IBM® Information Management Software

Managing DB2 for z/OS Utilities with DB2 Tools Solution Packs

- Simplify DB2 utilities definition and scheduling
- Choose the best recovery and disaster recovery plan
- Minimize window of execution for utilities

Paolo Bruni
Carlos Alberto Gomes da Silva Junior
Craig McKellar
Adilet Sabyrbaev
Tim Willging

Redbooks

ibm.com/redbooks

International Technical Support Organization

Managing DB2 for z/OS Utilities with DB2 Tools Solution Packs

July 2013

SG24-8046-01

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

Second Edition (July 2013)

This edition applies to IBM DB2 Utilities Solution Pack for z/OS, V1.1 (program number 5697-DUM) and IBM DB2 Fast Copy Solution Pack for z/OS, V1.1 (program number 5697-DFM) for use with IBM DB2 Version 10.1 for z/OS (program number 5605-DB2) and IBM DB2 Utilities Suite Version 10.1 for z/OS (program number 5655-V41).

Contents

Tables

Figures

Examples

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 grant 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 websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites 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.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

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

CICS®
DB2®
DS8000®
FlashCopy®
IBM®
IMS™
InfoSphere®
MVS™
OMEGAMON®
Optim™
RACF®
Redbooks®
Redbooks (logo) ®
Resource Measurement Facility™
RMF™
System Storage®
System z®
Tivoli®
z/OS®
z/VM®

The following terms are trademarks of other companies:

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

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

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

Preface

IBM® DB2® Tools for z/OS® support and take advantage of the latest versions of DB2 for z/OS. These tools are integral for the administration of the DB2 for z/OS environment and for optimization of data performance. In addition, the IBM portfolio addresses additional client requirements in the areas of data governance and version upgrade acceleration.

Underlying the operation of any database management system are the utilities. With the number of database objects growing exponentially, managing utility jobs, meeting service level agreements (SLAs), and ensuring recoverability can be overwhelming. IBM offers DB2 Tools solution packs that assist in the DB2 utilities management process. Solution packs combine several products into a single consolidated solution providing everything necessary to ensure the execution of a set of database administration functions. The goals are to reduce the operational complexity and reduce cost.

The objective of this IBM Redbooks® publication is to document the added value in terms of productivity and performance for database administrators when using the IBM DB2 Utilities Solution Pack and the IBM DB2 Fast Copy Solution Pack. We show the functions of the tools provided by the solution packs as used in real-life scenarios and adopting utilities best practices.

Authors

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

Paolo Bruni is a DB2 Information Management Project Leader at the International Technical Support Organization based in the Silicon Valley Lab since 1998. He has authored several IBM Redbooks publications about DB2 for z/OS and related tools and has conducted workshops and seminars worldwide. During his many years with IBM, in development and in the field, Paolo has worked mostly on database systems.

Carlos Alberto Gomes da Silva Junior is an IBM Brazil DB2 Systems Programmer providing Service Operations Delivery on server systems of large IBM clients. Carlos has seven years of experience on Mainframe Technology for DB2, IBM CICS®, and IBM IMS™. He has experience on DB2 migrations, disaster recovery, and monitoring systems and applications. He supports DB2 tools, such as DB2 High Performance Unload, DB2 Table Editor, DB2 Log Analysis, and DB2 Utilities Enhancement Tool. Carlos is certified in DB2 9 Fundamentals and DB2 9 for z/OS System Administration and works on course development and teaches DB2 for z/OS classes in Brazil.

Craig McKellar has been a DBA on IBM System z® for over 21 years, and over the last few years, he has used this experience in his role as an Accelerated Value Leader in Canberra, Australia. He provides hands-on support and consultation to large government clients for IM software. Craig has co-authored several IBM Redbooks publications on DB2 for z/OS tools.

Adilet Sabyrbaev is a Client Technical Professional (technical pre-sales) with IBM in Moscow, Russian Federation. Adilet holds a Masters in Computer Science from Tomsk State University of Control Systems and Radioelectronics and has seven years of experience in database development and administration (DB2 for z/OS, DB2 for LUW, and Oracle). He has recently been involved in three successful large DB2 benchmarks on IBM System z.

Tim Willging is a Database Tools Architect and Strategist at Rocket Software. Tim has been the product author and lead developer for DB2 Recovery Expert for z/OS. He has over 20 years of experience developing software for DB2 on z/OS and enterprise storage systems.

Special thanks to Ed Holt for helping remotely throughout this project.

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

Bob Haimowitz
International Technical Support Organization

Miguel Baez
Fai Chew
Calene Janacek
Jim MacLaren
Ravi Mahendrakar
Mary Petras
Kevin Poole
Haakon Roberts
Roger St Denis
IBM Silicon Valley Lab

Tim Ashmore
Reggie Culpepper
Jennifer Nelson
Kelly Smith
Denise Tabor
Rocket Software

Darren Bycof
Syncsort

Thanks to the authors of the previous editions of this book.

- Authors of the first edition, *Managing DB2 for z/OS Utilities with DB2 Tools Solution Packs*, SG24-8046-00, published in March 2013, were:

  Paolo Bruni, Edwin Holt, Jan Larsson, Craig McKellar, and Fabricio Pimentel

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out 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 books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

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

  ibm.com/redbooks

- Send your comments in an email to:

  redbooks@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

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:

http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

Summary of changes

This section describes the technical changes made in this edition of the book and in previous editions. This edition might also include minor corrections and editorial changes that are not identified.

Summary of Changes
for SG24-8046-01
for *Managing DB2 for z/OS Utilities with DB2 Tools Solution Packs*
as created or updated on July 16, 2013.

July 2013, Second Edition

This revision reflects the addition, deletion, or modification of new and changed information described below.

New information

- More information on the DB2 tools solution packs

Changed information

- Revised some scenarios of tools utilization and integration

Part 1

Overview and background information

In this part, we provide a general introduction to the use of DB2 tools for utilities management by including the following chapters:

1

DB2 Tools solution packs

In today's IT environments, data growth remains IT's biggest challenge, followed by scalability and performance, and cost containment. Data management must drive competitive advantage, business continuity, and availability, which drive strategic plans. Critical to success is the ability to maintain or improve user service levels and satisfaction while growing and adding data centers drives the need for consistent technology and architectural decisions to maximize purchase decisions.

For DB2 environments, DB2 tools and tools solution packs have these main objectives:

- Reducing costs
- Increasing responsiveness
- Maximizing IT staff productivity

Tools are evolving with functional improvements, more integration, and the incorporation of autonomic solutions and best practices.

Solution packs address the areas of common activities that the DBAs deal with on a regular basis:

- Managing the database
- Managing the data
- Managing the performance

Solution packs offer a complete solution for all needs in these areas rather than having to purchase multiple products with simplified installation and maintenance. They build intelligence into when and how actions are performed, optimizing the performance and resource utilization associated with DBA activities.

We look at the following areas:

- DB2 tools solution packs
- DB2 Utilities and Fast Copy solution packs
- Installing the DB2 Utilities Solution Pack
- Installing the DB2 Fast Copy Solution Pack

1.1 DB2 tools solution packs

Solution packs combine several products into a single consolidated solution providing everything necessary to ensure the execution of a set of database administration functions. The objectives are to reduce the operational complexity and reduce cost.

IBM currently offers the following four DB2 tools solution packs:

- IBM DB2 Administration Solution Pack for z/OS (program number: 5697-DAM), see Figure 1-1.

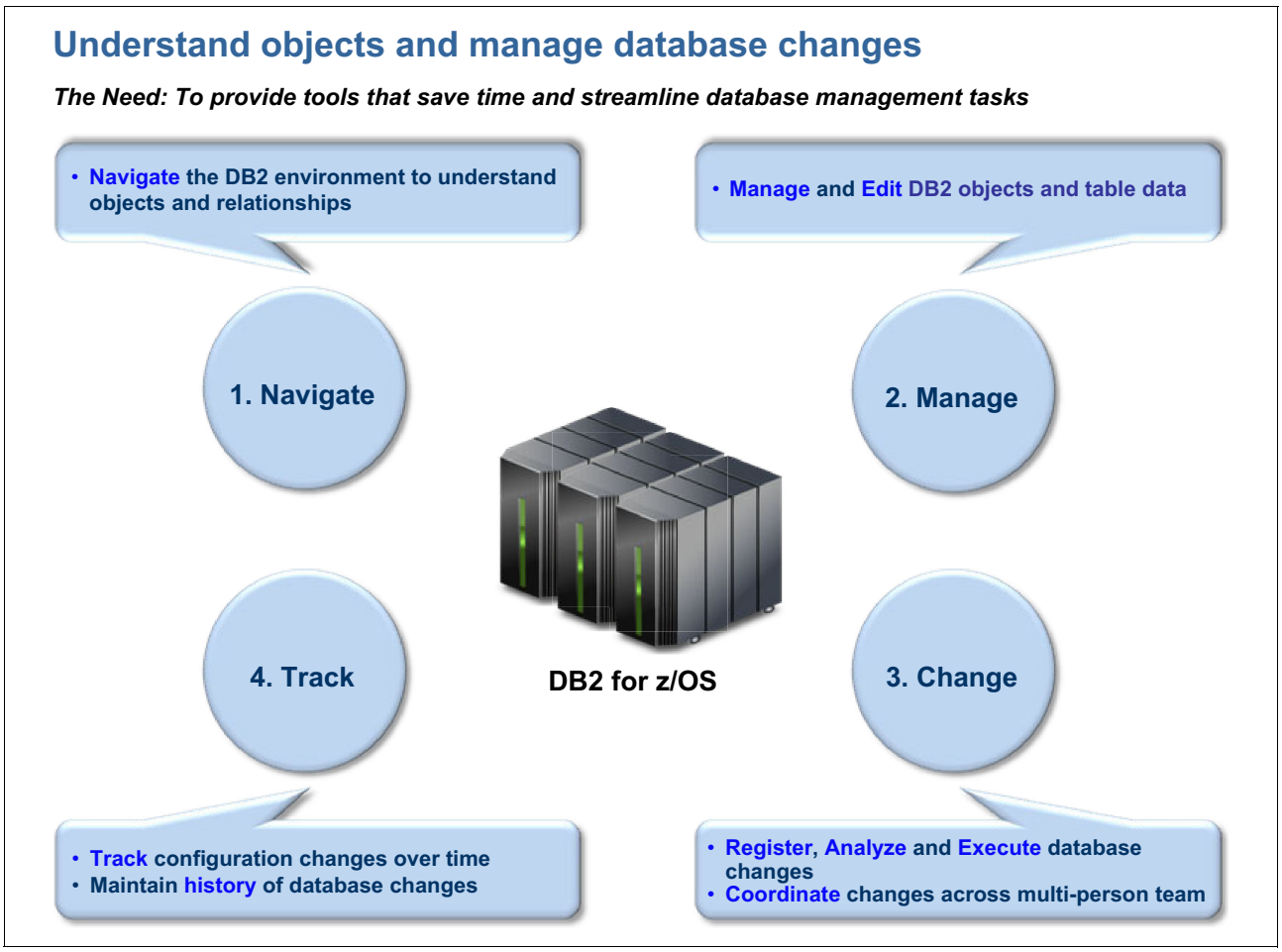

Figure 1-1 DB2 Administration Solution Pack

IBM DB2 Administration Solution Pack for z/OS is a product that combines a number of IBM components into a consolidated solution to help you manage your DB2 for z/OS environment. It includes the following components:

- DB2 Administration Tool for z/OS
- DB2 Object Comparison Tool for z/OS
- DB2 Table Editor
- IBM InfoSphere® IBM Optim™ Configuration Manager

- IBM DB2 Utilities Solution Pack for z/OS V1.1(program number: 5697-DUM), see Figure 1-2.

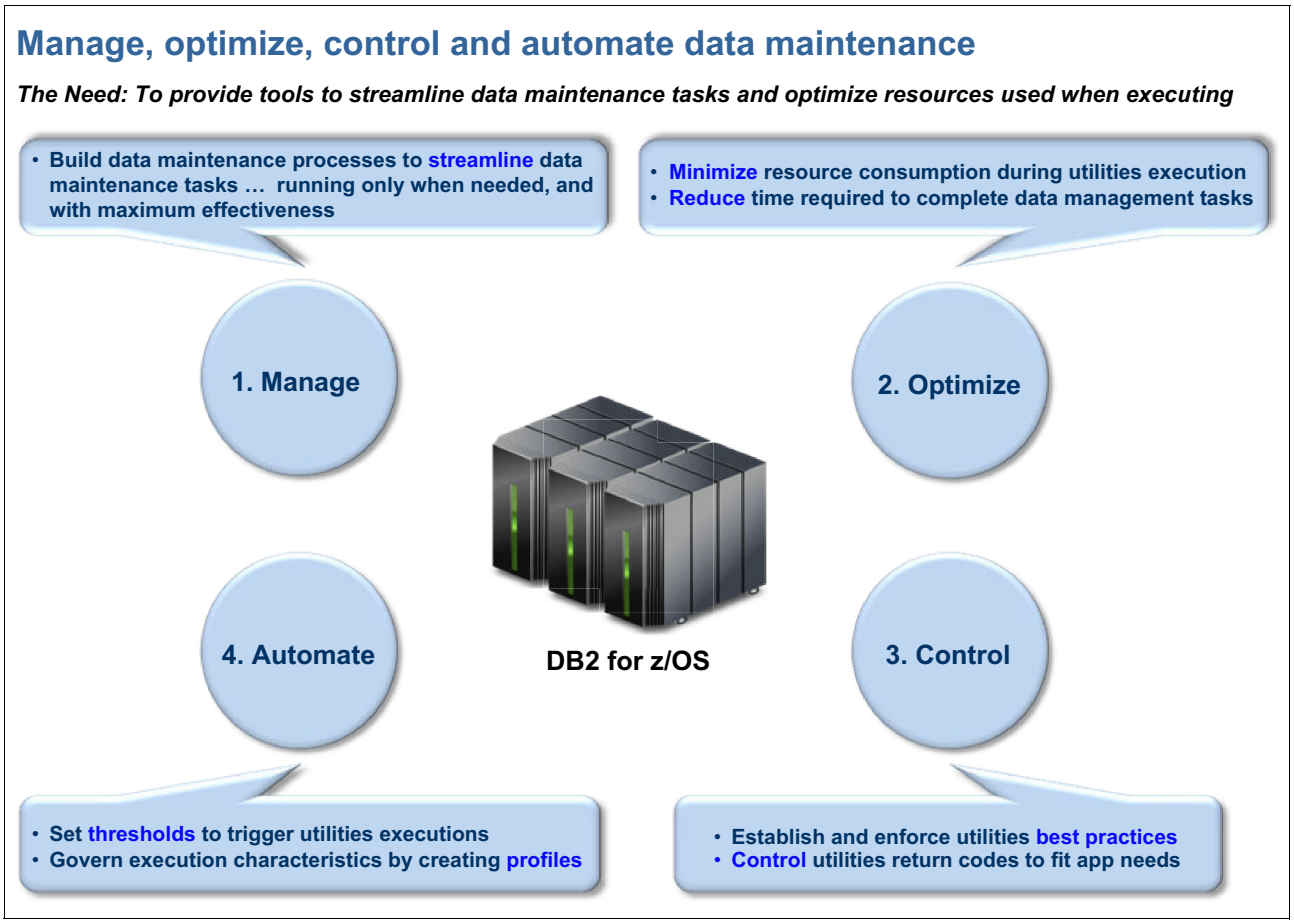

Figure 1-2 DB2 Utilities Solution Pack

IBM DB2 Utilities Solution Pack for z/OS is a product that combines a number of IBM components into a consolidated solution that enables enterprises to efficiently and intelligently run utilities, while optimizing the performance of daily utilities management activities. It combines the following components into a single offering:

- IBM DB2 Automation Tool for z/OS Version 4.1
- IBM DB2 High Performance Unload for z/OS Version 4.2
- IBM DB2 Sort for z/OS Version 1.3
- IBM DB2 Utilities Enhancement Tool for z/OS Version 2.2

- IBM DB2 Fast Copy Solution Pack for z/OS (program number: 5697-DFM) extends the DB2 Utilities Solution Pack by integrating the technology available with IBM DS8000® IBM FlashCopy®.

  Typically, FlashCopy tools are disk storage-related and do not directly address the needs of the database administrator. IBM DB2 Fast Copy Solution Pack for z/OS, Version 1 Release 1 combines several IBM components into a consolidated solution that leverages fast replication technology to optimize availability, performance, and resource utilization when you back up, recover, and clone DB2 subsystems or DB2 objects. This solution contains the following components:

  - IBM DB2 Cloning Tool for z/OS Version 3.1
  - IBM DB2 Recovery Expert for z/OS Version 3.1

- IBM DB2 Performance Solution Pack for z/OS V1.1 (program number: 5655-E74), see Figure 1-3.

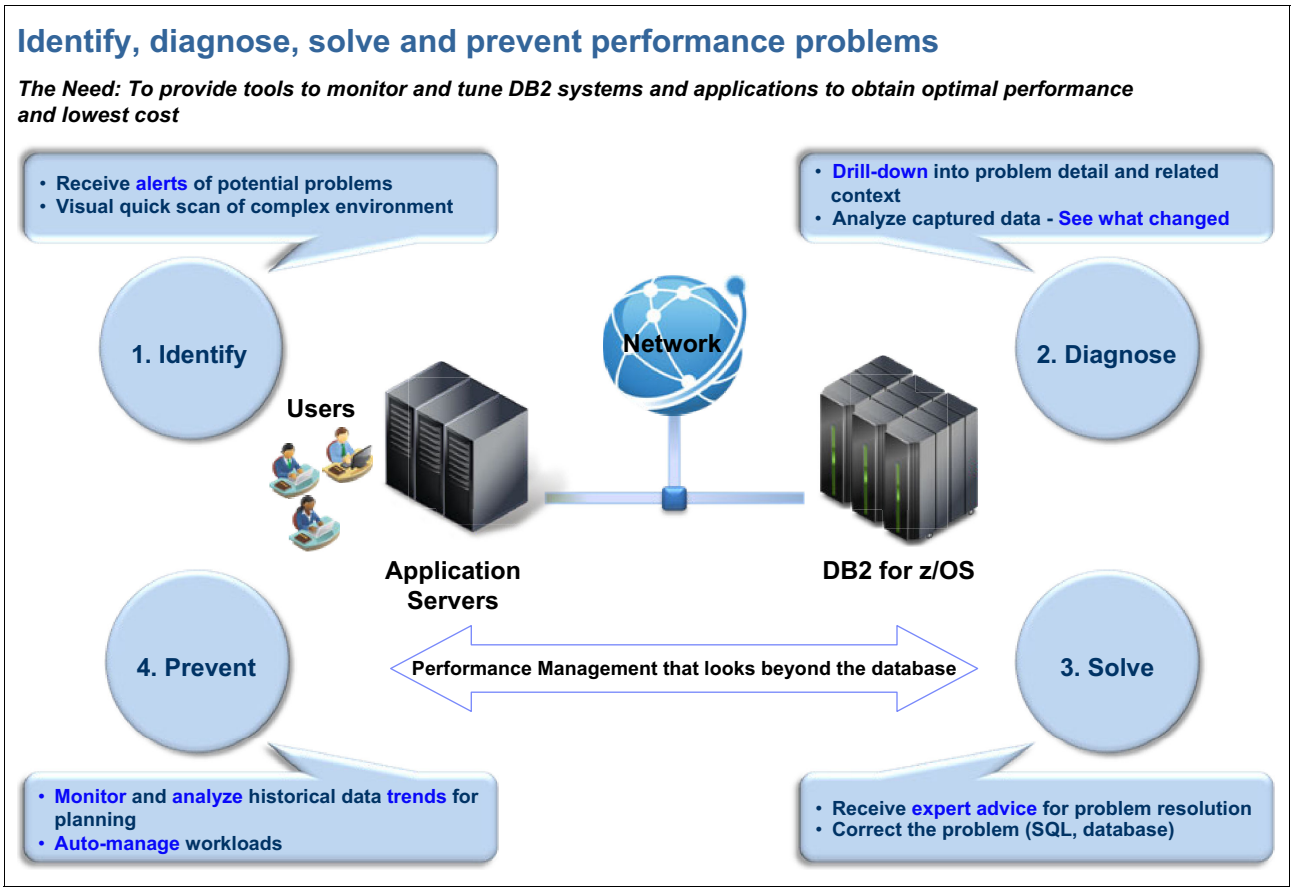

Figure 1-3 DB2 Performance Solution Pack

IBM DB2 Performance Solution Pack for z/OS, V1.1 delivers integrated performance management capabilities for DB2 for z/OS by combining the following functionality:

- IBM Tivoli® OMEGAMON® XE for IBM DB2 Performance Expert on z/OS, V5.1.1
- IBM DB2 Query Monitor for z/OS, V3.1
- IBM DB2 SQL Performance Analyzer for z/OS, V4.1
- IBM InfoSphere Optim Query Workload Tuner for DB2 for z/OS, V3.2

In this publication, we focus on the usage of the DB2 Utilities Solution Pack and the DB2 Fast Copy Solution Pack and explore scenarios for the execution of DB2 utilities.

1.2 DB2 Utilities and Fast Copy solution packs

Both solution packs require the IBM Tools Customizer for z/OS (Tools Customizer), program number 5655-V93, to be installed. Tools Customizer is a component of IBM Tools Base for z/OS, a no-charge product (program number 5655-V93) that is a prerequisite for the DB2 and IMS solution packs.

The solution packs require the same high-level qualifier (HLQ) for all components contained within the pack. So, for example, all components related to a Utilities Solution Pack must begin with BBY if using the default HLQ.

In our installation, the HLQ of DBTLSP was used for all solution packs with the MLQ identifying the component. For example, DBTLSP.SARYDENU is the Recovery Expert Metadata library within the FC Utilities Solution Pack.

Before attempting to install and customize these solution pack components, plan ahead to gain some component knowledge before you start your installation. Each component has to be customized before you can use the software. The customization assumes that you are familiar with the component you are installing and understand the implications of the associated parameters. To save time, read the relevant user guide and customization manual for each product; plan your installation for your site and how the component should be customized to meet your requirements.

Important: The components install out of the box and start using the default values. This "*quick*" install might not fit your local requirements, so "*read the manual*" and understand the defaults before installation.

Walk through a customization exercise. Map out how your selections of parameter values will work within your environment. Check out interfaces or associations with other products:

- Do I need an interface with the Automation Tool?
- Have I got the DB2 Administration Scheduler customized and working?
- What are the space requirements?
- What are the storage management subsystem (SMS) concerns and definitions?

  This particularly applies to the DB2 Recovery Expert and Cloning components of the Fast Copy Solution Pack because the customization requirements are considerable.

All the DB2 Tools documentation is available at this website:

`http://www.ibm.com/support/docview.wss?uid=swg27020910`

1.2.1 Overview of the solution packs and the IBM Tools Customizer

Tools Customizer provides a standard installation solution for installing the DB2 Tool products.

The IBM Tools Customizer (TCz in Figure 1-4 on page 8) assists in the post-SMP/E for z/OS customization of z/OS tool products. The goal is to provide a more consistent, usable, and simple solution for the customization of multiple z/OS products.

It is a common customization utility for all DB2 and IMS Tool products that provides these benefits:

- Consumability: Faster time to tools usage

  Due to automatic discovery of previous release customization parameters, there is less manual entry. Product templates are customized by Tools Customizer. It provides the job execution sequence.

- Easy customization of multiple tools

  Step-by-step, with HELP text, Interactive System Productivity Facility (ISPF) panel-driven dialog with a similar look and feel allows the specification of multiple product customization.

- Faster propagation of customization across multiple logical partitions (LPARs)

  Resulting product customized jobs can be saved on shared DASD or user can transmit to other systems for usage on other LPARs.

- Easier upgrades

  Parameters from the previous customization are saved for future new product releases and DB2 or IMS upgrades.

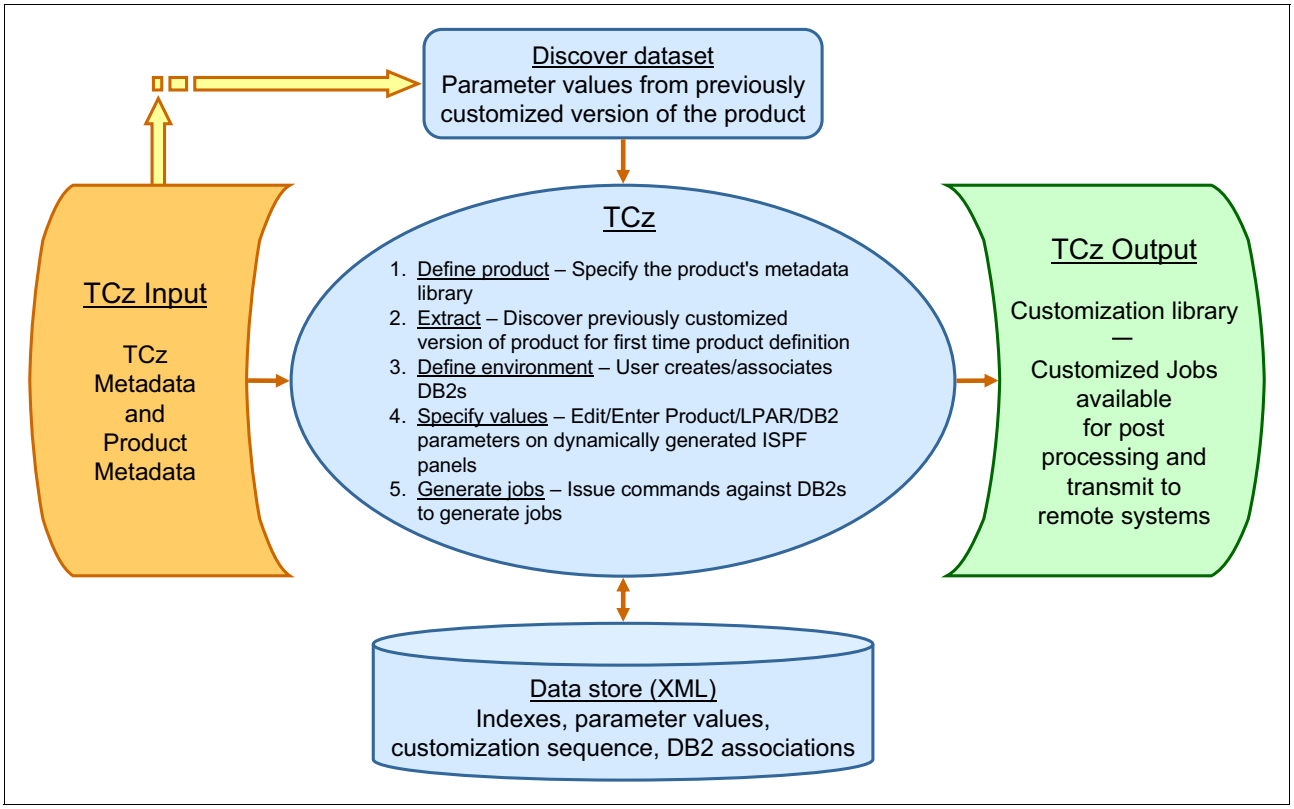

Figure 1-4 Tools Customizer framework

Tools Customizer offers these main features:

- A new standard for post-SMP/E customization of DB2 Tools and IMS Tools products
- A more consistent, usable, and simple solution for the customization of multiple products
- An ISPF user interface that guides the user through the customization process, assisting the user in the following main tasks:
  - Initial customization and re-customization of products
  - Discovery of the previous customization of a product
  - Guidance through the steps of customization with dynamically generated ISPF product panels
  - Entry of parameters' values required by the products that are being customized
  - Reuse and sharing of configuration settings across products via a common XML repository
  - Customization of product templates
  - Display of the job execution sequence

Before the solution pack packaging, the tools came as individual products, and they can still be ordered individually. The Tools Customizer installation requires each solution pack or product to provide a metadata file. This metadata file must conform to Tools Customizer requirements. It instructs Tools Customizer how to perform the installation, what parameters

are required, and what jobs must be generated to complete the installation. The approach to installing any product with Tools Customizer is to acquire the metadata file, which is the file that has a low-level qualifier (LLQ) starting SDENU, from the initial SMP/E installation and load it into Tools Customizer. Tools Customizer then adds that product or components to the Tools Customizer installation list. See Figure 1-5.

```
TCUSTMZR              Select the Components to Customize      Row 1 to 4 of 4
Command ===>                                                 Scroll ===> CSR

Select one or more components to customize. Press Enter to continue
or End to cancel.

Pack metadata library . : 'DBTLSP.SBBYDENU'
Pack to customize . . . : DB2 Utilities Solution Pack    > Version . : 1.1.0

Line commands: / - Select
  Cmd Name                                    Version Customization Status
  -   ------------------------------------> ------- --------------------
      DB2 Automation Tool for z/OS            4.1.0   Customized
      DB2 High Performance Unload for z/OS    4.2.0   Customized
      DB2 Sort for z/OS                       1.3.0   Customized
      DB2 Utilities Enhancement Tool for z/OS 2.2.0   Pending Customization
---------------------------- End of components ----------------------------
```

Figure 1-5 Components of the Utilities Solution Pack after loading the metadata file

From this point, the installation process is the same for each component or product. It follows the following list of tasks:

1. Select the component to customize.
2. Associate DB2 entries to the component.
3. Customize the component parameters.
4. Customize the LPAR parameters.
5. Customize the DB2 entry parameters.
6. Generate the job streams.
7. Submit the jobs.

