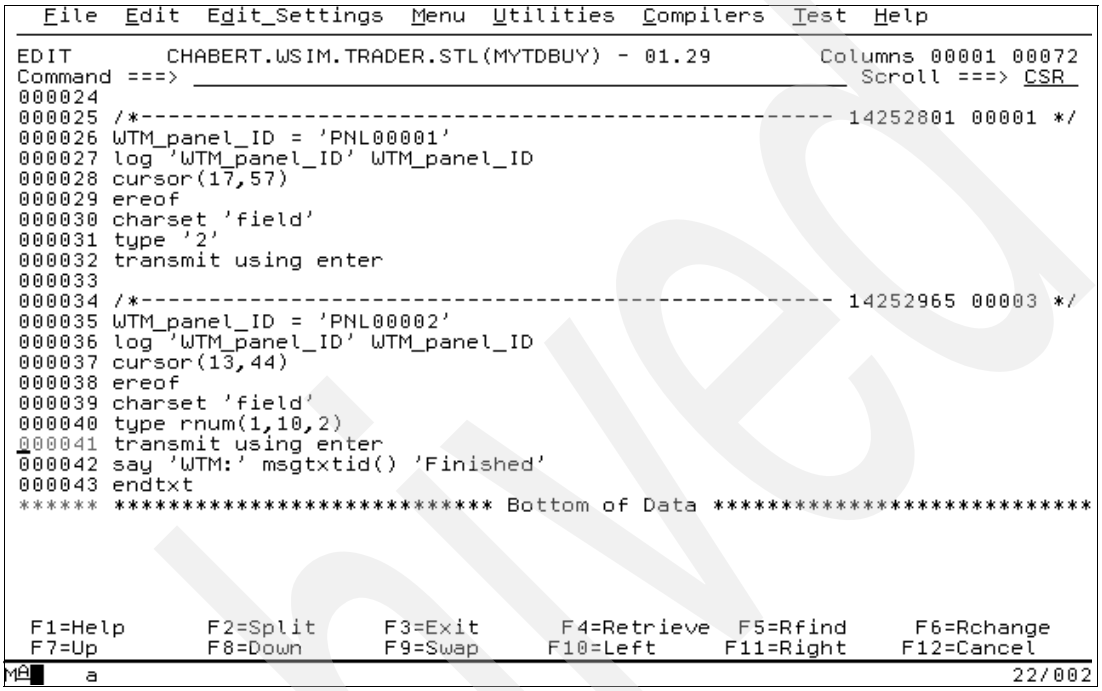


Figure 37-39 RNUM function

The RNUM function returns the string (EBCDIC) representation of a random number, as shown in Example 37-1.

Example 37-1 RNUM syntax

```
/*Syntax: RNUM(low,high,lenght)
rnum(1,10,2)
/* Generate a 2-digit random number between 1 and 10*/
```

The last definition we must do is the network and its associated workload.

37.3.9 Setting a delay before transmission

Because users do not think the same or type the same, in order to create a workload as close as possible of reality, we must introduce a user *operation* time. This can be done by using the DELAY function of WSim.

The DELAY statement specifies the delay after the next transmit by a simulated terminal. The value specified in this statement overrides the default delay for the terminal (specified with the DELAY operand for the terminal in the network definition). The actual delay (in hundredths of seconds) is the delay value specified in this statement multiplied by the terminal's UTI value. Example 37-2 shows the DELAY syntax.

Example 37-2 DELAY syntax

```
DELAY({'RATE',rate_table_number}[,uti_name])
      {fixed_time}
      {random_function}
```

Figure 37-40 shows how the DELAY function has been implemented to the MYTDBUY testcase. The same has to be done for the others testcases.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       CHABERT.WSIM.TRADER.STL(MYTDBUY) - 01.28        Columns 00001 00072
Command ===> _                                              Scroll ===> CSR
000024
000025 /*------------------------------------------------- 14252801 00001 */
000026 WTM_panel_ID = 'PNL00001'
000027 log 'WTM_panel_ID' WTM_panel_ID
000028 cursor(17,57)
000029 ereof
000030 charset 'field'
000031 type '2'
000032 delay(random(1,5))
000033 transmit using enter
000034
000035 /*------------------------------------------------- 14252965 00003 */
000036 WTM_panel_ID = 'PNL00002'
000037 log 'WTM_panel_ID' WTM_panel_ID
000038 cursor(13,44)
000039 ereof
000040 charset 'field'
000041 type rnum(1,10,2)
000042 delay(random(1,5))
000043 transmit using enter
000044 say 'WTM:' msgtxtid() 'Finished'
000045 endtxt
****** **************************** Bottom of Data ******************************


 F1=Help      F2=Split     F3=Exit      F4=Retrieve  F5=Rfind     F6=Rchange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
MA    a                                                              04/015
```

Figure 37-40 Delay function

Because we use a RANDOM function as an argument, the resulting random number is used as the delay value.

37.3.10 Creating a network and scheduling its associated workload

A schedule is a WSim network definition with supporting documentation and reporting facilities. To define a new schedule (or see the list of existing schedules), select option **4** or enter the `RUN` command from the WSim Test Manager main menu. A list of the schedules is displayed, as shown in Figure 37-41.

```
                          WSim Test Manager                      Invalid value

Select one of the following.  Then press Enter.

      Command   Action
4 _1. CASE      Create and Process Testcases
   2. GROUP     Create and Process Testgroups
   3. CYCLE     Create and Process Testcycles
   4. RUN       Create WSim Networks and Schedule WSim Simulation Runs
   D. DOC       Create Test Documentation
   P. PROJECT   Add/Change Project or Alternate HLI
   U. UTIL      Run WSim Test Manager Utilities
   W. WII       Invoke WSim/ISPF Interface

Project: TRADER                        Alternate HLI:




Command ===> ____________________________________________________________________




 F1=Help      F2=Split     F3=End       F4=          F5=          F6=
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Retrieve
MA    a                                                          06/004
```

Figure 37-41 Create a WSim network and schedule WSim simulation runs

A pop-up panel entitled Add New Test Schedule is displayed. Enter a new schedule name, an optional description, and the type of schedule. You must specify whether this is a VTAMAPPL, TCP/IP, or CPI-C schedule. To illustrate several options, we choose to define the ALLTRAD workload, as shown in Figure 37-42.

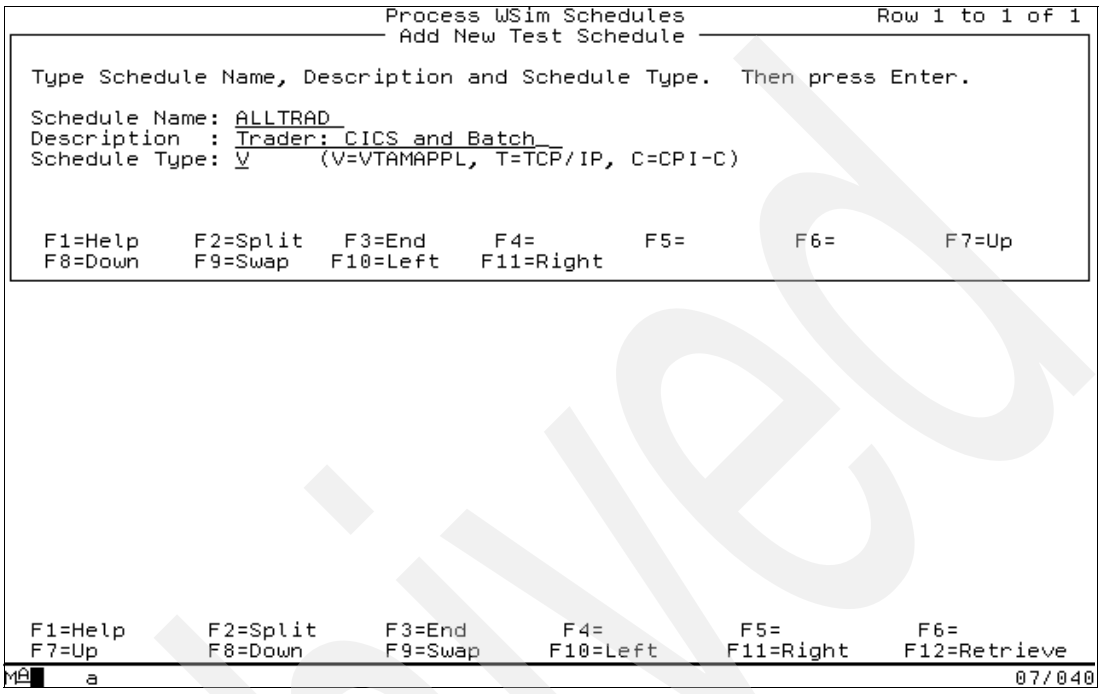


Figure 37-42 Network definition parameters

A resource list for this schedule is now displayed. Update this resource list as shown in Figure 37-43. The schedule resource list defines the number of WSim-simulated resources in this schedule and the test items referenced by each resource.

Note: If the resource list is not updated (no test items are selected), a WSim network is not created.

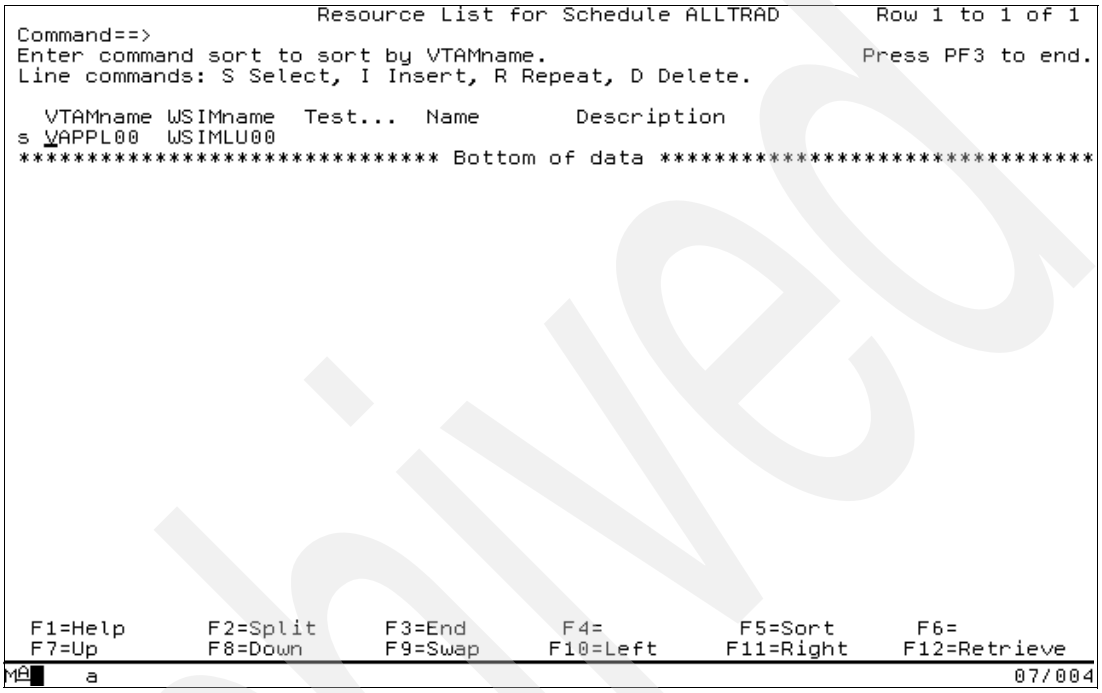


Figure 37-43 Network definition for ALLTRAD

To select the resource, use the **S** (Select) line command. This displays a pop-up panel entitled Specify Order for Resource WSIMname under VTAMname. This panel shows a list of all the test items available for this resource. Enter the number 1 to select a test item for this resource, as shown in Figure 37-44. To select multiple test items for this resource, specify a numerical reference order for each. Those resources that are not associated with a test item are not referenced by this schedule. When orders specification is complete, press PF3 to return to the resource list.

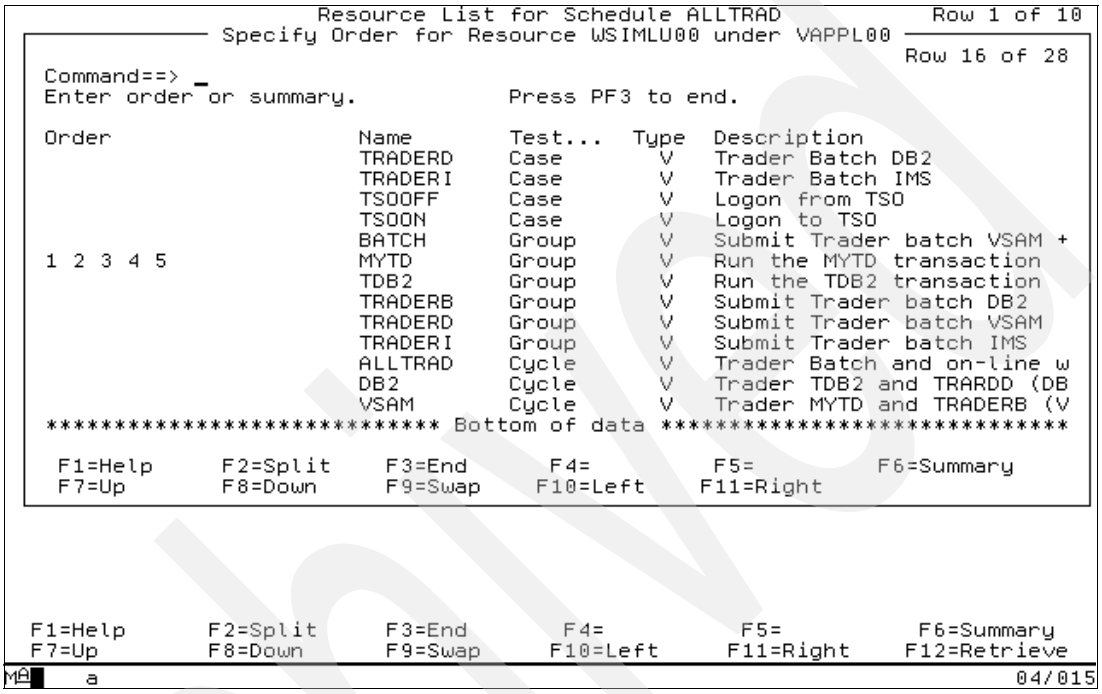


Figure 37-44 Select items (GROUP) for VAPPL00

When we press PF3, WSim associates the VTAM name (VAPPL00) with its workload definition, as shown in Figure 37-45.

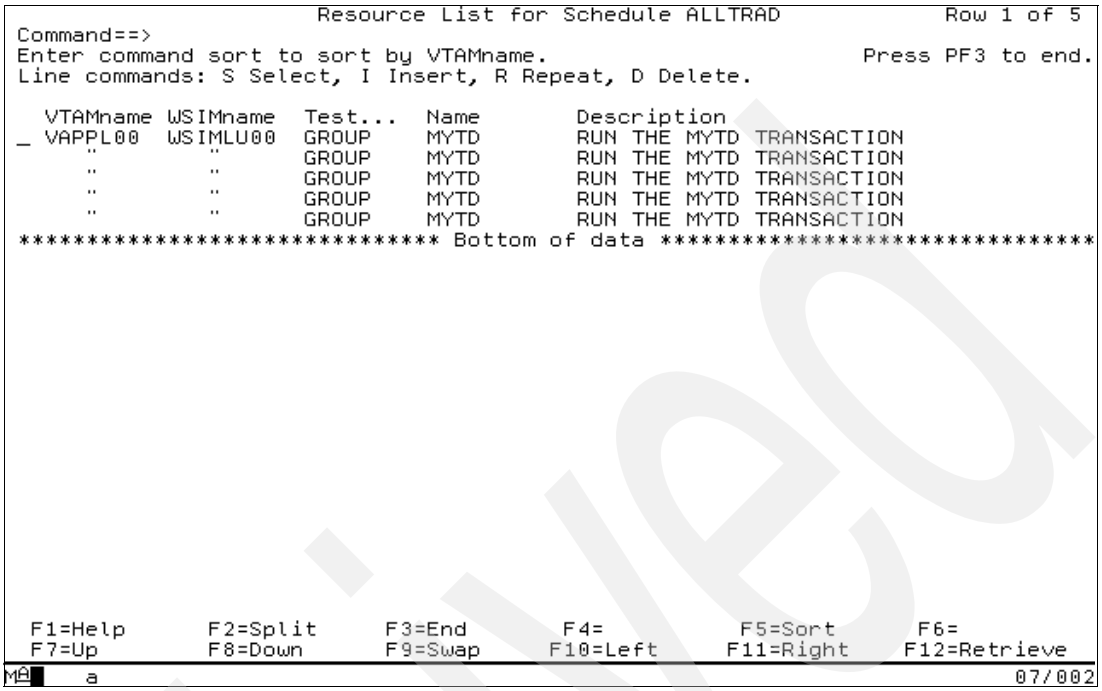


Figure 37-45 WSim: VTAM terminal and workload association

The first terminal (workload) has been defined, but to simulate a real workload, several terminals (users) have to submit their specific workloads at the same time. The workload we want to define has the following characteristics:

- Thirty-two users running only the CICS VSAM application (five times)
- Thirty-two users running only the CICS DB2 application (five times)
- Sixteen users running a mix of the CICS VSAM and the CICS DB2 application (five times both)
- One user running the Batch VSAM application (only once)
- One user running the Batch DB2 application (only once)
- One user running the Batch IMS application (only once)

To achieve that workload we must define others VTAM names (terminals) and associate them with their specific workload. The steps required are:

1. Add a new VTAMname.
2. Associate the new VTAMname with its specific workload.
3. Duplicate 16 times as a set of:
   - Two CICS VSAM terminals
   - Two CICS DB2 terminals
   - One CICS VSAM and DB2 terminal

We add batch applications in the middle of the CICS workload.

To add a new VTAMname, we use the **I** (Insert) line command, as shown in Figure 37-46.

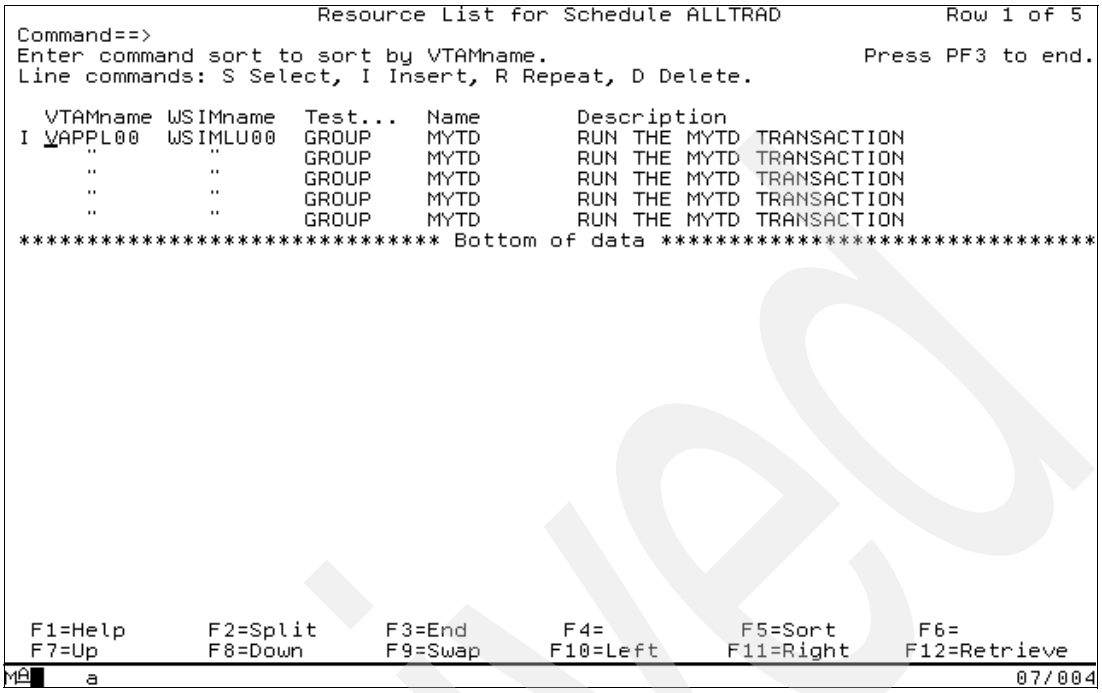


Figure 37-46 WSim: adding a new VTAMname

We enter numbers, specifying their reference order, to select multiple test items for this resource, as shown in Figure 37-47.

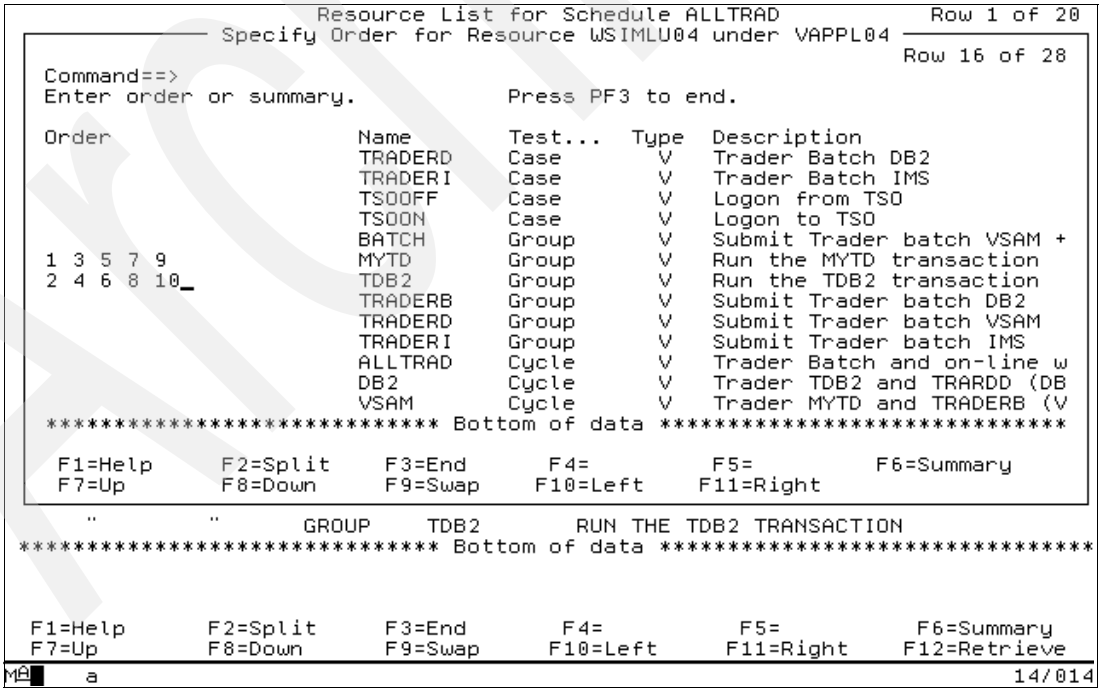


Figure 37-47 Select test items (GROUP) for VAPPL04

We continue the process until we have the following results, as shown in Figure 37-48:

- VAPPL00 and VAPPL02 running MYTD
- VAPPL01 and VAPPL03 running TDB2
- VAPPL04 running a mix of MYTD and TDB2

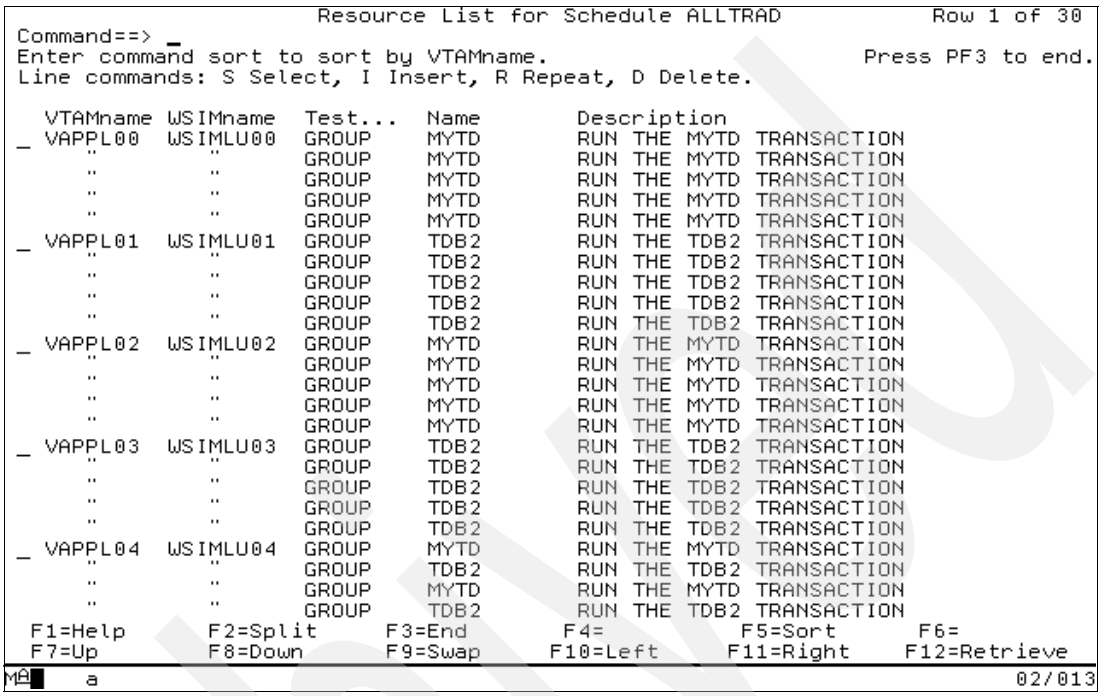


Figure 37-48 First part of resource for ALLTRAD

We now duplicate that set of VTAMname seven times to get the first part of our workload. We enter the **R** (Repeat) line command in front of each VTAMname and press Enter. WSim displays a pop-up window to specify the number of time we must repeat this.

We can now add the batch workload and complete the full scenario by duplicating the CICS workload.

When all workloads have been defined, define an Application Performance Analyzer Observation Session to monitor each of them.

38

Using Application Performance Analyzer with the WSim workload

In this chapter we first describe the observation sessions that we must define to monitor the ALLTRAD workload described in Chapter 37, “Using WSim to create workloads” on page 1179. Then we explain how to start the ALLTRAD workload, and we go through the main reports to be analyzed.

38.1 ALLTRAD workload and its observation sessions

First, you must define two types of observation sessions:

- One for CICS including specific transaction codes and subsystems
- One for each of our batches

38.1.1 Observation session for ALLTRAD: Definition for CICS

Only one observation session has to be defined to monitor all of our CICS transactions. Figure 38-1 shows the first panel.

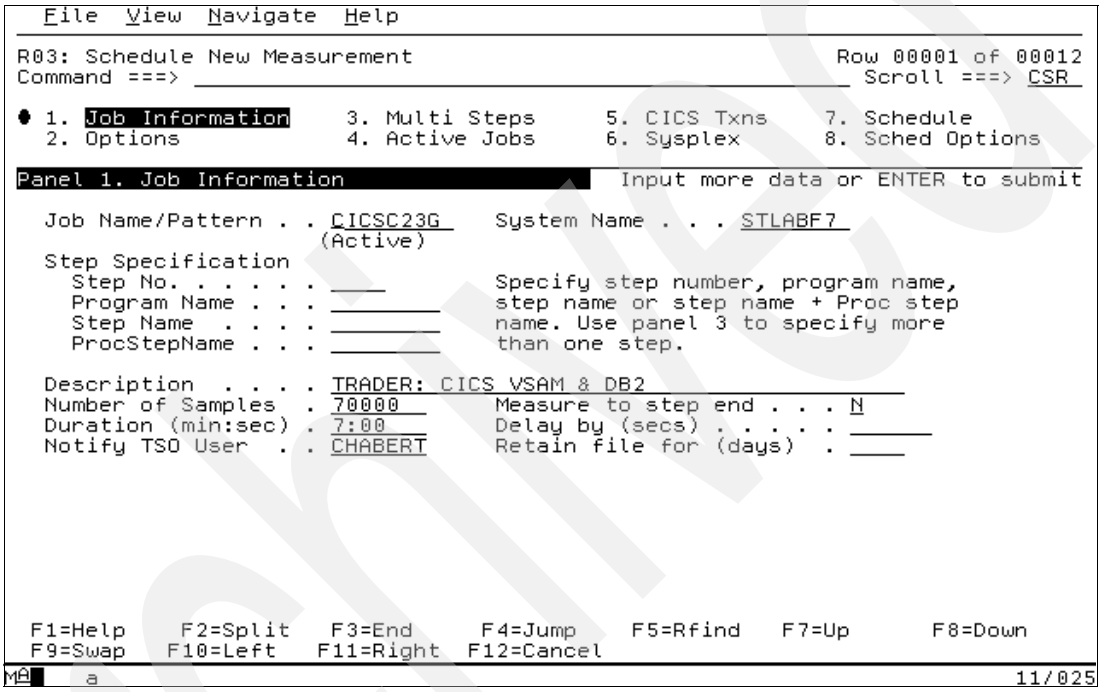


Figure 38-1 Observation session definition for ALLTRAD, panel 1

Then we have to select CICS, DB2, and DB2+, as shown in Figure 38-2.