Once these tasks have been performed, the component is ready for use. If you make an error, you can always change a parameter and regenerate the job stream. Some jobs are foolproof in that they can be rerun without change; others may require that you delete a file or database before rerunning.

An approach to running the job streams is to review all the jobs first before running the first one. Understand what each job does and determine any dependencies on previous jobs; or associations with other products. Does it refer to DB2 Automation, for example? Or, does it use other services, such as DB2 Administration Scheduler? Only when you are sure that you thoroughly understand the implications of running the job stream should you begin your installation.

The components of Utilities Solution Pack are listed in Table 1-1.

Table 1-1 5697-DUM Utilities Solution Pack

FMID	Component	HLQ	Notes
HBBY110	Utilities Solution Pack V1.1	DBTLSP	Default HLQ BBY
H25HKN0/H25H410	Automation V4.1	DBTLSP	Product MLQ HAA
HCNK130	DB2 Sort V1.3	DBTLSP	Product MLQ CNK
HINZ420	High Performance Unload V4.2	DBTLSP	Product MLQ INZ
H2AM220	Utilities Enhancement Tool 2.2	DBTLSP	Product MLQ ABP

The components of Fast Copy Utilities Solution Pack V1.1 are listed in Table 1-2.

Table 1-2 5697-DFM Fast Copy Utilities Solution Pack V1.1

FMID	Components	HLQ	Notes
HBBX110	Fast Copy Solution Pack V1.1	DBTLSP	Default HLQ BBX
H30RKN0/H30R310	Recovery Expert V3.1	DBTLSP	Product MLQ ARY
HCKZ310	Cloning Tool V3.1	DBTLSP	Product MLQ CKZ

To install solution packs, you also need the IBM Tools Customizer, product 5655-V93, which consists of the components listed in Table 1-3.

Table 1-3 5655-V93 IBM Tools Customizer product

FMID	Component	HLQ	Notes
HAHN130	IBM Tool Base for z/OS	DBTLSP	Default HLQ GLX
HTCZ110	IBM Tool Customizer	DBTLSP	Default HLQ CCQ

1.2.2 Installing the base for solution packs

Both solution packs are installed via the IBM Tool Customizer product. A local CLIST was used to start the Tools Customizer primary options panel. At our installation, we created a CLIST called DB2TOOLS (Figure 1-6), which was invoked from the TSO command line:

```
TSO DB2TOOLS CLIST
```

```
/* REXX */
ADDRESS ISPEXEC "SELECT PANEL(DB2TOOLS)";
EXIT
```

Figure 1-6 DB2TOOLS initial CLIST

This CLIST invoked the panel member in Figure 1-7 on page 11, which displayed our local DB2TOOLS panel. From here, we invoked the Tools Customizer startup CLIST from the installed CCQ.

```
-------------------- DB2 Tools Primary Options Menu -------------------
OPTION ===>
                                                                USERID - ADMR2
Select an option from below.                                    DATE   -13/01/31
                                                                TIME   - 09:22
     A Admin.       - Administration Tool
     T Automation   - Automation Tool
     C Cloning      - Cloning Tool
     I HPU          - High Performance Unload
     R RE           - Recovery Expert
     U UET          - Utilities Enhancement Tool

     L LP           - DB2 Launchpad
     Z TCZ          - Toolkit Customization

     X EXIT         - Exit
```

Figure 1-7 DB2Tools initial panel

The Tools Customizer option calls the default Tools Customizer initial CLIST (CCQTCZ) from the library CCQ.SCCQEXEC. This displays the Tools Customizer primary menu. See Figure 1-8. We set up our default options as per the instructions in *IBM DB2 Utilities Solution Pack for z/OS Version 1 Release 1 Overview and Customization*, SC19-3783.

```
TCUSTMZR                IBM Tools Customizer for z/OS               09:29:48
Option ===>

Select an option.



0. User settings for Tools Customizer
1. Customize a product

X. Exit
```

Figure 1-8 Tools Customizer initial CLIST

We selected Option 0 and set up our user settings as shown in Figure 1-9 on page 12.

```
TCUSTMZR                  Tools Customizer Settings                  09:34:23
Command ===>

Enter the settings for customizing a product or press End to save and exit.

Commands: SAVE - Save user settings

 Product Customization Settings

   Customization library qualifier . . DB2TOOLS.JCL
   Use DB2 group attach  . . . . . . . NO  (YES/NO)

 Tools Customizer Library Settings

   Metadata library . . . . . DBTLSP.SCCQDENU
   Discover output data set . DB2TOOLS.DISCOVER
   Data store data set  . . . DB2TOOLS.DATASTOR

 User Job Card Settings for Customization Jobs

 ===> //         JOB (999,POK),'RE',
 ===> //         REGION=0M,NOTIFY=&SYSUID,
 ===> //         MSGCLASS=X,CLASS=T
 ===> //PROCLIB JCLLIB ORDER=DBOAM.PROCLIB
 ===> /*JOBPARM SYSAFF=SC63
```

Figure 1-9 Tools Customizer user settings

Table 1-4 provides explanations of the field settings.

Table 1-4 User settings for Tools Customizer

Setting	Description	Sample value
Customization Library Qualifier	A high-level prefix that is used for the customization data set for each component.	DB2TOOLS.JCL.ssid.CKZ310
Metadata library	Tools Customizer metadata library for DB2 and LPAR parameters.	We use the IBM supplied library: DBTLSP.SCCQDENU
Discover output data set	Used by the DISCOVER EXEC for the discovery of information from previous installations.	We are not using the DISCOVER EXEC so although this field is required, it does not apply to our installation.
Data store data set	Tools Customizer data stores information and LPAR, DB2, and parameter values.	We specified: DB2TOOLS.DATASTOR

After we save these options, we continue with the installation of the solution packs.

1.3 Installing the DB2 Utilities Solution Pack

The DB2 Utilities Solution Pack combines powerful DB2 tools that, when used together, can improve utility performance and simplify utility management. They include the following functionally compatible products:

- Automation Tool

  Used to set up recurring utility jobs for both conditional and routine DB2 maintenance. It automates common tasks and generates JCL for more complex tasks with analytical capabilities.

- High Performance Unload

  Offers performance options for unloading and extracting large amounts of data across an enterprise. It works outside of DB2 so it does not compete for DB2 resources.

- DB2 Sort

  Provides optimized sort product for sorting large amounts of data. Using zIIP technology, it optimizes the memory usage and sort work disk space. It reduces sort overhead on the general CPU processor.

- Utilities Enhancement Tool

  Provides additional “features” for DB2 LOAD, REORG, and CHECK DATA utilities. It offers “Thread” block and cancel functions.

By using the DB2 Utilities Solution Pack, you can simplify the management and running your utilities while at the same time improving the utility performance.

Every component installed through Tools Customizer requires a metadata library. This is normally shipped as a DENU-type library. For this solution pack, the DENU library is called DBTLSP.SBBYDENU. To begin the installation process, we specify option 1 on the Tools Customizer primary menu and enter the name of the metadata library. We are presented with the panel in Figure 1-10, which shows the components that need to be customized to install this solutions pack.

```
TCUSTMZR              Select the Components to Customize      Row 1 to 4 of 4
Command ===>                                                 Scroll ===> CSR

Select one or more components to customize. Press Enter to continue
or End to cancel.

Pack metadata library . : 'DBTLSP.SBBYDENU'
Pack to customize . . . : DB2 Utilities Solution Pack    > Version . : 1.1.0

Line commands: / - Select
  Cmd Name                                  Version Customization Status
  -   ----------------------------------> ------- --------------------
      DB2 Automation Tool for z/OS          4.1.0   Not Found
      DB2 High Performance Unload for z/OS  4.2.0   Not Found
      DB2 Sort for z/OS                     1.3.0   Not Found
      DB2 Utilities Enhancement Tool for z/OS 2.2.0   Not Found
---------------------------- End of components ----------------------------
```

Figure 1-10 Tools Customizer Customization panel for the DB2 Utilities Solution Pack

You can select one or multiple tools to customize. If you select multiple tools, the tools are customized in the order of the selection list. Selecting DB2 Automation Tool, we are presented with the following panel. This panel provides information about the DISCOVER EXEC, which can be invoked to “discover” customized detail from a previous installation of the Automation Tool component (if it was installed with Tools Customizer and the metadata was written to the previous STORDATA and DISCOVER files). In our case, we are customizing for the first time so the DISCOVER EXEC is not relevant. We press F3 (END) to continue. See Figure 1-11.

```
TCUSTMZR                    Customizer Workplace
Command ===>                                                Scroll ===> CSR

Use the Generate jobs line command to select the DB2 entries on which to
customize the component, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST
Pack to Customize .: DB2 Utilities Solution Pack        >  Version . : 1.1.0
  Component metadata library : 'DBTLSP.SHAADENU          > LPAR  . . : SC63
  Component name . . . . . . : DB2 Automation Tool for   > Version . : 4.1.0

EssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssN
e TCUSTMZR                          Run Discover EXEC                        e
e                                                                            e
e If DB2 Automation Tool for z/OS                                            e
e has been customized before, you can run the Discover EXEC to retrieve      e
e information from that customized version. Otherwise, if you are either     e
e customizing this component for the first time or you have not customized   e
e the component yet, continue the customization process because information  e
e to be discovered does not exist.                                           e
e                                                                            e
e To go to the Discover Customized Component Information panel and run the   e
e Discover EXEC, press Enter. To return to the Customizer Workplace panel    e
e without running the Discover EXEC, press End.                              e
e                                                                            e
e  F1=HELP     F2=SPLIT    F3=END      F4=RETURN   F5=RFIND    F6=RCHANGE    e
e  F7=UP       F8=DOWN     F9=SWAP     F10=LEFT    F11=RIGHT                 e
DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 1-11 Initial customization panel with the DISCOVER EXEC option

Installation of a solution pack begins with associating the DB2 subsystem to the components within the packs. You can create the necessary DB2 entries through the ASSOCIATE command on the Customizer Workplace panel. See Figure 1-12 on page 15.

```
TCUSTMZR                    Customizer Workplace
Command ===>                                                Scroll ===> CSR

Use the Generate jobs line command to select the DB2 entries on which to
customize the component, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST
Pack to Customize .: DB2 Utilities Solution Pack        >  Version . : 1.1.0
  Component metadata library : 'DBTLSP.SHAADENU          > LPAR  . . : SC63
  Component name . . . . . . : DB2 Automation Tool for   > Version . : 4.1.0

Component and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
    Component parameters.: Incomplete
    LPAR parameters . . .: Ready to Customize

Associated DB2 Entries and Parameter Status
  Line commands: G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID GrpAttch Lvl Mode User ID  Date          Status
    ------------------------- End of DB2 entries-----------------------------
```

Figure 1-12 Create the DB2 entries by using the ASSOCIATE command

In our panel, you can see that three entries have already been created by using the CREATE command. These are the DB2 subsystems in our environment. We associate the entries for DB0A and DB0C to solution pack components by selecting those entries with “A” for the Associate line command. See Figure 1-13.

```
TCUSTMZR            Associate DB2 Entry for Component         Row 1 to 3 of 3
Command ===>                                                 Scroll ===> CSR

Select any of the following DB2 entries to add them to the Customizer
Workplace panel. You use the Customizer Workplace panel to choose the DB2
subsystems, data sharing members, and group attach names on which to
customize the component.

Commands:  CREATE - Create a new DB2 entry


Pack to Customize .: DB2 Utilities Solution Pack        >  Version . : 1.1.0
  Component metadata library : 'DBTLSP.SHAADENU          > LPAR  . . : SC63
  Component name . . . . . . : DB2 Automation Tool for   > Version . : 4.1.0

Line commands: A - Associate  C - Copy

Cmd SSID GrpAttch
 A  DB0A   --
 A  DB0C   --
    DB0D   --
------------------------------ End of DB2 entries -----------------------------
```

Figure 1-13 Creating and associating DB2 entries to a solution pack

Once associated, customization of a component can begin. The parameter status field provides information about two areas of customization: Component and LPAR. These parameters must be provided (incomplete) or checked and edited before continuing with the component customization. See Figure 1-14.

```
Command ===>                                                  Scroll ===> CSR

Use the Generate jobs line command to select the DB2 entries on which to
customize the component, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST
Pack to Customize .: DB2 Utilities Solution Pack        >  Version . : 1.1.0
  Component metadata library : 'DBTLSP.SHAADENU          > LPAR  . . : SC63
  Component name . . . . . . : DB2 Automation Tool for   > Version . : 4.1.0

Component and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
  E Component parameters.: Incomplete
    LPAR parameters . . .: Ready to Customize

Associated DB2 Entries and Parameter Status
  Line commands: G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID GrpAttch Lvl Mode User ID  Date          Status
      DBOA  --      101 NFM  ADMR2    2012/09/24    Ready to Customize
      DBOC  --      101 NFM  ADMR3    2012/05/02    Ready to Customize
    ------------------------- End of DB2 entries -----------------------------
```

Figure 1-14 Component and LPAR status

Complete the component customization by entering the “E” (for Edit) line command. This takes you to a panel with the related component parameters, which must be customized to your installation. For details about the parameters, consult the related component installation and customization guide. This information can be found at the following link:

http://www.ibm.com/support/docview.wss?uid=swg27020910#db2lat-lib

As an example, Figure 1-15 on page 17 shows the first page from the DB2 Automation Tool Component Parameter panel. When the LPAR and Component parameters have been updated and saved, the component is ready to have the individual DB2 entries customized. Select an E against each subsystem. The DB2 Parameters panel is displayed. Ensure that you have added SDSNLOAD to the RUN execution libraries if this library is not in your LINKLIST. This parameter entry is in the Component Parameter panel.

```
TCUSTMZR                          DB2 Parameters                    Top of data
Command ===>                                                     Scroll ===> CSR

Enter values for all of the DB2 parameters. Press End to save and exit.

Commands: SAVE - Save parameter values

 Pack to Customize .: DB2 Utilities Solution Pack     >   Version . : 1.1.0
   Component metadata library : 'DBTLSP.SHAADENU       >  LPAR  . . : SC63
   Component name . . . . . . : DB2 Automation Tool fo >  Version . : 4.1.0

                                                                  More:     +

 DB2 subsystem ID  . . . . . . . . . . . . : DBOA
 Group attach name . . . . . . . . . . . . :

 General DB2 Information
  Mode . . . . . . . . . . . . . . . . . . . NFM (CM, CM8, CM9, NFM)
  Level Number . . . . . . . . . . . . . . . 101 (810, 910, 101)

 DB2 Libraries
  Run Library  . . . . . . . . . . . . . . . DBOAM.RUNLIB.LOAD   >     Add...
  Bootstrap data set . . . . . . . . . . . . DBOAB.BSDS01        >     More...

 DB2 Bufferpools
  Name of the 4 KB bufferpool  . . . . . . . BPO
  Name of the 16 KB bufferpool . . . . . . . BP16KO
  Name of the 32 KB bufferpool . . . . . . . BP32K
```

Figure 1-15 DB2 Parameters panel for individual DB2 entries

After the individual DB2 entries have been customized, the GENERATEALL command can be issued to GENERATE the JCL to customize the product. If you want to generate customization jobs for specific DB2 entries, specify the “G” line command against the DB2 entry. See Figure 1-16 on page 18.

```
TCUSTMZR                Finish Component Customization       Row 1 to 13 of 19
Command ===>                                                  Scroll ===> CSR

Submit the members in the order in which they apply to all DB2 entries. To
submit the job, browse the member and issue the TSO SUBMIT command, or browse
the customized library and submit the jobs from there.

Pack to Customize .: DB2 Utilities Solution Pack        >  Version . : 1.1.0
  Component metadata library : 'DBTLSP.SHAADENU          > LPAR  . . : SC63
  Component name . . . . . . : DB2 Automation Tool for   > Version . : 4.1.0

Line Commands: E - Edit  B - Browse

    Component customization library: DB2TOOLS.JCL.£SC63£.HAA410

 Cmd Member   SSID GrpAttch Template Date       Description
 -   -------- ---- ----     -------- ---------- ------------------------------>
     AOV41     --   --      HAAV41   2013/02/01 Configure Startup CLIST 1
     A1V41C    --   --      HAAV41C  2013/02/01 Configure Startup CLIST 2
     A2EXECS   --   --      HAAEXECS 2013/02/01 HAA Required Execs
     A3#DDLAA DBOA  --      HAA#DDL  2013/02/01 Create DB2 Objects
     A3#DDLAB DBOC  --      HAA#DDL  2013/02/01 Create DB2 Objects
     B3#B10AA DBOA  --      HAA#B10N 2013/02/01 V10 NFM Binds
     B3#B10AB DBOC  --      HAA#B10N 2013/02/01 V10 NFM Binds
     COGRANAA DBOA  --      HAAGRANT 2013/02/01 Grant Execute Authority
     COGRANAB DBOC  --      HAAGRANT 2013/02/01 Grant Execute Authority
     C2CNTFL   --   --      HAACNTFL 2013/02/01 Create Control File
     C3PROC    --   --      HAAPROC  2013/02/01 Job Tracking Started Task Proc
     C4PARMS   --   --      HAAPARMS 2013/02/01 Job Tracking Started Task Parms
     C5CF1UP   --   --      HAACF1UP 2013/02/01 Update Control File 1
```

Figure 1-16 Output from the GENERATEALL command

The jobs can now be submitted to customize the component. Select the Edit line, correct the JCL if necessary, and submit the job. From the System Display and Search Facility (SDSF), save the job output to an output library for future reference. Use the “XDC” line command in SDSF to set up a partitioned data set (PDS) member to contain the job output. See Figure 1-17 on page 19.

```
TCUSTMZR                 Component Parameters                        06:47:40
Command ===>                                                  Scroll ===> CSR

Complete the following tasks to customize the components. The required tasks
and steps are preselected. Ensure that all parameters are specified for
selected step within a task. Press End to save and exit.

Commands: SAVE - Save parameter values
Line Commands: / - Select

 Pack to Customize .: DB2 Utilities Solution Pack     >   Version . : 1.1.0
   Component metadata library : 'DBTLSP.SHAADENU       >  LPAR  . . : SC63
   Component name . . . . . . : DB2 Automation Tool fo >  Version . : 4.1.0

Component customization library: DB2TOOLS.JCL.£SC63£.HAA410
                                                                More:     +

IMPORTANT:
 - If you have edited the DB2 parameters, some of the parameters on this
   panel might be moved from their original sections to the Required
   parameters section.  To avoid this situation, create and edit DB2 entries
   first.

 Required parameters
  Startup CLIST library  . . . . . . . . . . DBTLSP.SHAASAMP              >
  HAA production load library  . . . . . . . DBTLSP.SHAALOAD              >
  HAA ISPF panel library . . . . . . . . . . DBTLSP.SHAAPENU              >
  User ISPF skeleton library . . . . . . . .                              >
  HAA ISPF skeleton library  . . . . . . . . DBTLSP.SHAASLIB              >
  FEC common code load library . . . . . . . DBTLSP.SFECLOAD              >
```

Figure 1-17 Example of the customization job list

Check each job carefully before submitting.

If you need to change the JCL, do not: First, return to the component parameters to try to locate the parameter that relates to the change that you want to make. Correct that parameter and GENERATE the jobs again. This saves you from having to cancel the installation later when you find that subsequent jobs fail because of a related parameter that you have modified manually in the generated JCL.

1.3.1 DB2 Automation Component customization

The following information provides details about how we installed and customized the DB2 automation component. An existing version already existed on our platform at the 4.1 level. Because we wanted to test all the functions of the solution pack, we chose to perform a full installation and create a repository copy from the original database repository. That meant we had to select some additional options when performing our installation:

- Change the database to HAADB. The default and recommended database is DLCDB.
- Change the HAA configuration value to HAA. The default is DLC.
- Change the Qualifier for HAA object names to HAA. The default is DLC.

- Selected the option in the components parameters to upgrade the HAA repository, HAA V4.1 Maintenance Upgrade PM70641. This creates a job in the job stream to upgrade the original DLC database and adds a new column to table DLC.OBJECTS_V13.
- Run the copy repository job to copy the tables from the DLC schema to the new HAA scheme.

Using this method meant that we saved all of our original profile data from a previous installation of DB2 Automation.

Ensure that you specify a member name on the job tracking PROC and PARM location.

After we completed the customization, we were able to start the DB2 Automation product from the DB2 Launchpad. See Figure 1-18.

```
 ------------------------- DB2 Tools Launchpad ----------------- Row 1 from 10
Command ===>                                                   Scroll ===> CSR


Specify DB2 SSID (opt) ===>  DBOA  (Enter '?' for a list of active SSIDs)

Select the DB2 tool you wish to launch or enter its code in the command line.

Sel Code  Tool Name                                        Rel  Prog No.
    ---   ------------ ADMINISTRATION TOOLS -------------  ---  --------
    ADM   DB2 Administration Tool                          101  1097-L90
    HAA   DB2 Automation Tool                              410  5655-T59
    OBJ   DB2 Object Comparison Tool for z/OS             100  5655-W36
    ---   --------- APPLICATION MANAGEMENT TOOLS --------  ---  --------
    HPU   HPU High Performance Unload                      410  5655-AA1
    ---   --------- PERFORMANCE MANAGEMENT TOOLS --------  ---  --------
          No table entries in this category
    ---   -- RECOVERY AND REPLICATION MANAGEMENT TOOLS --  ---  --------
          No table entries in this category
****************************** Bottom of data ********************************
```

Figure 1-18 DB2 Launchpad with the DB2 Automation component

From the DB2 Tools Launchpad, we can access the DB2 Automation Primary Option by selecting the HAA entry. Figure 1-19 on page 21 is displayed.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2013/02/01  09:40:04
Option  ===>
--------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Disaster Recovery
                3 - Exception Profiles      11 - Stand Alone Utilities
                4 - Job Profiles            12 - DB2 Admin Scheduler
                5 - Quick Build             13 - DB2 Autonomic Statistics
                6 - Execution Reports        X - Exit
                7 - DB2 Command Processor

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DBOA         (1-4 Character Subsystem ID or ? for list)
 Current SQLID:    ADMR2        User: ADMR2   - HAA
--------------------------------------------------------------------------------









sssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssN
 IBM* Rocket**                                                           e
 Licensed Materials - Property of IBM. 5655-E37                          e
 Copyright IBM Corporation 2000, 2011  All Rights Reserved.              e
 Copyright Rocket Software, Inc. 2000, 2011  All Rights Reserved.        e
 *Trademark of International Business Machines                           e
 **Trademark of Rocket Software, Inc.                                    e
```

Figure 1-19 DB2 Automation primary options panel

1.3.2 DB2 High Performance Unload for z/OS

We continue now with the DB2 High Performance Unload component of the solution pack. The process is the same as for DB2 Automation. We ASSOCIATE the DB2 subsystems to the component, ensure that the LPAR and component parameters are correct, generate the relevant job stream, and submit the jobs. See Figure 1-20 on page 22.

For more information about specifying values for parameters, see the *IBM DB2 High Performance Unload for z/OS Version 4 Release 2 User's Guide*, SC19-3777.

```
TCUSTMZR                Finish Component Customization       Row 1 to 11 of 11
Command ===>                                                  Scroll ===> CSR

Submit the members in the order in which they apply to all DB2 entries. To
submit the job, browse the member and issue the TSO SUBMIT command, or browse
the customized library and submit the jobs from there.

Pack to Customize .: DB2 Utilities Solution Pack        >  Version . : 1.1.0
  Component metadata library : 'DBTLSP.SINZDENU          > LPAR  . . : SC63
  Component name . . . . . . : DB2 High Performance Unl  > Version . : 4.2.0

Line Commands: E - Edit  B - Browse

    Component customization library: DB2TOOLS.JCL.£SC63£.INZ420

 Cmd Member   SSID GrpAttch Template Date       Description
 -   -------- ---- ----     -------- ---------- ------------------------------>
     A0TVAR    --   --      INZTVAR  2013/02/01 Generates the INZUTIL member co
     A1PARM    --   --      INZPARM  2013/02/01 Defines the dsname of the DB2 H
     A2HPUCL   --   --      INZHPUCL 2013/02/01 Generates the members required
     A3LAUNC   --   --      INZLAUNC 2013/02/01 Generates the CLIST to be run t
     A4ADT00   --   --      INZADT00 2013/02/01 Generates a member with instruc
     A5BINDAA DB0A  --      INZBIND  2013/02/01 Binds the plans and packages th
     A5BINDAB DB0C  --      INZBIND  2013/02/01 Binds the plans and packages th
     A6EXEUAA DB0A  --      INZEXEUE 2013/02/01 Generates a sample job for runn
     A6EXEUAB DB0C  --      INZEXEUE 2013/02/01 Generates a sample job for runn
     A8CHECAA DB0A  --      INZCHECK 2013/02/01 Generates members with the DB2-
     A8CHECAB DB0C  --      INZCHECK 2013/02/01 Generates members with the DB2-
--------------------------- End of customized jobs ----------------------------
```

Figure 1-20 Job list for High Performance Unload

Before running the jobs, set up a HLQ.SINZPARM and ensure that the component parameters reflect this. This library is not allocated during the SMP/E install but it is required to run the job list. Do not set HLQ.SINZSAMP.

After running through the job stream, verify that the DB2 High Performance Unload component is working. There is a sample batch job, INZCHECK, which runs a batch DB2 High Performance Unload. See Figure 1-21 on page 23.

```
 Display  Filter  View  Print  Options  Search  Help
 -------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY INZCHEAA JOB15421  DSID     2 LINE 0       COLUMNS 02- 81
 COMMAND INPUT ===>                                            SCROLL ===> CSR
******************************** TOP OF DATA *********************************
                  J E S 2  J O B  L O G  --  S Y S T E M  S C 6 3  --  N O D E

11.16.46 JOB15421 ---- FRIDAY,    01 FEB 2013 ----
11.16.46 JOB15421  IRR010I  USERID ADMR2    IS ASSIGNED TO THIS JOB.
11.16.46 JOB15421  ICH70001I ADMR2    LAST ACCESS AT 11:16:39 ON FRIDAY, FEBRUAR
11.16.46 JOB15421  £HASP373 INZCHEAA STARTED - INIT TWS  - CLASS T - SYS SC63
11.16.46 JOB15421  IEF403I INZCHEAA - STARTED - TIME=11.16.46 - ASID=0047 - SC63
11.16.46 JOB15421  -                                          --TIMINGS (MINS.)--
11.16.46 JOB15421  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK
11.16.46 JOB15421  -INZCHEAA          INZDDB0A    00     47    .00    .00    .00
11.16.46 JOB15421  -INZCHEAA          INZTDB0A    00     73    .00    .00    .00
11.16.46 JOB15421  IGD01008I STORCLAS SET TO
11.16.46 JOB15421  INZX006 SYSTSDBA TABLESPACE UNLOAD PHASE STARTED
11.16.46 JOB15421  INZX090 SYSREC1  : 1 RECORDS WRITTEN IN 00:00:00, UNLOAD DONE
11.16.46 JOB15421  INZU222I SYSREC1 , TOTAL NUMBER OF RECORDS WRITTEN 1
11.16.46 JOB15421  -INZCHEAA          DB2NDB0A    00   1435    .00    .00    .01
11.16.47 JOB15421  IGD01008I STORCLAS SET TO
11.16.47 JOB15421  INZX090 SYSREC1  : 1 RECORDS WRITTEN IN 00:00:00, UNLOAD DONE
11.16.47 JOB15421  INZU222I SYSREC1 , TOTAL NUMBER OF RECORDS WRITTEN 1
11.16.47 JOB15421  -INZCHEAA          DB2FDB0A    00   1386    .00    .00    .00
11.16.47 JOB15421  IEF404I INZCHEAA - ENDED - TIME=11.16.47 - ASID=0047 - SC63
11.16.47 JOB15421  -INZCHEAA ENDED.  NAME-RE                   TOTAL CPU TIME=
11.16.47 JOB15421  £HASP395 INZCHEAA ENDED
```

Figure 1-21 Sample output from DB2 High Performance Unload check job

1.3.3 DB2 Sort for z/OS

In customizing the DB2 Sort component, only the component parameters and DB2 entries need to be updated. There is no LPAR parameter customization required for this component. The job list generated from the GENERATEALL subcommand is listed in Figure 1-22 on page 24.

Ensure that you set the correct VCAT HLQ in the DB2 entries for the DB2 storage group and that the correct Authid is set. This is used by the IVP jobs.

```
TCUSTMZR                Finish Component Customization         Row 1 to 5 of 5
Command ===>                                                  Scroll ===> CSR

Submit the members in the order in which they apply to all DB2 entries. To
submit the job, browse the member and issue the TSO SUBMIT command, or browse
the customized library and submit the jobs from there.

Pack to Customize .: DB2 Utilities Solution Pack        >  Version . : 1.1.0
  Component metadata library : 'DBTLSP.SCNKDENU          > LPAR  . . : SC63
  Component name . . . . . . : DB2 Sort for z/OS         > Version . : 1.3.0

Line Commands: E - Edit  B - Browse

    Component customization library: DB2TOOLS.JCL.£SC63£.CNK130

 Cmd Member   SSID GrpAttch Template Date       Description
 -   -------- ---- ----     -------- ---------- ------------------------------>
     AORLNKJ   --   --      CNKRLNKJ 2013/02/01 Customize DB2 Sort library.
     A1IVP1AA DBOA  --      CNKIVP1J 2013/02/01 DB2 Sort IVP job 10B/40B.
     A1IVP1AB DBOC  --      CNKIVP1J 2013/02/01 DB2 Sort IVP job 10B/40B.
     A2IVP2AA DBOA  --      CNKIVP2J 2013/02/01 DB2 Sort IVP job 8B/25B.
     A2IVP2AB DBOC  --      CNKIVP2J 2013/02/01 DB2 Sort IVP job 8B/25B.
--------------------------- End of customized jobs ----------------------------




      EssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssN
      e CCQC000I The jobs were generated on the selected DB2 entries. e
 F1=H DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM ANGE
```

Figure 1-22 Job list from DB2 Sort GENERATEALL

1.3.4 DB2 Utilities Enhancement Tool

As with the other components, we customize the DB2 Utilities Enhancement Tool product by following the same procedure. First, we ALLOCATE the DB2 entries, and then we customize the component parameters. Again, there are no LPAR parameters for this product. After we complete the customization, we can GENERATEALL the jobs. The job list for our DB2 Utilities Enhancement Tool implementation is shown in Figure 1-23 on page 25.

```
TCUSTMZR                Finish Component Customization       Row 1 to 12 of 12
Command ===>                                                  Scroll ===> CSR

Submit the members in the order in which they apply to all DB2 entries. To
submit the job, browse the member and issue the TSO SUBMIT command, or browse
the customized library and submit the jobs from there.

Pack to Customize .: DB2 Utilities Solution Pack        >  Version . : 1.1.0
  Component metadata library : 'DBTLSP.SABPDENU          > LPAR  . . : SC63
  Component name . . . . . . : DB2 Utilities Enhancemen  > Version . : 2.2.0

Line Commands: E - Edit  B - Browse

    Component customization library: DB2TOOLS.JCL.£SC63£.ABP220

 Cmd Member   SSID GrpAttch Template Date       Description
 -   -------- ---- ----     -------- ---------- ------------------------------>
     A00BJCAA DB0A  --      ABPOBJCR 2013/02/01 Create DB2 UET DB2 objects
     A00BJCAB DB0C  --      ABPOBJCR 2013/02/01 Create DB2 UET DB2 objects
     A5BNDCAA DB0A  --      ABPBNDCR 2013/02/01 Bind the DB2 UET plan
     A5BNDCAB DB0C  --      ABPBNDCR 2013/02/01 Bind the DB2 UET plan
     A7OPTCAA DB0A  --      ABPOPTCR 2013/02/01 Create the options module
     A7OPTCAB DB0C  --      ABPOPTCR 2013/02/01 Create the options module
     A8MODCAA DB0A  --      ABPMODCR 2013/02/01 Create BCAN, BGLB, PROC, PLCY m
     A8MODCAB DB0C  --      ABPMODCR 2013/02/01 Create BCAN, BGLB, PROC, PLCY m
     A9SMPCAA DB0A  --      ABPSMPCR 2013/02/01 Create maintenance members
     A9SMPCAB DB0C  --      ABPSMPCR 2013/02/01 Create maintenance members
     B0RUNCAA DB0A  --      ABPRUNCR 2013/02/01 Create product CLISTs
     B0RUNCAB DB0C  --      ABPRUNCR 2013/02/01 Create product CLISTs
--------------------------- End of customized jobs ----------------------------
```

Figure 1-23 The Utilities Enhancement Tool Job list built from the solution pack

Check that the SVC number specified in the component parameters is not in use. By default, it is 255. We changed ours to 254. Also, the ABP Server Address Space Procedure ABP1PROC needs to be in SYS1.PROCLIB and started before the Utilities Enhancement Tool interface can be used. The procedure is created in the SABPSAMP library and needs to be copied to PROCLIB.

1.4 Installing the DB2 Fast Copy Solution Pack

The IBM DB2 Fast Copy Solution Pack combines two DB2 Tool products: DB2 Cloning and DB2 Recovery Expert. These two products help to maintain highly available and fully recoverable DB2 production environments with the ability to copy in “near real time” for testing, training, and so on with minimal or no disruption to the original source environment.

These products utilize the storage-aware fast replication tools, such as IBM FlashCopy, to drastically reduce the downtime and copying overhead of backing up or creating cloned environments. By using “storage processor” cycles rather than z/OS CPU cycles, you can back up and recover within a much shorter and less disruptive time frame than traditional methods.

The combination of these two tools means that you can perform these functions:

- Offload copy operations to the storage processor.
- Take more frequent backups.
- Create cloned environments easily for testing purposes.
- Save CPU and I/O by removing the need to create Image Copies from production systems.
- Improved restore options. Restore from system level backup (SLB) or image copy (IC), and then perform log-based recovery to a point in time (PIT).
- Use Recovery Analysis to determine the fastest possible recovery of DB2 objects.
- Build alternative recovery scenarios by using Recovery Analysis.
- Utilize DBA resources more productively.

Similar to the Utilities Solution Pack, we start the Fast Copy Solution Pack installation by loading up the DENU metadata file in the Tools Customizer. Specify HLQ.SBBXDENU as the input metadata library to Tools Customizer.

1.4.1 DB2 Cloning Tool for z/OS

Start from the Fast Copy Solution Pack customization panel shown in Figure 1-24.

```
TCUSTMZR              Select the Components to Customize      Row 1 to 2 of 2
Command ===>                                                 Scroll ===> CSR

Select one or more components to customize. Press Enter to continue
or End to cancel.

Pack metadata library . : 'DBTLSP.SBBXDENU'
Pack to customize . . . : DB2 Fast Copy Solution Pack f  > Version . : 1.1.0

Line commands: / - Select
  Cmd Name                                  Version Customization Status
  -   ----------------------------------> ------- --------------------
      DB2 Cloning Tool for z/OS             3.1.0   Pending Customization
      DB2 Recovery Expert for z/OS          3.1.0   Pending Customization
---------------------------- End of components ----------------------------
```

Figure 1-24 The Fast Copy Solution Pack components

Select the component and modify the component parameters and DB2 parameters. As you can see from Figure 1-25 on page 27, there are no LPAR parameter modifications.

```
TCUSTMZR                    Customizer Workplace              Row 1 to 2 of 2
Command ===>                                                 Scroll ===> CSR

Use the Generate jobs line command to select the DB2 entries on which to
customize the component, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST
Pack to Customize .: DB2 Fast Copy Solution Pack for z/ >  Version . : 1.1.0
  Component metadata library : 'DBTLSP.SCKZDENU          > LPAR  . . : SC63
  Component name . . . . . . : DB2 Cloning Tool for z/O  > Version . : 3.1.0

Component and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
    Component parameters.: Ready to Customize
    LPAR parameters . . .: Not Required

Associated DB2 Entries and Parameter Status
  Line commands: G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID GrpAttch Lvl Mode User ID  Date          Status
      DBOA  --      101 NFM  ADMR2    2013/02/02    Ready to Customize
      DBOC  --      101 NFM  ADMR2    2013/02/01    Customized
    -------------------------- End of DB2 entries -----------------------------









      EsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssN
      e CCQB001I The DB2 parameter data was saved in the data store. e
 F1=H DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM HANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 1-25 Preparing to modify the component parameters for the DB2 Cloning Tool

After customization, generate the job list through the GENERATEALL command. This creates the JCL in the customization library that has been defined in the Tools Customizer customization panel. It follows the format of HLQ.JCL.£ssid£.componentid, for example, DB2TOOLS.JCL.£SC63£.INZ420. The job list for the cloning tool is shown in Figure 1-26 on page 28.

```
TCUSTMZR                Finish Component Customization       Row 1 to 13 of 23
Command ===>                                                  Scroll ===> CSR

Submit the members in the order in which they apply to all DB2 entries. To
submit the job, browse the member and issue the TSO SUBMIT command, or browse
the customized library and submit the jobs from there.

Pack to Customize .: DB2 Fast Copy Solution Pack for z/ >  Version . : 1.1.0
  Component metadata library : 'DBTLSP.SCKZDENU          > LPAR  . . : SC63
  Component name . . . . . . : DB2 Cloning Tool for z/O  > Version . : 3.1.0

Line Commands: E - Edit  B - Browse

    Component customization library: DB2TOOLS.JCL.£SC63£.CKZ310

 Cmd Member   SSID GrpAttch Template Date       Description
 -   -------- ---- ----     -------- ---------- ------------------------------>
     A0IMRG    --   --      CKZIMRG  2013/02/02 Run INIMERGE to merge a previou
     A1BNDPAA DB0A  --      CKZBNDPS 2013/02/02 Subsystem Cloning plan bind
     A1BNDPAB DB0C  --      CKZBNDPS 2013/02/02 Subsystem Cloning plan bind
     A2BNDSAA DB0A  --      CKZBNDSS 2013/02/02 Subsystem Cloning package bind
     A2BNDSAB DB0C  --      CKZBNDSS 2013/02/02 Subsystem Cloning package bind
     A3DROSAA DB0A  --      CKZDROSP 2013/02/02 Drop Subsystem Cloning stored p
     A3DROSAB DB0C  --      CKZDROSP 2013/02/02 Drop Subsystem Cloning stored p
     A4DEFSAA DB0A  --      CKZDEFSP 2013/02/02 Define Subsystem Cloning stored
     A4DEFSAB DB0C  --      CKZDEFSP 2013/02/02 Define Subsystem Cloning stored
     A5BNDSAA DB0A  --      CKZBNDSP 2013/02/02 Bind the Subsystem Cloning stor
     A5BNDSAB DB0C  --      CKZBNDSP 2013/02/02 Bind the Subsystem Cloning stor
     A6SSP1    --   --      CKZSSP1  2013/02/02 Allocate the Subsystem Cloning
     A7SSP2    --   --      CKZSSP2  2013/02/02 Allocate the Subsystem Cloning
```

Figure 1-26 Job list for installing the DB2 Cloning Tool

Run through the installation jobs to complete the installation of the DB2 Cloning Tool. The CLISTs copied in job CKZ should update your local CLIST library with the CKZISPF1 driver CLISTs. Be aware that the job uses DISP=OLD, which causes a wait until the CLIST library is allocated.

Test the component installation by executing the driver CLIST CKZ. The initial panel (Figure 1-27 on page 29) appears.

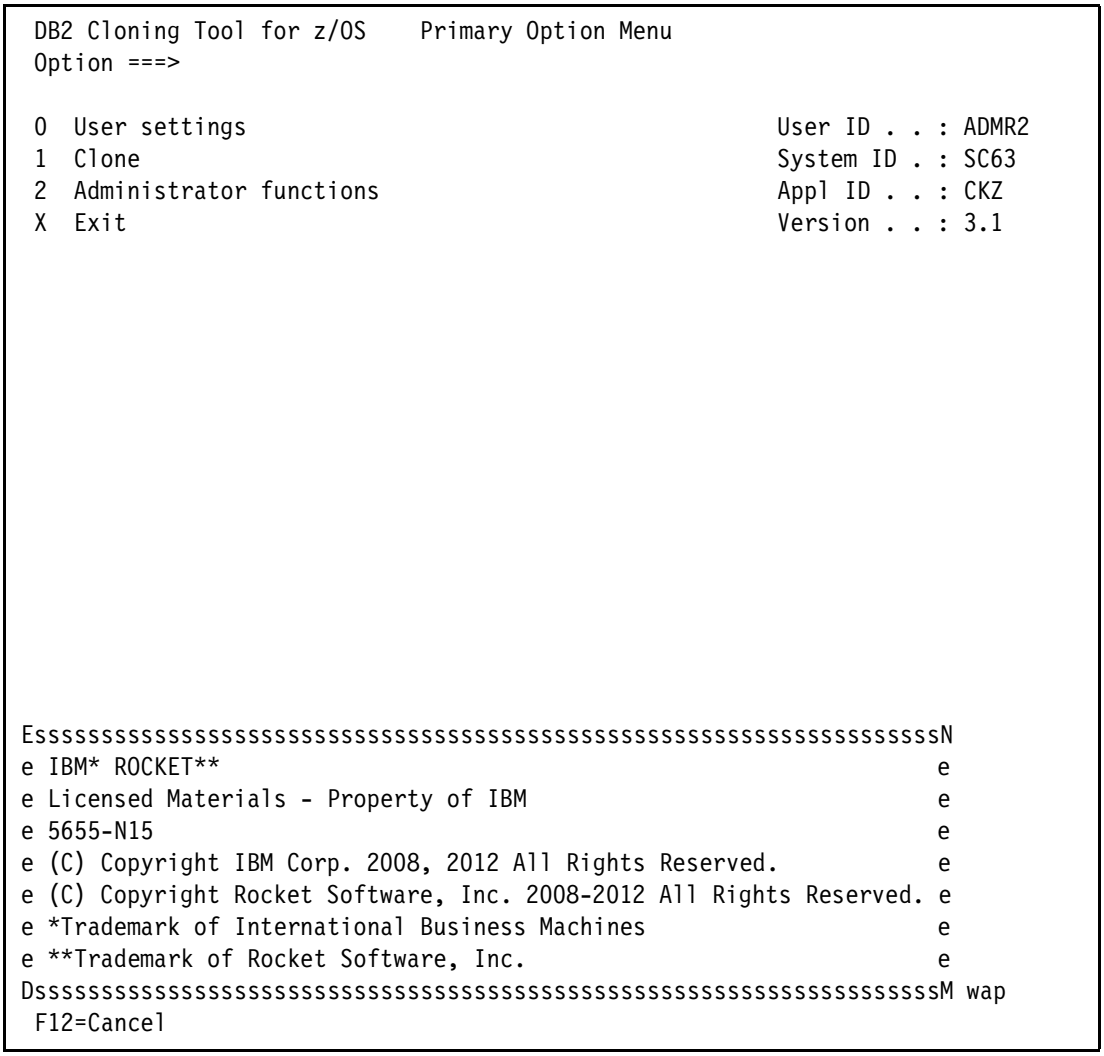

Figure 1-27 DB2 Cloning Tool for z/OS Primary Option Menu

For the DB2 Cloning Tool, if the subsystem cloning stored procedure is to be used, the DB2 administrative task scheduler must be configured and available on the DB2 systems where the stored procedures are run.

DB2 Cloning Tool does not have a Launchpad interface, although you can write one. The instructions are in the *IBM DB2 Administration Tool for z/OS Version 10 Release 2 User's Guide and Reference*, SC19-3774. For ease of use, we have combined all the tools' products through a local ISPF interface shown in Figure 1-28 on page 30.

```
 -------------------- DB2 Tools Primary Options Menu ------------------
OPTION ===>
                                                              USERID - ADMR2
Select an option from below.                                  DATE   - 13/02/02
                                                              TIME   - 06:12
     A Admin.       - Administration Tool
     T Automation   - Automation Tool
     C Cloning      - Cloning Tool
     I HPU          - High Performance Unload
     R RE           - Recovery Expert
     U UET          - Utilities Enhancement Tool

     L LP           - DB2 Launchpad
    Z Tools Customizer        - Toolkit Customization

     X EXIT         - Exit
```

Figure 1-28 Local ISPF interface for tool products

1.4.2 DB2 Recovery Expert for z/OS

The second component in the Fast Copy solution pack is DB2 Recovery Expert. Select the component from the solutions entries and the customization panel is shown. As with all the components installed, the DISCOVER EXEC is not used. This EXEC is used to copy components' parameter information from previous releases of tool products. The customization requires you to ASSOCIATE DB2 entries and then to customize the component and LPAR parameters.

This component has many parameters. It is essential that you understand the component and how the parameters need to be customized to fit within your organization *before you perform customization*. Failure to perform pre-investigative work and installation planning leads to an installation failure. Other chapters in this book help you to determine how to install and customize Recovery Expert. Read those first, plus the related user customization and installation manual, before proceeding with the installation.

After the parameters have been customized, generate the job stream through the GENERATEALL command. See Figure 1-29 on page 31.

```
TCUSTMZR                Finish Component Customization       Row 1 to 13 of 47
Command ===>                                                  Scroll ===> CSR

Submit the members in the order in which they apply to all DB2 entries. To
submit the job, browse the member and issue the TSO SUBMIT command, or browse
the customized library and submit the jobs from there.

Pack to Customize .: DB2 Fast Copy Solution Pack for z/ >  Version . : 1.1.0
  Component metadata library : 'DBTLSP.SARYDENU          > LPAR  . . : SC63
  Component name . . . . . . : DB2 Recovery Expert for   > Version . : 3.1.0