```
  File  View  Navigate  Help
R03: Schedule New Measurement                          Row 00001 of 00022
Command ===> _____________________________________________ Scroll ===> CSR

● 1. Job Information    3. Multi Steps      5. CICS Txns      7. Schedule
  2. Options            4. Active Jobs      6. Sysplex        8. Sched Options

Panel 2. Measurement Options

 Data Extractors. '/' to select extended measurement options:
      /  CICS  CICS information
      /  DB2   SQL call information
      _  IMS   DLI call information
      _  MQ    MQSeries call information
      /  DB2+  SQL service/CPU time/counts
      _  IMS+  DLI service/CPU time/counts

 Specify up to 10 load libraries to be searched by IBM APA for z/OS for
 external symbol information. These are applicable only when sampled modules
 are fetched from dynamically allocated load libraries.

    1  _____________________________________________
    2  _____________________________________________
    3  _____________________________________________
    4  _____________________________________________
    5  _____________________________________________
    6  _____________________________________________
    7  _____________________________________________
    8  _____________________________________________
 F1=Help    F2=Split   F3=End     F4=Jump    F5=Rfind   F7=Up      F8=Down
 F9=Swap    F10=Left   F11=Right  F12=Cancel
MA  a                                                              17/008
```

Figure 38-2 Observation session definition for ALLTRAD, panel 2

The last panel allows us to specify which CICS transaction we must monitor, as shown in Figure 38-3.

```
  File  View  Navigate  Help
R03: Schedule New Measurement                          Row 00001 of 00006
Command ===> _____________________________________________ Scroll ===> CSR

● 1. Job Information    3. Multi Steps      5. CICS Txns      7. Schedule
● 2. Options            4. Active Jobs      6. Sysplex        8. Sched Options

Panel 5. CICS Transactions

 Specify up to 16 CICS trancodes for which measurement data is to be
 recorded. Wildcard character '*' can be specified at the end of a
 partial name. '*' by itself specifies all transactions.

   01 MYTD      02 TDB2      03 _____     04 _____
   05 _____     06 _____     07 _____     08 _____
   09 _____     10 _____     11 _____     12 _____
   13 _____     14 _____     15 _____     16 _____

 Include CICS system transactions in measurement(Y/N): N

 F1=Help    F2=Split   F3=End     F4=Jump    F5=Rfind   F7=Up      F8=Down
 F9=Swap    F10=Left   F11=Right  F12=Cancel
MA  a                                                              15/034
```

Figure 38-3 Observation session definition for ALLTRAD, panel 5

38.1.2 Observation session for ALLTRAD: Definition for batch jobs

For the batch jobs, we define the observation sessions by adding a new step to each of them. This first step calls the CAZBATCH program to define the observation session required for that specific batch job.

The new TRADERB batch job is listed in Example 38-1.

Example 38-1 New TRADERB batch job (with its own observation session definition)

```
//TRADERB  JOB 1,RACFUSER,TIME=1440,NOTIFY=&SYSUID,REGION=4M,
//             CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//PROCLIB JCLLIB ORDER=CHABERT.BOOK2005.JCL
//*******************************************************************
//* LICENSED MATERIALS - PROPERTY OF IBM                            *
//* 5655-ADS (C) COPYRIGHT IBM CORP. 2004                           *
//* ALL RIGHTS RESERVED                                             *
//*******************************************************************
//CAZBATCH EXEC PGM=CAZBATCH,PARM='STCID=CAZ0'
//STEPLIB  DD   DISP=SHR,DSN=ADTOOLS.APA.V1R1M0.SCAZAUTH
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 NEW
  JOBNAME=(TRADERB)
  DURATION=(1:00)
  SAMPLES=(10000)
  DESCR="TRADER VSAM BATCH"
  RUNTOEOS=(Y)
  NOTIFY=(CHABERT)
  ACTIVE=(Y)
  SYSTEMS=(*)
  RUNAGAIN=(5,FOR=5)
 ;
//TRADERB  EXEC PGM=TRADERB,PARM=RPTOPTS(YES)
//STEPLIB  DD DISP=SHR,DSN=CHABERT.BOOK2005.LOAD
//SYSPRINT DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//COMPFILE DD DISP=SHR,DSN=CHABERT.BOOK2005.COMPFILE.BATCH
//CUSTFILE DD DISP=SHR,DSN=CHABERT.BOOK2005.CUSTFILE.BATCH
//TRANSACT DD DISP=SHR,DSN=CHABERT.BOOK2005.TRANFILE.VSAM
//REPOUT   DD DUMMY
//TRANREP  DD DUMMY
//*
```

The new TRADERB batch job is listed in Example 38-2.

Example 38-2 New TRADERD batch job (with its own observation session definition)

```
//TRADERD  JOB ,CLASS=A,NOTIFY=&SYSUID,MSGCLASS=H,MSGLEVEL=(1,1)
//JOBLIB   DD DISP=SHR,DSN=SYS1.D81G.SDSNLOAD
//CAZBATCH EXEC PGM=CAZBATCH,PARM='STCID=CAZ0'
//STEPLIB  DD   DISP=SHR,DSN=ADTOOLS.APA.V1R1M0.SCAZAUTH
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 NEW
  JOBNAME=(TRADERD)
  DURATION=(1:00)
  SAMPLES=(10000)
  DESCR="TRADER DB2 BATCH"
  RUNTOEOS=(Y)
```

```
  NOTIFY=(CHABERT)
  ACTIVE=(Y)
  SYSTEMS=(STLABF7)
  FEATURES=(DB2,DB2+)
 ;
//TRADERD  EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
//DBRMLIB  DD DISP=SHR,DSN=CHABERT.D81G.DBRMLIB.DATA
//TRANSACT DD DISP=SHR,DSN=CHABERT.BOOK2005.TRANFILE.DB2
//REPOUT   DD DUMMY
//TRANREP  DD DUMMY
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//DSNTRACE DD SYSOUT=*
//SYSTSIN  DD *
 DSN SYSTEM(D81G)
 RUN  PROGRAM(TRADERD) PLAN(TRADERD) +
      LIB('CHABERT.BOOK2005.LOAD')
 END
```

The new TRADERI batch job is listed in Example 38-3.

Example 38-3 New TRADERI batch job (with its own observation session definition)

```
//TRADERI  JOB ,CLASS=A,NOTIFY=&SYSUID,MSGCLASS=H,MSGLEVEL=(1,1)
//PROCLIB JCLLIB ORDER=CHABERT.BOOK2005.JCL
//*******************************************************************
//* LICENSED MATERIALS - PROPERTY OF IBM                            *
//* 5655-ADS (C) COPYRIGHT IBM CORP. 2004                           *
//* ALL RIGHTS RESERVED                                             *
//*******************************************************************
//CAZBATCH EXEC PGM=CAZBATCH,PARM='STCID=CAZ0'
//STEPLIB  DD   DISP=SHR,DSN=ADTOOLS.APA.V1R1M0.SCAZAUTH
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 NEW
  JOBNAME=(TRADERI)
  DURATION=(1:00)
  SAMPLES=(10000)
  DESCR="TRADER IMS BATCH"
  RUNTOEOS=(Y)
  NOTIFY=(CHABERT)
  ACTIVE=(Y)
  SYSTEMS=(STLABF7)
  FEATURES=(IMS,IMS+)
 ;
//TRADERI EXEC PROC=IMSGO,
//         MBR=TRADERI,
//         PSB=PTRDI
//TRANSACT DD DISP=SHR,DSN=CHABERT.BOOK2005.TRANFILE.IMS
//REPOUT   DD SYSOUT=DUMMY
//TRANREP  DD SYSOUT=DUMMY
 //*
```

When the Application Performance Analyzer observation session definition has been done for the CICS part of the Trader application, and when batch jobs have been modified, we have to start the ALLTRAD workload to determine where the issues are, if any.

38.1.3 ALLTRAD workload and its submission

From the Workload Simulator Test Manager main panel, we choose option **4**. WSim displays the list of all available workload that can be executed. We use the **X** line command for the ALLTRAD workload to run it. WSim first checks the network configuration, as shown in Figure 38-4.

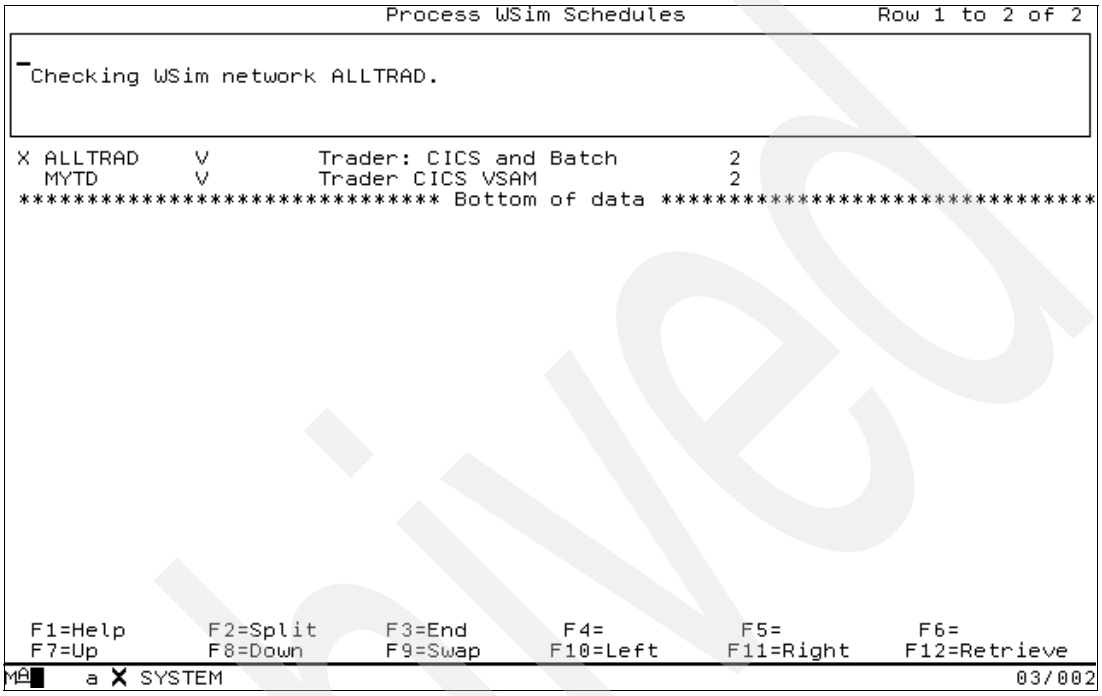


Figure 38-4 ALLTRAD: network verification

We now have to specify the name of the WSim log for the simulation run about to be initialized. If the WSim log specified does not exist, the WSim Test Manager creates it. We set the Completion Report field to **Y**, as shown in Figure 38-5.

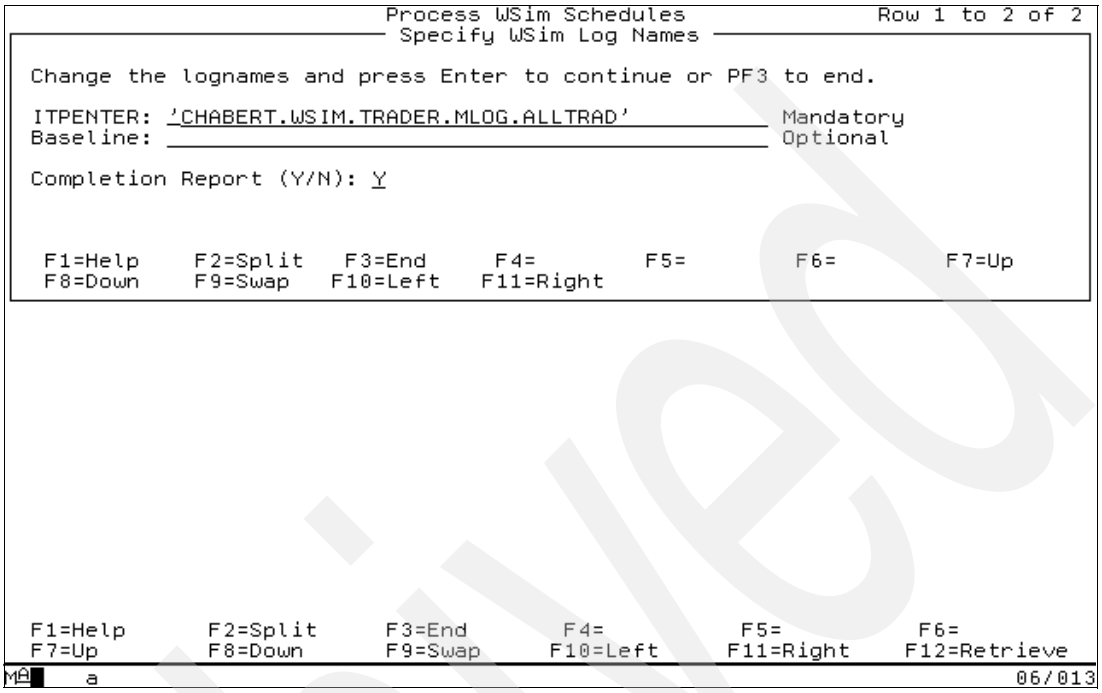


Figure 38-5 ALLTRAD: log name definition

Now we press Enter to run the schedule (WSim network).

Figure 38-6 shows an extract of the WSim log messages issued during the execution of the workload.

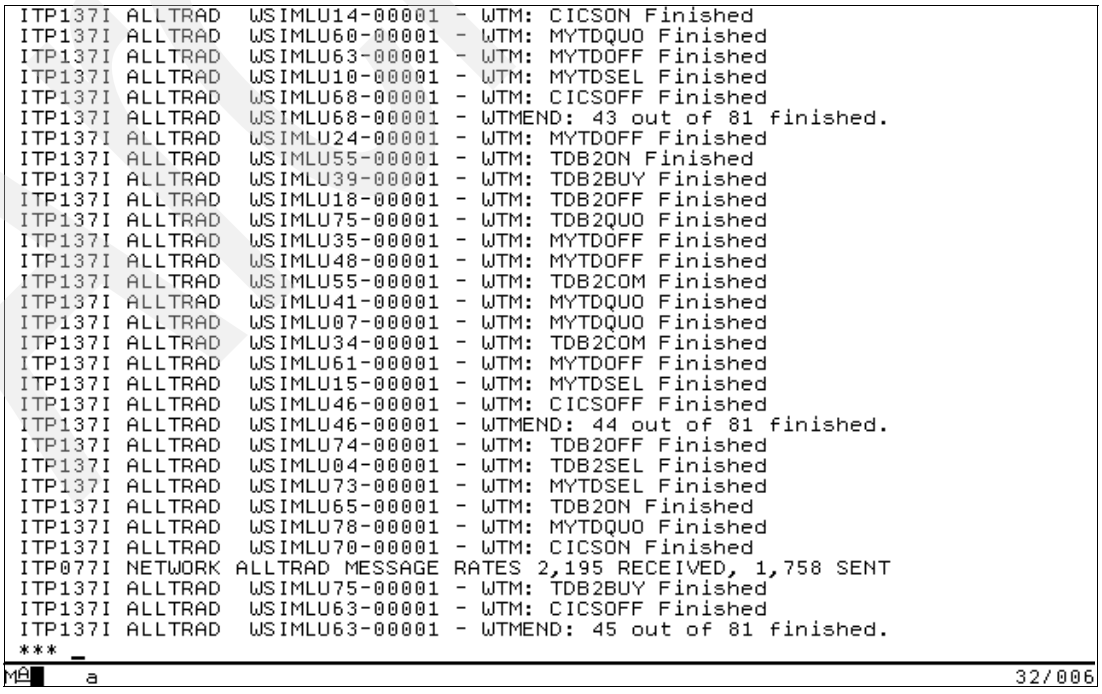


Figure 38-6 WSim log messages

38.2 APA reports for the CICS application

In this section we discuss APA reports for the CICS application.

38.2.1 APA reports for the first run

When the workload ends, four new reports (5037, 5038, 5039, and 5040) are available in the Observation List, as shown in Figure 38-7.

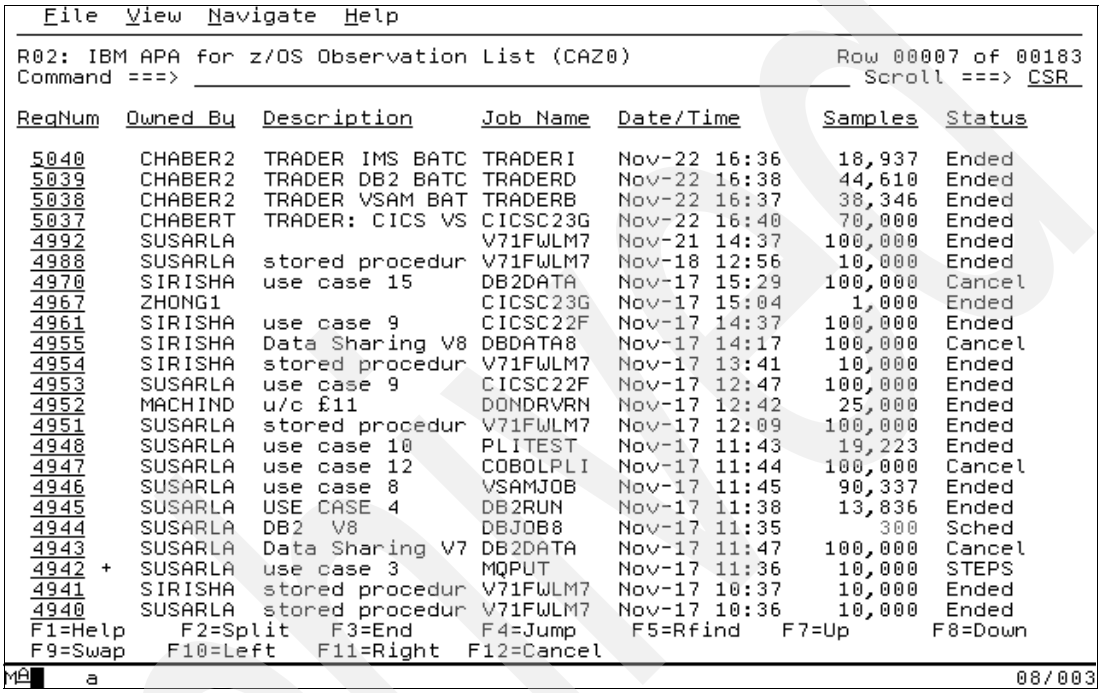


Figure 38-7 APA reports for the first run

We concentrate on 5037, the online TRADER application (CICS - DB2 - VSAM). The first report to analyze is the E01: CICS Session Statistics, as shown in Figure 38-8 and in Figure 38-9.

```
  File  View  Navigate  Help
E01: CICS Session Statistics (5037)                        Row 00001 of 00046
Command ===> _                                              Scroll ===> CSR

Environmental Information
   CICS Release                          CICS/TS 2.3

Transaction Statistics
   First Transaction TaskId              0042706
   Last Transaction TaskId               0049792
   Number of TaskId Increments           7,086
   Number of Observed Transactions       1,639
   Transaction Rate (per sec)            16.91
   Peak Active Txns (Observed)           51
   Peak Active Txns (Overall)            80
   MaxTask                               150

Mean Transaction Time (Execution + Suspend + Delay = Service)
   Execution Time                        0.002456
   Suspend Time                          6.646036
   CICS Dispatch Delay Time              0.001878
   MVS Dispatch Delay Time               0.000776
   Service Time                          6.651146

Service Statistics
   Program Requests                      16,872
   Terminal Messages                     1,054
   Storage Getmains                      188,284
   Storage Freemains                     180,403
   File I/O Requests                     3,783
   Temporary Storage Requests            17,252
MA    a                                                              04/015
```

Figure 38-8 E01: CICS Session Statistics (5037) (1 of 2)

```
  File  View  Navigate  Help
E01: CICS Session Statistics (5037)                        Row 00021 of 00046
Command ===> _                                              Scroll ===> CSR

Service Statistics
   Program Requests                      16,872
   Terminal Messages                     1,054
   Storage Getmains                      188,284
   Storage Freemains                     180,403
   File I/O Requests                     3,783
   Temporary Storage Requests            17,252
   Transient Data Requests               9,352
   Journal Write Requests                0

Exception or Critical Conditions
   System Dumps                          0
   System Dumps Suppressed               0
   Transaction Dumps                     0
   Transaction Dumps Suppressed          0
   Storage Violations                    0
   Short on Storage occurrences          0
   Times at MaxTask                      0
   Times at Class MaxTask                0

Transaction Counts

      TranId     Count
       TDB2       796
       MYTD       843

MA    a                                                              04/015
```

Figure 38-9 E01: CICS Session Statistics (5037) (2 of 2)

Table 38-1 contains a summary of the main information that we can have from E01.

Table 38-1 CICS Statistics summary (1 of 11)

	Report number 5037
Mean service time	6.651146
Transaction rate	16.91
Number of tasks	7086
Peak active transactions	51

We can see that the mean service time is very high for the CICS transactions, and the peak active transactions should not exceed six for the workload we have defined. We display E04: Mean Service Time by Transaction to get more details, as shown in Figure 38-10.

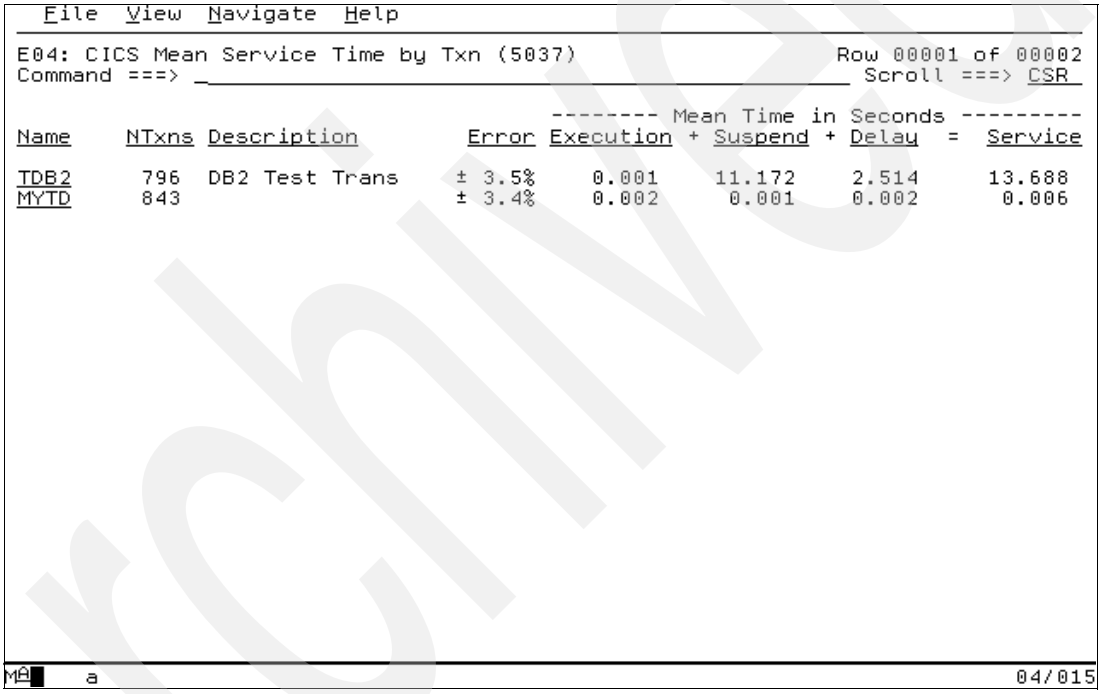


Figure 38-10 E04: CICS Mean Service Time by Transaction (5037)

Note: It is paramount to know how much we can trust the APA reports than we analyze. Margin of Error displays on each APA report below 3.5% that makes APA reports reliable.

We have two transactions in the workload. E04's report shows that MYTD (CICS - VSAM) ran fine, and that we have an issue with TDB2 (CICS - DB2). By expanding the three levels of the first row, Application Analyzer Performance displays details about that transaction, as shown in Figure 38-11.

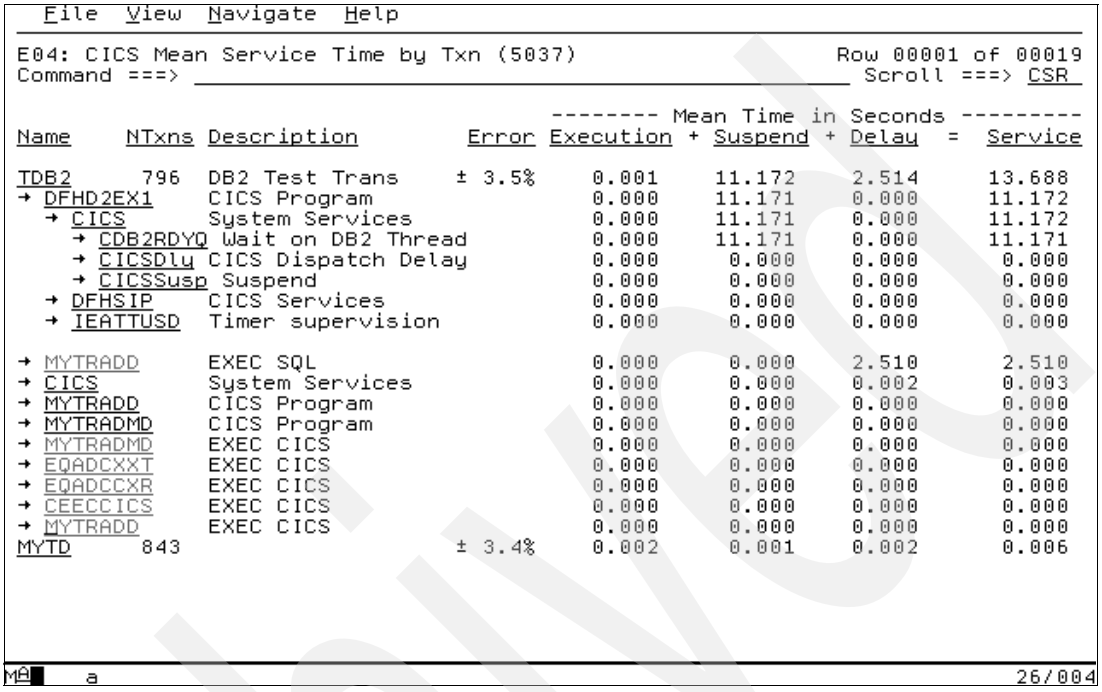


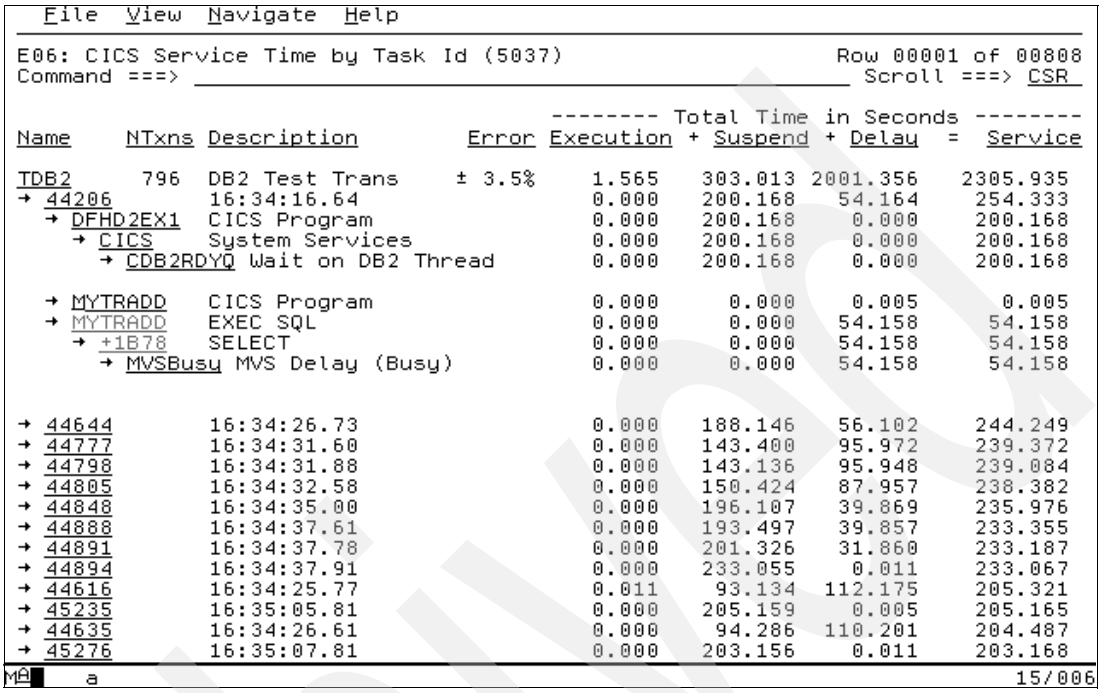
Figure 38-11 E04: CICS Mean Service Time by Transaction (5037) (expanded)

It appears that the response depends on Wait on Thread. Table 38-2 now provides more information about that run.

Table 38-2 CICS summary (2 of 11)

	Report number 5037
Mean service time	6.651146 sec.
Transaction rate	16.91 per sec.
Number of tasks	7086
Peak active transactions	51
TDB2: mean suspend time	11.172 sec.
TDB2: mean delay time	2.514 sec.

We know that the mean response time is very high because of a few number of task-IDs. To see the range of the response time for TDB2, we use report E06. Figure 38-12 shows an expanded (four levels) version of it.