Line Commands: E - Edit  B - Browse

   Component customization library: DB2TOOLS.JCL.£SC63£.ARY310

 Cmd Member   SSID GrpAttch Template Date       Description
 -   -------- ---- ----     -------- ---------- ------------------------------>
     AOEXECS   --   --      ARYEXECS 2013/02/02 ARY Required Execs
     A1V31     --   --      ARYV31   2013/02/02 Configure Startup Clist 1
     A2V310    --   --      ARYV310  2013/02/02 Configure Startup Clist 2
     A3DRPDAA DBOA  --      ARYDRPDB 2013/02/02 Drop the IBM Recovery Expert DB
     A3DRPDAB DBOC  --      ARYDRPDB 2013/02/02 Drop the IBM Recovery Expert DB
     A4DDLSAA DBOA  --      ARYDDLSG 2013/02/02 Create IBM Recovery Expert SG
     A4DDLSAB DBOC  --      ARYDDLSG 2013/02/02 Create IBM Recovery Expert SG
     A5DDLDAA DBOA  --      ARYDDLDB 2013/02/02 Create IBM Recovery Expert DB
     A5DDLDAB DBOC  --      ARYDDLDB 2013/02/02 Create IBM Recovery Expert DB
     A9DDL1AA DBOA  --      ARYDDL1  2013/02/02 Create DB2 Objects
     A9DDL1AB DBOC  --      ARYDDL1  2013/02/02 Create DB2 Objects
     BODDL2AA DBOA  --      ARYDDL2  2013/02/02 Create views on SLR
     BODDL2AB DBOC  --      ARYDDL2  2013/02/02 Create views on SLR
```

Figure 1-29 Recovery Expert Job Stream from the GENERATEALL command

Now, continue to install the product by running the jobs that have been generated by the Tools Customizer customization process.

2

FlashCopy technology

IBM DB2 for z/OS optimizes the use of disk storage functions in z/OS environments to productively use the synergy with I/O subsystems on IBM System z. These functions are also used by DB2 for z/OS tools, allowing ease of use and better interactions between the DB2 administrator and the storage administrator.

To start, see *DB2 9 for z/OS and Storage Management,* SG24-7823. This book offers the DB2 administrator information about how to use DFSMS for managing DB2 data sets. This book offers the storage administrator information about the characteristics of DB2 data sets and how DB2 uses the disks.

In this chapter, we explain the basic characteristics of FlashCopy when used in a System z environment with the DS8000 as a prerequisite for its use by DB2 utilities, DB2 Recovery Expert, and DB2 Cloning Tool. How FlashCopy works is helpful in understanding the advantages that it provides for cloning, backup, and recovery. The complexity of invoking and managing the FlashCopy commands and relationships is handled automatically by DB2 Recovery Expert and the DB2 Cloning Tool.

The main task of FlashCopy is to create a copy of a volume or data set at a specific point-in-time, which is also known as a Point-in-Time copy, instantaneous copy, or time-zero copy (t0 copy).

This chapter describes the following topics:

- Operational environments
- Basic concepts
- FlashCopy in combination with other Copy Services
- FlashCopy for z/OS data sets
- DB2 for z/OS and FlashCopy

2.1 Operational environments

It takes less than a second to a few seconds to establish the FlashCopy relationships for tens to hundreds or more volume pairs. The copy is then immediately available for both read and write access. In a 24x7 production environment, you can use the speed of the FlashCopy operation to take multiple FlashCopies of the same volume for use with different applications or operations. Some of the uses of FlashCopy are shown in Figure 2-1.

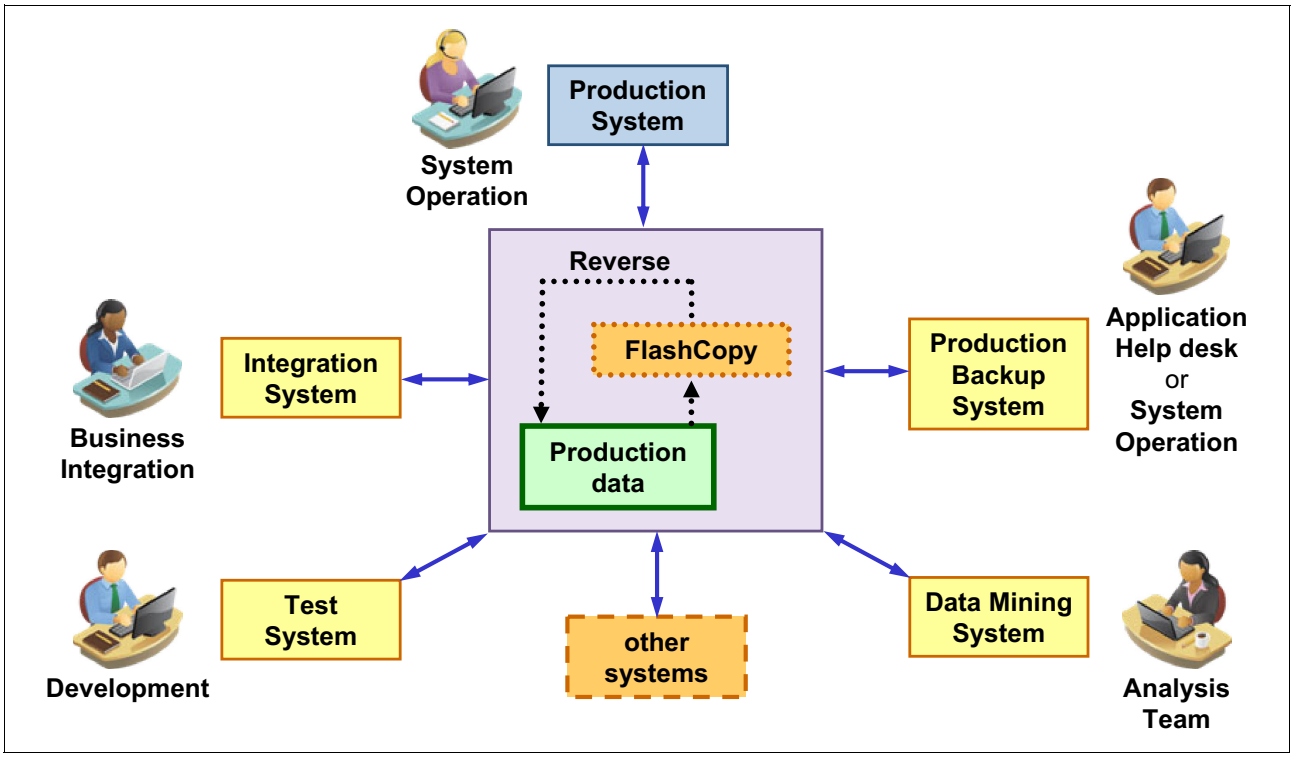

Figure 2-1 FlashCopy uses

FlashCopy is suitable for the following operational environments:

- Production backup system

  A periodic FlashCopy of the production data allows data recovery from an earlier version of data. This action might be necessary because of a user error or a logical application error.

  Assume that a user accidentally deletes a customer record. The production backup system could work with one of the periodic FlashCopy copies of the data. The necessary part of the customer data can be exported and then be imported into the production environment. Thus, production continues while a specific problem is being fixed, and most users continue to work without any knowledge of this issue.

  The FlashCopy of the data can also be used by another operating system to re-establish production in case there are any server errors.

  A FlashCopy of the production data allows the client to create backups with no application outage. An additional reason for data backup is to provide protection in case there is source data loss because of a disaster, hardware failure, or software failure.

- Data warehousing system

  A FlashCopy of the data can be used as the foundation to create a clone system for data analysis, thus avoiding performance impacts for the production system because of long running queries.

- Test system

  Test environments that are created with FlashCopy can be used by the development team to test new application functions with real production data, which leads to a faster test setup process.

- Integration system

  New application releases (for example, SAP releases) are likely to be tested before you implement them onto a production server. By using FlashCopy, a copy of the production data can be established and used for integration tests.

With the capability to reverse a FlashCopy, a previously created FlashCopy can be used within seconds to bring production back to the point-in-time when the FlashCopy was taken.

2.2 Basic concepts

In a discussion about Metro Mirror, Global Copy, and Global Mirror, the following terms are frequently used interchangeably:

- The terms *local*, *production*, *application*, *primary*, or *source* denote the site where the production applications run while in normal operation. These applications create, modify, and read the application data. The meaning is extended to the storage system that holds the data, as well as to its components, that is, volumes and logical subsystems (LSS).
- The terms *remote*, *recovery*, *backup*, *secondary*, or *target* denote the site to where the data is replicated (the copy of the application data). The meaning is extended to the storage system that holds the data, as well as to its components (volumes and LSS).

When we describe FlashCopy, we use the term *source* to refer to the original data that is created by the application, and we use the term *target* to refer to the point-in-time backup copy.

By creating a FlashCopy, a relationship is *established* between source and target volumes. The two volumes form a FlashCopy *pair*.

As a result of the FlashCopy, all physical blocks from the source volume are copied to the target volume or, when you use the **`no copy`** option, only the data that changed in the source volume since the FlashCopy was established is copied. The target volume must be the same size or larger than the source volume and be in the same storage facility image when FlashCopy is used to copy the entire volume.

Three types of FlashCopy are available:

- Standard FlashCopy uses a fully provisioned volume as a target volume.
- Space-efficient FlashCopy (FlashCopy SE) uses track space-efficient (TSE) volumes as FlashCopy target volumes and must be in a background no copy relationship. A space-efficient volume has a virtual size that is equal to the source volume size. However, space is not allocated when the volume is initially created and the FlashCopy initiated. Space is allocated in a repository when the first update is made to a track on the source volume, which causes the source track to be copied to the FlashCopy SE target volume to maintain the t0 copy. Writes to the FlashCopy SE target also use repository space.

For more information about Space-Efficient volumes and the concept of repository, see Chapter 11, “IBM FlashCopy SE” on page 119 of *IBM System Storage DS8000 Copy Services for IBM System z,* SG24-6787-06.

- FlashCopy is supported by thin-provisioned Extent-Space-Efficient (ESE) volumes with LMC 6.6.20.*nnn* for DS8700 and LMC 7.6.20.*nnn* for DS8800. Space is not allocated when the thin-provisioned volume is initially created. Extents are allocated from an extent pool when the first update is made to an extent on the thin-provisioned volume. Thin provisioning does not use tracks from a repository, but rather uses extents from the extent pool. ESE thin provisioning supports only fixed block (FB) volumes.

FlashCopy, FlashCopy SE, and thin provisioning are optional and distinct licensed features of the DS8000. All features can coexist on a DS8000.

Note: DB2 Cloning Tool and DB2 Recovery Expert only operate on count key data (CKD) volumes (z/OS format volumes).

Typically, large databases have their data spread across multiple volumes. If these volumes are copied while data on them is being updated, the order of dependent writes must be maintained to ensure that the target volumes have consistent data. Consistent data allows a database restart, as opposed to a database recovery, which could take a long time to complete. Consistency Group FlashCopy can maintain the order of dependent writes and create volume copies that have consistent data.

DB2 Cloning Tool and DB2 Recovery Expert are “storage-aware” database tools that automate the process of driving the FlashCopy processes described as Consistency Group FlashCopy. This allows the FlashCopy target volumes to be used effectively as a backup and the basis of a cloned DB2 system.

Terminology: Whenever *source* or *target* is used in this chapter without further specifying what is meant, it refers to both a volume or a data set.

The following characteristics are basic to the FlashCopy operation:

- Establishing a FlashCopy relationship

  When a FlashCopy is started, the relationship between source and target is established within seconds by creating a pointer table, including a bitmap for the target.

  While the FlashCopy relationship is being created, the DS8000 holds off the I/O activity to the volume for a period of time. No user intervention is required. I/O activity resumes when the FlashCopy *establish* process is completed.

  If all bits for the bitmap of the target are set to their initial values, this configuration means that no data block is copied so far. A bitmap entry of 1 indicates that the track is not copied yet, and a 0 indicates that it is copied. The data in the target is not modified during the setup of the bitmaps. At this first step, the bitmap and the data look as illustrated in Figure 2-2 on page 37.

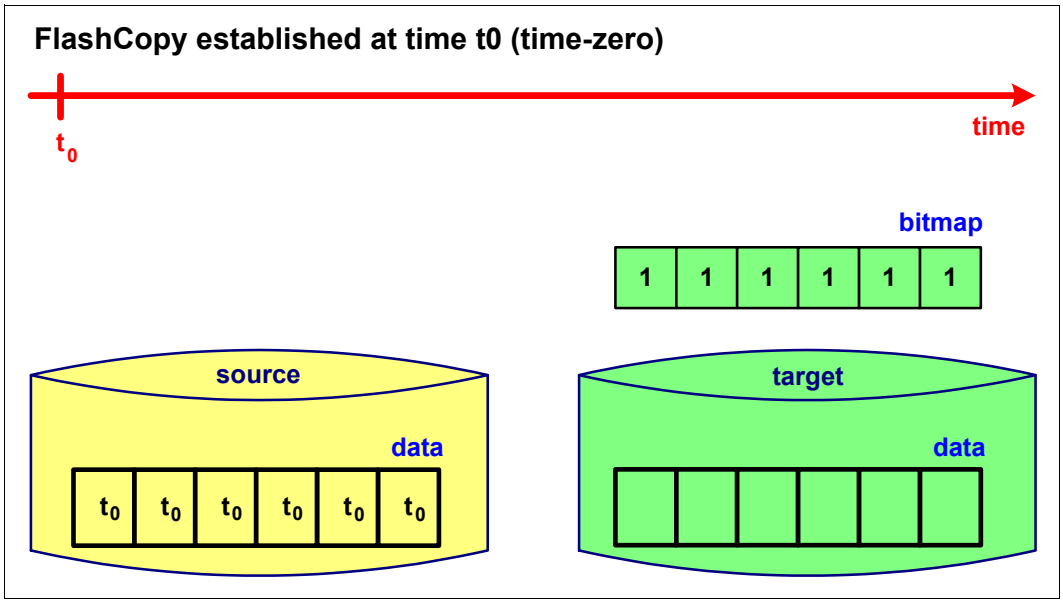

Figure 2-2 FlashCopy at time t0

The target volume, as depicted in various figures in this section, can be a normal volume or a space-efficient volume. In both cases, the logic is the same.

The difference between standard FlashCopy and FlashCopy SE is where the physical storage is. For standard FlashCopy, it is a fully provisioned volume; for IBM FlashCopy SE, it is a repository (see Figure 2-4 on page 39).

When the relationship is established, it is possible to perform read and write I/Os on both the source and the target. Assuming that the target is used for reads only while production is ongoing, the environment looks as illustrated in Figure 2-3 on page 38.

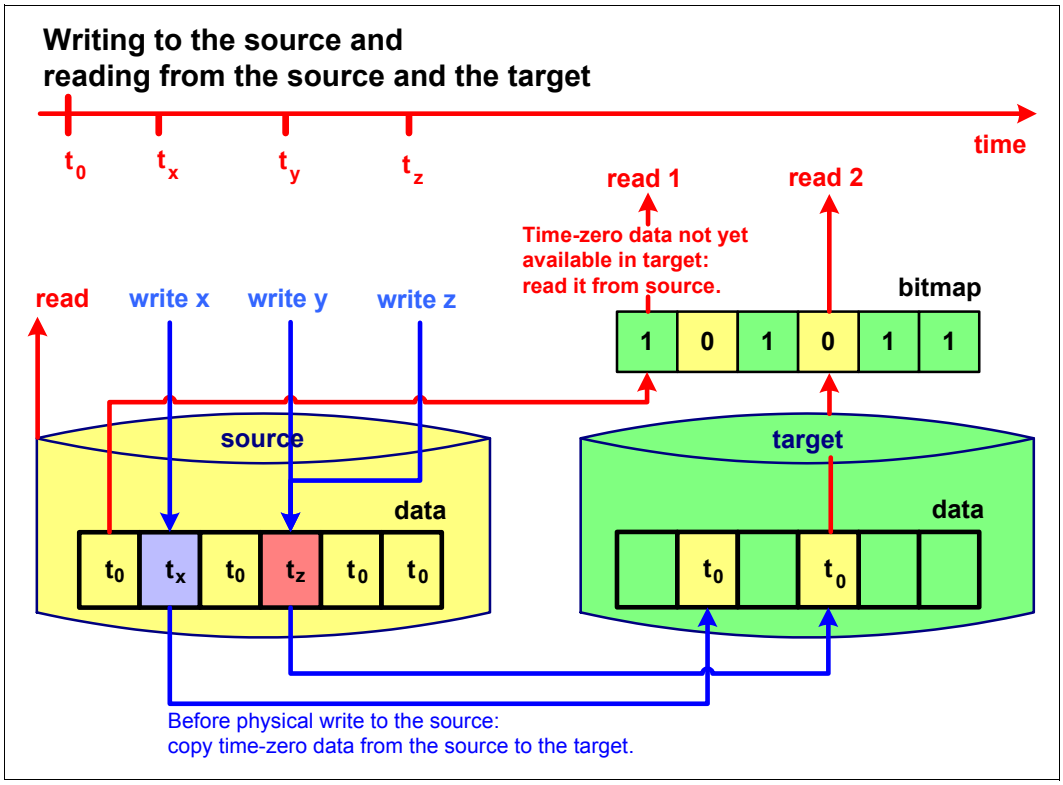

Figure 2-3 Reads from source and target volumes and writes to source volume

Figure 2-3 reading from the source and target volumes and writing to the source volume:

- Reading from the source

  The data is read immediately from the source volume.

- Writing to the source

  Whenever data is written to the source volume while the FlashCopy relationship exists, the storage system makes sure that the t0 data is copied to the target volume before you overwrite it in the source volume. When the target volume is a space-efficient volume, the data is written to a repository.

  To determine if the data of the physical block on the source volume must be copied to the target volume, the bitmap is analyzed. If it determines that the t0 data is not available on the target volume, the data is copied from the source to the target. If it states that the t0 data is already copied to the target volume, no further action is done. If the bitmap is 0, original data is copied to the target volume, so this I/O is written straight to the source volume. See Figure 2-3.

  The target volume is immediately available for reading data and for writing data.

- Reading from the target

  Whenever a read-request goes to the target while the FlashCopy relationship exists, the bitmap is used to identify whether the data must be retrieved from the source or from the target. If the bitmap states that the t0 data is not yet copied to the target, the physical read is directed to the source. If the t0 data is copied to the target, the read is performed immediately against the target. See Figure 2-3 or Figure 2-4 on page 39.

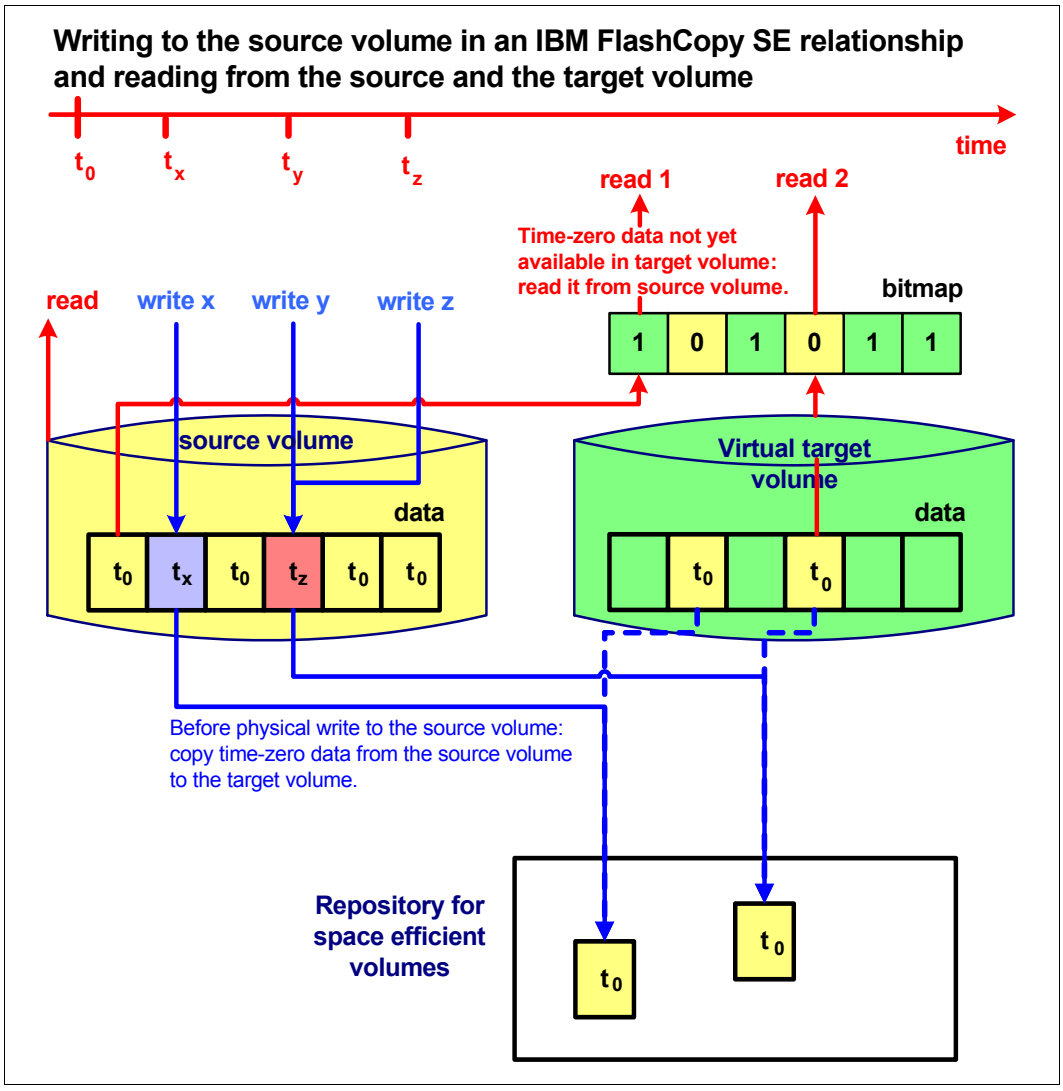

Figure 2-4 Reads from source and target volumes and writes to source volume for IBM FlashCopy SE relationships

- Writing to the target

  Whenever data is written to the target volume while the FlashCopy relationship exists, the storage system ensures that the bitmap is updated. This way, the t0 data from the source volume never overwrites updates that are done directly to the target volume. So, if the bitmap is 1, it is set to 0 to prevent the data from being overwritten by source data in the future. See Figure 2-5 on page 40.

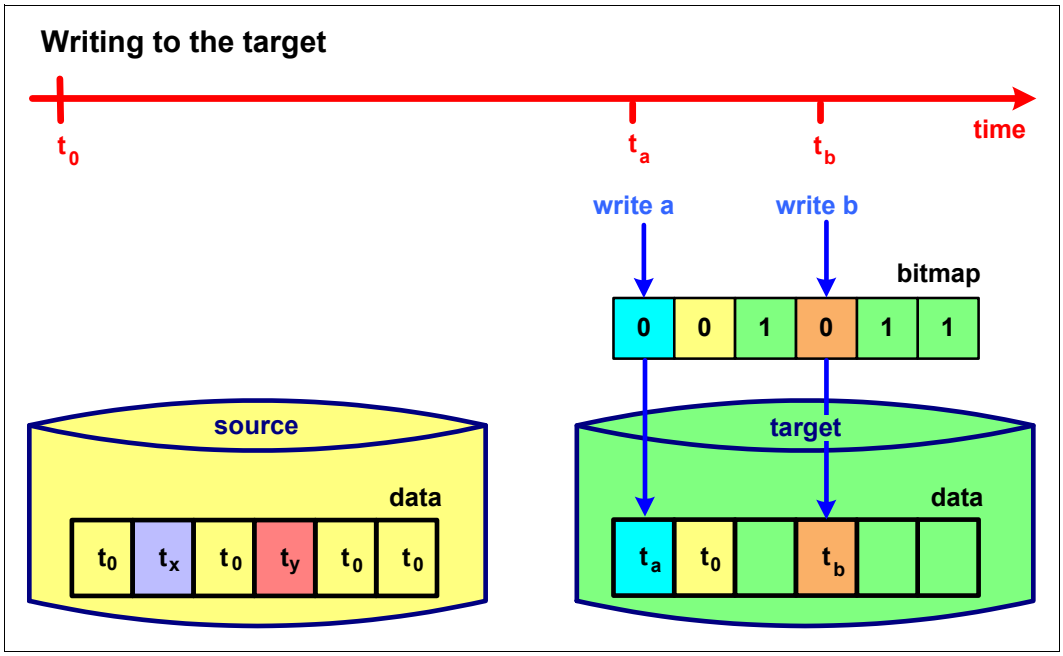

Figure 2-5 Writes to the target volume

- Terminating the FlashCopy relationship

  The FlashCopy relationship is *automatically removed* when all tracks are copied from the source volume to the target volume. The relationship can also be *explicitly withdrawn* by running the relevant commands. If the **-persistent** option is specified, the FlashCopy relationship continues until it is explicitly withdrawn.

  An IBM FlashCopy SE relationship must be explicitly withdrawn. When the relationship is withdrawn, there is an option to release the allocated space of the space-efficient volume.

  The main advantage of FlashCopy SE is that it allows you to use much less space for each clone or each backup copy of DB2 that is created. Because of this, you will be able to create more backup points or clones without incurring the cost of doubling the storage requirement for each backup or clone.

  Both DB2 Recovery Expert and DB2 Cloning Tool automatically discover if a space-efficient FlashCopy volume is being used and automatically manage the FlashCopy SE relationship for you.

2.2.1 Full volume copy

When the **copy** option is invoked and the establish process completes, a background process is started that copies all data from the source to the target. When this process is finished and if there are no updates on the target, you see a chart similar to the one in Figure 2-6 on page 41.

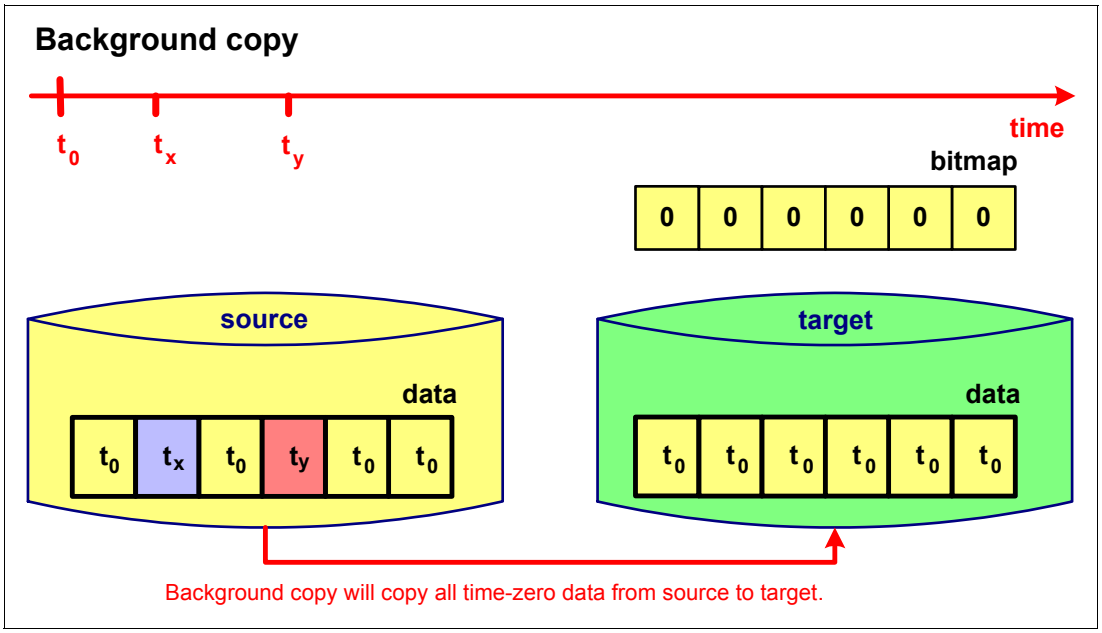

Figure 2-6 Target volume after a FlashCopy relationship ends

If not explicitly defined as *persistent*, the FlashCopy relationship ends as soon as all the data is copied.

Only the traditional FlashCopy allows a full copy; IBM FlashCopy SE has no such function. Remember that both features can coexist.

If there are writes to the target, you see a chart that is similar to the one in Figure 2-7.

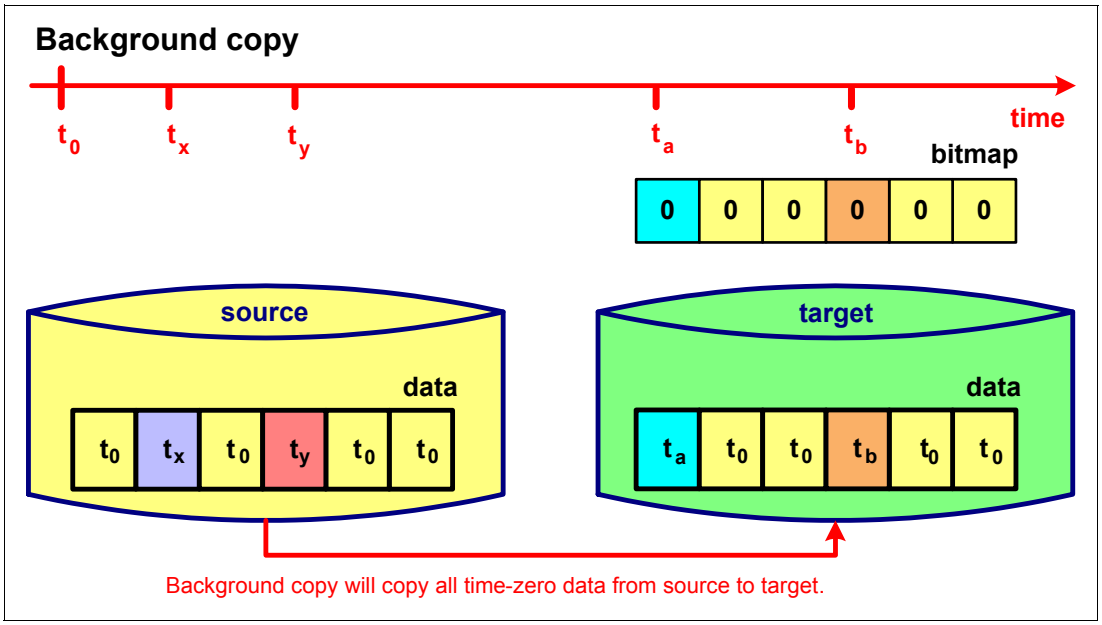

Figure 2-7 FlashCopy after updates to the target volume

2.2.2 No copy option

If FlashCopy is established by using the **nocopy** option, the result is the one shown in Figure 2-3 on page 38 and Figure 2-5 on page 40. The relationship lasts until it is explicitly withdrawn or until all data in the source volume is modified. Blocks for which no write occurred on the source or on the target stay as they were at the time when the FlashCopy was established.

If the **persistent** FlashCopy option was specified, the FlashCopy relationship must be withdrawn explicitly.

The **nocopy** option is the default for IBM FlashCopy SE. The main use of NOCOPY in the Fast Copy Pack is by DB2 Recovery Expert. One set of backup volumes can be used to back up multiple DB2 systems. This is accomplished by setting the number of generations to zero in the backup copypool storage group. Recovery Expert detects this setting and automatically uses the NOCOPY option when it drives FlashCopy to copy the DB2 system. The backup can then be copied to tape. When this process is complete, the same set of target volumes can be used to back up a different DB2. The use of NOCOPY avoids having the DS8*nnnn* go through the unnecessary process of copying all the data to the target volumes when the only desire is to have a copy on tape (or virtual tape).

2.3 FlashCopy in combination with other Copy Services

Volume-based FlashCopy can be used in various combinations with other Copy Services functions, while the most suitable option depends on the characteristics of the environment and the requirements.

Note: The scenarios that are described in the present section do not apply to data set FlashCopy.

2.3.1 FlashCopy with Metro Mirror and Global Copy

You can use this option to establish a FlashCopy relationship where the target is also a Metro Mirror or a Global Copy primary volume, which enables the user to create full or incremental point-in-time copies at a local site and then use remote mirroring to copy the data to a remote site.

The issue with this approach is when the user initiates a FlashCopy onto a Metro Mirror/Global Copy primary volume that is in a FULL DUPLEX mode, the mode switches to Copy PENDING state during the FlashCopy background copy operation. After FlashCopy completes, the primary volume returns to FULL DUPLEX. But, while the configuration is in the COPY PENDING state, the system is vulnerable to disaster because there is no disaster recovery protection until FlashCopy is finished and the state returns to the FULL DUPLEX state.

This issue is resolved with Remote Pair FlashCopy (see 2.3.2, “Remote Pair FlashCopy” on page 43).

Figure 2-8 on page 43 shows the system as vulnerable during the COPY PENDING state.

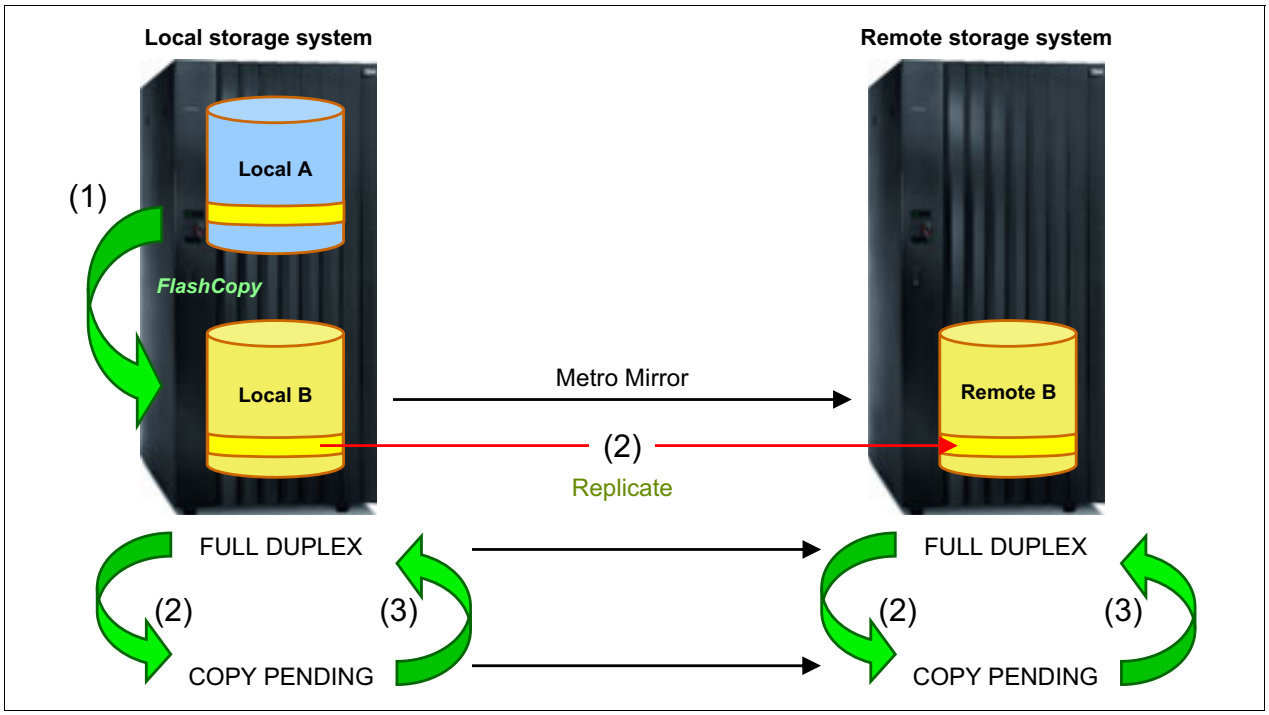

Figure 2-8 FlashCopy and Metro Mirror

2.3.2 Remote Pair FlashCopy

Remote Pair FlashCopy or Preserve Mirror is available with Licensed Machine Code (LMC) 6.5.10.*nnn* on DS8700 and is also available with LMC 7.6.00.*nnn* on the DS8800.

Remote Pair FlashCopy overcomes the shortcomings of the previous solution to FlashCopy onto a Metro Mirror primary volume and the loss of the disaster recovery capability during the FlashCopy copy operation. Figure 2-8 illustrates this behavior.

The function preserves the existing Metro Mirror status of FULL DUPLEX during the copy operation. Figure 2-9 on page 44 shows this approach, which ensures that there is no loss of disaster recovery functionality:

1. A FlashCopy command is issued by an application or by the customer to Local A with Local B volumes as the FlashCopy target. The DS8000 firmware propagates the FlashCopy command through the Metro Mirror links from the local storage system to the remote storage system. This inband propagation of a Copy Services command is only possible for FlashCopy commands.
2. Independently of each other, the local storage system and the remote storage system then run the FlashCopy operation. The local storage system coordinates the activities at its end and acts when the FlashCopies do not succeed at both storage systems. Remote Pair FlashCopy (also called *Preserve Mirror*)[1] supports both full volume and data set FlashCopy. The key is that disaster recovery protection is not absent at any time and FlashCopy operations can occur freely within the disk storage configuration.

[1] See *IBM System Storage DS8000: Remote Pair FlashCopy (Preserve Mirror)*, REDP-4504.

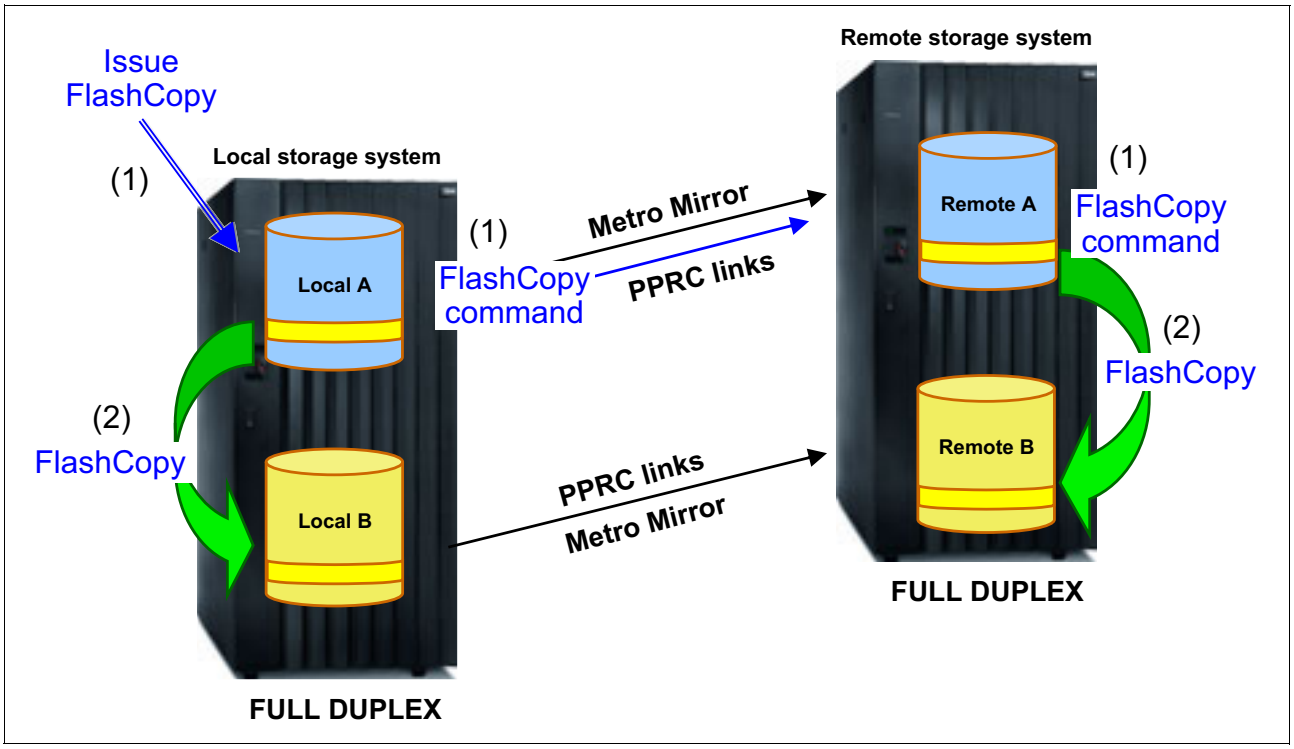

Figure 2-9 Remote Pair FlashCopy preserves Metro Mirror FULL DUPLEX state

The following conditions are required to establish Remote Pair FlashCopy:

- Both the Local A/Remote A and the Local B/Remote B Metro Mirror pairs are in the FULL DUPLEX state. LMC 6.6.20.*nnn* and 7.6.20.*nnn* or later allow Remote Pair FlashCopy to Metro Mirror pairs that are SUSPENDED or COPY PENDING.
- The Remote A and Remote B volumes are in the same DS8000 Storage Facility Image (SFI).
- The required microcode level must be installed on both the local and remote storage systems.

Remote Pair FlashCopy can be initiated by using various interfaces:

- Time Sharing Option (TSO)
- Device Support Facilities (ICKDSF)
- DFSMSdss
- DS CLI
- DS GUI
- Tivoli Storage Productivity Center for Replication

DB2 Recovery Expert automatically executes Remote Pair FlashCopy based on the original settings for the Recovery Expert installation.

2.3.3 FlashCopy and Global Mirror

FlashCopy in combination with Global Mirror supports only one type of relationship at the primary site (see Figure 2-10 on page 45).

A FlashCopy source volume can become a Global Mirror primary volume and vice versa. The relationships can be established in any sequence. A FlashCopy target volume cannot become a Global Mirror primary volume.

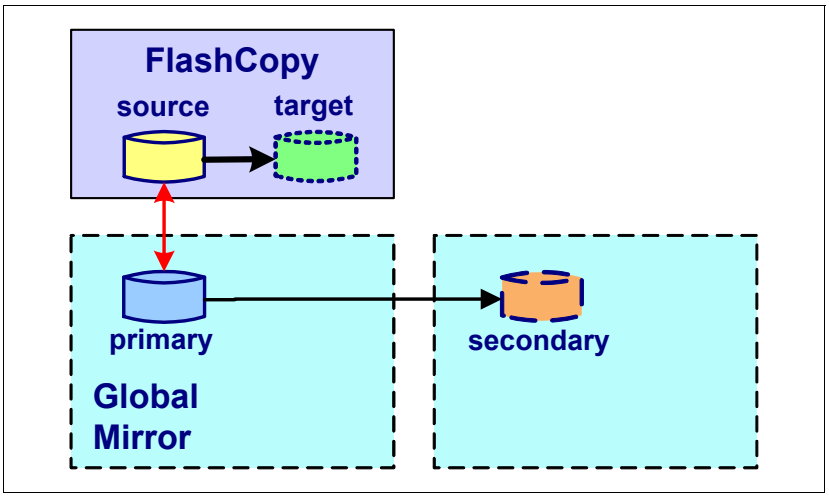

Figure 2-10 FlashCopy and Global Mirror

On the Global Mirror secondary site, the Global Mirror secondary volume cannot be used as a FlashCopy source or FlashCopy target unless the Global Mirror pair is first suspended.

To run FlashCopy in a Global Mirror site, the target volumes must not be Primary Global Mirror volumes and be removed from the Global Mirror Relationship. This should only be done if the clone is not required at the Global Mirror secondary site. For backup purposes, the backup can be offloaded to tape or a replicated virtual tape library to deliver it to the Global Mirror secondary site. DB2 Recovery Expert has automated processes to perform this copy to a replicated virtual tape library (VTL) and deliver the backup to the Global Mirror secondary site.

2.4 FlashCopy for z/OS data sets

The following rules apply when you use FlashCopy for z/OS data sets (see Figure 2-11 on page 46):

- All types of z/OS data sets are supported (sequential, partitioned, and VSAM data sets).
- The source data set and the target data set can be in the same or in different volumes.
- Within the volumes to which they belong, the source data set and the target data set can have different relative locations.

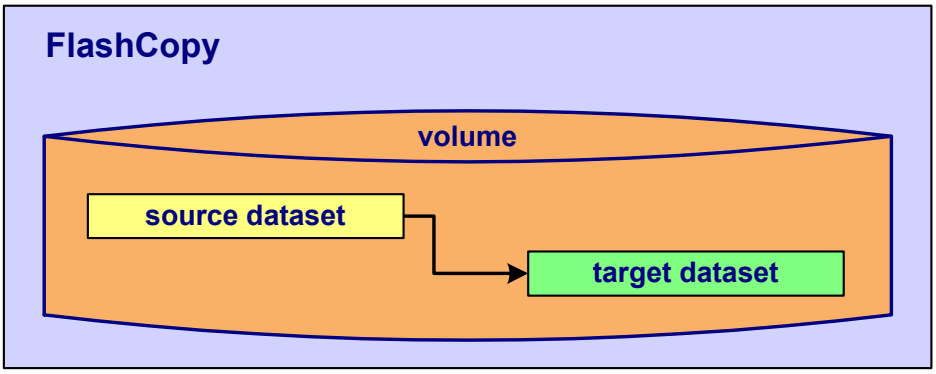

Figure 2-11 Source data set and target data set can be in the same volume

2.5 DB2 for z/OS and FlashCopy

The use of FlashCopy for data copy operations can reduce elapsed and CPU times spent in z/OS. A z/OS FlashCopy client, such as the DFSMSdss ADRDSSU fast replication function, waits only until the FlashCopy operation has logically completed, which normally takes a much shorter time than the actual physical FlashCopy operation spends inside the DS8000 disk subsystem. No CPU is charged to z/OS clients for the DS8000 physical data copy operation.

2.5.1 DB2 for z/OS and FlashCopy use

FlashCopy use in DB2 for z/OS began with Version 8 with the use of disk volume FlashCopy for the DB2 SYSTEM BACKUP and RESTORE utility, which helped to reduce the z/OS elapsed time that is required to back up and restore the entire DB2 system. For example, backing up a few terabytes of data consisting of active logs, bootstrap data sets, catalog, directory, and user table and index spaces can become a matter of only a few seconds to logically complete in z/OS without application unavailability.

In DB2 9, the BACKUP and RESTORE SYSTEM utilities were enhanced to support functions that are available with DFSMShsm V1R8. For example, in DB2 9, you can keep several SYSTEM BACKUP versions on DASD or on tape, you can use the SYSTEM BACKUP utility to perform incremental track-level FlashCopy operations, and you can use a SYSTEM BACKUP utility-created backup to recover individual table spaces or index spaces. Also in DB2 9, the CHECK INDEX SHRLEVEL CHANGE utility performs consistency checks on a table and index space shadow that the utility creates using the DFSMShsm ADRDSSU fast replication function, which implicitly uses FlashCopy.

For more information about FlashCopy usage in DB2 for z/OS, see *DB2 9 for z/OS and Storage Management*, SG24-7823.

2.5.2 FlashCopy use by the DB2 COPY utility

In DB2 10, the COPY utility is enhanced to provide an option to use the DFSMShsm fast replication function for taking full image copies by the COPY utility or the inline COPY function of the REORG and LOAD utilities. The DFSMShsm fast replication function invokes FlashCopy to perform the physical data copy operation, which in turn offloads the physical data copy operation to the DS8000 disk subsystem. As a result, no data pages need to be read into the table space buffer pool, which by itself reduces CPU usage that is normally

caused by buffer pool getpage processing. For more information about FlashCopy use by the DB2 10 COPY utility, see 11.1, “Support FlashCopy enhancements” on page 426 of *DB2 10 for z/OS Technical Overview*, SG24-7892.

To illustrate the performance benefit that the new FlashCopy use can provide, we created and populated a sample table with 100,000 rows in a table space. We defined the table space with MAXROWS 1 to force DB2 to allocate 100,000 data pages with one row per page. We then performed two COPY utility executions, one using FlashCopy and one not using FlashCopy, to compare COPY utility performance.

Figure 2-12 shows the accounting report highlights of the utility execution using the FLASHCOPY NO COPY utility option. In the buffer pool activity section, DB2 reads all data pages into the local buffer pool for image copy processing.