```
  File  View  Navigate  Help
E06: CICS Service Time by Task Id (5037)                    Row 00001 of 00808
Command ===> _____________________________________________ Scroll ===> CSR

                                                 -------- Total Time in Seconds --------
Name     NTxns Description          Error  Execution + Suspend + Delay  =  Service

TDB2       796 DB2 Test Trans     ± 3.5%      1.565   303.013 2001.356   2305.935
→ 44206        16:34:16.64                    0.000   200.168   54.164    254.333
  → DFHD2EX1   CICS Program                   0.000   200.168    0.000    200.168
    → CICS     System Services                0.000   200.168    0.000    200.168
      → CDB2RDYQ Wait on DB2 Thread           0.000   200.168    0.000    200.168

  → MYTRADD    CICS Program                   0.000     0.000    0.005      0.005
  → MYTRADD    EXEC SQL                       0.000     0.000   54.158     54.158
    → +1B78    SELECT                         0.000     0.000   54.158     54.158
      → MVSBusy MVS Delay (Busy)              0.000     0.000   54.158     54.158


→ 44644        16:34:26.73                    0.000   188.146   56.102    244.249
→ 44777        16:34:31.60                    0.000   143.400   95.972    239.372
→ 44798        16:34:31.88                    0.000   143.136   95.948    239.084
→ 44805        16:34:32.58                    0.000   150.424   87.957    238.382
→ 44848        16:34:35.00                    0.000   196.107   39.869    235.976
→ 44888        16:34:37.61                    0.000   193.497   39.857    233.355
→ 44891        16:34:37.78                    0.000   201.326   31.860    233.187
→ 44894        16:34:37.91                    0.000   233.055    0.011    233.067
→ 44616        16:34:25.77                    0.011    93.134  112.175    205.321
→ 45235        16:35:05.81                    0.000   205.159    0.005    205.165
→ 44635        16:34:26.61                    0.000    94.286  110.201    204.487
→ 45276        16:35:07.81                    0.000   203.156    0.011    203.168
MA    a                                                                    15/006
```

Figure 38-12 E06: CICS Service Time by Task Id (5037) (expanded)

We can add more information into our CICS summary table, as shown in Table 38-3.

Table 38-3 CICS summary (3 of 11)

	Report number 5037
Mean Service Time	6.651146 sec.
Transaction Rate	16.91 per sec.
Number of Task	7086
Peak Active Transactions	51
TDB2: Mean Suspend Time	11.172 sec.
TDB2: Mean Delay Time	2.514 sec.
TDB2: Worst Time	Task Id 44206: Suspend Time: 200.168 sec. Task Id 44616 Delay Time: 112.175 sec.

Report E07: CICS Wait by Transaction confirms what we already know, as shown in Figure 38-13.

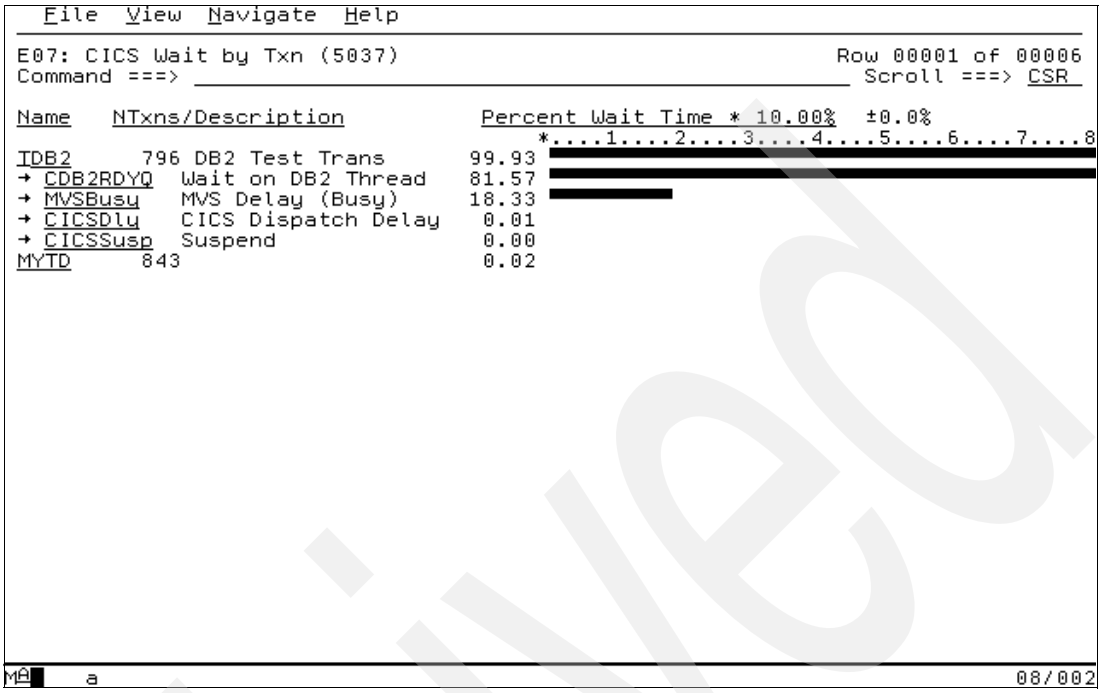


Figure 38-13 E07: CICS Wait by Transaction (5037) (expanded)

Table 38-4 lists the latest information that we must keep in mind about our first run.

Table 38-4 CICS summary (4 of 11)

	Report number 5037
Mean Service Time	6.651146 sec.
Transaction Rate	16.91 per sec.
Number of Task	7086
Peak Active Transactions	51
TDB2: Mean Suspend Time	11.172 sec.
TDB2: Mean Delay Time	2.514 sec.
TDB2: Worst Time	Task Id 44206: Suspend Time: 200.168 sec. Task Id 44616 Delay Time: 112.175 sec.
TDB2: Wait on DB2 Thread	81.57% of 99.93% of the total Wait Time
TDB2: MVS Delay (Busy)	18.33% of 99.93% of the total Wait time

38.2.2 Actions taken

We know that we must improve how DB2 thread is managed. First we display the DB2 Connection parameters, as shown in Figure 38-14.

```
 I DB2C
 STATUS:  RESULTS - OVERTYPE TO MODIFY
   Accountrec( None )             Planexitname( DSNCUEXT )
   Authid( TMROSS   )             Priority( High )
   Authtype(              )       Purgecyclem( 00 )
   Comauthid(          )          Purgecycles( 30 )
   Comauthtype( Cuserid )         Resyncmember(              )
   Comthreadlim( 0001 )           Signid( CICSC23G )
   Comthreads(0000)               Security(              )
   Connecterror( Sqlcode )        Standbymode( Reconnect )
   Connectst( Connected )         Statsqueue( CDB2 )
   Db2groupid(      )             Tcblimit( 0012 )
   Db2id( D81G )                  Tcbs(0000)
   Db2release(0810)               Threaderror(N906d)
   Drollback(Rollback)            Threadlimit( 0003 )
   Msgqueue1( CDB2 )              Threads(0000)
   Msgqueue2(      )              Threadwait( Twait )
   Msgqueue3(      )
   Nontermrel( Release )
   Plan(          )

                                                 SYSID=C23G APPLID=CICSC23G
  RESPONSE: NORMAL                          TIME:  16.59.49  DATE: 11.22.05
PF 1 HELP       3 END       5 VAR       7 SBH 8 SFH 9 MSG 10 SB 11 SF
MA    b                                                              01/010
```

Figure 38-14 CEMT I DB2C (current definition)

Two of these parameters have to be modified, Tcblimit and Threadlimit. Figure 38-15 shows the changes.

```
 I DB2C
 STATUS:  RESULTS - OVERTYPE TO MODIFY
   Accountrec( None )             Planexitname( DSNCUEXT )
   Authid( TMROSS   )             Priority( High )
   Authtype(              )       Purgecyclem( 00 )
   Comauthid(          )          Purgecycles( 30 )
   Comauthtype( Cuserid )         Resyncmember(              )
   Comthreadlim( 0001 )           Signid( CICSC23G )
   Comthreads(0000)               Security(              )
   Connecterror( Sqlcode )        Standbymode( Reconnect )
   Connectst( Connected )         Statsqueue( CDB2 )
   Db2groupid(      )             Tcblimit( 0050 )
   Db2id( D81G )                  Tcbs(0000)
   Db2release(0810)               Threaderror(N906d)
   Drollback(Rollback)            Threadlimit( 0030 )
   Msgqueue1( CDB2 )              Threads(0000)
   Msgqueue2(      )              Threadwait( Twait )
   Msgqueue3(      )
   Nontermrel( Release )
   Plan(          )

                                                 SYSID=C23G APPLID=CICSC23G
  RESPONSE: NORMAL                          TIME:  17.04.19  DATE: 11.22.05
PF 1 HELP       3 END       5 VAR       7 SBH 8 SFH 9 MSG 10 SB 11 SF
MA    b                                                              01/010
```

Figure 38-15 CEMT I DB2C (new definition)

The next CICS resource that we have to concentrate on is the DB2Entry, as shown in Figure 38-16.

```
 I DB2E
 STATUS:  RESULTS - OVERTYPE TO MODIFY
  Db2e(DBMYTDB2) Uow Sig Ena Poo Equ Pro( 0000 ) Pth(0000)
     Threadl( 0005 ) Threads(0000) Tpo Plan( MYTRADD  )
  Db2e(DBMYTDB3) Uow Sig Ena Poo Equ Pro( 0000 ) Pth(0000)
     Threadl( 0005 ) Threads(0000) Tpo Plan( MYTRADD3 )
  Db2e(IBSIENT )         Ena Poo Hig Pro( 0000 ) Pth(0000)
     Threadl( 0000 ) Threads(0000) Tpo Plan( IGINSUR  )
  Db2e(NSSIENT )         Ena Poo Hig Pro( 0000 ) Pth(0000)
     Threadl( 0000 ) Threads(0000) Tpo Plan( NGINSUR  )









                                                SYSID=C23G APPLID=CICSC23G
  RESPONSE: NORMAL                          TIME:  16.59.23  DATE: 11.22.05
PF 1 HELP       3 END       5 VAR       7 SBH 8 SFH 9 MSG 10 SB 11 SF
MA    b                                                            01/010
```

Figure 38-16 CEMT I DB2E (current definition)

We can see that no *protected threads* are defined. In order to improve the response, we permit up to 25 protected threads for the TDB2 transaction, as shown in Figure 38-17.

```
 I DB2E
 STATUS:  RESULTS - OVERTYPE TO MODIFY
  Db2e(DBMYTDB2) Uow Sig Ena Poo Equ Pro( 0025 ) Pth(0000)
     Threadl( 0030 ) Threads(0000) Tpo Plan( MYTRADD  )
  Db2e(DBMYTDB3) Uow Sig Ena Poo Equ Pro( 0000 ) Pth(0000)
     Threadl( 0005 ) Threads(0000) Tpo Plan( MYTRADD3 )
  Db2e(IBSIENT )         Ena Poo Hig Pro( 0000 ) Pth(0000)
     Threadl( 0000 ) Threads(0000) Tpo Plan( IGINSUR  )
  Db2e(NSSIENT )         Ena Poo Hig Pro( 0000 ) Pth(0000)
     Threadl( 0000 ) Threads(0000) Tpo Plan( NGINSUR  )









                                                SYSID=C23G APPLID=CICSC23G
  RESPONSE: NORMAL                          TIME:  17.04.52  DATE: 11.22.05
PF 1 HELP       3 END       5 VAR       7 SBH 8 SFH 9 MSG 10 SB 11 SF
MA    b                                                            01/010
```

Figure 38-17 CEMT I DB2E (new definition)

The last CICS resource that we can check, but just for information, is the DB2Transaction, as shown in Figure 38-18.

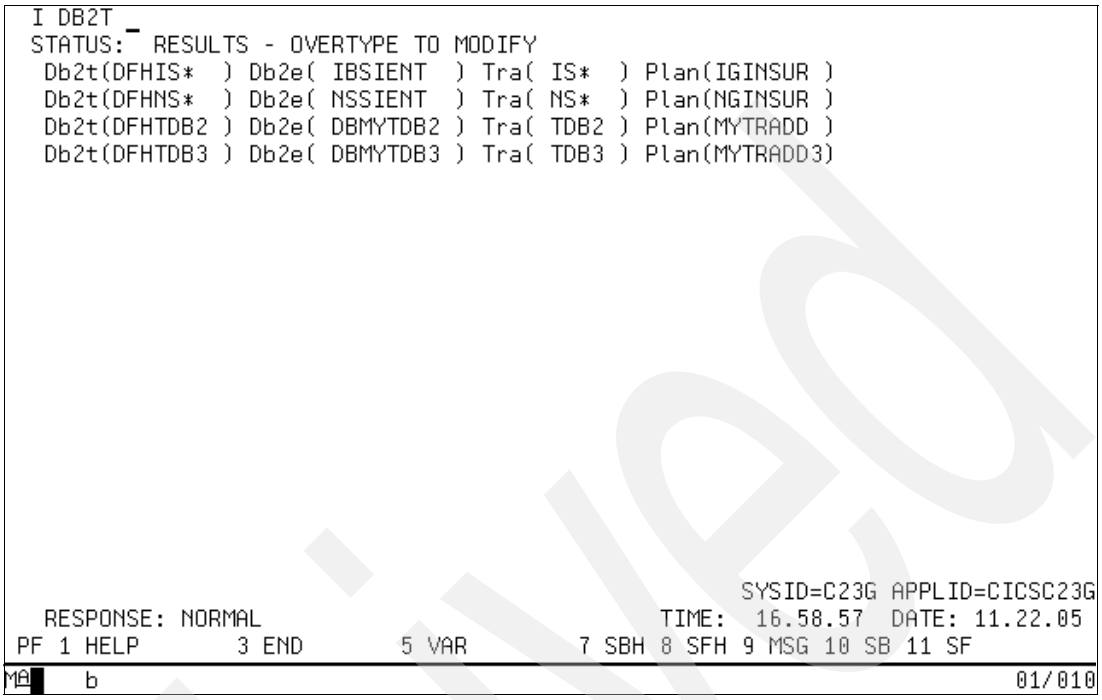
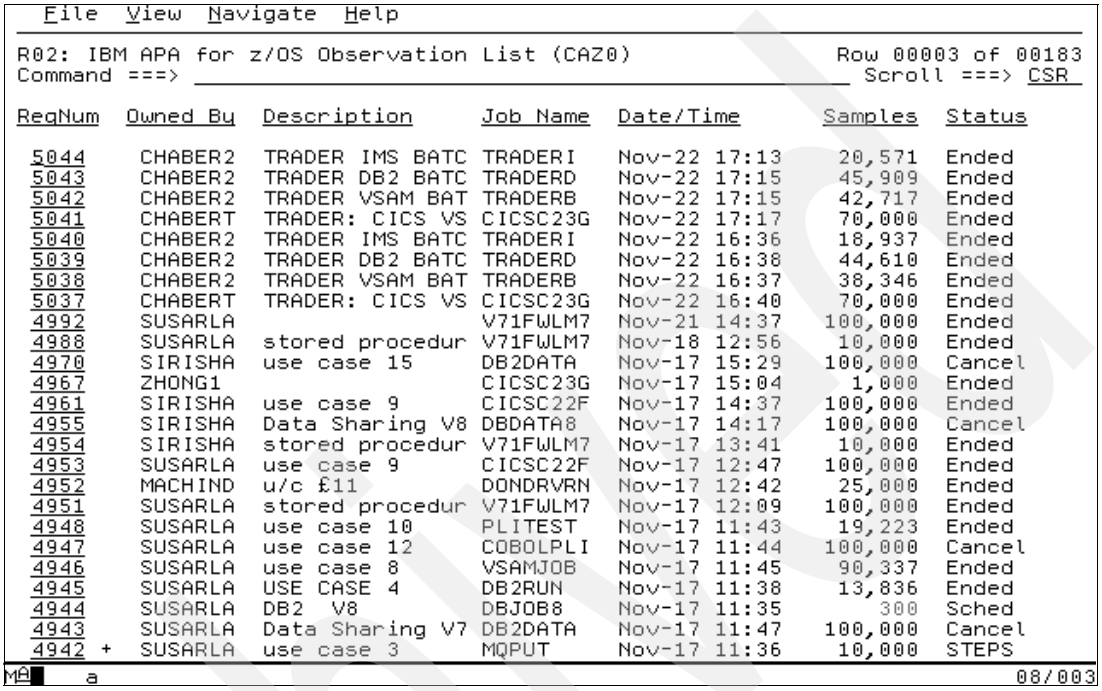


Figure 38-18 CEMT I DB2T

38.2.3 APA reports for the second run

We now rerun exactly the same scenario as before (ALLTRAD) to get another set of Application Performance Analyzer reports (5041, 5042, 5043, and 5044), as shown in Figure 38-19.



File View Navigate Help

R02: IBM APA for z/OS Observation List (CAZ0) Row 00003 of 00183
Command ===> Scroll ===> CSR

ReqNum		Owned By	Description	Job Name	Date/Time	Samples	Status
5044		CHABER2	TRADER IMS BATC	TRADERI	Nov-22 17:13	20,571	Ended
5043		CHABER2	TRADER DB2 BATC	TRADERD	Nov-22 17:15	45,909	Ended
5042		CHABER2	TRADER VSAM BAT	TRADERB	Nov-22 17:15	42,717	Ended
5041		CHABERT	TRADER: CICS VS	CICSC23G	Nov-22 17:17	70,000	Ended
5040		CHABER2	TRADER IMS BATC	TRADERI	Nov-22 16:36	18,937	Ended
5039		CHABER2	TRADER DB2 BATC	TRADERD	Nov-22 16:38	44,610	Ended
5038		CHABER2	TRADER VSAM BAT	TRADERB	Nov-22 16:37	38,346	Ended
5037		CHABERT	TRADER: CICS VS	CICSC23G	Nov-22 16:40	70,000	Ended
4992		SUSARLA		V71FWLM7	Nov-21 14:37	100,000	Ended
4988		SUSARLA	stored procedur	V71FWLM7	Nov-18 12:56	10,000	Ended
4970		SIRISHA	use case 15	DB2DATA	Nov-17 15:29	100,000	Cancel
4967		ZHONG1		CICSC23G	Nov-17 15:04	1,000	Ended
4961		SIRISHA	use case 9	CICSC22F	Nov-17 14:37	100,000	Ended
4955		SIRISHA	Data Sharing V8	DBDATA8	Nov-17 14:17	100,000	Cancel
4954		SIRISHA	stored procedur	V71FWLM7	Nov-17 13:41	10,000	Ended
4953		SUSARLA	use case 9	CICSC22F	Nov-17 12:47	100,000	Ended
4952		MACHIND	u/c £11	DONDRVRN	Nov-17 12:42	25,000	Ended
4951		SUSARLA	stored procedur	V71FWLM7	Nov-17 12:09	100,000	Ended
4948		SUSARLA	use case 10	PLITEST	Nov-17 11:43	19,223	Ended
4947		SUSARLA	use case 12	COBOLPLI	Nov-17 11:44	100,000	Cancel
4946		SUSARLA	use case 8	VSAMJOB	Nov-17 11:45	90,337	Ended
4945		SUSARLA	USE CASE 4	DB2RUN	Nov-17 11:38	13,836	Ended
4944		SUSARLA	DB2 V8	DBJOB8	Nov-17 11:35	300	Sched
4943		SUSARLA	Data Sharing V7	DB2DATA	Nov-17 11:47	100,000	Cancel
4942	+	SUSARLA	use case 3	MQPUT	Nov-17 11:36	10,000	STEPS

MA a 08/003

Figure 38-19 APA reports for the second run

As for the first run, we focus on the CICS part of the Trader application (report 5041) to see if the changes have made any differences. The first report to display is E01, as shown in Figure 38-20 and in Figure 38-21.

```
  File  View  Navigate  Help
E01: CICS Session Statistics (5041)                          Row 00001 of 00046
Command ===> _                                               Scroll ===> CSR

Environmental Information
   CICS Release                          CICS/TS 2.3

Transaction Statistics
   First Transaction TaskId              0052122
   Last Transaction TaskId               0059908
   Number of TaskId Increments           7,786
   Number of Observed Transactions       1,813
   Transaction Rate (per sec)            18.58
   Peak Active Txns (Observed)           33
   Peak Active Txns (Overall)            80
   MaxTask                               150

Mean Transaction Time (Execution + Suspend + Delay = Service)
   Execution Time                        0.002108
   Suspend Time                          3.746837
   CICS Dispatch Delay Time              0.002472
   MVS Dispatch Delay Time               0.001185
   Service Time                          3.752602

Service Statistics
   Program Requests                      18,519
   Terminal Messages                     549
   Storage Getmains                      202,908
   Storage Freemains                     194,613
   File I/O Requests                     4,103
   Temporary Storage Requests            13,474
MA    a                                                          04/015
```

Figure 38-20 E01: CICS Session Statistics (5041) (1 of 2)

```
  File  View  Navigate  Help
E01: CICS Session Statistics (5041)                          Row 00021 of 00046
Command ===> _                                               Scroll ===> CSR

Service Statistics
   Program Requests                      18,519
   Terminal Messages                     549
   Storage Getmains                      202,908
   Storage Freemains                     194,613
   File I/O Requests                     4,103
   Temporary Storage Requests            13,474
   Transient Data Requests               9,909
   Journal Write Requests                0

Exception or Critical Conditions
   System Dumps                          0
   System Dumps Suppressed               0
   Transaction Dumps                     0
   Transaction Dumps Suppressed          0
   Storage Violations                    0
   Short on Storage occurrences          0
   Times at MaxTask                      0
   Times at Class MaxTask                0

Transaction Counts

      TranId     Count
       MYTD       969
       TDB2       844

MA    a                                                          04/015
```

Figure 38-21 E01: CICS Session Statistics (5041) (2 of 2)

The numbers from the two runs are listed in the next few tables (Table 38-5 to Table 38-8). The mean service time (response time) has been reduced 50%.

First we consider the data shown in Table 38-5.

Table 38-5 CICS summary (5 of 11)

	Report number 5037	**Report number 5041**
Mean Service Time	6.651146 sec.	3.752602 sec.
Transaction Rate	16.91 per sec.	18.58 per sec.
Number of Task	7086	7786
Peak Active Transactions	51	33
TDB2: Mean Suspend Time	11.172 sec.	
TDB2: Mean Delay Time	2.514 sec.	
TDB2: Worst Time	Task Id 44206: Suspend Time: 200.168 sec. Task Id 44616 Delay Time: 112.175 sec.	
TDB2: Wait on DB2 Thread	81.57% of 99.93% of the total Wait Time	
TDB2: MVS Delay (Busy)	18.33% of 99.93% of the total Wait time	

Report E04: CICS Mean Service Time by Transaction shows that mean suspend time has been removed for TDB2, but the mean delay time jumps from 2.5 seconds to 8.0 seconds. This can be explained because the ALLTRAD workload runs both CICS and BATCH jobs and the DB2 Thread contention we had has been removed, making CICS more CPU requester. Because we have CPU limited, CICS must from time to time be the CPU hold by the batch job, as shown in Figure 38-22.

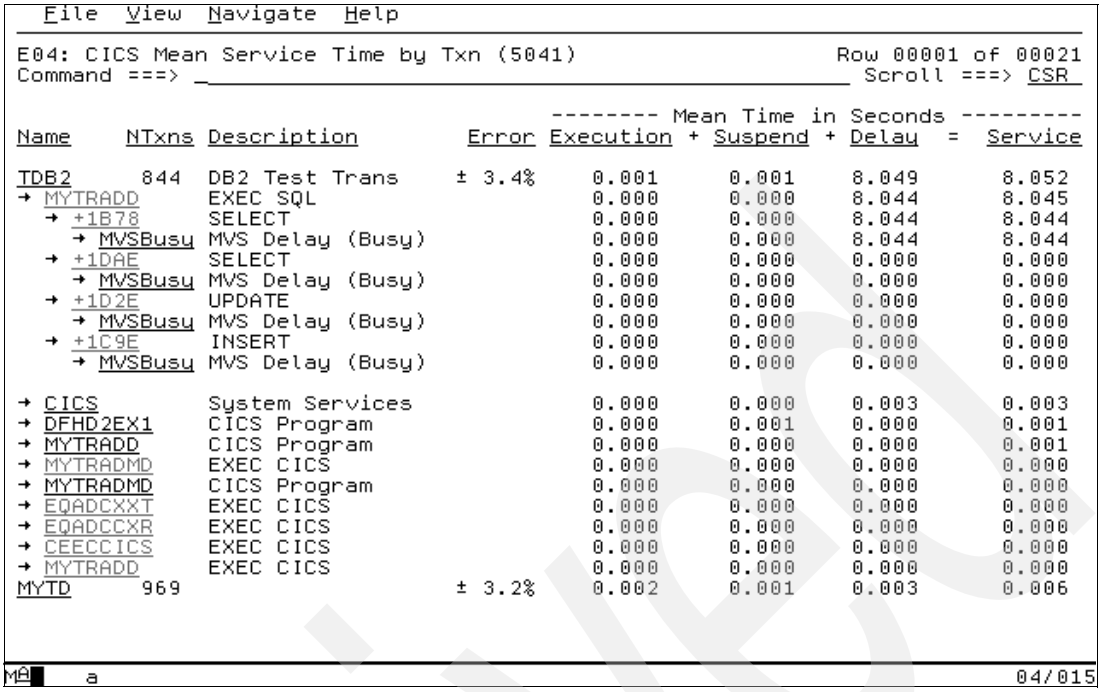


Figure 38-22 E04: CICS Mean Service Time by Transaction (5041) (expanded)

We can fill our CICS summary table with more data, as shown in Table 38-6.

Table 38-6 CICS summary (6 of 11)

	Report number 5037	**Report number 5041**
Mean Service Time	6.651146 sec.	3.752602 sec.
Transaction Rate	16.91 per sec.	18.58 per sec.
Number of Task	7086	7786
Peak Active Transactions	51	33
TDB2: Mean Suspend Time	11.172 sec.	0.001 sec.
TDB2: Mean Delay Time	2.514 sec.	8.049 sec.
TDB2: Worst Time	Task Id 44206: Suspend Time: 200.168 sec. Task Id 44616 Delay Time: 112.175 sec.	
TDB2: Wait on DB2 Thread	81.57% of 99.93% of the total Wait Time	
TDB2: MVS Delay (Busy)	18.33% of 99.93% of the total Wait time	

We know that the mean response time is very high, but this could be because of a low number of task IDs. To see the range of the response times for TDB2, we use report E06. Figure 38-23 shows an expanded (four levels) version of it.

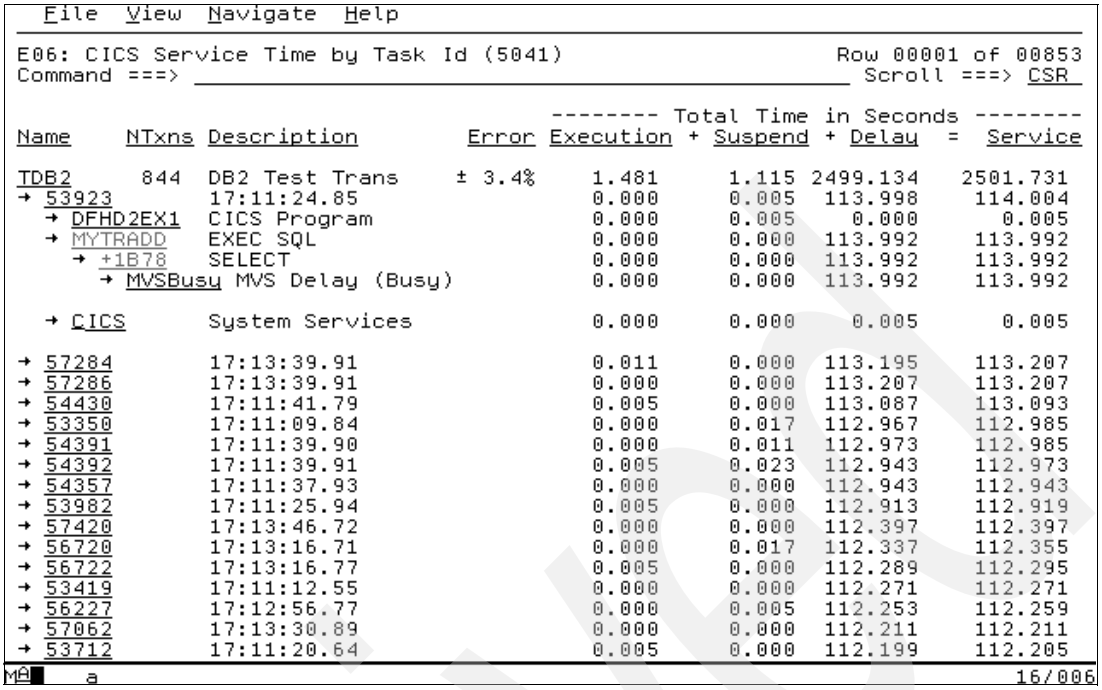


Figure 38-23 E06: CICS Service Time by Task Id (5041) (expanded)

Table 38-7 points out the reduction of the suspend time, keeping its delay time the same.

Table 38-7 CICS summary (7 of 11)

	Report number 5037	**Report number 5041**
Mean Service Time	6.651146 sec.	3.752602 sec.
Transaction Rate	16.91 per sec.	18.58 per sec.
Number of Task	7086	7786
Peak Active Transactions	51	33
TDB2: Mean Suspend Time	11.172 sec.	0.001 sec.
TDB2: Mean Delay Time	2.514 sec.	8.049 sec.
TDB2: Worst Time	Task Id 44206: Suspend Time: 200.168 sec. Task Id 44616 Delay Time: 112.175 sec.	Task Id 54392 Suspend Time: 0.023 sec. Task Id 53923 Delay Time: 113.998 sec.
TDB2: Wait on DB2 Thread	81.57% of 99.93% of the total Wait Time	
TDB2: MVS Delay (Busy)	18.33% of 99.93% of the total Wait time	

Report E07: CICS Wait by Transaction confirms what we already know, as shown in Figure 38-24.

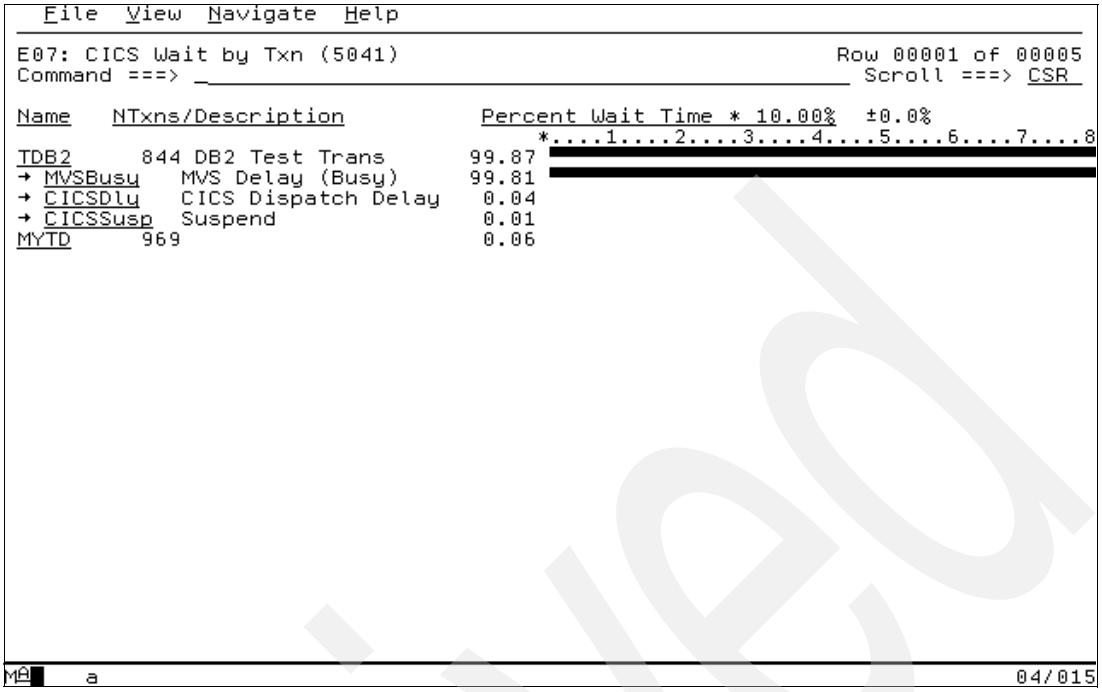


Figure 38-24 E07: CICS Wait by Transaction (5041) (expanded)

Table 38-8 lists the data from the second run.

Table 38-8 CICS summary (8 of 11)

	Report number 5037	**Report number 5041**
Mean Service Time	6.651146 sec.	3.752602 sec.
Transaction Rate	16.91 per sec.	18.58 per sec.
Number of Task	7086	7786
Peak Active Transactions	51	33
TDB2: Mean Suspend Time	11.172 sec.	0.001 sec.
TDB2: Mean Delay Time	2.514 sec.	8.049 sec.
TDB2: Worst Time	Task Id 44206: Suspend Time: 200.168 sec. Task Id 44616 Delay Time: 112.175 sec.	Task Id 54392 Suspend Time: 0.023 sec. Task Id 53923 Delay Time: 113.998 sec.
TDB2: Wait on DB2 Thread	81.57% of 99.93% of the total Wait Time	NO WAIT
TDB2: MVS Delay (Busy)	18.33% of 99.93% of the total Wait time	99.81% of 99.87% of the total Wait time.

38.2.4 Actions taken

In order to avoid any conflict between CICS and batch processing, we can either:

- Use WLM to limit the CPU usage of the Trader batch application.
- Reschedule the batch window later.

In our case we do not allow Trader batch to run at the same time as Trader CICS.

The last workload has the same online transactions as previous runs, but no batch. This is why in Figure 38-25 only one new report (5047) appears.

```
  File  View  Navigate  Help
R02: IBM APA for z/OS Observation List (CAZ0)              Row 00001 of 00183
Command ===> _                                             Scroll ===> CSR

ReqNum  Owned By  Description     Job Name  Date/Time       Samples  Status

 5047    CHABER2  TRADER: CICS VS CICSC23G  Nov-22 18:53     70,000  Ended
 5045    ZHONG    db2+ explain te CICSC23G  Nov-22 17:49     99,999  Ended
 5044    CHABER2  TRADER IMS BATC TRADERI   Nov-22 17:13     20,571  Ended
 5043    CHABER2  TRADER DB2 BATC TRADERD   Nov-22 17:15     45,909  Ended
 5042    CHABER2  TRADER VSAM BAT TRADERB   Nov-22 17:15     42,717  Ended
 5041    CHABERT  TRADER: CICS VS CICSC23G  Nov-22 17:17     70,000  Ended
 5040    CHABER2  TRADER IMS BATC TRADERI   Nov-22 16:36     18,937  Ended
 5039    CHABER2  TRADER DB2 BATC TRADERD   Nov-22 16:38     44,610  Ended
 5038    CHABER2  TRADER VSAM BAT TRADERB   Nov-22 16:37     38,346  Ended
 5037    CHABERT  TRADER: CICS VS CICSC23G  Nov-22 16:40     70,000  Ended
 4992    SUSARLA                  V71FWLM7  Nov-21 14:37    100,000  Ended
 4988    SUSARLA  stored procedur V71FWLM7  Nov-18 12:56     10,000  Ended
 4970    SIRISHA  use case 15     DB2DATA   Nov-17 15:29    100,000  Cancel
 4967    ZHONG1                   CICSC23G  Nov-17 15:04      1,000  Ended
 4961    SIRISHA  use case 9      CICSC22F  Nov-17 14:37    100,000  Ended
 4955    SIRISHA  Data Sharing V8 DBDATA8   Nov-17 14:17    100,000  Cancel
 4954    SIRISHA  stored procedur V71FWLM7  Nov-17 13:41     10,000  Ended
 4953    SUSARLA  use case 9      CICSC22F  Nov-17 12:47    100,000  Ended
 4952    MACHIND  u/c £11         DONDRVRN  Nov-17 12:42     25,000  Ended
 4951    SUSARLA  stored procedur V71FWLM7  Nov-17 12:09    100,000  Ended
 4948    SUSARLA  use case 10     PLITEST   Nov-17 11:43     19,223  Ended
 4947    SUSARLA  use case 12     COBOLPLI  Nov-17 11:44    100,000  Cancel
 4946    SUSARLA  use case 8      VSAMJOB   Nov-17 11:45     90,337  Ended
 4945    SUSARLA  USE CASE 4      DB2RUN    Nov-17 11:38     13,836  Ended
 4944    SUSARLA  DB2  V8         DBJOB8    Nov-17 11:35        300  Sched
MA    a                                                          04/015
```

Figure 38-25 APA Report for the third run

Report E01: CICS Session Statistics (Figure 38-26 and Figure 38-27 on page 1238) shows that service time is almost perfect.