```
HIGHLIGHTS
--------------------
PARALLELISM: UTILITY

TIMES/EVENTS  APPL(CL.1)  DB2 (CL.2) TOTAL  BPOOL ACTIVITY   TOTAL
------------  ----------  ---------- -------------------- ------
ELAPSED TIME   11.381112    0.047218 GETPAGES              100299
CP CPU TIME     0.981997    0.472171 BUFFER UPDATES            58
 AGENT          0.022562    0.006620 SEQ. PREFETCH REQS       1564
 PAR.TASKS      0.959435    0.465551 PAGES READ ASYNCHR.     100093
```

Figure 2-12 FLASHCOPY NO COPY utility accounting report

Figure 2-13 shows the accounting report highlights of the utility execution using the FLASHCOPY YES COPY utility option. In the buffer pool activity section, DB2 does not read the data pages that are to be processed into the local buffer pool. Instead, DB2 invokes the DFSMShsm ADRDSSU fast replication function, which in this particular situation results in a 97% z/OS CPU time and a 94% elapsed time reduction compared to the COPY utility execution, as illustrated in Figure 2-12.

```
HIGHLIGHTS
--------------------
PARALLELISM: NO

TIMES/EVENTS  APPL(CL.1)  DB2 (CL.2) TOTAL  BPOOL ACTIVITY   TOTAL
------------  ----------  ---------- -------------------- ------
ELAPSED TIME    0.731253    0.034548 GETPAGES                 103
CP CPU TIME     0.020734    0.004625 BUFFER UPDATES            25
 AGENT          0.020734    0.004625 SEQ. PREFETCH REQS         0
 PAR.TASKS      0.000000    0.000000 PAGES READ ASYNCHR.        0
```

Figure 2-13 FLASHCOPY YES COPY utility accounting report

CPU and elapsed time savings: The CPU and elapsed time savings that you can achieve by using the COPY utility FlashCopy exploitation can vary, depending on I/O configuration and table space size. For a fair comparison, you need to evaluate the IBM Resource Measurement Facility™ (RMF™) report to include the resources spent to execute the DFSMS functions invoked by DB2.

For utilities measurements, see *DB2 10 for z/OS Performance Topics,* SG24-7942.

2.5.3 DSNZPARMs for FlashCopy use by utilities

The fields on the DSNTIP6 DB2 installation panel configure the default behavior of enhancements to the BACKUP SYSTEM, RESTORE SYSTEM, RECOVER, and other utilities, including the default enablement of FlashCopy use. See Figure 2-14.

```
DSNTIP6         INSTALL DB2 - DB2 UTILITIES PARAMETERS
===>

Enter system-level backup options for RESTORE SYSTEM and RECOVER below:
 1 SYSTEM-LEVEL BACKUPS ===> NO         As a recovery base: NO or YES
 2 RESTORE/RECOVER      ===> NO         From dump: NO or YES
 3 DUMP CLASS NAME      ===>            For RESTORE/RECOVER from dump
 4 MAXIMUM TAPE UNITS   ===> NOLIMIT    For RESTORE SYSTEM: NOLIMIT or 1-255
Enter default settings for the DB2 Utilities FLASHCOPY options below:
 5 FAST REPLICATION     ===> REQUIRED   DSScopy replication type for CHECK
                                        utilities: PREFERRED or REQUIRED
 6 FAST RESTORE         ===> PREFERRED  For RECOVERY: NONE, PREFERRED, REQUIRED
 7 FLASHCOPY PPRC       ===> REQUIRED   FLASHCOPY peer to peer remote copy:
                                        blank, NONE, PREFERRED, or REQUIRED
 8 DEFAULT TEMPLATE     ===> DBOAI.&DB..&SN..N&DSNUM..&UQ.

 9 COPY                 ===> YES        Use FLASHCOPY defaults for COPY
10 LOAD                 ===> YES        Use FLASHCOPY defaults for LOAD
11 REORG TABLESPACE     ===> YES        Use FLASHCOPY defaults for REORG TS
12 REBUILD INDEX        ===> YES        Use FLASHCOPY defaults for REBUILD IX
13 REORG INDEX          ===> YES        Use FLASHCOPY defaults for REORG IX

PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 2-14 DB2 utilities parameters

For details, see a recent version of the *DB2 10 for z/OS Installation and Migration Guide*, GC19-2974.

3

Backup and recovery concepts and functions

In this chapter, we provide the fundamental concepts related to the functions of backup and recovery.

This chapter describes the following topics:

- Image copy backups
- DB2 recovery considerations
- Introduction to recovery
- Planning your recovery plan
- Important topics about DB2 recovery
- Choosing the best recovery for each situation

3.1 Image copy backups

First, we explain the recovery time objective (RTO) and the recovery point objective (RPO):

- Recovery time objective (RTO)

  Maximum allowable downtime. Period of time within which systems, applications, or functions must be recovered after an outage.

- Recovery point objective (RPO)

  Maximum allowable data loss. A point in time (PIT) to which systems and data must be recovered after an outage.

Today's backup media for DB2 is to either tape or disk. The format of that backup is either a traditional full or incremental image copy or a volume-based backup using the BACKUP SYSTEM and RESTORE SYSTEM utilities. Either way, your backup/recovery plans are a subset of the overall business continuity plan. Consequently, the business depends on those backup/recovery plans working and meeting the required RTO/RPO metrics.

In the following topic, we explore backup strategies and look at recommendations and considerations. *Do your homework*. Image copies are the fundamental recovery object for application data. But, do not forget that the catalog, directory, logs, and most important, the bootstrap data set (BSDS) must also be backed up. These objects need a backup strategy, too. Before considering application data, ensure that you have a strategy in place that enables you to recover these vital system components in a variety of recovery scenarios, including disaster recovery.

Image copies that are taken with the DB2 COPY utility can be either a FULL copy, with all pages copied, or an INCREMENTAL copy, which copies only pages that have changed since the last full copy. Image copies can be of table spaces or copyable index spaces. The COPY utility can be used in combination with LISTDEF and TEMPLATE statements to perform the image copy backups. LISTDEF statements allow you to group a set of objects and the TEMPLATE statement allows the target data sets' names to be defined on the contents of the LISTDEF input.

Incremental copies can sometimes take longer than full image copies because incrementals are copying a subset of pages. Full image copies have the performance benefit of sequential prefetch and fast hardware. If only a small proportion of pages changed, it might be quicker to perform an incremental copy. You can use CHANGELIMIT to automatically control whether a full or incremental copy is taken. The keyword, ANY, has been added in DB2 9 as an operand of COPY CHANGELIMIT to allow a user to take a full image copy if any page has changed since the last image copy for large objects with few changed pages.

You can perform some benchmarks to determine at what percentage you should do a full copy as opposed to an incremental. Determining what type of copy to perform, or even if a copy is necessary in the case of static objects, is time-consuming, and can even waste resources. These decisions can easily be configured and automated through Exception Profiles in IBM DB2 Automation Tool for z/OS.

DB2 10 supports the following enhancements in integrating FlashCopy technology in the image copy and recover utilities:

- Support for FlashCopy image copies that are produced at the data set level using the FlashCopy technology and registered in SYSIBM.SYSCOPY as any other image copy. The LOAD, REBUILD INDEX, and REORG utilities can use these image copies to produce inline image copies of data and indexes.

- The exploitation of FlashCopy image copies by the COPY, REORG, LOAD, REBUILD INDEX, and RECOVER utilities.
- The ability to create consistent COPY SHRLEVEL CHANGE image copies.
- FlashCopy 2 is required; otherwise, DB2 reverts to traditional methods. All data sets have to be on storage management subsystem (SMS-managed) volumes.

Your backup strategy should include the following considerations:

- DASD mirroring technologies, Peer-to-Peer Remote Copy Extended Remote Copy (PPRC XRC), are meant for disaster recovery, but they are not the complete answer to backing up your DB2 subsystem. These are “physical” backup strategies. In the case of a “rolling disaster”, where you receive a dropped object or other “logical” error that might have happened two hours ago, which you have only just discovered, it is going to be reflected in your mirrored DASD environment. Most DB2 recoveries are performed due to logically damaged data, not physical errors.
- If no mirroring technology is in place, you must plan for a disaster recovery strategy using either Image Copies (ICs) or a system level backup (SLB) to perform the disaster recovery. Recovery Expert can help users set up a disaster recovery plan using combinations of ICs, SLBs, and mirroring.
- Periodically take a full backup of every object regardless.
- Base your strategy on the business dictated RTO/RPO values and ensure that your strategy meets the business recovery objectives. Remember, as a DBA, it is not for you to decide the RTO/RPO values of various applications. That decision must come from the business; your job is to ensure that the backup strategy meets those RTO/RPO values during a recovery.
- The length of time that you keep your archive log is very important to consider when setting a backup frequency. If your plan is to only keep 30 days of archive logs, you must image copy each table space at least once in that 30-day period. You might decide to set up an exception processing to image copy each space at least twice within your archive log period.
- When your backup strategy is in place, use IBM DB2 Automation Tool for z/OS exception processing to determine if and when an image copy is required. There is no point in creating image copies for every object if no or very few pages have changed. Also, use DB2 Automation Tool to REORG only when required. Use real-time statistics[1] (RTS) in the DB2 Automation Tool exception processing and the administrative task scheduler to automate maintenance and reduce the number of jobs scheduled to run in your batch window.
- Copy your large indexes. Ensure that you have defined your indexes with COPY YES; otherwise, they can only be rebuilt from the corresponding table space. Recovering a large index from an image copy is much quicker than rebuilding it from the table space. Indexes can be recovered in parallel with table space recoveries.
- If you are creating many INCREMENTAL copies, be aware that in a recovery scenario all of these incrementals are required in a recovery situation. Use MERGECOPY to create a single INCREMENTAL or consolidate a FULL copy with INCREMENTALS to create a new FULL copy.
- Perform partition-level copies so that you can recover individual partitions. Partition-level copies and recoveries can be restored in parallel.

[1] For setting real-time statistics, see this website:
http://pic.dhe.ibm.com/infocenter/dzichelp/v2r2/index.jsp?topic=%2Fcom.ibm.db2z10.doc.perf%2Fsrc%2Ftpc%2Fdb2z_setup4realtimestatistics.htm

- Exercise your backup and recovery scenarios frequently. Use cloned environments to test recovery procedures and processes. Document and ensure that the business “signs off on” or approves your recovery procedures as meeting their RPO/RTO requirements. If you have multiple applications in one subsystem, ensure that you understand the recovery priorities associated with each application.

Maintain the SYSCOPY and SYSLGRNX catalog tables by running frequent MODIFY utilities. This utility performs necessary housekeeping by removing old entries.

3.2 DB2 recovery considerations

In this section, we look at the following topics:

- Backup and recovery design
- COPY/RECOVER versus SLB/RECOVER
- DB2 administrative task scheduler
- Backup best practice summary

Backup and recovery design

The majority of DB2 users perform daily/weekly full image copies to tape, possibly to an IBM System Storage® 3592 tape drive, Virtual Tape Server (VTS) or equivalent, and take daily incremental image copies to disk. This particular backup/recovery design recovery strategy is based on a COPY/RECOVER (application level using image copies) methodology rather than a BACKUP/RECOVER (system-level) methodology. This is because a staged recovery must be implemented in order to meet the RTO/RPO metrics of the business.

In a disaster recovery scenario, critical business applications normally need an RTO/RPO of 24/12. Other less important applications have an RTO/RPO of 36/24 or more. The RTO/RPO numbers for each application dictate whether a daily full IC or weekly full IC is taken and also determine the number of daily incremental ICs that are taken. Full copy ICs are written directly to tape media of some sort and can be stacked by using the COPY utility.

Image copy also optimizes performance by allowing DB2 to store image copies using 256 KB blocks. This became available in DB2 9. Using 256 KB instead of 28 KB increases throughput to read and write image copies by between 30% - 50%. The image copies are stacked to optimize the use of the 3592 cartridges and reduce tape mount times in RECOVERY mode. SHRLEVEL CHANGE is used and allows work to continue while the COPY is executing. Although this does not produce a consistent copy, the RECOVER utility recovers to a point in time and produces a recovered object by applying log records to the object after the share level change copy has been restored. To get the best performance when recovering a group of objects, it is better to use the COPY utility to organize the image copies by recovery group. Use the TAPEUNITS and PARALLEL keywords for optimal performance.

VTS is a viable option instead of 3592 because of its disk cache but remember, it is still a tape drive to z/OS and therefore conforms to the same serialization I/O standards as other tape drives.

The COPY utility can also be used to write incremental ICs to disk. If tape was used for incrementals, the RECOVERY time will be compromised because of potential multiple tape drive mounts. During RECOVERY, the utility opens all incremental copies at the same time as the full image copy in order to merge them. Having incrementals on tape, even indirectly, through Data Facility Hierarchical Storage Management System (DFHSM), can cause severe problems. For this reason, the small amount of disk space required for incrementals justifies the use of DASD rather than tape. We can of course write all ICs to DASD and let Hierarchical

Storage Management (HSM) migrate to tape but again that would increase RECOVERY time. It might be slower writing the objects out to tape using COPY than disk, but RECOVERY would then have to wait for HSM to recall. In RECOVERY mode, time is of the essence.

As a rule, if the aggregate size of all the incrementals is above 20% of the size of the full image copy, generate a new full copy. Also, do not perform image copies unless necessary. The criteria is that unless there have been changes, no new image copy is taken. The only exception is a time-based one. Regardless of any changes, a full image copy is always taken every month. These rules can be built into the Exception Profiles of DB2 Automation Tool and can be used to control how and when image copies are taken.

COPY/RECOVER versus SLB/RECOVER

The recovery utility can recover objects from image copies, FlashCopy image copies, a system-level backup, or the log. The largest object is a table or index space, and the smallest object is a page. After recovery, objects are left in a consistent state with any uncommitted work being backed out. This particular design methodology is focused on COPY/RECOVER because an application recover strategy is employed. This is because objects can be grouped as a “recovery group” and several DB2 applications run within the same subsystem. The benefit is that tape mounts are reduced to a minimum and “recovery groups” speed up the recovery process.

As an example, 10,000 objects are distributed over 200 volumes. The 200 volumes are dumped to 50 tapes. If we have a group of 100 corrupted objects, which were dumped in the same recovery group, only one tape mount is required to recover. However, if these 100 objects were distributed across the 200 volumes, it is likely that RECOVER with system-level backup (SLB) would require access to all 50 tapes. System-level backup cannot form “recovery groups”.

DB2 administrative task scheduler

IBM DB2 Tools can be used to optimize the DB2 maintenance jobs that are run during the maintenance window. Not all DB2 subsystems have the same window at the same time. Some critical applications may only have a maintenance window during the weekend. Other applications may have a nightly maintenance window, for example, between 01:00 a.m. and 03:00 a.m.

By using the DB2 Tools package, all the maintenance jobs are scheduled via DB2 Automation Tool, to run under the DB2 administrative task scheduler. Various maintenance windows or schedules are defined to DB2 administrative task scheduler to meet the batch window requirements of different DB2 applications. Also, by using the Exception Profiles provided with DB2 Automation Tool, the number of jobs to run can be determined: REORG (with inline copy), RUNSTATS, and COPIES.

The old methodology that some companies still employ is to schedule maintenance jobs regardless of whether they are required. This takes up a considerable resource and wastes time. Only performing the necessary maintenance reduces the number of jobs that have to be run and therefore maximizes the work that can be done in the maintenance window. The use of DB2 Sort also reduces elapsed and CPU times of the sort phases and also reroutes CPU-intensive work to zIIP hardware.

Backup best practice summary

Consider the following backup best practices:

- Full IC every month, and then only when necessary
- Incremental copies kept on disk, and monitor space
- Consider copy also for critical indexes

- Consider FlashCopy Image Copy for large table spaces and the CONSISTENT option
- COPY/RECOVER design
- DB2 Automation Tool used to control image copy generation
- Archive logs maintained on disk
- Make use of LISTDEF to form recovery groups

Maximize batch window summary

Use IBM DB2 Tools to maximize DB2 maintenance during the DB2 batch window. Also, use SHRLEVEL CHANGE wherever possible to avoid application outage, re-creating consistent PIT RPOs during the RECOVER phase.

By using DB2 Automation Tool, you can determine which objects required maintenance and therefore only schedule the required maintenance. The old method was to schedule maintenance jobs regardless of whether they were required. This took up considerable resources and elapsed time. Only performing the necessary maintenance reduces the number of jobs that are run. Using the DB2 administrative task scheduler also means many maintenance jobs, REORGS, RUNSTATS, COPIES and so on, are automated and scheduled in the background. Using DB2 Sort and High Performance Unload tools also reduces CPU times and improves “back-end” DB2 processing.

3.3 Introduction to recovery

Probably one of the hardest tasks for a DBA or a system programmer is *recovery analysis*. Recovery analysis is the process of asking and answering a series of questions to determine what the failure is, the appropriate list of related objects to recover, and the best method of recovery. You might face some of the following questions as a database administrator responsible for backup and recovery tasks:

- Can a transaction be reversed or does the entire database have to be recovered?
- How can you determine which objects have been affected?
- Do you have the necessary resources to recover to a point in time?
- Are you prepared for a disaster?
- Can you recover your subsystem?
- How much data are you willing to lose?
- Can you recover from a dropped object?

Having the correct resources to perform a recovery is critical. Unfortunately in many cases, this is not addressed until after data is already lost. Database backup and recovery solutions include recovering from a dropped object to bouncing back from a major disaster, and everything in between.

Recoveries that are done manually can be error-prone, time-consuming, and resource-intensive. Having a well-defined business continuity plan helps you minimize the risks of recovery.

Why recovery is not an easy task:

- Lack of DBA experience

  This is not a usual task. There are DB2 for z/OS users who do not run a single recovery in a year.
- Lack of knowledge of the objects that need to be recovered together.
- No recovery plan.
- Lack of documentation.

Reasons to have a recovery plan and when to recover data:

- Application error
- Disk failure
- Human error

The development of backup and recovery procedures at your site is critical to avoid costly and time-consuming data losses.

You should develop procedures to perform these tasks:

- Create a point of consistency
- Recover the DB2 system and application objects to a point of consistency
- Back up the DB2 catalog and directory and your data
- Recover the DB2 catalog and directory and your data
- Recover from out-of-space conditions
- Recover from a hardware or power failure
- Recover from a z/OS component failure
- Recover from an accidental drop of an object
- Recover from a disaster to an offsite location

This list includes common recovery terms used to understand and help build a recovery plan at your site. Most of them apply to disaster or local recovery.

- Business continuity

  Describes the processes and procedures that an organization puts in place to ensure that essential functions can continue during and after a disaster. Business Continuity Planning seeks to prevent interruption of mission-critical services, and to reestablish full functioning as swiftly and smoothly as possible.