```
  File  View  Navigate  Help
E01: CICS Session Statistics (5047)                        Row 00001 of 00046
Command ===> _                                             Scroll ===> CSR

Environmental Information
   CICS Release                        CICS/TS 2.3

Transaction Statistics
   First Transaction TaskId            0009215
   Last Transaction TaskId             0018290
   Number of TaskId Increments         9,075
   Number of Observed Transactions     1,806
   Transaction Rate (per sec)          21.65
   Peak Active Txns (Observed)         4
   Peak Active Txns (Overall)          79
   MaxTask                             150

Mean Transaction Time (Execution + Suspend + Delay = Service)
   Execution Time                      0.002076
   Suspend Time                        0.001518
   CICS Dispatch Delay Time            0.001711
   MVS Dispatch Delay Time             0.001372
   Service Time                        0.006677

Service Statistics
   Program Requests                    21,563
   Terminal Messages                   212
   Storage Getmains                    233,565
   Storage Freemains                   214,837
   File I/O Requests                   4,722
   Temporary Storage Requests          9,301
MA    a                                                          04/015
```

Figure 38-26 E01: CICS Session Statistics (5047) (1 of 2)

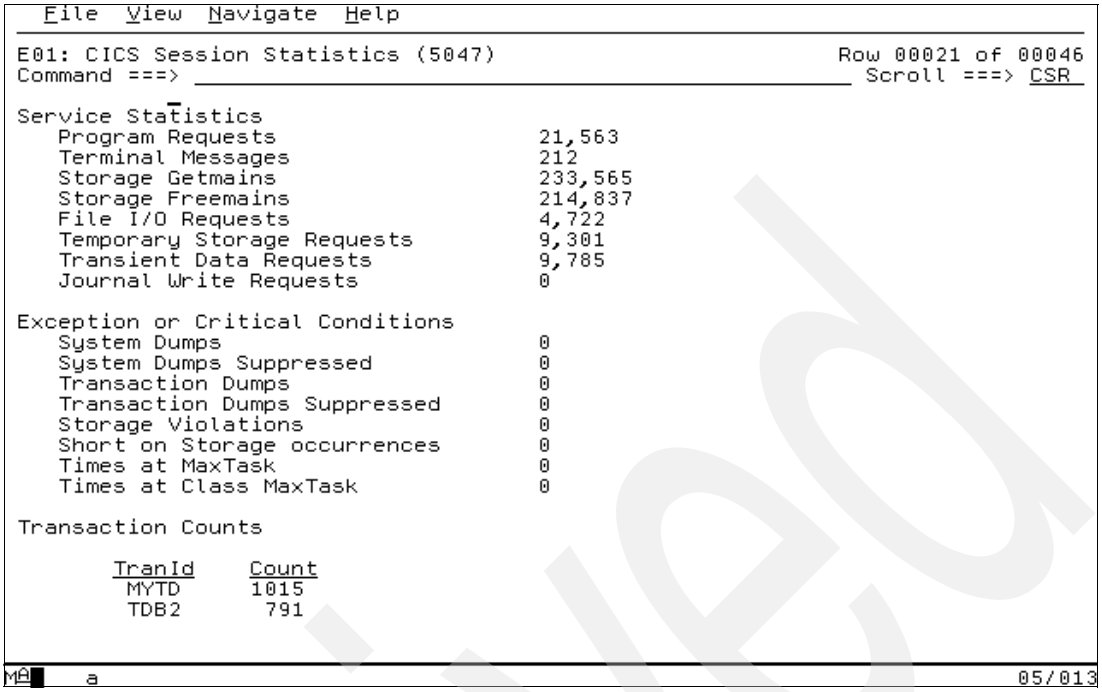


Figure 38-27 E01: CICS Session Statistics (5047) (2 of 2)

Table 38-9 lists the first data of the third run. The peak active transaction is online with what was expected for that workload.

Table 38-9 CICS summary (9 of 11)

	Report number 5037	Report number 5041	Report number 5047
Mean Service Time	6.651146 sec.	3.752602 sec.	0.006677 sec.
Transaction Rate	16.91 per sec.	18.58 per sec.	21.65
Number of Task	7086	7786	9075
Peak Active Transactions	51	33	4
TDB2: Mean Suspend Time	11.172 sec.	0.001 sec.	
TDB2: Mean Delay Time	2.514 sec.	8.049 sec.	
TDB2: Worst Time	Task Id 44206: Suspend Time: 200.168 sec. Task Id 44616 Delay Time: 112.175 sec.	Task Id 54392 Suspend Time: 0.023 sec. Task Id 53923 Delay Time: 113.998 sec.	
TDB2: Wait on DB2 Thread	81.57% of 99.93% of the total Wait Time	NO WAIT	
TDB2: MVS Delay (Busy)	18.33% of 99.93% of the total Wait time	99.81% of 99.87% of the total Wait time.	

Report E04: CICS Mean Service Time by Transaction (Figure 38-28) shows that there is no significant suspend time or delay time impact on the transaction response time.

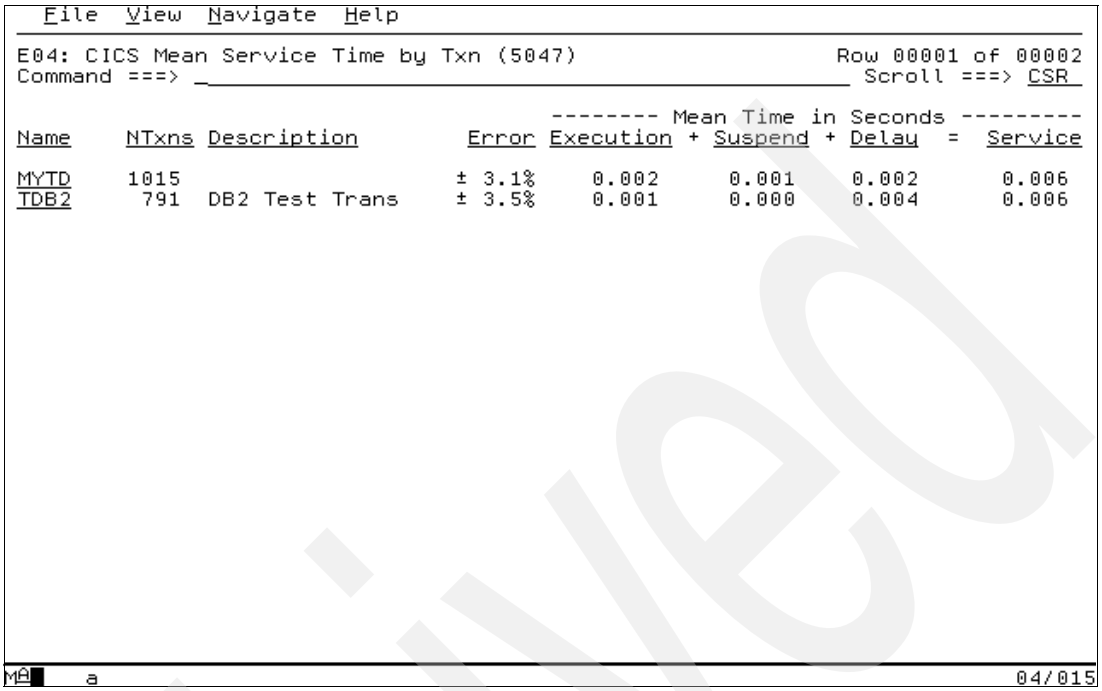


Figure 38-28 E04: CICS Mean Service Time by Transaction (5047)

Table 38-10 summarizes the three runs at that point.

Table 38-10 CICS summary (10 of 11)

	Report number 5037	Report number 5041	Report number 5047
Mean Service Time	6.651146 sec.	3.752602 sec.	0.006677 sec.
Transaction Rate	16.91 per sec.	18.58 per sec.	21.65
Number of Task	7086	7786	9075
Peak Active Transactions	51	33	4
TDB2: Mean Suspend Time	11.172 sec.	0.001 sec.	0.001 sec.
TDB2: Mean Delay Time	2.514 sec.	8.049 sec.	0.002 sec.
TDB2: Worst Time	Task Id 44206: Suspend Time: 200.168 sec. Task Id 44616 Delay Time: 112.175 sec.	Task Id 54392 Suspend Time: 0.023 sec. Task Id 53923 Delay Time: 113.998 sec.	
TDB2: Wait on DB2 Thread	81.57% of 99.93% of the total Wait Time	NO WAIT	
TDB2: MVS Delay (Busy)	18.33% of 99.93% of the total Wait time	99.81% of 99.87% of the total Wait time	

Report E06: CICS Service Time by Task Id (Figure 38-29) drilled down the schematic of the MYTD service time, which is now, after the TDB2 optimization, the most expensive one.

```
  File  View  Navigate  Help
E06: CICS Service Time by Task Id (5047)                    Row 00001 of 01023
Command ===> _____________________________________________ Scroll ===> CSR

                                                 -------- Total Time in Seconds --------
Name     NTxns Description          Error  Execution + Suspend + Delay  =  Service

MYTD     1015                     ± 3.1%      2.333     1.721     2.957      7.012
→ 12030        18:47:52.31                     0.005     0.077     0.000      0.083
  → CEECCICS   EXEC CICS                       0.005     0.000     0.000      0.005
  → MYTRADS    EXEC CICS                       0.000     0.077     0.000      0.077
    → +1AAE    REWRITE FILE(CUSTFILE)          0.000     0.077     0.000      0.077
      → FCIOWAIT Wait on VSAM File I/O         0.000     0.077     0.000      0.077


→ 16724        18:50:12.26                     0.005     0.077     0.000      0.083
→ 16722        18:50:12.26                     0.000     0.077     0.000      0.077
→ 17452        18:51:22.34                     0.000     0.071     0.000      0.071
→ 16726        18:50:12.29                     0.000     0.059     0.000      0.059
→ 16728        18:50:12.29                     0.005     0.053     0.000      0.059
→ 12031        18:47:52.34                     0.000     0.047     0.000      0.047
→ 12033        18:47:52.37                     0.005     0.017     0.000      0.023
→ 10267        18:47:00.20                     0.005     0.011     0.005      0.023
→ 10265        18:47:00.20                     0.011     0.005     0.000      0.017
→ 11577        18:47:36.31                     0.000     0.017     0.000      0.017
→ 16424        18:50:00.26                     0.000     0.017     0.000      0.017
→ 16901        18:50:20.28                     0.000     0.017     0.000      0.017
→ 13631        18:48:38.48                     0.000     0.000     0.011      0.011
→ 10007        18:46:50.20                     0.000     0.005     0.005      0.011
→ 16409        18:49:59.54                     0.000     0.011     0.000      0.011
→ 12967        18:48:16.13                     0.005     0.000     0.005      0.011
MA    a                                                                    11/006
```

Figure 38-29 E06: CICS Service Time by Task Id (5047) (expanded)

Table 38-11 lists the with new values from report E06.

Table 38-11 CICS summary (11 of 11)

	Report number 5037	Report number 5041	Report number 5047
Mean Service Time	6.651146 sec.	3.752602 sec.	0.006677 sec.
Transaction Rate	16.91 per sec.	18.58 per sec.	21.65 per sec.
Number of Task	7086	7786	9075
Peak Active Transactions	51	33	4
TDB2: Mean Suspend Time	11.172 sec.	0.001 sec.	0.001 sec.
TDB2: Mean Delay Time	2.514 sec.	8.049 sec.	0.002 sec.
Worst Time	TDB2 Task Id 44206: Suspend Time: 200.168 sec. Task Id 44616 Delay Time: 112.175 sec.	TDB2 Task Id 54392 Suspend Time: 0.023 sec. Task Id 53923 Delay Time: 113.998 sec.	MYTD Task Id 12030 Suspend Time: 0.077 sec. Task Id 13631 Delay Time: 0.011 sec.
TDB2: Wait on DB2 Thread	81.57% of 99.93% of the total Wait Time	NO WAIT	NO WAIT
TDB2: MVS Delay (Busy)	18.33% of 99.93% of the total Wait time	99.81% of 99.87% of the total Wait time	

Now that the CICS-DB2 interface has been optimized (and it was normal to start with it because it was responsible, for the first run, of more than 13 seconds out of 17 of the response time) we can check whether we can still optimize the batch application.

38.3 APA reports for the batch application

We run the batch application without any CICS activities to avoid delay or suspend due to the CICS transactions.

38.3.1 APA report for the first run

We do not have to manually create an APA request, as it is now part of the batch application itself. The request number assigned to our batch job by APA is 5059, as shown in Figure 38-30.

```
  File  View  Navigate  Help
R02: IBM APA for z/OS Observation List (CAZ0)              Row 00002 of 00184
Command ===> _                                             Scroll ===> CSR

ReqNum  Owned By  Description     Job Name  Date/Time       Samples  Status
 5059    CHABERT  TRADER DB2 BATC TRADERD   Nov-28  6:09     84,957  Ended
 5053    ZHONG    ims / db2 compa CICSC23G  Nov-23 10:55     99,999  Ended
 5047    CHABER2  TRADER: CICS VS CICSC23G  Nov-22 18:53     70,000  Ended
 5045    ZHONG    db2+ explain te CICSC23G  Nov-22 17:49     99,999  Ended
 5044    CHABER2  TRADER IMS BATC TRADERI   Nov-22 17:13     20,571  Ended
 5043    CHABER2  TRADER DB2 BATC TRADERD   Nov-22 17:15     45,909  Ended
 5042    CHABER2  TRADER VSAM BAT TRADERB   Nov-22 17:15     42,717  Ended
 5041    CHABERT  TRADER: CICS VS CICSC23G  Nov-22 17:17     70,000  Ended
 5040    CHABER2  TRADER IMS BATC TRADERI   Nov-22 16:36     18,937  Ended
 5039    CHABER2  TRADER DB2 BATC TRADERD   Nov-22 16:38     44,610  Ended
 5038    CHABER2  TRADER VSAM BAT TRADERB   Nov-22 16:37     38,346  Ended
 5037    CHABERT  TRADER: CICS VS CICSC23G  Nov-22 16:40     70,000  Ended
 4992    SUSARLA                  V71FWLM7  Nov-21 14:37    100,000  Ended
 4988    SUSARLA  stored procedur V71FWLM7  Nov-18 12:56     10,000  Ended
 4970    SIRISHA  use case 15     DB2DATA   Nov-17 15:29    100,000  Cancel
 4961    SIRISHA  use case 9      CICSC22F  Nov-17 14:37    100,000  Ended
 4955    SIRISHA  Data Sharing V8 DBDATA8   Nov-17 14:17    100,000  Cancel
 4954    SIRISHA  stored procedur V71FWLM7  Nov-17 13:41     10,000  Ended
 4953    SUSARLA  use case 9      CICSC22F  Nov-17 12:47    100,000  Ended
 4952    MACHIND  u/c £11         DONDRVRN  Nov-17 12:42     25,000  Ended
 4951    SUSARLA  stored procedur V71FWLM7  Nov-17 12:09    100,000  Ended
 4948    SUSARLA  use case 10     PLITEST   Nov-17 11:43     19,223  Ended
 4947    SUSARLA  use case 12     COBOLPLI  Nov-17 11:44    100,000  Cancel
 4946    SUSARLA  use case 8      VSAMJOB   Nov-17 11:45     90,337  Ended
 4945    SUSARLA  USE CASE 4      DB2RUN    Nov-17 11:38     13,836  Ended
MA  a                                                              04/015
```

Figure 38-30 APA report for the first run (batch job)

We first display the S01 report to have an overview. It points out, as shown in Figure 38-31, that the DB2 SQL processing consumes more than 72 percent of the active CPU. We must concentrate on the DB2 part first to see if it is possible to improve it.

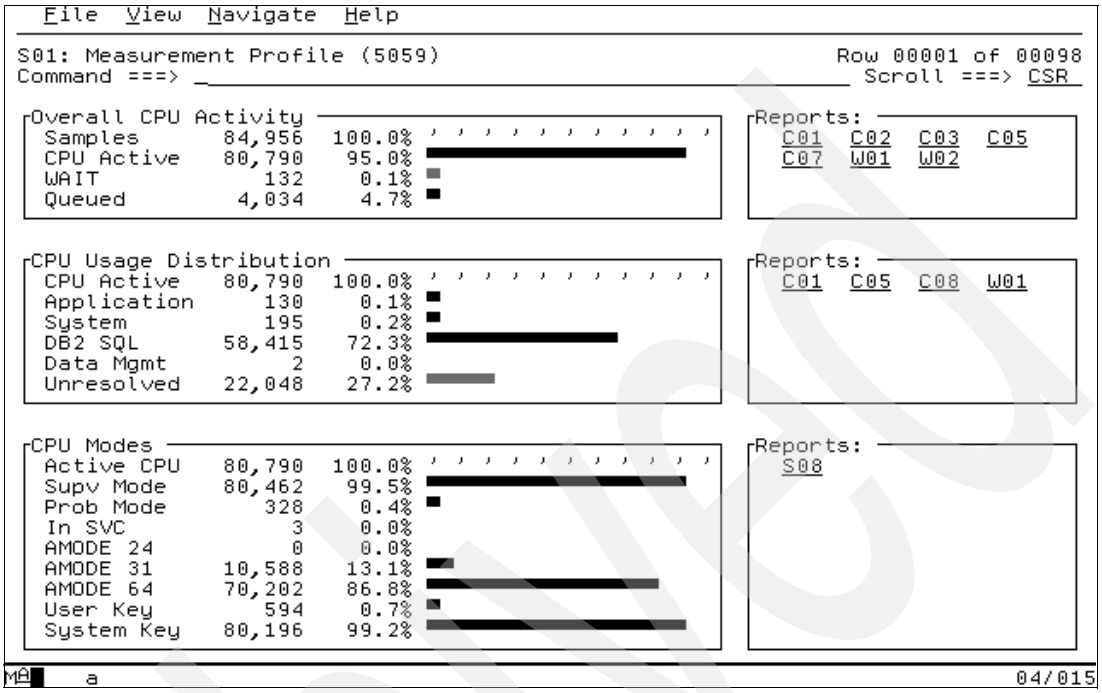


Figure 38-31 S01: Measurement Profile (5059)

By scrolling vertically towards the bottom of the report, we can see that the batch job duration was 8 minutes 29.73 seconds. We can also deduce it from the sample rate we defined for this report (10,000 samples per minute, and the total number of samples we have is 84,957). By dividing 84,957 by 10,000 we obtain 8.4957 minutes, very close to 8.50 minutes, or 8 minutes and 30 seconds.

By displaying the F01: DB2 Measurement Profile report, as shown in Figure 38-32, we can see that precompiler statements number 825, 742, and 774 are the most active and the most CPU consumptive SQL statements.

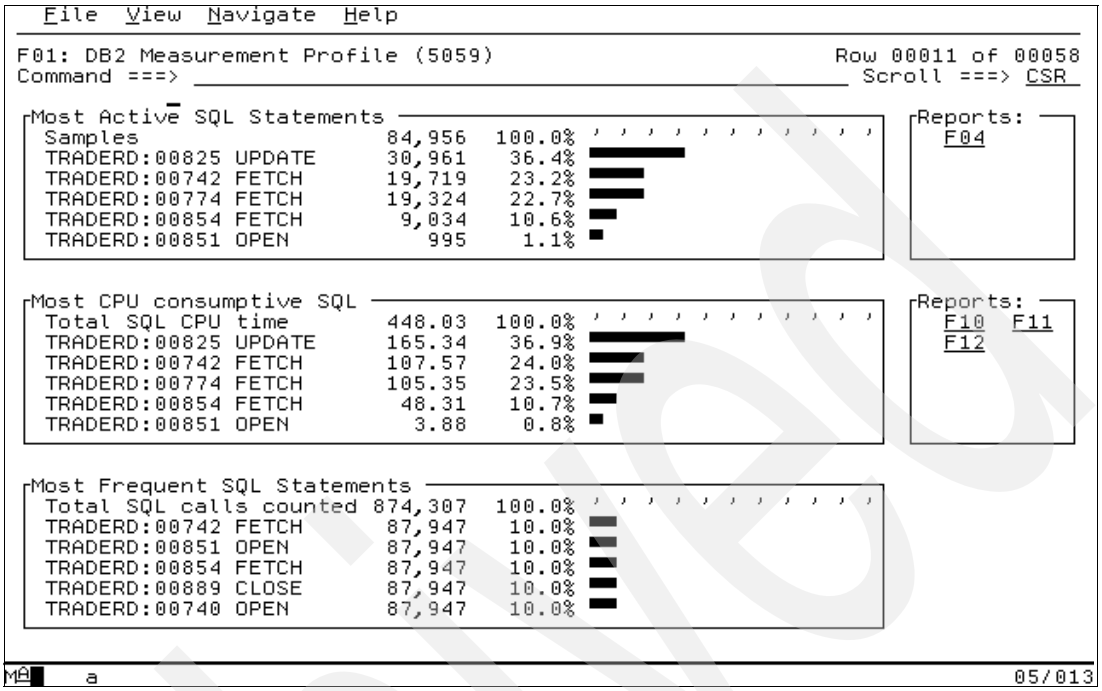


Figure 38-32 F01: DB2 Measurement Profile (5059)

We put the cursor on the F11 hot spot to drill down into more details, as shown in Figure 38-33.

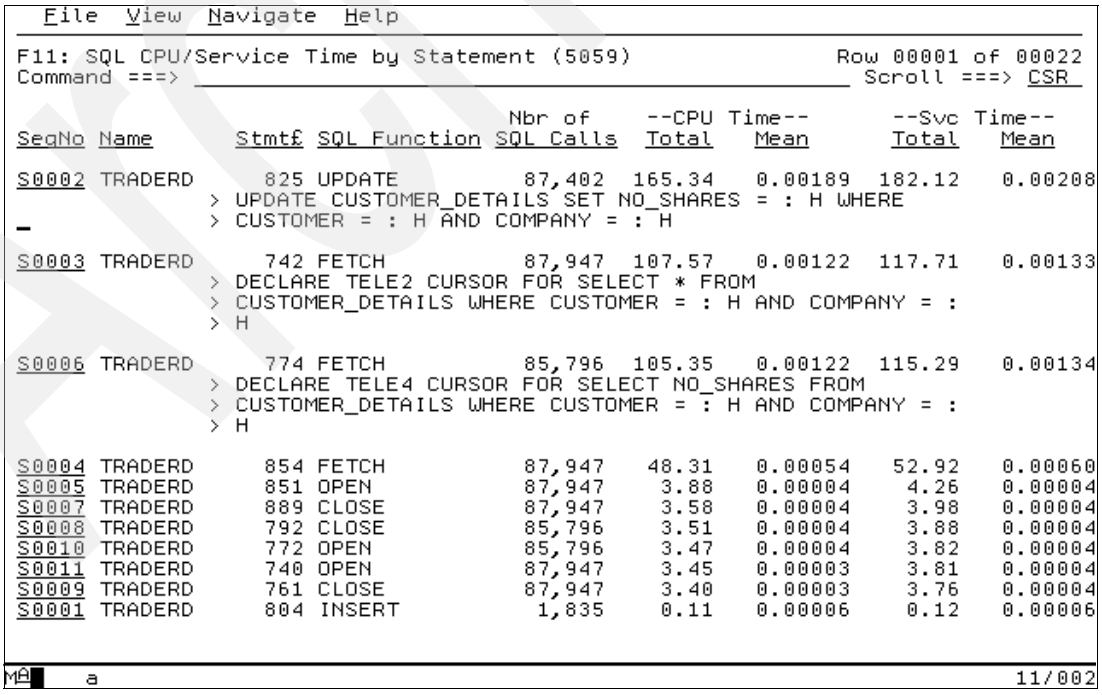


Figure 38-33 F11: SQL CPU/Service Time by Statement (expanded) (5059)

It shows that one UPDATE and two SELECT statements use a lot of CPU. Using the **P** line command, it is easy to display a listing that shows where these statements are located in the source code, as shown in Figure 38-34.

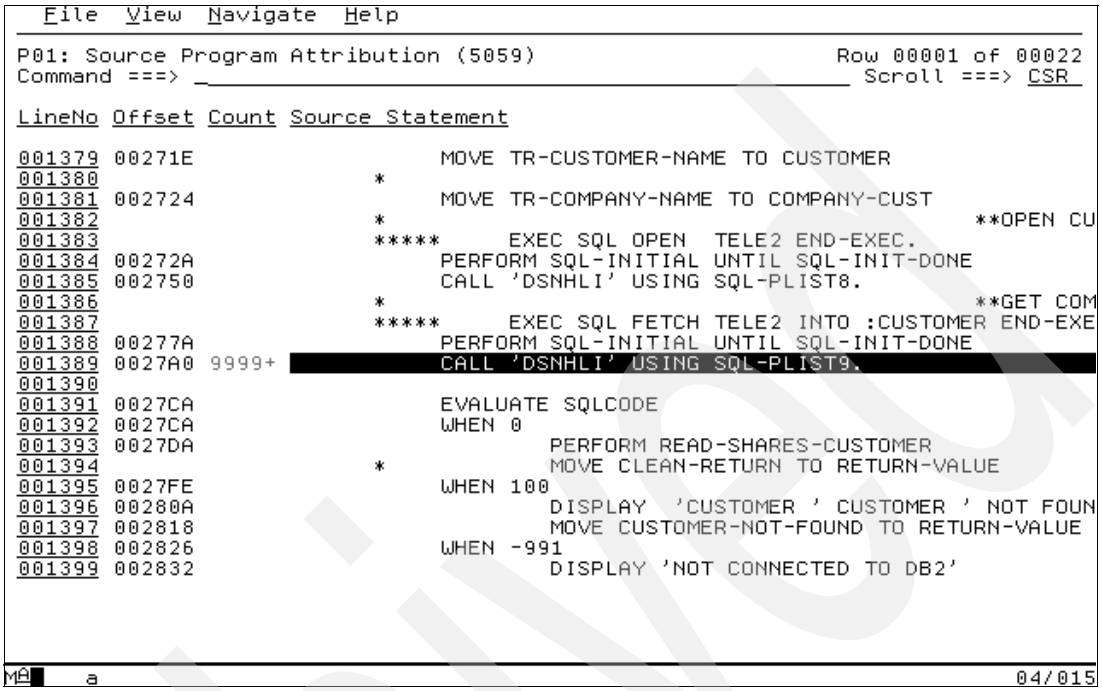


Figure 38-34 P01: Source Program Attribution (5059)

Note: The associations between Source Program Mapping (SPM) files and observation sessions has been done previously through the A01 - Source Program Mapping panel.

But it is not very useful to tune the application. In our case a more relevant Application Performance Analyzer feature is the DB2 dynamic explain. By typing the **Ex** line command, as shown in Figure 38-35, for the statement number S0003, APA displays the panel shown in Figure 38-36.

```
  File  View  Navigate  Help
F11: SQL CPU/Service Time by Statement (5059)              Row 00001 of 00022
Command ===> _____________________________________________ Scroll ===> CSR

                                      Nbr of     --CPU Time--       --Svc Time--
SeqNo Name      Stmt£ SQL Function  SQL Calls   Total    Mean       Total   Mean

S0002 TRADERD     825 UPDATE          87,402  165.34   0.00189  182.12   0.00208
              > UPDATE CUSTOMER_DETAILS SET NO_SHARES = : H WHERE
              > CUSTOMER = : H AND COMPANY = : H

EX003 TRADERD     742 FETCH           87,947  107.57   0.00122  117.71   0.00133
              > DECLARE TELE2 CURSOR FOR SELECT * FROM
              > CUSTOMER_DETAILS WHERE CUSTOMER = : H AND COMPANY = :
              > H

S0006 TRADERD     774 FETCH           85,796  105.35   0.00122  115.29   0.00134
              > DECLARE TELE4 CURSOR FOR SELECT NO_SHARES FROM
              > CUSTOMER_DETAILS WHERE CUSTOMER = : H AND COMPANY = :
              > H

S0004 TRADERD     854 FETCH           87,947   48.31   0.00054   52.92   0.00060
S0005 TRADERD     851 OPEN            87,947    3.88   0.00004    4.26   0.00004
S0007 TRADERD     889 CLOSE           87,947    3.58   0.00004    3.98   0.00004
S0008 TRADERD     792 CLOSE           85,796    3.51   0.00004    3.88   0.00004
S0010 TRADERD     772 OPEN            85,796    3.47   0.00004    3.82   0.00004
S0011 TRADERD     740 OPEN            87,947    3.45   0.00003    3.81   0.00004
S0009 TRADERD     761 CLOSE           87,947    3.40   0.00003    3.76   0.00004
S0001 TRADERD     804 INSERT           1,835    0.11   0.00006    0.12   0.00006

MA  a                                                                  13/004
```

Figure 38-35 Explain line command for statement S0003

By pressing Enter, Application Performance Analyzer displays the DB2 Dynamic Explain information shown in Figure 38-36.

```
  File  View  Navigate  Help
  File  View  Navigate  Help
DB2 EXPLAIN (5059)                                    Row 00001 of 00001
Command ===> _____________________________________________ Scroll ===> CSR

Blk  Plan  Mix  Join  Acc  Match  Index  Sort New  Sort Comp  Table
Num. Num.  Op   Mthd  Type Cols   Only   U J O G   U J O G    Name

0001    1    0    0    R       0     N    N N N N   N N N N   CUSTOMER_DETA

MA  a                                                                  06/017
```

Figure 38-36 DB2 explain for statement S0003

If we are not DB2 specialists, an easier way to read the DB2 explain report is to expand it by placing the cursor on the block number field and pressing Enter, as shown in Figure 38-37.

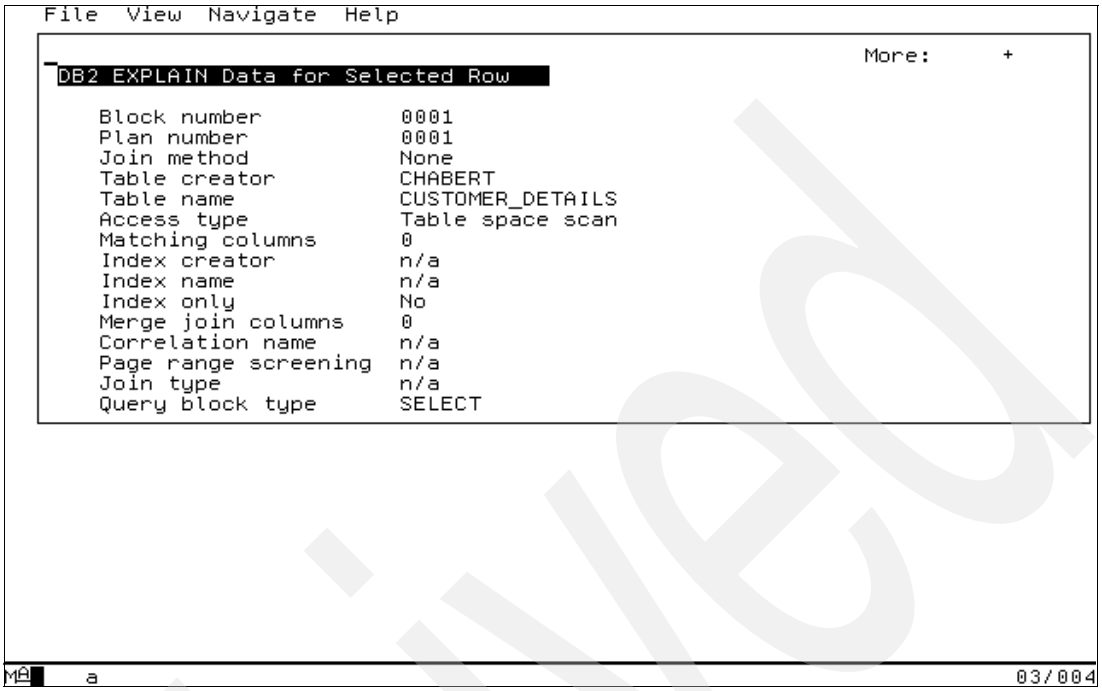


Figure 38-37 DB2 explain for S0003 expanded (1 of 3)

This report shows that the access type is *table space scan*, which is the worst case that we can have to access DB2 data. It is because no index has been created for that table.

Table 38-12 summarizes the data we have for statement number S0003 of our first run, and we have to remember that the total elapse time for the batch was 8 minutes and 30 seconds and the DB2 SQL CPU usage was 72 percent of the active CPU.

Table 38-12 APA data for statement S0003

	First run of DB2 batch
Number of calls (Report F01)	87 947
CPU Time: Total (Report F11)	105.57 sec.
CPU Time: Mean (Report F11)	0.00122 sec.
Service Time: Total (Report F11)	117.71 sec.
Service Time: Mean (Report F11)	0.00133 sec.
Access type (DB2 Explain)	Tablespace Scan
Index name (DB2 Explain)	No index
Matching columns (DB2 Explain)	0
Index only (DB2 Explain)	No

38.3.2 Action taken

We now create a unique index, as shown in Example 38-4.

Example 38-4 Unique index definition for customer details

```
CREATE TABLE CUSTOMER_DETAILS
  (
   CUSTOMER                       CHAR(25) NOT NULL,
   COMPANY                        CHAR(20) NOT NULL,
   NO_SHARES                      INTEGER,
   PRIMARY KEY(CUSTOMER,COMPANY) )
   IN DSNDB04.TRADER;

 CREATE UNIQUE INDEX CUST_IND ON
   CUSTOMER_DETAILS(CUSTOMER,COMPANY) USING STOGROUP FBIGROUP;
```

38.3.3 APA reports for the second run

We rerun the same batch. The tables have been re-initialized to the same status as the one they have for the first run. Figure 38-38 shows that the new report is number 5061.

```
  File  View  Navigate  Help
R02: IBM APA for z/OS Observation List (CAZ0)              Row 00001 of 00109
Command ===> _                                             Scroll ===> CSR