- Business impact analysis (BIA)

  Performed to determine the impacts associated with disruptions to specific functions or assets in a firm, including the operating, financial, and legal or regulatory impacts. For example, if billing, receivables, and collections business functions are crippled by inaccessibility to information, the cash flow to the business suffers. Additional risks are that lost customers never return, the business' credit rating may suffer, and significant costs may be incurred for hiring temporary help. Lost revenues, additional costs to recover, fines and penalties, overtime, applications and hardware, lost goodwill, and delayed collection of funds might be the business impact of a disaster.

- Risk analysis

  Identifies important functions and assets that are critical to a firm's operations and then subsequently establishes the probability of a disruption to those functions and assets. When the risk is established, objectives and strategies to eliminate avoidable risks and to minimize impacts of unavoidable risks can be set. A list of critical business functions and assets should first be compiled and prioritized. Then, determine the probability of specific threats to business functions and assets. For example, a certain type of failure may occur once in 10 years. From a risk analysis, you should develop a set of objectives and strategies to prevent, mitigate, and recover from disruptive threats.

- Disaster recovery plan (DRP)

  IT-focused plan designed to restore operability of the target systems, applications, or computer facility at an alternative site after an emergency. A DRP addresses major site disruptions that require site relocation. The DRP applies to major, usually catastrophic, events that deny access to the normal facility for an extended period. Typically, Disaster Recovery Planning involves an analysis of business processes and continuity needs; it may also include a significant focus on disaster prevention.

- Disaster tolerance

  Defines an environment's ability to withstand major disruptions to systems and related business processes. Disaster tolerance at various levels should be built into an environment and can take the form of hardware redundancy, high availability/clustering solutions, multiple data centers, eliminating single points of failure, and distance solutions.

- DR hot site

  A data center facility with sufficient hardware, communications interfaces, and environmentally controlled space capable of providing relatively immediate backup data processing support.

- DR warm site

  A data center or office facility that is partially equipped with hardware, communications interfaces, electricity, and environmental conditioning capable of providing backup operating support.

- DR cold site

  One or more data center or office space facilities equipped with sufficient pre-qualified environmental conditioning, electrical connectivity, communications access, configurable space, and access to accommodate the installation and operation of equipment by critical staff required to resume business operations.

- Bare metal recovery

  Describes the process of restoring a complete system, including system and boot partitions, system settings, applications, and data to their original state at some point prior to a disaster.

- High availability

  Describes a system's ability to continue processing and functioning for a certain period of time, normally a very high percentage of time, for example, 99.999%. High availability can be implemented in your IT infrastructure by reducing any single points-of-failure (SPOF) and using redundant components. Similarly, clustering and coupling applications between two or more systems can provide a highly available computing environment.

- Recovery time objective (RTO)

  Time needed to recover from a disaster or, alternatively, how long you can afford to be without your systems.

- Recovery point objective (RPO)

  Describes the age of the data that you want the ability to restore in the event of a disaster. For example, if your RPO is six hours, you want to be able to restore systems back to the state they were in, as of no longer than six hours ago. To achieve this, you need to be making backups or other data copies at least every six hours. Any data created or modified inside your recovery point objective is either lost or must be recreated during a recovery. If your RPO is to *not* lose any data, synchronous remote copy solutions are your only choice.

- Network recovery objective (NRO)

  Indicates the time required to recover or fail over network operations. Keep in mind that systems level recovery is not fully complete if customers cannot access the application services via network connections. Therefore, the NRO includes the time required to bring online alternative communication links, reconfigure routers and domain name servers (DNS), and alter client system parameters for alternative TCP/IP addresses. Comprehensive network failover planning is of equal importance to data recovery in a Disaster Recovery scenario.

3.4 Planning your recovery plan

We live in a globalized world where your data can be accessed and used at anytime and everywhere. More and more applications are becoming critical to your business. Having a well-designed recovery plan is important to your business continuity management.

You must consider many requirements when you build your plan. The IT and business areas must work together to understand how critical and important each application is and decide the RTO and RPO.

Table 3-1 shows an example or suggestion to define RTO and RPO based on the criticality of the application.

Table 3-1 Example of definition of RTO and RPO based on criticality

Criticality	RTO	RPO
High	< 1 hour	Near zero
Medium	Between 1 and 4 hours	< 2 hours
Low	Between 4 and 12 hours	< 8 hours
None	> 12 hours	> 8 hours

With this definition, you can start mapping your applications to the values defined in the RTO/RPO matrix. Table 3-2 shows an example.

Table 3-2 Mapping applications to defined RTO/RPO

Application	RTO	RPO
App A	High	High
App B	High	Medium
App C	Low	High
App D	None	Low

To help you decide the RPO/RTO of each application, you need to understand the behavior of the business objects of it. A *business object* is a set of tables (and everything is related, such as indexes, views, plans, and packages) related to each other. This relationship normally is placed through referential integrity, forced or not.

The RPO and RTO can help you decide how often you take and save backups of your table. A table that is heavily updated should require intermediate backups during the day to help achieve the RPO. With this mapped, you can decide how often you need an entire database backup and the frequency, if necessary, of each objective, and if it is a full or an incremental backup.

3.5 Important topics about DB2 recovery

To build a recovery plan, it is important understand how DB2 tracks data changes, how changes are recorded in the log, and where you can find this information. This helps with your planning of data and system backup, as well as identifying the best frequency of the backups.

This information is used in recovery situations, and with a well-designed and well-tested plan, it is possible to reduce the downtime of your data.

You need to understand the following important DB2 objects and concepts before building your recovery plan:

Unit of recovery A unit of recovery is the work that changes DB2 data from one point of consistency to another. This work is done by a single DB2 DBMS for an application. The *point of consistency* (also referred to as *sync point* or *commit point*) is a time when all recoverable data that an application program accesses is consistent with other data.

Log The DB2 log registers data changes and significant events as they occur. DB2 writes each log record to the active log, which is a disk data set. When the active log data set is full, DB2 copies its contents to the archive log, which is a disk or a tape data set. This process is called *offloading*.

BSDS The bootstrap data set (BSDS) is a repository of information about the data sets that contains the log. The BDSD contains an inventory of all active and archive log data sets that are known to DB2, an inventory of all recent checkpoint activity that DB2 uses during restart processing, a record of the system backups taken and related checkpoints, a distributed data facility (DDF) communication record, and information about buffer pools.

RBA Relative byte address (RBA) is the offset of a data record or control interval from the beginning of the storage space that is allocated to the data set or file to which it belongs.

LRSN Log record sequence number (LRSN) is an identifier for a log record that is associated with a data sharing member. DB2 uses the LRSN for recovery in the data sharing environment.

DB2 utilities DB2 provides online and offline utilities to read logs, report recovery situations, recover a page set, or recover an entire subsystem.

3.5.1 Understanding DB2 logs

As you make changes to your tables, DB2 writes the appropriate records to the DB2 log, allowing DB2 to back out the changes if a unit of recovery fails, or to apply these changes during a recovery. The DB2 log is mapped onto data sets. Each DB2 system has a predefined fixed set of active log data sets on disk. Log records are first written by DB2 into a log buffer. They are subsequently written from there onto the active log data sets. As an active log data set fills up, DB2 moves onto the next active log data set.

When all active log data sets have been filled, DB2 wraps around to the first active log data set and uses it again. In order not to lose log records that may be required for a backout or recovery, the active log data sets are automatically offloaded as they fill up.

They are offloaded by DB2 to archive log data sets, which may be on disk but frequently reside on tapes. In contrast to the limited set of active log data sets (93 for each of the two log copies), there is effectively an endless set of archive log data sets. However, only those archive logs that are still identified in the BSDS (10,000 with DB2 10) can be utilized for backout or recovery operations.

This process shows how DB2 records information into the log:

1. DB2 registers changes to data and significant events in recovery log records.
2. DB2 processes recovery log records and breaks them into segments, if necessary.
3. Log records are placed sequentially in output log buffers, which are formatted as VSAM control intervals (CIs). Each log record is identified by a continuously increasing RBA in the range 0 to 2^{48}-1, where 2^{48} represents 2 to the 48th power. (In a data sharing environment, a log record sequence number (LRSN) is also used to identify log records).
4. The CIs are written to a set of predefined disk active log data sets, which are used sequentially and recycled.
5. As each active log data set becomes full, its contents are automatically offloaded to a new archive log data set.

Where to find log information

You can find information to recover your data from the following locations:

- SYSIBM SYSCOPY

  SYSIBM.SYSCOPY is a catalog table that contains information about full and incremental image copies. If concurrent updates were allowed when making the copy, the log RBA corresponds to the image copy start time; otherwise, it corresponds to the end time. The RECOVER utility uses the log RBA to look for log information after restoring the image copy. The SYSCOPY catalog table also contains information that is recorded by the COPYTOCOPY utility. SYSCOPY also contains entries with the same kinds of log RBAs that are recorded by the utilities QUIESCE, REORG, LOAD, REBUILD INDEX, RECOVER TOCOPY, and RECOVER TOLOGPOINT.

- SYSIBM.SYSLGRNX

  SYSIBM.SYSLGRNX is a directory table that contains records of the log RBA ranges that are used during each period of time that any recoverable page set is open for update. Those records speed up recovery by limiting the scan of the log for changes that must be applied. In addition to any tool you might have in your installation, DB2 has mechanisms to allow you to recover your information.

3.5.2 DB2 utilities resources

DB2 provides online and offline utilities to help you in a recovery situation. There are utilities from report information about your recovery assets to actual recovering an object to an entire subsystem.

Several important DB2 utilities are described.

REPORT RECOVERY

The REPORT RECOVERY online utility provides information that is needed to recover a table space, an index, or a table space and all of its indexes. You can also use the REPORT utility to obtain recovery information about the catalog.

Use the REPORT RECOVERY utility to find out the following information:

- Which image copies and archive log data sets (if any) you need for the recovery of a table space
- Information that is stored in the SYSIBM.SYSCOPY catalog table
- Log range information from SYSIBM.SYSLGRNX
- Archive log data set from BSDS

DSNJU004 (print log map)

DSNJU004 is an offline utility where you can print your log map to list the following information:

- Log data set name, log RBA association, and log LRSN for both copy 1 and copy 2 of all active and archive log data sets
- Active log data sets that are available for new log data
- Status of all conditional restart control records in the bootstrap data set
- Contents of the queue of checkpoint records in the bootstrap data set
- The communication record of the BSDS, if one exists
- Contents of the quiesce history record
- System and utility time stamps
- Contents of the checkpoint queue
- Archive log command history
- BACKUP SYSTEM utility history
- System CCSID information
- System-level backup information

RECOVER

The RECOVER online utility is used to recover data to the current state or to a previous point in time by restoring a copy, and then by applying log records.

The largest unit of data recovery is the table space or index space; the smallest is the page. You can recover a single object or a list of objects. The RECOVER utility recovers an entire table space, index space, a partition or data set, pages within an error range, or a single page. You can recover data from sequential image copies of an object, a FlashCopy image copy of an object, a system-level backup, or the log. Point-in-time recovery is executed with consistency starting with DB2 10. Recovery to an RBA automatically detects the uncommitted transactions running at the recovery point in time and rolls back their changes on the recovered objects. So after recovery, objects are left in their transactionally consistent state.

There are two main RECOVER phases:

RESTORE The RESTORE phase merges the full image copy with its incremental image copies to form the basis for the restored table space. Pages in an incremental image copy replace the appropriate pages in the full image copy or in earlier incremental image copies.

LOGAPPLY During the LOGAPPLY phase, the appropriate log records are read from the DB2 log and applied to the restored table space.

REBUILD INDEX

The REBUILD INDEX online utility is used to reconstruct indexes from the table that they reference. You can rebuild the entire index, a partition of a partitioned index, or a logical partition of a non-partitioned index. A logical partition always corresponds to a physical partition of the associated partitioned table space. It contains all index entries whose target rows are contained in that physical partition.

With a single REBUILD INDEX statement, you can rebuild multiple indexes, index partitions, or logical partitions. All indexes or index partitions that are rebuilt must be for tables of the same table space.

RESTORE SYSTEM

The RESTORE SYSTEM utility is a special conditional restart that invokes z/OS DFSMShsm to recover a DB2 subsystem or a data sharing group to a previous point in time. To perform the recovery, the utility uses data that is copied by the BACKUP SYSTEM utility. All data sets that you want to recover must be SMS-managed data sets.You can run the RESTORE SYSTEM utility from any member in a data sharing group, even one that is usually quiesced when any backups are taken. Any member in the data sharing group that is active at or beyond the log truncation point must be restarted, and its logs are truncated to the SYSPITR LRSN point. You can specify the SYSPITR LRSN point in the CRESTART control statement of the DSNJU003 (change log inventory) utility. Any data sharing group member that is usually quiesced at the time that the backups are taken and is not active at or beyond the log truncation point does not have to be restarted.

Important: RESTORE SYSTEM does not restore logs; the utility only applies the logs. If you specify BACKUP SYSTEM FULL to create copies of both the data and the logs, you can restore the logs by using DFSMShsm.

DB2 Recovery Expert can perform a RESTORE from an SLB that has been taken with the BACKUP SYSTEM or one of its own SLBs. It automates many processes around the restore. It can restore the log volumes. It automatically clears coupling facility structures when restoring a data sharing system. It also automatically detects when objects need further attention after the log has been applied. This can happen if the log range being applied contains an online reorg for a table space. This table space is in recovery pending after the system restore. DB2 Recovery Expert can detect this situation and automatically generate a RECOVER utility to recover these table spaces from an image copy.

3.5.3 DB2 system-level backup considerations

There are several operational advantages to using DB2 system-level backup (SLB), such as speed and cost reductions, but it requires a sophisticated infrastructure and metadata to manage the DB2 and storage processor coordination.

The following activities can affect the ability to recover an object to a prior point in time from a system-level backup:

- Migrating to new disk storage or redistributing data sets for performance reasons. The movement of data sets for disk management should be restricted or limited when system-level backups are taken. When movement of data sets occurs, a new system-level backup or object-level backup should immediately be taken.
- Using the DFSMShsm migrate and recall feature.
- Using REORG TABLESPACE or LOAD REPLACE.
- Using REORG INDEX or REBUILD INDEX using RECOVER from an image copy or concurrent copy.

All of these considerations are extremely important when restoring your data, especially when using fast copy options that DB2 might not notice when some volume offload or migration occurs.

This infrastructure is called *DB2 system and storage coordinator* (DSS), which can be a person, group, or even a solution (software, hardware, or both).

The DSS should be able to perform these functions:

- DB2 system configuration management facilities that discover your DB2 system and determine the volumes on which it resides. It identifies DB2 layout issues that might conflict with or inhibit an SLB approach. It provides reports about storage volume contents and moves data sets to support your SLB and recovery objectives.
- DSS performs an SLB and restores operations by invoking the correct fast replication functions in the storage system. Backup and restore operations are performed instantaneously as perceived by the application and DB2 system.
- DSS validates that a complete system backup is performed by checking all the storage volumes on which the DB2 system resides. It ensures all catalogs are included in the backup so that the correct recovery functions can be performed.
- Recovery Expert validates that all objects are in the correct state to be backed up. This includes reporting on objects that are in error states when the system backup is performed.
- DSS supports application and object level recovery from a system backup. This allows an SLB to be used for local site application recovery. Object recovery uses data set level fast replication facilities to snap data sets from the backup volumes to the source DB2 volumes. DB2 recovery is performed in parallel to the data set restore process to speed overall recovery time.
- DSS has an integrated metadata repository. It maintains information about the DB2 system, storage volumes, backup volumes, backup time, and corresponding DB2 values that are used to maintain the system and perform DB2 recovery functions.
- DSS supports the tape offloading of disk-based system backups. The tapes can be encrypted and can use Data Facility Data Set Services (DFDSS) or Fast Dump Restore (FDR) to perform the archival process.
- DSS provides DB2 disaster recovery support for all common storage processor vendors.
- DSS provides expanded support for older versions of DB2.

DB2 Recovery Expert, which is described in this section, is a DSS.

3.5.4 DB2 10 utilities enhancements for recovery

DB2 10 includes a variety of improvements to the utilities. The major enhancements for the utilities for recovery are listed:

- Deeper integration with FlashCopy utilities
- RECOVER with BACKOUT YES

This section describes these enhancements.

Deeper integration with FlashCopy

Several utilities integrate the option to use FlashCopy at the table space level, providing improvements in availability and performance.

DB2 10 supports the following enhancements in integrating FlashCopy technology in the image copy and recover utilities:

- Support for FlashCopy image copies that are produced at the data set level by using the FlashCopy technology and that are registered in SYSIBM.SYSCOPY as any other image copy. The LOAD, REBUILD INDEX, and REORG utilities can use these image copies to produce inline image copies of data and indexes.
- The exploitation of FlashCopy image copies by the COPY, REORG, LOAD, REBUILD INDEX, and RECOVER utilities.
- The ability to create consistent COPY SHRLEVEL CHANGE image copies.

FlashCopy 2 is required; otherwise, DB2 reverts to traditional methods. If the target volume is the primary volume on a Metro Mirror remote copy, the Remote Pair FlashCopy function (also called *preserve mirror*) that is provided by microcode and DFSMSdss OA24811 are also required. A new DSNZPARM FLASHCOPY_PPRC (APAR PM26762) controls the options.

Because FlashCopy works at a single controller level, defining separate storage pools for source and target objects is important.

FlashCopy image copy is useful for making image copies of large DB2 objects; however, using FlashCopy does not mean that you can obtain better performance than sequential image copies with small DB2 objects. For small objects, you might get better performance by using an SLB, which avoids the cost of data set allocations.

Because FlashCopy image copy does not support incremental copies, do not consider FlashCopy image copy as a replacement for incremental image copies. Also consider that, if the hardware does not support FlashCopy, the system uses IDCAMS Repro to copy the data set, in which case the only advantage is that Repro does not affect the DB2 buffer pool.

The following DB2 for z/OS utilities can use FlashCopy to create image copies:

- COPY utility
- LOAD utility
- REBUILD INDEX utility
- REORG INDEX utility
- REORG TABLESPACE utility

The following utilities accept the VSAM data sets produced by FlashCopy as input:

- COPYTOCOPY
- DSN1COMP
- DSN1COPY
- DSN1PRNT
- RECOVER

You can recover an object to a specific FlashCopy image copy by specifying the RECOVER utility with the TOCOPY, TOLASTCOPY, or TOLASTFULLCOPY options. If the object is partitioned, you must specify the data set number on the DSNUM parameter in the RECOVERY utility control statement for each partition that is being recovered.

Note: A FlashCopy image copy with consistency consumes more processing resources when the image copy is created, and when the image copy is used for recovery. For recovery to a log point or full recovery, if uncommitted work was backed out from the FlashCopy image copy during consistency processing, recovery requires more analysis of the logs during the preliminary LOGCSR phase (PRELOGC). The preliminary log apply phase (PRELOGA) and the other log phases also require more analysis.

RECOVER with BACKOUT YES

Whether you recover to the current point or a point-in-time recovery, up to DB2 9 for z/OS, the RECOVER utility always identifies the recovery base that fits best and then performs a forward log recovery up to the point on the log that you specify as the target point in time.

The next illustrations show the difference between traditional recovery and recovery with BACKOUT.

Let us assume that the most recent recovery base is about 24 hours old and that the recovery base is a regular sequential image copy as shown in Figure 3-1. Also, assume that you plan to use the RECOVER utility to remove the inserts that started about one hour ago.

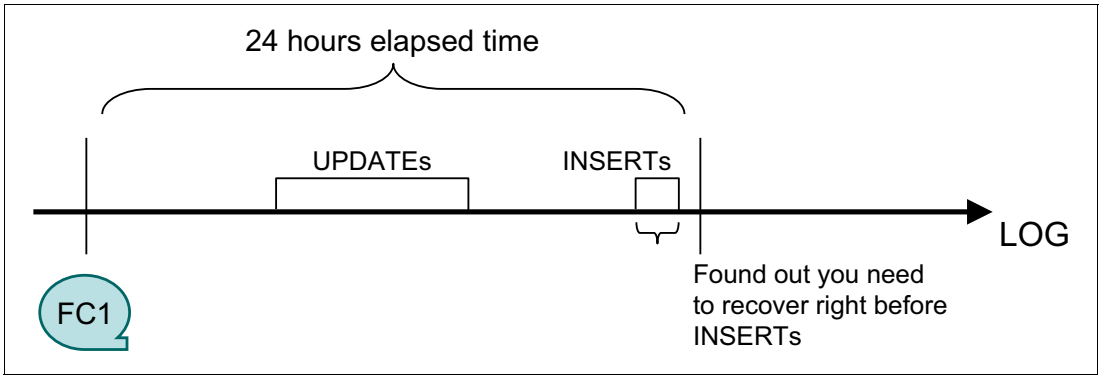

Figure 3-1 RECOVER BACKOUT YES: Base situation

In DB2 9, you can remove the inserts by recovering the entire table space to an RBA that is previous to the first insert or during the inserts, as shown in Figure 3-2. DB2 performs these tasks:

1. Creates a new VSAM cluster.
2. Restores the image copy data set.
3. Reads SYSIBM.SYSLGRNX to find RBA ranges during which the cluster was involved in updates and applies the needed log records reading forward in the log.

All this activity can take a significant amount of elapsed time and resources.

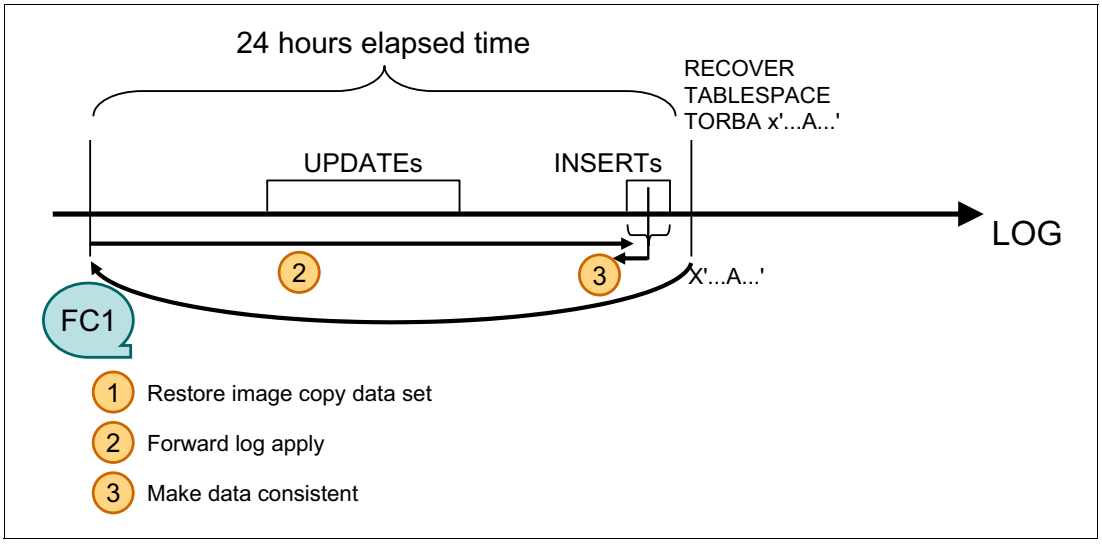

Figure 3-2 RECOVER TABLESPACE without BACKOUT

With DB2 10, you can now back out changes starting from the current operational page set using the RECOVER TABLESPACE statement and the BACKOUT YES option. You can use BACKOUT YES together only with TOLOGPOINT or TORBA. As shown in Figure 3-3, DB2 performs the following steps to handle the BACKOUT YES request:

1. First, DB2 identifies the latest checkpoint that occurred prior to the point in time (RBA) that you specified on the RECOVER TABLESPACE statement.
2. Starting from there, DB2 performs a current status rebuild to identify open units of recovery (URs) that it needs to handle as part of the BACKOUT YES process. We call this the LOGCSR phase.
3. When completed, DB2 lists all open URs in the RECOVER TABLESPACE job output.
4. With this knowledge, DB2 can now back out data up to the RBA that you specified as the point in time RBA and can go back even further, if needed, to make the data consistent.

In our example, the point-in-time RBA x'...A...' was in the middle of the stream of inserts. To make this UR consistent, DB2 continues to back out all inserts that belong to this URID.

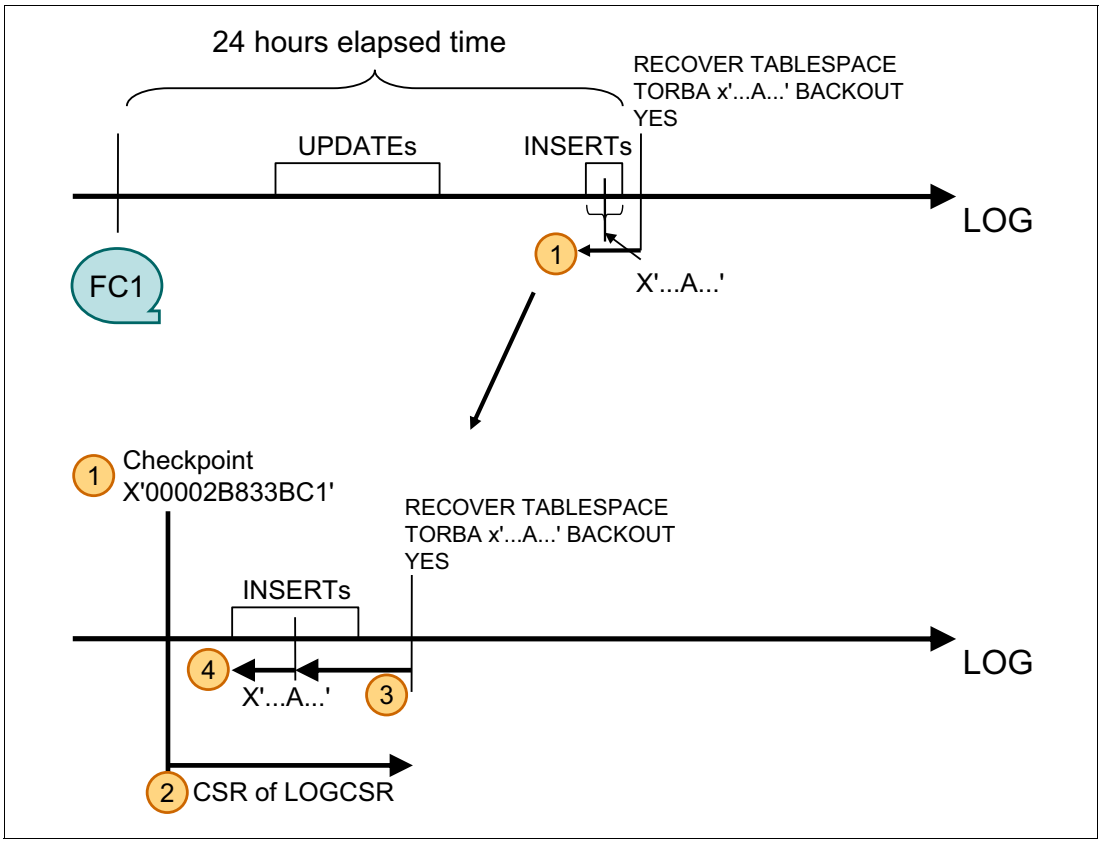

Figure 3-3 RECOVER TABLESPACE... BACKOUT YES

Depending on the amount of time from the last good image copy to the point you want to recover, the use of the option BACKOUT YES can save a considerable amount of execution time because reading and applying changes from a table can take a long time.

3.6 Choosing the best recovery for each situation

Determining what to use for recovering depends on several factors, including time to recovery and how much data you can afford to lose. Because there are many ways to recover an object and its application or referential related dependent objects, your RTO and RPO must be considered.

You can use the following methods to recover an object:

- Restore from an SLB and RECOVER LOGONLY
- RECOVER to a copy, current, or point in time
- Restore of a VSAM data set and RECOVER LOGONLY
- DSN1COPY and RECOVER LOGONLY
- RECOVER or DSN1COPY to an image copy data set and redo SQL
- RECOVER with FLASHCOPY and log apply
- UNDO SQL from current
- RECOVER BACKOUT

All of these options require various resources and should be considered when choosing the best recovery plan.

Table recovery

If you need to recover a single table only, you must consider the relationships to other application objects and whether this procedure might affect data integrity. If it has no relationship, forced or not, probably the easiest method is to perform a simple recovery to a consistent point.

However, you must understand the relationship between all tables and the indexes that are involved with the table that you want to recover. First, you need to find a consistent point for all tables and recover them in order so that you do not affect data integrity. In this case, it is possible to recover a set of tables in parallel, reducing downtime.

You should have the information about tables and their relationships in your recovery plan, especially if the referential integrity is not implemented by DB2. It is also important to understand how often a table is updated or affected by a utility, such as LOAD REPLACE or LOAD RESUME.

Index recovery

Normally, the recovery of an index is executed through the REBUILD INDEX DB2 utility. Depending on its size, consider taking regular index backups together with the related table space. An index must have the COPY YES attribute set in order to allow recovery of the index. In many cases, the recovery of an index can be faster than rebuilding it, depending on the size of the index and the amount of log data to be applied.

DB2 subsystem recovery

For DB2 systems where the entire DB2 is one application (for example, a customer relationship management (CRM) system), the recommended recovery path is to restore the entire system. This ensures the entire application is restored to a common point in time.

Possible scenarios for subsystem corruption include physical data errors, problems with storage volumes containing the DB2 system catalog and data, or an application that has corrupted several table spaces. It is possible that recovering the entire subsystem is faster than attempting to recover a large set of damaged objects.

There are two choices when restoring from an SLB.

If the SLB is a full system backup that includes active logs, archive logs, and BSDS data sets, and you want to restore the DB2 system to the point of the SLB, follow these steps:

1. Stop the DB2 system.
2. Perform a hierarchical storage management (HSM) **frrecov** of the data pool.
3. Perform an HSM **frrecov** of the LOG pool.
4. Restart the DB2 system.

If you want to restore the system and roll forward the log to a point after the SLB, follow these steps:

1. Stop the DB2 system.
2. Create a SYSPITR conditional restart record of the RBA or LRSN of the time to which you want to restore the system.
3. Start the DB2 system.
4. Perform a RESTORE SYSTEM utility.

Check for objects in recover/rebuild pending and recover those objects from the image copies.

Part 2

IBM DB2 Utilities Solution Pack

In this part, we describe the tools included in the Utilities Solution Pack:

4

IBM DB2 High Performance Unload for z/OS

In this chapter, we provide technical information related to High Performance Unload V4.2 (DB2 High Performance Unload), its benefits, and also its integration with DB2 Administration Tool. We show a simple scenario about how to unload with the new Format Internal parameter and loading after an ALTER COLUMN in a large table.

This chapter contains the following topics:

- DB2 High Performance Unload technical overview
- Using DB2 Administration Tool to generate a DB2 High Performance Unload job
- DB2 High Performance Unload and FORMAT INTERNAL
- DB2 High Performance Unload from FlashCopy
- Generating DB2 High Performance Unload jobs from DB2 Automation Tool

4.1 DB2 High Performance Unload technical overview

DB2 High Performance Unload is part of DB2 Utilities Solution Pack and is a high-speed DB2 utility for unloading DB2 tables from either a table space or from an image copy. Tables are unloaded to one or more files based on a format that you specify.

The main DB2 High Performance Unload advantage is that it works outside of DB2, working directly on VSAM or sequential files that contain the table space or image copy data set.

DB2 High Performance Unload uses VSAM buffering capabilities, which allow an entire cylinder to be read in a single I/O, avoiding DB2 buffer pool usage, also.

Regarding parallelism, whenever possible, DB2 High Performance Unload processes requests to unload data from the same table space in parallel. You can create different output files during the same unload process at almost no additional cost. For example, you can unload a list of customers who have payments due this week and another list of customers whose birthdays are on the first day of the week. You can create these lists in a single execution of DB2 High Performance Unload at a fraction of the cost that is required by traditional dual unload executions.

Another advantage of DB2 High Performance Unload is that, starting in V4R2, it is integrated into Tools Customizer, which can facilitate the installation process.

DB2 High Performance Unload also offers some limited compatibility with the syntaxes that other independent software providers have in their products. DB2 High Performance Unload supports the JCL that is used with Fast Unload for DB2, Version 3.1 and the JCL that is used with UNLOAD PLUS for DB2, Version 2.1.01. However, some features of the Fast Unload and UNLOAD PLUS products might be ignored or might be interpreted differently when they are issued by DB2 High Performance Unload. In most cases, the amount of work that is required to convert Fast Unload JCL and UNLOAD PLUS JCL to DB2 High Performance Unload JCL is reduced. DB2 High Performance Unload supports the Fast Unload and UNLOAD PLUS syntaxes only to the extent that DB2 High Performance Unload can perform processing that is like the processing that is described in the Fast Unload and UNLOAD PLUS syntaxes. Many keywords are ignored, and some options are automatically converted to DB2 High Performance Unload syntax.

4.2 Using DB2 Administration Tool to generate a DB2 High Performance Unload job

Our first test involves unloading a table space of about 12,000,000 records with no selective criteria. We use the DB2 Administration Tool to access the DB2 High Performance Unload panels.

First, we access the DB2 Administration Panel and navigate down to locate our table space. Access to the DB2 Administration Panel is through the TSO command:

```
TSO DB2TOOLS
```

We are presented with the selection panel in Figure 4-1 on page 73 from where we can access the Administration Tool.

```
-------------------- DB2 Tools Primary Options Menu -------------------
OPTION ===>
                                                                USERID - ADMR2
Select an option from below.                                    DATE   - 12/04/24
                                                                TIME   - 17:21
     A Admin.       - Administration Tool
     T Automation   - Automation Tool
     C Cloning      - Cloning Tool
     I HPU          - High Performance Unload
     R RE           - Recovery Expert
     U UET          - Utilities Enhancement Tool

     L TCZ          - DB2 Launchpad
     Z TCZ          - Toolkit Customization

     X EXIT         - Exit
```

Figure 4-1 DB2 Tools Primary Option panel

From here, we show the Administration Menu by selecting option A. The panel in Figure 4-2 opens.

```
DB2 Admin -------------- DB2 Administration Menu 10.1.0 ----------------- 17:15
Option ===>

   1 - DB2 system catalog                                DB2 System: DBOA
   2 - Execute SQL statements                            DB2 SQL ID: ADMR2
   3 - DB2 performance queries                           Userid    : ADMR2
   4 - Change current SQL ID                             DB2 Schema: ADMR2
   5 - Utility generation using LISTDEFs and TEMPLATEs   DB2 Rel   : 1015
   P - Change DB2 Admin parameters
  DD - Distributed DB2 systems
   E - Explain
   Z - DB2 system administration
  SM - Space management functions
   W - Manage work statement lists
   X - Exit DB2 Admin

  CM - Change management

Interface to other DB2 products and offerings:


  CP  - DB2 Object Comparison Tool
```

Figure 4-2 DB2 Administration Menu panel

From within the DB2 system catalog, we can navigate down to our table space. In this case, it is table space DSN00037. See Figure 4-3 on page 74.

```
DB2 Admin --------------- DBOA Tables, Views, and Aliases ---- Row 1 to 1 of
Command ===>                                                  Scroll ===> CSR

Commands: GRANT  MIG  ALL
Line commands:
 C - Columns  A - Auth  L - List  X - Indexes  S - Table space  D - Database
 V - Views  T - Tables  P - Plans  Y - Synonyms  SEL - Select prototyping
 ? - Show all line commands

Sel   Name                Schema   T DB Name  TS Name    Cols        Rows Chks
      *                   *        * *        *             *           *    *
----- ------------------- -------- - -------- -------- ------ ----------- ----
      TABLE1              DB2R1    T DSN00037 TABLE1        6    12731474    0
```

Figure 4-3 Administration selection panel

Click on ? to show all line commands. You will be presented with the characters chosen by you in the High Performance Unload integration CLIST, for example, HPU. Using the line command, HPU, against our DSN00037.TABLE1 drives the High Performance Utility ISPF interface. We are presented with the panel in Figure 4-4 on page 75.

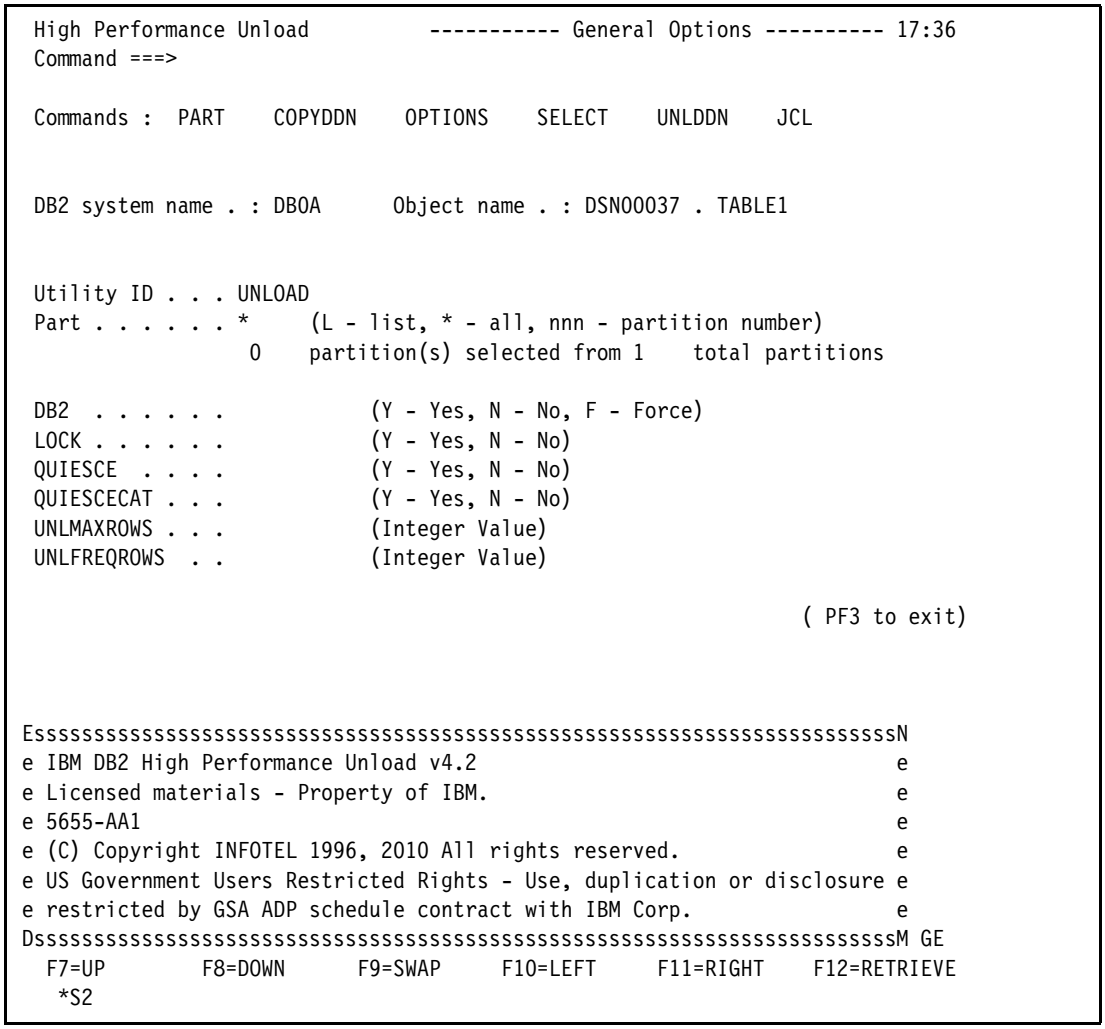

Figure 4-4 High Performance Unload - Example of unload selection from Administration tool

For this scenario, as with all the other scenarios, we get High Performance Unload to create the JCL to run the unload. Again, to reiterate the advantage of using the DB2 Tools, in all the scenarios we demonstrated so far we have not actually written any JCL. Everything has been generated for us by the invoked DB2 Tool product. This provides a level of consistency that reduces potential errors.

The object that we are unloading has already been primed in the panel. We only need to provide a utility ID. Here, we use UNLOAD, and then we enter the UNLDDN command in the command line. This presents us with a panel that allows us to define the data set for the unloaded table space. See Figure 4-5 on page 76.

```
HPU --------- DB0A DSN00037 . TABLE1   - UNLDDN File Parameters ------- 08:46
Command ===>

Commands :  GENERAL    PART    COPYDDN    OPTIONS    SELECT    JCL
                                                     *

           Data set with physical UNLOAD of table space or partition
Data set name . . . . . ADMR3.DSN00037.TABLE1.HPU.UNLOAD
Disposition . . . . . ( N , C , D )  (New/Old/Mod) (Del/Keep/Catlg)
Generic unit  . . . . . SYSDA
Volume serial . . . . .        -        -        -        -
Space units . . . . . . CYLS         (BLKS, TRKS, CYLS)
Primary quantity  . . . 50
Secondary quantity  . . 50
Record format . . . . .              (F, FB, V, VB)
Record length . . . . .
Block size  . . . . . .
SMS
Data class  . . . . . .
Management class  . . .
Storage class . . . . .
for tape unit
Label . . . . . .
                                                                ( PF3 to exit)
```

Figure 4-5 High Performance Unload - Specifying UNLDDN information

In this panel, you can see that we have entered a data set name and the generic unit of VT3590. Our unloaded table space is to be written to a Virtual Tape Server (VTS) tape. The data set name (DSN) also has the disposition of New, Catalog, or Delete. To generate the JCL, we enter JCL on the command line and press Enter. We are presented with the small selection box shown in Figure 4-6 on page 77.

```
HPU --------- DBOA DSN00037 . TABLE1   - UNLDDN File Parameters ------- 17:44
Command ===> JCL

Commands :  GENERAL    PART    COPYDDN    OPTIONS    SELECT    JCL
                                                              EssssssssssssN
                                                              e   EDIT      e
           Data set with physical UNLOAD of table space or    e   SUBmit    e
Data set name . . . . . ADMR2.DSN00037.TABLE1.HPU.UNLOAD      e   SaveJCL   e
Disposition . . . . . ( N , C , D )  (New/Old/Mod) (Del/Kee   e   JobCard   e
Generic unit  . . . . . VT3590                                DssssssssssssM
Volume serial . . . . .        -        -        -        -
Space units . . . . . .              (BLKS, TRKS, CYLS)
Primary quantity  . . .
Secondary quantity  . .
Record format . . . . .              (F, FB, V, VB)
Record length . . . . .
Block size  . . . . . .
SMS
Data class  . . . . . .
Management class  . . .
Storage class . . . . .
for tape unit
Label . . . . . .
                                                                 ( PF3 to exit)
```

Figure 4-6 High Performance Unload - Submitting unload job

We select EDIT so we can see the JCL generated. We are presented with an ISPF edit panel from which we can edit the JCL before submitting the unload job. Our JCL panel is shown in Figure 4-7 on page 78.

```
File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
sssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
EDIT       ADMR2.SC63.SPFTEMP2.CNTL                        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 //ADMR2D JOB (999,POK),'DB2 UTILITY',NOTIFY=&SYSUID,
000002 // MSGCLASS=X,CLASS=A
000003 /*JOBPARM SYSAFF=SC63
000004 //PROCLIB JCLLIB ORDER=DBOAM.PROCLIB
000005 //*************************************************************
000006 //*                   H P U   P R O D U C T                   *
000007 //*************************************************************
000008 //*                                                           *
000009 //* UNLOAD GENERATED BY ADMR2     ON   12/04/24  17:51        *
000010 //*                                                           *
000011 //*************************************************************
000012 //UNLOAD   EXEC PGM=INZUTILB,PARM='DBOA,UNLOAD',
000013 //             REGION=0M
000014 //STEPLIB  DD DSN=INZ.SINZLINK,DISP=SHR
000015 //         DD DSN=DBOAT.SDSNEXIT,DISP=SHR
000016 //         DD DSN=DBOAT.SDSNLOAD,DISP=SHR
000017 //UNLDD    DD DSN=ADMR2.DSN00037.TABLE1.HPU.UNLOAD,
000018 //            UNIT=VT359O,
000019 //            DISP=(NEW,CATLG,DELETE)
000020 //UTPRINT  DD SYSOUT=*
000021 //SYSPRINT DD SYSOUT=*
000022 //SYSIN    DD *
000023 UNLOAD TABLESPACE DSN00037.TABLE1
000024 UNLDDN UNLDD
000025 /*
****** *************************** Bottom of Data****************************
```

Figure 4-7 High Performance Unload - Editing the JCL created by UNLOAD

We submit the job through the normal SUBMIT command. Next, we want to run a standard DB2 UNLOAD utility. This is generated from the DB2 Administration panel using line option U.U against the table space with which we want to work. A workstation list is built and from that we can submit the IBM UNLOAD. Figure 4-8 on page 79 shows an example of the utility statements generated by DB2 Administration.

```
//SYSTSIN  DD *
  DSN SYSTEM(DBOA)
 RUN  PROGRAM(ADBTEP2) PLAN(ADBTEP2) -
   LIB('ADB.V10ROMO.SADBLLIB')  -
   PARMS('/SSID(DBOA) WORKLIST(UNLOAD)   -
     -
 ')
 END
//ADBTEPIN DD *
BINDERROR='MAXE'
,AC='NO'
,RESTREP='NO'
,RESTENV='CKPT'
;
//SYSIN    DD DATA,DLM='@@'
-- Created by ADMR2 on 2012/04/24 at 18:29:06.29
TSODELETE 'ADMR2.DBOA.CNTL.DSN00037.TABLE1.PTALL';
TEMPLATE UTLPUNCH DSN 'ADMR2.DBOA.CNTL.DSN00037.TABLE1.PTALL'
  UNIT SYSDA;
TSODELETE 'ADMR2.DBOA.UNLD.DSN00037.TABLE1.PTALL';
TEMPLATE UTLREC DSN 'ADMR2.DBOA.UNLD.DSN00037.TABLE1.PTALL'
  SPACE CYL
  UNIT VT35;
--#RESTART 1
UNLOAD TABLESPACE DSN00037.TABLE1
  PUNCHDDN(UTLPUNCH)
  UNLDDN(UTLREC);
--#RESTART 2
@@
//*
```

Figure 4-8 DB2 UNLOAD - Example of generated job

Having run both unloads, we can now compare the results, in particular the total CPU and elapsed times found in the job output. Table 4-1 provides the numbers.

Table 4-1 DB2 High Performance Unload - Timings from UNLOAD

Utility	Elapsed time in seconds	CPU time in seconds
DB2 High Performance Unload	53	3.99
DB2 UNLOAD	57	9.12

The numbers reveal a 7% improvement in elapsed time and a 56.25% reduction in CPU time. One of the reasons behind this performance improvement is that DB2 High Performance Unload is working outside of DB2 and does not incur the overhead on internal DB2 processes. It is a simple test meant to show the performance benefits of DB2 High Performance Unload. With larger tables, the improvement can be much greater.

Using DB2 High Performance Unload with selective criteria

How else can we apply some best practices in the use of DB2 High Performance Unload? We try another scenario where we force DB2 High Performance Unload to interface with DB2 in selecting records and compare this with the standard DB2 Utility unload program DSNUTILB. In this scenario, we apply a SELECT statement to extract only records with a particular character value in one of the columns. The total number of records extracted is 4,240,642.

Figure 4-9 shows the JCL and control statements that we used in the two runs. Again, the JCL was generated by the DB2 High Performance Unload and DB2 Administration interfaces. Figure 4-9 shows the generated DB2 High Performance Unload statements.

```
//UNLOAD   EXEC PGM=INZUTILB,PARM='DBOA,UNLOAD',
//             REGION=0M
//STEPLIB  DD DSN=INZ.SINZLINK,DISP=SHR
//         DD DSN=DBOAT.SDSNEXIT,DISP=SHR
//         DD DSN=DBOAT.SDSNLOAD,DISP=SHR
//O11      DD DSN=ADMR2.HPU.SELECT,
//            UNIT=VT3590,
//            DISP=(NEW,CATLG,DELETE)
//UTPRINT  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
UNLOAD TABLESPACE DSN00037.TABLE1

SELECT  *
 FROM
DB2R1.
TABLE1
WHERE T_B_CODE='MT'
OUTDDN ( O11 )
FORMAT DSNTIAUL STRICT
/*
```

Figure 4-9 Using DB2 High Performance Unload with selective criteria

Figure 4-10 shows the JCL for DSNUTILB.

```
//*********************************************************************
//* STEP UNLD1: UNLOAD TABLES
//*********************************************************************
//UNLD1 EXEC DSNUPROC,SYSTEM=DBOA,
//             LIB='DBOAT.SDSNLOAD',
//             UID=''
//SYSPUNCH  DD DSN=ADMR2.DBOA.CNTL.DSN00037.TABLE1,
//             SPACE=(TRK,(5,5),RLSE),
//             UNIT=SYSDA,
//             DISP=(,CATLG,DELETE)
//SYSREC    DD DSN=ADMR2.DBOA.UNLD.DSN00037.TABLE1,
//             DISP=(,CATLG,DELETE),
//             DCB=(BLKSIZE=8192),
//             UNIT=VT3590
//SYSIN  DD  *
UNLOAD TABLESPACE DSN00037.TABLE1
  FROM TABLE
"DB2R1"."TABLE1" WHEN (T_B_CODE='MT')
/*
```

Figure 4-10 Using DSNUTIL UNLOAD with selective criteria

The results from these two runs are shown in Table 4-2 on page 81.

Table 4-2 Comparing Unload performance

Utility	Elapsed time in seconds	CPU time in seconds
DB2 High Performance Unload	19	3.97
DB2 UNLOAD	42	11.31

We see a large reduction of elapsed time with DB2 High Performance Unload against DSNUTILB: an improvement of 54% in elapsed time and 64% in CPU time.

4.3 DB2 High Performance Unload and FORMAT INTERNAL

DB2 High Performance Unload can create outputs in several formats, depending on user needs, including the Internal Format, which is strongly recommended when the source and the target are the same. Performance expectations are to reduce CPU and elapsed time up to 85%. Unloads generated by DB2 High Performance Unload can be used by the DB2 LOAD utility and the reduction in elapsed time can be up to 77% and 56% in CPU time. The difference between UNLOAD and LOAD is because of the number of indexes, and it certainly can vary depending on the underlying data. The more indexes you have on a table decrease the CPU usage and elapsed time percentages on the LOAD.

Both the LOAD and UNLOAD utilities can take advantage of the new FORMAT INTERNAL parameter. DB2 table data is stored on disk in an internal, proprietary DB2 format. During the unload process, this internal format is converted to an external format in the SYSREC data set. During a subsequent load process, this external representation of data is changed back to the DB2 internal format within the data page. Performing this data conversion uses CPU and elapsed time for both the LOAD and UNLOAD utilities. Using DB2 FORMAT INTERNAL can avoid all this CPU and elapsed time consumption, but you must ensure that the column definitions are the same. You do not have to worry about unlike attributes, such as compression, OBID translation, segment sizes, reordered row format, or basic row format.

In DB2 9 for example, FORMAT INTERNAL might be useful when you need to change a table space attribute, such as DSSIZE. Then, unload the data from TABLEA, drop the table space for TABLEA, re-create the table space with the new DSSIZE, and reload the data into TABLEA, re-establishing indexes, authorizations, triggers, and referential integrity (RI) relationships as necessary. In DB2 10, this process can be simplified by using the ALTER command followed by a REORG. This deferred ALTER is only for table spaces defined as a universal table space (UTS) in DB2 10.

For a LOAD statement that uses FORMAT INTERNAL, field specification is not allowed, which considerably simplifies the LOAD statement, enhances usability, aids productivity, and delivers performance savings on both elapsed time and CPU usage.

4.3.1 Running DB2 High Performance Unload with format internal

We describe a scenario about how to use DB2 High Performance Unload to unload and generate Load cards in the case of a table space attribute change, in our case, DSSIZE.

The table space chosen is GLWSEPA, from our test database GLWSAMP. The table space is partitioned by growth, with four partitions, and one table with 806,000 rows.

We use the same job generated on previous pages, but we change the SYSIN as shown in Figure 4-11. The DB2 High Performance Unload job generates the load statements on the GLWSAMP.GLWSEPA.SYSPUNCH data set, calculating SORTKEYS, and stating `RESUME NO SHRLEVEL NONE REPLACE LOG NO.` Users can specify load statements according to the environment's needs by using the LOADOPT card. Also, since the structure of the table to receive the load is the same as the one being unloaded, we are using format internal.

```
//INZHPUCL JOB (999,POK),'RE',
//         REGION=0M,NOTIFY=&SYSUID,
//         MSGCLASS=X,CLASS=T
//PROCLIB JCLLIB ORDER=DBOAM.PROCLIB
/*JOBPARM SYSAFF=SC63
//*************************************************************
//*                    H P U   P R O D U C T                  *
//*************************************************************
//*                                                           *
//* UNLOAD GENERATED BY ADMR3     ON   13/02/25  13:08        *
//*                                                           *
//*************************************************************
//UNLOAD   EXEC PGM=INZUTILB,PARM='DBOA,TESTE'
//STEPLIB  DD DSN=DBTLSP.SINZLINK,DISP=SHR
//         DD DSN=DBOAT.SDSNEXIT,DISP=SHR
//         DD DSN=DBOAT.SDSNLOAD,DISP=SHR
//SYSPUNCH DD DSN=DBOAS.GLWSAMP.GLWSEPA.SYSPUNCH,
//            SPACE=(TRK,(5535,5535)),RECFM=VB,LRECL=109,
//            BLKSIZE=0,
//            UNIT=SYSDA,
//            DISP=(NEW,CATLG,DELETE)
//UTPRINT  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
GLOBAL
OPTIONS LOADINDDN YES;
TEMPLATE OUTFILE DSN DBOAS.&DB..&TS..P&PART. UNIT WORK BUFNO 30
UNLOAD TABLESPACE GLWSAMP.GLWSEPA DB2 NO LOCK NO QUIESCE YES
PARALLELISM 10
SELECT * FROM GLWSAMP.GLWTEPA
 OUTDDN (OUTFILE)
 FORMAT INTERNAL LOADDDN SYSPUNCH
 LOADOPT(SORTKEYS &SORTKEYS RESUME NO SHRLEVEL NONE REPLACE LOG NO)
/*
```

Figure 4-11 DB2 High Performance Unload format internal generated job

We submitted the job, and it finished with rc=0. We analyzed the job output in the next figures. We see that since parallelism 10 was used, all partitions were read at the same time, as shown in Figure 4-12 on page 83.

```
16.31.22 JOB19504  INZXO05 GLWSEPA  PARTITION NO. 1 IS BEING READ
16.31.22 JOB19504  INZXO05 GLWSEPA  PARTITION NO. 2 IS BEING READ
16.31.22 JOB19504  INZXO05 GLWSEPA  PARTITION NO. 4 IS BEING READ
16.31.22 JOB19504  INZXO05 GLWSEPA  PARTITION NO. 3 IS BEING READ
16.31.22 JOB19504  INZX009 GLWSEPA  PARTITION NO. 3 READ, 0 ROWS PROCESSED
16.31.22 JOB19504  INZX009 GLWSEPA  PARTITION NO. 4 READ, 0 ROWS PROCESSED
16.31.22 JOB19504  INZX090 SYS00003 : 0 RECORDS WRITTEN IN 00:00:00, UNLOAD DONE
16.31.22 JOB19504  INZX090 SYS00004 : 0 RECORDS WRITTEN IN 00:00:00, UNLOAD DONE
16.31.22 JOB19504  INZX009 GLWSEPA  PARTITION NO. 2 READ, 0 ROWS PROCESSED
16.31.22 JOB19504  INZX090 SYS00002 : 0 RECORDS WRITTEN IN 00:00:00, UNLOAD DONE
16.31.23 JOB19504  INZX009 GLWSEPA  PARTITION NO. 1 READ, 806717 ROWS PROCESSED
16.31.23 JOB19504  INZX090 SYS00001 : 806717 RECORDS WRITTEN IN 00:00:00, UNLOAD DONE
16.31.23 JOB19504  INZU222I SYS00001, TOTAL NUMBER OF RECORDS WRITTEN 806717
16.31.23 JOB19504  INZU222I SYS00002, TOTAL NUMBER OF RECORDS WRITTEN 0
16.31.23 JOB19504  INZU222I SYS00003, TOTAL NUMBER OF RECORDS WRITTEN 0
16.31.23 JOB19504  INZU222I SYS00004, TOTAL NUMBER OF RECORDS WRITTEN 0
16.31.23 JOB19504  -                                        --TIMINGS (MINS.)-
16.31.23 JOB19504  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOC
16.31.23 JOB19504  -INZHPUCL          UNLOAD      00   4983    .00    .00    .0
```

Figure 4-12 DB2 High Performance Unload job output - Parallelism messages

On the same output, we can find the number of unloaded rows by partition, as shown in Figure 4-13.

```
- TABLESPACE   GLWSAMP.GLWSEPA - DB2 HIGH PERFORMANCE UNLOAD - STATISTICS -
* CREATOR.TABLE               * PART NUM  *   ROWS READ   *   ROWS KEPT   *
*----------------------------*-----------*---------------*---------------*-
* GLWSAMP.GLWTEPA             *         1 *        806717 *        806717 *
* OBID=27                     *         2 *             0 *             0 *
*                             *         3 *             0 *             0 *
*                             *         4 *             0 *             0 *
*----------------------------*-----------*---------------*---------------*-
* TOTAL UNLOAD STATISTICS ....*         4 *        806717 *        806717 *
* INVALID ROWS................*                         0
* NUMBER OF PAGES IN ERROR....*                         0

INZU260I GENERATING LOAD STATEMENT FOR SELECT STARTING AT POS(6, 1)
INZU222I SYS00001, TOTAL NUMBER OF RECORDS WRITTEN 806717
INZU376I SELECT 1 PARTITION 1 NUMBER OF RECORDS WRITTEN 806717
INZU222I SYS00002, TOTAL NUMBER OF RECORDS WRITTEN 0
INZU376I SELECT 1 PARTITION 2 NUMBER OF RECORDS WRITTEN 0
INZU222I SYS00003, TOTAL NUMBER OF RECORDS WRITTEN 0
INZU376I SELECT 1 PARTITION 3 NUMBER OF RECORDS WRITTEN 0
INZU222I SYS00004, TOTAL NUMBER OF RECORDS WRITTEN 0
INZU376I SELECT 1 PARTITION 4 NUMBER OF RECORDS WRITTEN 0
```

Figure 4-13 DB2 High Performance Unload output job - Number of records unloaded/partition

The next step in our scenario is to drop and re-create the table space changing DSSIZE. Before dropping the table space, we use DB2 Administration Tool to generate the Data Definition Language (DDL) statements that are used to re-create all the objects.