ReqNum  Owned By  Description     Job Name  Date/Time       Samples  Status

 5061    CHABERT  TRADER DB2 BATC TRADERD   Nov-28  6:24     9,802  Ended
 5059    CHABERT  TRADER DB2 BATC TRADERD   Nov-28  6:09    84,957  Ended
 5053    ZHONG    ims / db2 compa CICSC23G  Nov-23 10:55    99,999  Ended
 5047    CHABER2  TRADER: CICS VS CICSC23G  Nov-22 18:53    70,000  Ended
 5045    ZHONG    db2+ explain te CICSC23G  Nov-22 17:49    99,999  Ended
 5044    CHABER2  TRADER IMS BATC TRADERI   Nov-22 17:13    20,571  Ended
 5043    CHABER2  TRADER DB2 BATC TRADERD   Nov-22 17:15    45,909  Ended
 5042    CHABER2  TRADER VSAM BAT TRADERB   Nov-22 17:15    42,717  Ended
 5041    CHABERT  TRADER: CICS VS CICSC23G  Nov-22 17:17    70,000  Ended
 5040    CHABER2  TRADER IMS BATC TRADERI   Nov-22 16:36    18,937  Ended
 5039    CHABER2  TRADER DB2 BATC TRADERD   Nov-22 16:38    44,610  Ended
 5038    CHABER2  TRADER VSAM BAT TRADERB   Nov-22 16:37    38,346  Ended
 5037    CHABERT  TRADER: CICS VS CICSC23G  Nov-22 16:40    70,000  Ended
 4992    SUSARLA                  V71FWLM7  Nov-21 14:37   100,000  Ended
 4988    SUSARLA  stored procedur V71FWLM7  Nov-18 12:56    10,000  Ended
 4970    SIRISHA  use case 15     DB2DATA   Nov-17 15:29   100,000  Cancel
 4961    SIRISHA  use case 9      CICSC22F  Nov-17 14:37   100,000  Ended
 4955    SIRISHA  Data Sharing V8 DBDATA8   Nov-17 14:17   100,000  Cancel
 4954    SIRISHA  stored procedur V71FWLM7  Nov-17 13:41    10,000  Ended
 4953    SUSARLA  use case 9      CICSC22F  Nov-17 12:47   100,000  Ended
 4952    MACHIND  u/c £11         DONDRVRN  Nov-17 12:42    25,000  Ended
 4951    SUSARLA  stored procedur V71FWLM7  Nov-17 12:09   100,000  Ended
 4948    SUSARLA  use case 10     PLITEST   Nov-17 11:43    19,223  Ended
 4947    SUSARLA  use case 12     COBOLPLI  Nov-17 11:44   100,000  Cancel
 4946    SUSARLA  use case 8      VSAMJOB   Nov-17 11:45    90,337  Ended
MA    a                                                          04/015
```

Figure 38-38 R02: PAP for z/OS Observation List

The number of samples is only 9 802. Because we have the same options as for the first run, we can deduce that the total elapse time is less than 1 minute (the sample rate was 10,000 records per minute). We do not even have to analyze any other report to know that the index we have created before is very useful to improve the batch response time.

Nevertheless, we display the same reports as for the first run to make a complete comparison. Report S01 shown in Figure 38-39 points out that the DB2 SQL process now only use 56% of the active CPU (it was 72 percent for the first run).

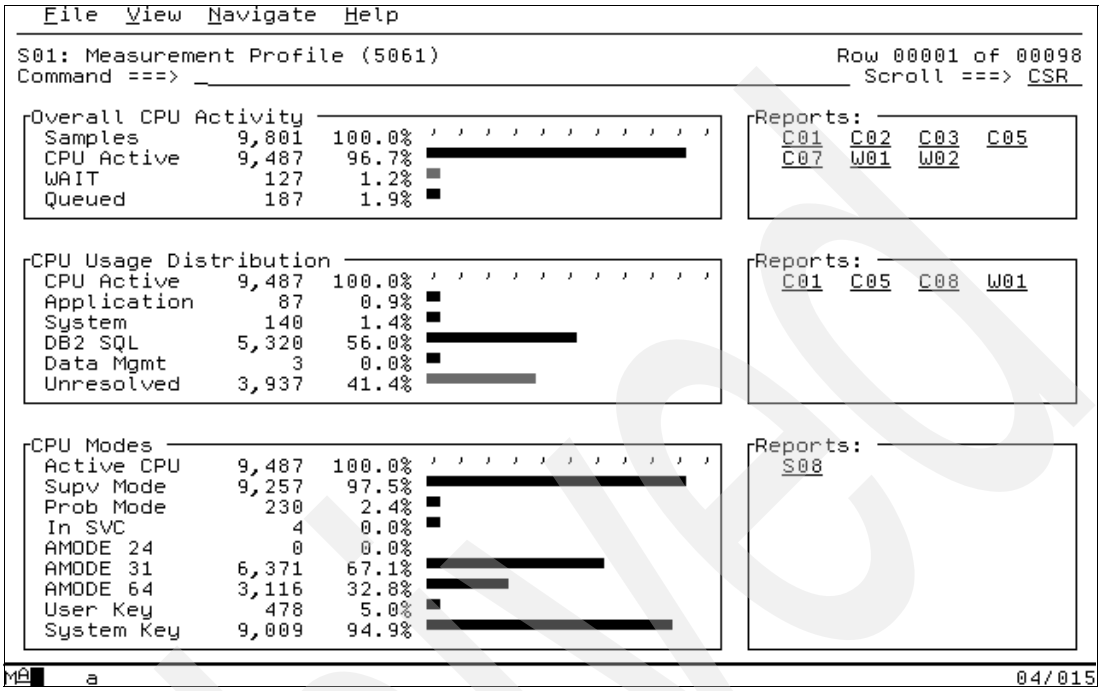


Figure 38-39 S01: Measurement Profile (5061)

The next report, F01 (Figure 38-40) shows that the number of calls (87,479) for statement TRADER:00742 is about the same as before (87,947), but its CPU consumption is now only 4.65 seconds (11.6% of the total SQL CPU time). It was 107.57 seconds before (24.0% of the total SQL CPU time).

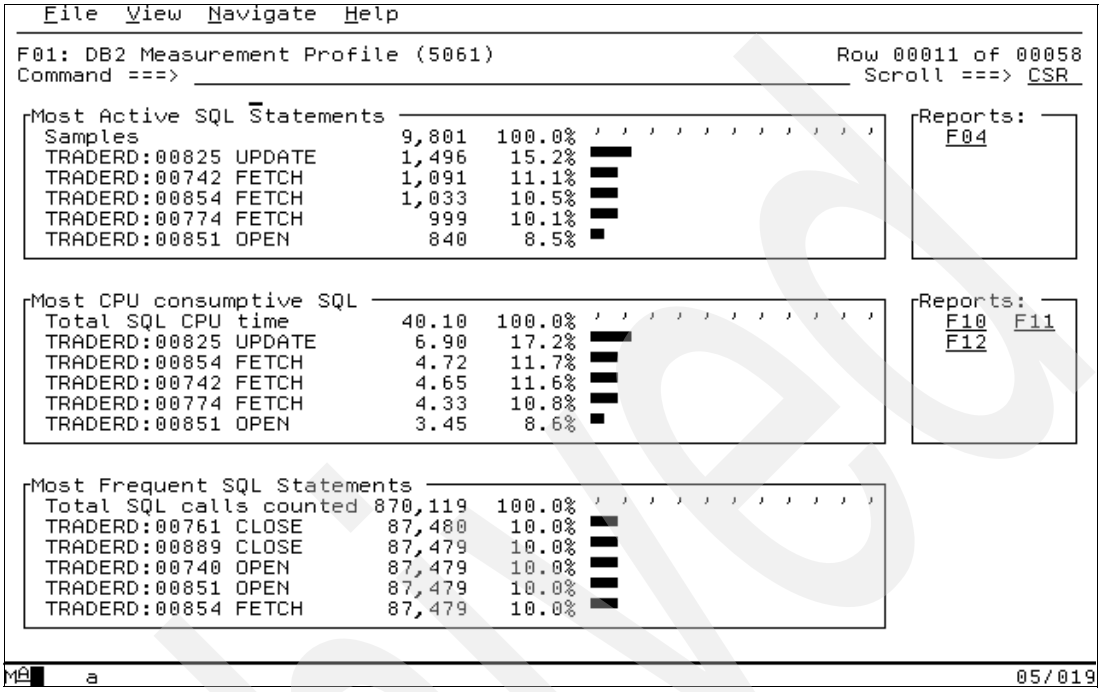


Figure 38-40 F01: DB2 Measurement Profile (5061)

We put the cursor on the F11 hot spot to drill down into more detail, as shown in Figure 38-41.

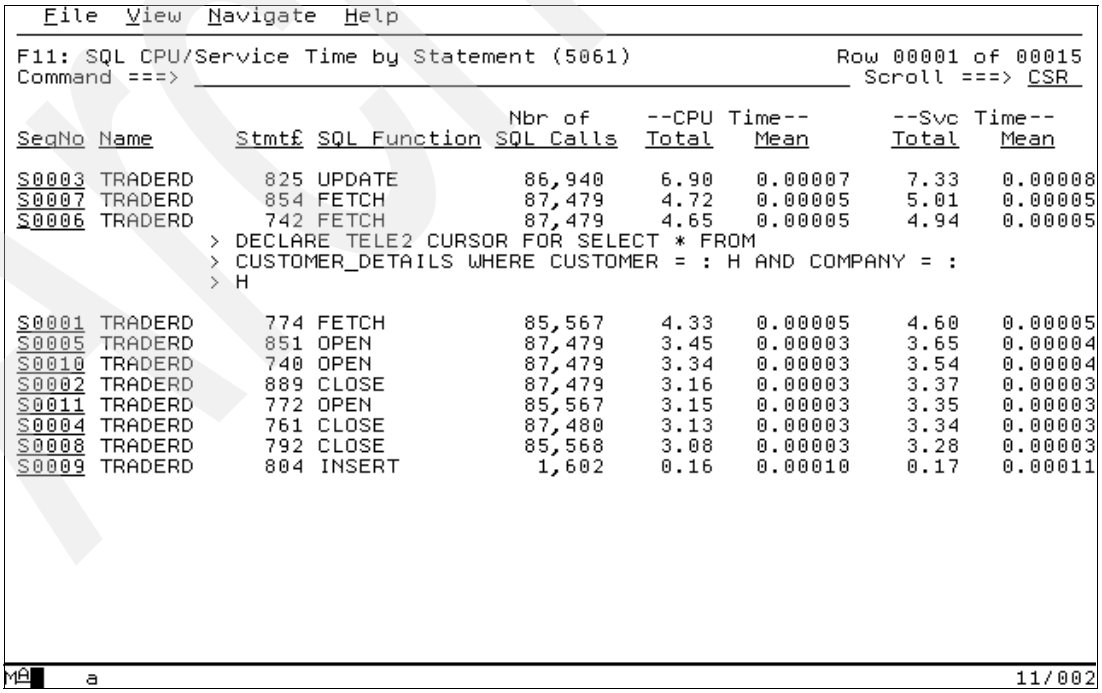


Figure 38-41 F11: SQL CPU/Service Time by Statement (5061)

Both the CPU time and the service time have been enormously reduced. They are now about *125 times shorter* than before.

The last report is the DB2 Explain report, as shown in Figure 38-42. It proves that the index we created before has been used.

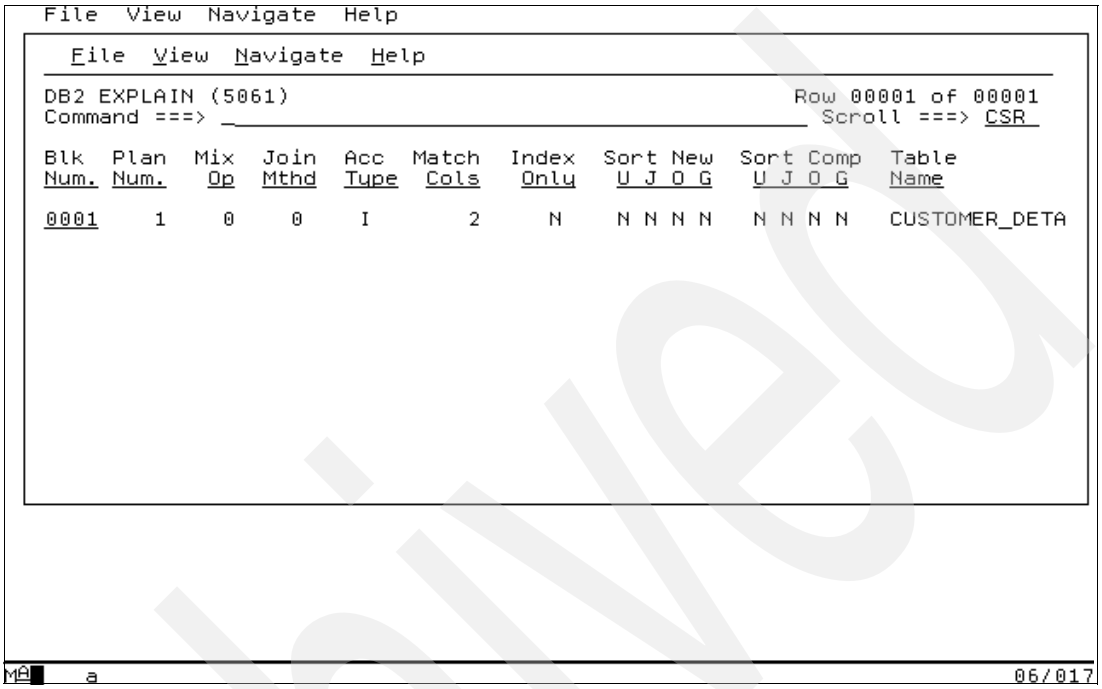


Figure 38-42 DB2 Explain

The expanded version of it is shown in Figure 38-43.

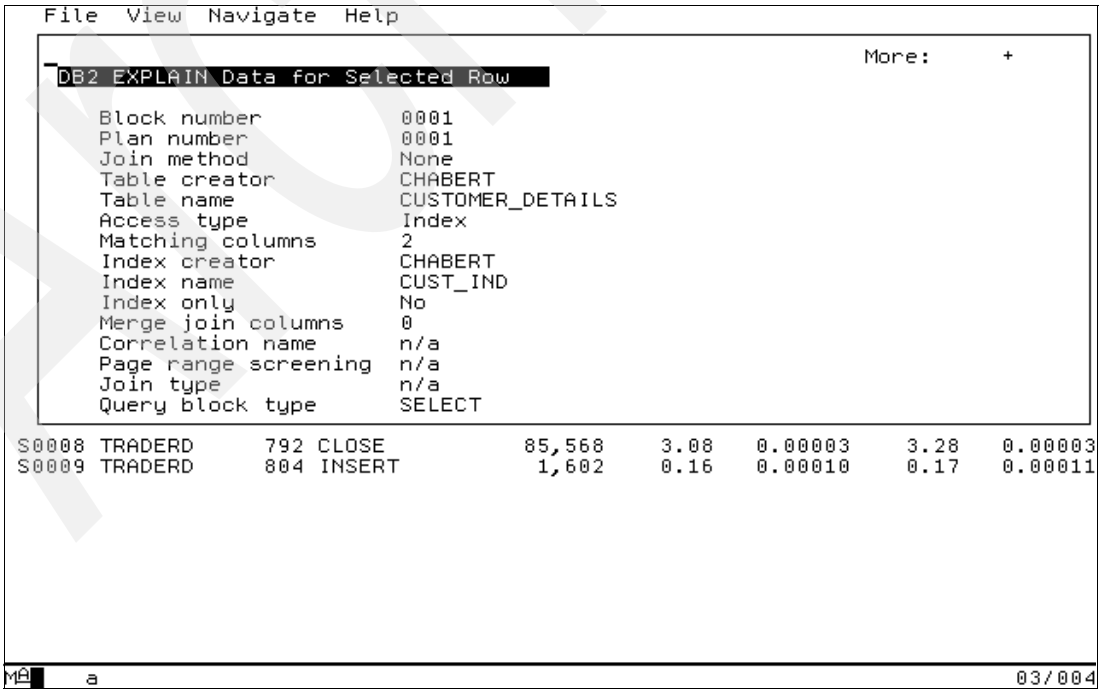


Figure 38-43 DB2 Explain (expanded)

Table 38-13 summarizes the data we have for the same SQL statement for our two runs.

*Table 38-13 APA data for the DECLARE TELE2 CURSOR FOR SELECT **

	First run of DB2 batch	Second run of DB2 batch
Number of calls (Report F01)	87 947	87 479
CPU Time: Total (Report F11)	105.57 sec.	4.65 sec.
CPU Time: Mean (Report F11)	0.00122 sec.	0.00005 sec.
Service Time: Total (Report F11)	117.71 sec.	4.95 sec.
Service Time: Mean (Report F11)	0.00133 sec.	0.00005 sec.
Access type (DB2 Explain)	Tablespace Scan	Index
Index name (DB2 Explain)	No index	CUST_IND
Matching columns (DB2 Explain)	0	2
Index only (DB2 Explain)	No	No

A

Additional material

This book refers to additional material that can be downloaded from the Internet as described below.

Locating the Web material

The Web material associated with this book is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

ftp://www.redbooks.ibm.com/redbooks/SG247372

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the Redbooks form number, SG247372.

Using the Web material

The additional Web material that accompanies this book includes the following file:

File name	Description
SG247372.zip	Zipped code samples

The following system configuration is recommended:

Hard disk space:	4 MB for the downloaded zip file and unpacked files
Operating System:	Windows 2000/XP
Processor:	Pentium
Memory:	128 MB

System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space:	4 MB for the downloaded zip file and unpacked files
Operating System:	Windows 2000/XP
Processor:	Pentium
Memory:	128 MB

How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder.

The extracted files are all in binary format. They are the output of the TSO TRANSMIT command.

Use your mainframe file transfer protocol to upload the binary files. You must use the following attributes: FB, LRECL=80, BLKSIZE=3120.

After each file is uploaded, issue the following command from the TSO READY prompt:

RECEIVE INDA(*xxxx*)

In this command, *xxxx* is the name of the file.

You will receive the following messages, if you issue the command against the **source** file as shown in Example A-1.

Example: A-1 Receive INDA(xxxx) messages

```
INMR901I Dataset DAVINR1.REDBOOK.PDPAK.SOURCE from DAVINR1 on NODENAME
INMR906A Enter restore parameters or 'DELETE' or 'END' +
```

You can reply as shown in Example A-2.

Example: A-2 Receive INDA(xxxx) reply

```
indsn(pdtools.redbook.source)
```

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see “How to get IBM Redbooks” on page 1256. Note that some of the documents referenced here may be available in softcopy only.

- *Supporting On Demand Business Applications with the IBM Problem Determination Tools (APA, DT, DT with Advanced Facilities, FA, File Export, FM, WS)*, SG24-7192

Other publications

These publications are also relevant as further information sources:

- *Fault Analyzer User Guide and Reference,* SC19-1088-02
- *Application Performance Analyzer User Guide,* SC19-1065-00
- *Debug Tool and Debug Tool Utilities and Advanced Functions for z/OS,* SC19-1071-02
- *File Export User Guide,* SC18-9178-09
- *File Manager User Guide and Reference,* SC19-1037-00
- *Workload Simulator User Guide*, SC31-8948-00

Online resources

These Web sites are also relevant as further information sources:

- Application Performance Analyzer documentation and support:
  http://www-306.ibm.com/software/awdtools/apa/
- Debug Tool documentation and support:
  http://www-306.ibm.com/software/awdtools/debugtool/
- Fault Analyzer documentation and support:
  http://www-306.ibm.com/software/awdtools/faultanalyzer/
- File Export documentation and support:
  http://www-306.ibm.com/software/awdtools/fileexport/
- File Manager documentation and support:
  http://www-306.ibm.com/software/awdtools/filemanager/
- Workload Simulator documentation and support:
  http://www-306.ibm.com/software/awdtools/workloadsimulator/
- Other IBM publications:
  http://www-306.ibm.com/software/awdtools/

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

D

F

O

R

S

T

V

W

Redbooks

IBM Application Development and Problem Determination Tools V7 for System z

(2.0” spine)
2.0” <-> 2.498”
1052 <-> 1314 pages

IBM Application Development and Problem Determination Tools V7 for System z

Application Performance Analyzer, Debug Tool Utilities and Advanced Functions, Fault Analyzer, File Export, File Manager, and Workload Simulator

With features and functionality from the latest versions of each product

New scenarios and practical advice

Completely new File Export sections

This IBM Redbooks publication introduces the IBM Application Development and Problem Determination Tools V7. The tools covered are:

- IBM Application Performance Analyzer for z/OS Version 1 Release 1
- IBM Debug Tool for z/OS Version 7 Release 1
- IBM Debug Tool and Advanced Facilities for z/OS Version 7 Release 1
- IBM Fault Analyzer for z/OS Version 7 Release 1
- IBM File Manager for z/OS Version 7 Release 1
- IBM File Export for z/OS Version 1 Release 2
- IBM Workload Simulator for z/OS and OS/390 Version1 Release 1

This comprehensive suite of powerful yet easy-to-use tools helps developers efficiently build, test, and service applications while minimizing errors and increasing cost-effectiveness. The tools allow users to quickly identify and resolve problems that occur in batch, IMS, DB2, CICS, and UNIX System Services applications.

For each tool, this book provides step-by-step instructions for installation and customization, a detailed discussion of the features and capabilities, and guidance for using the tool in your environment.

Practical scenarios demonstrate how to apply the tools to monitor, manage, debug, and analyze applications in a real-world environment.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-7372-00 ISBN 0738486302