Accessing the DB2 Administration Panel, we choose the DB2 system catalog option as shown in Figure 4-14 on page 84.

```
DB2 Admin -------------- DB2 Administration Menu 10.1.0 -----------------
Option ===> 1

   1 - DB2 system catalog                                DB2 System: DBOA
   2 - Execute SQL statements                            DB2 SQL ID: ADMR3
   3 - DB2 performance queries                           Userid    : ADMR3
   4 - Change current SQL ID                             DB2 Schema: ADMR3
   5 - Utility generation using LISTDEFs and TEMPLATEs   DB2 Rel   : 1015
   P - Change DB2 Admin parameters
  DD - Distributed DB2 systems
   E - Explain
   Z - DB2 system administration
  SM - Space management functions
   W - Manage work statement lists
   X - Exit DB2 Admin

  CM - Change management

Interface to other DB2 products and offerings:


  CP  - DB2 Object Comparison Tool
```

Figure 4-14 DB2 Administration Tool system catalog option

We search for our table space GLWSEPA, as shown in Figure 4-15.

```
DB2 Admin --------------------- DBOA System Catalog --------------------- 17:28
Option ===> s


Object options:                                        DB2 System: DBOA
  AO - Authorization options                           DB2 SQL ID: ADMR3
   G - Storage groups                P - Plans
   D - Databases                     L - Collections
   S - Table spaces                  K - Packages
   T - Tables, views, and aliases
   V - Views                         H - Schemas
   A - Aliases                       E - User defined data types
   Y - Synonyms                      F - Functions
   X - Indexes                       O - Stored procedures
   C - Columns                       J - Triggers
   N - Constraints                   Q - Sequences
  DS - Database structures          DSP - DS with plans and packages
 PDC - DB2 pending definition changes

Enter standard selection criteria (Using a LIKE operator, criteria not saved):
Name     ===> GLWSEPA%           >  Grantor  ===>          >
Owner    ===>          >           Grantee  ===>          >
In D/L/H ===>                   >
And/or other selection criteria (option xC shows you columns for option x)
Column   ===>                   >  Operator ===>        Value ===>
```

Figure 4-15 GLWSEPA table space selection

Then, all GLWSEPA table spaces on DB0A are displayed. We type GEN in front of GLWSAMP.GLWSEPA, as shown in Figure 4-16.

```
DB2 Admin ------------------ DB0A Table Spaces --------------- Row 1 to 8 of 8
Command ===>                                                  Scroll ===> PAGE

Commands: GRANT  MIG  DIS  STA  STO  ALL
Line commands:
 T - Tables  D - Database  A - Auth  G - Storage group  ICS - Image copy status
 DIS - Display table space  STA - Start table space  STO - Stop table space
 ? - Show all line commands

Select Name     DB Name   Parts Bpool  L E S I C Tables  Act. pages  Segsz T L
       *        *             * *      * * * * *      *           *      * * *
------ -------- -------- ------ ------ - - - - - ------ ----------- ------ - -
       GLWSEPA  ADMR4         1 BP15   A N A N Y      1        7236     64 G Y
       GLWSEPA  DB2R7T        1 BP15   A N A N Y      1         396     64 G Y
       GLWSEPA  DB2R7TST      1 BP15   A N A N Y      1          36     64 G Y
       GLWSEPA  GLW0A         1 BP15   A N A N Y      1        4858     64 G Y
       GLWSEPA  GLWS002       6 BP15   A N S N Y      1     5577824     64 G Y
       GLWSEPA  GLWS003       6 BP15   A N T N Y      0          -1     64 G Y
GEN    GLWSEPA  GLWSAMP       4 BP15   A N A N Y      1       14418     64 G Y
       GLWSEPA  GLWTEST       1 BP15   A N A N Y      1          36     64 G Y
```

Figure 4-16 GENDDL for GLWSAMP.GLWSEPA

On the next panel, we changed the data set name to DB0AS.GLWSAMP.DDL. DB2 Administration Tool puts the generated DDL in this data set. We also changed Create storage group and Grant use of storage group to N, as shown in Figure 4-17 on page 86.

```
DB2 Admin -------------- DBOA Generate SQL from DB2 catalog ------------- 18:09
Option ===>

 Generate SQL statements for tablespace GLWSAMP.GLWSE >  DB2 System: DBOA
                                                        DB2 SQL ID: ADMR3
                                                                  More:     +
 SQL statement types to be generated from the DB2 catalog:

   CREATE TABLESPACE  . . . Y (Y,N)   GRANT access ON TABLESPACE . Y (Y,N,A,R)
   CREATE TABLE . . . . . . Y (Y,N)   GRANT access ON TABLE  . . . Y (Y,N,A,R)
   CREATE VIEW  . . . . . . Y (Y,N,D) GRANT access ON VIEW . . . . Y (Y,N,A,R)
   CREATE INDEX . . . . . . Y (Y,N)   ALTER TABLE ADD FOREIGN KEY. Y (Y,N,D)
   CREATE SYNONYM . . . . . Y (Y,N)   LABEL ON . . . . . . . . . . Y (Y,N)
   CREATE ALIAS . . . . . . Y (Y,N)   COMMENT ON . . . . . . . . . Y (Y,N)
   CREATE TRIGGER . . . . . Y (Y,N,D) REBIND PLAN/PACKAGE  . . . . Y (Y,N,D)
   CREATE MASK  . . . . . . Y (Y,N)   ALTER TABLE ACTIVATE CONTROL Y (Y,N)
   CREATE PERMISSION  . . . Y (Y,N)
   CREATE STORAGE GROUP . . N (Y,N)   GRANT use OF STORAGE GROUP . N (Y,N,A,R)

 New names/values for generated SQL: (leave blank to use current values)
   Object schema  . . . . .          >  Run SQLID  . . . . . . . .
   Object grantor . . . . .          >
   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: 1015)
   Use Masking  . . . . . . NO          (Yes/No)
   Generate catalog stats . NO          (Yes/No/Only)
     Target cat qualifier .          >  (Default is SYSIBM)
     Statistics tables  . . ALL         (All or Select. Default is All)
   Include DB2 pending chgs             (Yes/No/Alter/Only)

 SQL output data set and execution mode:
   Add to a WSL . . . . . . NO          (Yes/No)
   Data set name  . . . . . DBOAS.GLWSAMP.GLWSEPA.DDL
     Data set disposition . SHR         (OLD, SHR, or MOD)
   Execution mode . . . . . BATCH       (BATCH or TSO)
   Commit statements per  .             (Db, tS, Tb, All, None. Default is All)
   DB2 defaults handling  .             (Keep, or Remove. Default is Keep)
   Prompt to run SQL  . . . NO          (Yes/No. For TSO mode and no WSL)

 DB2 Command output data set:
```

Figure 4-17 GEN options

DB2 Administration Tool generates a batch job, as shown in Figure 4-18 on page 87. The account information on the JCL and region should be changed to meet the defaults of your environment. If you do not have a data set created to store the DDL, you need to edit the JCL to specify the necessary information.

```
//ADMR3D JOB (ACCOUNTING INFO),'DB2 UTILITY',
//*       RESTART=STEPNAME, <== FOR RESTART REMOVE * AND ENTER STEP NAME
//         REGION=XXXXK,NOTIFY=ADMR3,
//         MSGCLASS=X,
//         CLASS=A
//*
//*******************************************************************
//*
//* DB2 ADMIN GENERATED JOB
//*
//*******************************************************************
//*
//*************************************************************ADB2GEN2**
//* STEP GENSQL: GENERATE SQL FROM DB2 CATALOG
//*******************************************************************
//GENSQL   EXEC PGM=IKJEFT01,DYNAMNBR=100
//STEPLIB  DD DISP=SHR,DSN=GOC.V10ROM0.SGOCLLIB
//         DD DISP=SHR,DSN=ADB.V10ROM0.SADBLLIB
//         DD DISP=SHR,DSN=DBOAT.SDSNEXIT
//         DD DISP=SHR,DSN=DBOAT.SDSNLOAD
//         DD DISP=SHR,DSN=DBOAT.SCNKLINK
//         DD DISP=SHR,DSN=DBOAT.SCNKLPA
//         DD DISP=SHR,DSN=ADB.V10ROM0.SADBLINK
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
  DSN SYSTEM(DBOA)
  RUN PROG(ADB2GEN) PLAN(ADB2GEN) -
  PARMS('/REBIND')
  END
/*
//SYSPRINT DD SYSOUT=*
//SQL      DD DSN=DBOAS.GLWSAMP.GLWSEPA.DDL,
//            DISP=(NEW,CATLG,CATLG),
//            SPACE=(CYL,(20,5),RLSE)
//IN       DD *
  DB2SYS   = 'DBOA',
  DB2ALOC  = '',
  DB2SERV  = 'DBOA',
```

Figure 4-18 GEN generated job

We changed the account information and region size in the JCL and submitted the job. It completed with rc=0. On the job output, we can find a summary of the objects that were generated, as shown in Figure 4-19 on page 88.

```
ADB2GEN - Create DDL from catalog info
------------------------------------------------------------------------------
-


          Processing Table space GLWSEPA In Database GLWSAMP

------------------------------------------------------------------------------
-
ADB2GEN - Create DDL from catalog info
------------------------------------------------------------------------------
-


         ADB2GEN - Summary of objects created

         Number of Storage groups created  . . :       0
         Number of Databases created . . . . . :       0
         Number of Table Spaces created  . . . :       1
         Number of Tables created  . . . . . . :       1
         Number of Table Synonyms created  . . :       0
         Number of Table Aliases created . . . :       0
         Number of Labels on Tables created  . :       0
         Number of Comments on Tables created  :       1
         Number of Indexes created . . . . . . :       2
         Number of Comments on Indexes created :       0
         Number of Views created . . . . . . . :       1
         Number of View Synonyms created . . . :       0
         Number of View Aliases created  . . . :       0
         Number of Labels on Views created . . :       0
         Number of Comments on Views created . :       0
         Number of Labels on Aliases created . :       0
         Number of Comments on Aliases created :       0
         Number of Labels on Columns created . :       0
         Number of Comments on Columns created :       0
         Number of Referential constraints . . :       2
         Number of Referential constraints(ext):       0
         Number of Triggers  . . . . . . . . . :       0
         Number of Trigger Comments  . . . . . :       0
         Number of Distinct Types  . . . . . . :       0
         Number of Distinct Type Comments  . . :       0
```

Figure 4-19 GEN summary

If we browse DB0AS.GLWSAMP.GLWSEPA.DDL, we see that the DDL statements were created, as shown in Figure 4-20 on page 89. This data set is edited further to change DSSIZE.

```
-------------------------------------------------------------------------
-- Database=GLWSAMP   Stogroup=DBOAXSG (for partition 1)
-- Table space=GLWSAMP.GLWSEPA
-------------------------------------------------------------------------
--
  SET CURRENT SQLID='DB2R2';
--
  CREATE TABLESPACE GLWSEPA
    IN GLWSAMP
    USING STOGROUP DBOAXSG
    PRIQTY 120 SECQTY 120
    FREEPAGE 0 PCTFREE 5
    GBPCACHE CHANGED
    TRACKMOD YES
    MAXPARTITIONS 1000
    LOGGED
    DSSIZE 4 G
    NUMPARTS 4
    SEGSIZE 64
    BUFFERPOOL BP15
    LOCKSIZE ANY
    LOCKMAX SYSTEM
    CLOSE YES
    COMPRESS NO
    CCSID      EBCDIC
    DEFINE YES
    MAXROWS 255;
--
  COMMIT;
```

Figure 4-20 GEN created data set

Now, we return to DB2 Administration Tool and navigate until we go through the table space options and issue DROP in front of GLWSAMP.GLWSEPA, as shown in Figure 4-21 on page 90.

```
DB2 Admin ------------------ DBOA Table Spaces ------------- Row 1 to 10 of 10
Command ===>                                                  Scroll ===> PAGE

Commands: GRANT  MIG  DIS  STA  STO  ALL
Line commands:
 T - Tables  D - Database  A - Auth  G - Storage group  ICS - Image copy status
 DIS - Display table space  STA - Start table space  STO - Stop table space
 ? - Show all line commands

Select Name     DB Name   Parts Bpool  L E S I C Tables  Act. pages  Segsz T L
       *        *             * *      * * * * *      *           *      * * *
------ -------- -------- ------ ------ - - - - - ------ ----------- ------ - -
       GLWSO01  GLWSAMP       0 BP15   A N A N Y      9          47      4   Y
       GLWSO02  GLWSAMP       0 BP15   A N A N Y      4          18      4   Y
       GLWSO03  GLWSAMP       0 BP15   A N A N Y      1           2     16   Y
       GLWSDPT  GLWSAMP       0 BP15   A N A N Y      1         269     32   Y
       GLWSEMP  GLWSAMP       4 BP15   A N A N Y      1        2537     32 R Y
DROP   GLWSEPA  GLWSAMP       4 BP15   A N A N Y      1       14418     64 G Y
       GLWSPJA  GLWSAMP       0 BP15   A N A N Y      1        5816     32   Y
       GLWSPRJ  GLWSAMP       0 BP15   A N A N Y      1        2565     32   Y
       GLWSSPL  GLWSAMP       4 BP15   A N A N Y      1          40     32 R Y
       XGLW0000 GLWSAMP       4 BP16K0 X N A Y Y      1        1300     32 P Y
```

Figure 4-21 Table Space drop

A confirmation panel is displayed, as shown in Figure 4-22. We only change Display object impact report to NO and press Enter.

```
DB2 Admin -------------------- DBOA Drop Table Space -------------------- 17:01
Command ===>

 DROP TABLESPACE

 Database ===> GLWSAMP  (? to look up)
 Name     ===> GLWSEPA  (? to look up)

 Display Drop Impact Report ===> NO    (Yes, No, or Batch)
```

Figure 4-22 Drop confirmation panel

A drop confirmation message is displayed, as shown in Figure 4-23.

```
DB2 Admin -------------------- DBOA Drop Table Space -------------------- 17:06
Command ===>
DROP stmt executed
 DROP TABLESPACE

 Database ===> GLWSAMP  (? to look up)
 Name     ===> GLWSEPA  (? to look up)

 Display Drop Impact Report ===> NO    (Yes, No, or Batch)
```

Figure 4-23 Drop confirmation message

Now, we change the DDL data set before running it. In our case, we change table space DSSIZE to 5G and NUMPARTS to 2, as you can see on Figure 4-24.

```
--
  SET CURRENT SQLID='DB2R2';
--
  CREATE TABLESPACE GLWSEPA
    IN GLWSAMP
    USING STOGROUP DBOAXSG
    PRIQTY 120 SECQTY 120
    FREEPAGE 0 PCTFREE 5
    GBPCACHE CHANGED
    TRACKMOD YES
    MAXPARTITIONS 1000
    LOGGED
    DSSIZE 2 G
    NUMPARTS 2
    SEGSIZE 64
    BUFFERPOOL BP15
    LOCKSIZE ANY
    LOCKMAX SYSTEM
    CLOSE YES
    COMPRESS NO
    CCSID      EBCDIC
    DEFINE YES
    MAXROWS 255;
--
  COMMIT;
--
```

Figure 4-24 Modified table space DDL

The next step is to run the DDL to re-create the objects. We use DB2 Administration Tool to execute the SQL, as shown in Figure 4-25 on page 92.

```
DB2 Admin -------------- DB2 Administration Menu 10.1.0 ----------------- 17:39
Option ===> 2

   1 - DB2 system catalog                                  DB2 System: DBOA
   2 - Execute SQL statements                              DB2 SQL ID: ADMR3
   3 - DB2 performance queries                             Userid    : ADMR3
   4 - Change current SQL ID                               DB2 Schema: ADMR3
   5 - Utility generation using LISTDEFs and TEMPLATEs     DB2 Rel   : 1015
   P - Change DB2 Admin parameters
  DD - Distributed DB2 systems
   E - Explain
   Z - DB2 system administration
  SM - Space management functions
   W - Manage work statement lists
   X - Exit DB2 Admin

  CM - Change management

Interface to other DB2 products and offerings:


  CP  - DB2 Object Comparison Tool
```

Figure 4-25 Execute SQL statements

On the next panel, we choose the option to Run or Explain SQL statements, as shown in Figure 4-26.

```
DB2 Admin -------------------- Execute SQL Statements ------------------- 17:42
Option ===> 2

   1 - Edit/run SQL statements                             DB2 System: DBOA
   2 - Run or Explain SQL statements                       DB2 SQL ID: ADMR3
   3 - Build SQL SELECT, INSERT, UPDATE or DELETE prototype
   4 - Create/drop/label/comment on objects
   5 - Grant/revoke privileges on objects
```

Figure 4-26 Run or Explain SQL statements

On the next panel, we enter the data set name where the DDL is stored. We also changed the Edit first option to NO in order to run the DDL directly, as shown in Figure 4-27 on page 93.

```
DB2 Admin ---------------- Run or Explain SQL Statements ---------------- 17:56
Option ===> 1


   1 - Run SQL statements from a data set                DB2 System: DBOA
        EDIT first ===> NO  (Yes/No)                     DB2 SQL ID: ADMR3
   2 - Run or Explain SQL located in a program
        Program type ===>   (1=COBOL, 2=PL/I)

ISPF library:
   Project ===>
   Group   ===>           ===>           ===>           ===>
   Type    ===>
   Member  ===>                    (blank for member selection list)

Other partitioned or sequential data set:
   Data Set Name  ===> 'DBOAS.GLWSAMP.GLWSEPA.DDL'
   Volume Serial  ===>             (if not cataloged)

Alternative pre-allocated DD name:
   DD name ===>                    (use ddname(member) for members)
```

Figure 4-27 Running DDL from a data set

On the first run, we received -551, as shown in Figure 4-28, because the GLWSAMP authid did not have authority to create tables. This way, only the table space was created.

```
DB2 Admin --------------------- DB2 Error Display 1 --------------------- 18:35
Command ===>
Rollback done
   SQLCODE : -551                          DSNTIAR CODE :  0

DSNT408I SQLCODE = -551, ERROR:  GLWSAMP DOES NOT HAVE THE PRIVILEGE TO PERFORM
         OPERATION CREATE TABLE ON OBJECT GLWSAMP
DSNT418I SQLSTATE   = 42501 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXODD2 SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = 50  0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'00000032'  X'00000000'  X'00000000'  X'FFFFFFFF'
         X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
```

Figure 4-28 -551 error

On DB2 Administration Tool, we noticed that table space GLWSAMP.GLWSEPA was created, as shown in Figure 4-29 on page 94.

```
DB2 Admin ------------------ DBOA Table Spaces ------------- Row 1 to 10 of 10
Command ===>                                                  Scroll ===> PAGE

Commands: GRANT  MIG  DIS  STA  STO  ALL
Line commands:
 T - Tables  D - Database  A - Auth  G - Storage group  ICS - Image copy status
 DIS - Display table space  STA - Start table space  STO - Stop table space
 ? - Show all line commands

Select Name     DB Name   Parts Bpool  L E S I C Tables  Act. pages  Segsz T L
       *        *             * *      * * * * *      *           *      * * *
------ -------- -------- ------ ------ - - - - - ------ ----------- ------ - -
       GLWSEPA  ADMR4         1 BP15   A N A N Y      1        7236     64 G Y
       GLWSEPA  DB2R7T        1 BP15   A N A N Y      1         396     64 G Y
       GLWSEPA  DB2R7TST      1 BP15   A N A N Y      1          36     64 G Y
       GLWSEPA  GLWOA         1 BP15   A N A N Y      1        4858     64 G Y
       GLWSEPA  GLWDB1        1 BP15   A N A N Y      1        7236     64 G Y
       GLWSEPA  GLWNEW        1 BP15   A N P N Y      1          -1     64 G Y
       GLWSEPA  GLWSOO2       6 BP15   A N A N Y      1     5577824     64 G Y
       GLWSEPA  GLWSOO3       6 BP15   A N T N Y      0          -1     64 G Y
       GLWSEPA  GLWSAMP       2 BP15   A N A N Y      1          -1     64 G Y
       GLWSEPA  GLWTEST       1 BP15   A N A N Y      1          36     64 G Y
```

Figure 4-29 GLWSAMP.GLWSEPA on DB2 Administration Tool

We dropped GLWSAMP.GLWSEPA, as shown in Figure 4-30, before fixing the DDL data set.

```
DB2 Admin ------------------ DBOA Table Spaces --------------- Row 1 to 9 of 9
Command ===>                                                  Scroll ===> PAGE

Commands: GRANT  MIG  DIS  STA  STO  ALL
Line commands:
 T - Tables  D - Database  A - Auth  G - Storage group  ICS - Image copy status
 DIS - Display table space  STA - Start table space  STO - Stop table space
 ? - Show all line commands

Select Name     DB Name   Parts Bpool  L E S I C Tables  Act. pages  Segsz T L
       *        *             * *      * * * * *      *           *      * * *
------ -------- -------- ------ ------ - - - - - ------ ----------- ------ - -
       GLWSEPA  ADMR4         1 BP15   A N A N Y      1        7236     64 G Y
       GLWSEPA  DB2R7T        1 BP15   A N A N Y      1         396     64 G Y
       GLWSEPA  DB2R7TST      1 BP15   A N A N Y      1          36     64 G Y
       GLWSEPA  GLWOA         1 BP15   A N A N Y      1        4858     64 G Y
       GLWSEPA  GLWNEW        1 BP15   A N A N Y      1        9756     64 G Y
       GLWSEPA  GLWSOO2       6 BP15   A N S N Y      1     5577824     64 G Y
       GLWSEPA  GLWSOO3       6 BP15   A N T N Y      0          -1     64 G Y
DROP   GLWSEPA  GLWSAMP       2 BP15   A N T N Y      0          -1     64 G Y
       GLWSEPA  GLWTEST       1 BP15   A N A N Y      1          36     64 G Y
```

Figure 4-30 Second drop on GLWSAMP.GLWSEPA

The next panel displays the table space information. We change the Display Drop Impact Report option to NO and press Enter, as shown in Figure 4-31 on page 95.

```
DB2 Admin -------------------- DBOA Drop Table Space -------------------- 18:41
Command ===>

 DROP TABLESPACE

 Database ===> GLWSAMP  (? to look up)
 Name     ===> GLWSEPA  (? to look up)

 Display Drop Impact Report ===> NO    (Yes, No, or Batch)
```

Figure 4-31 Drop Impact Report option

A confirmation is displayed, as shown in Figure 4-32.

```
DB2 Admin -------------------- DBOA Drop Table Space -------------------- 18:44
Command ===>
DROP stmt executed
 DROP TABLESPACE

 Database ===> GLWSAMP  (? to look up)
 Name     ===> GLWSEPA  (? to look up)

 Display Drop Impact Report ===> NO    (Yes, No, or Batch)
```

Figure 4-32 Drop statement confirmation panel

We change the current SQLID to DB2R2 on the DDL data set, as shown in Figure 4-33 on page 96.

```
--    Table=GLWSAMP.GLWTEPA In GLWSAMP.GLWSEPA
------------------------------------------------------------------------
--
  SET CURRENT SQLID='DB2R2';
--
  CREATE TABLE GLWSAMP.GLWTEPA
     (EMP_NO               INTEGER NOT NULL,
      PROJ_NO              INTEGER NOT NULL,
      ACT_NO               INTEGER NOT NULL,
      EMPTIME              DECIMAL(5, 2) WITH DEFAULT NULL,
      EMSTDATE             DATE NOT NULL,
      EMENDATE             DATE WITH DEFAULT NULL,
      CREATED_TS           TIMESTAMP (6) WITHOUT TIME ZONE NOT NULL
        WITH DEFAULT,
      CREATED_BY           CHAR(8) FOR SBCS DATA NOT NULL
        WITH DEFAULT,
      UPDATED_TS           TIMESTAMP (6) WITHOUT TIME ZONE NOT NULL
        WITH DEFAULT,
      UPDATED_BY           CHAR(8) FOR SBCS DATA NOT NULL
        WITH DEFAULT,
      CONSTRAINT GLWPEPA
      PRIMARY KEY (PROJ_NO,
                   ACT_NO,
                   EMP_NO))
    IN GLWSAMP.GLWSEPA
    PARTITION BY SIZE
    AUDIT NONE
    DATA CAPTURE NONE
    CCSID      EBCDIC
    NOT VOLATILE
    APPEND NO  ;
--
```

Figure 4-33 Changing SQLID

Now, we use DB2 Administration Tool again to run the DDL, as shown in Figure 4-34 on page 97.

```
DB2 Admin ---------------- Run or Explain SQL Statements ---------------- 12:01
Option ===> 1


   1 - Run SQL statements from a data set               DB2 System: DBOA
        EDIT first ===> NO  (Yes/No)                    DB2 SQL ID: ADMR3
   2 - Run or Explain SQL located in a program
        Program type ===>   (1=COBOL, 2=PL/I)

ISPF library:
   Project ===>
   Group   ===>           ===>           ===>           ===>
   Type    ===>
   Member  ===>                    (blank for member selection list)

Other partitioned or sequential data set:
   Data Set Name  ===> 'DBOAS.GLWSAMP.GLWSEPA.DDL'
   Volume Serial  ===>             (if not cataloged)

Alternative pre-allocated DD name:
   DD name ===>                    (use ddname(member) for members)
```

Figure 4-34 DDL second run

We received a message after running the DDL, as shown in Figure 4-35. It is a set statement confirmation message. We check the DB2 Administration Tool panel afterward to verify that all objects were created.

```
DB2 Admin ---------------- Run or Explain SQL Statements ---------------- 12:01
Option ===>
SET stmt executed

   1 - Run SQL statements from a data set               DB2 System: DBOA
        EDIT first ===> NO  (Yes/No)                    DB2 SQL ID: DB2R2
   2 - Run or Explain SQL located in a program
        Program type ===>   (1=COBOL, 2=PL/I)

ISPF library:
   Project ===>
   Group   ===>           ===>           ===>           ===>
   Type    ===>
   Member  ===>                    (blank for member selection list)

Other partitioned or sequential data set:
   Data Set Name  ===> 'DBOAS.GLWSAMP.GLWSEPA.DDL'
   Volume Serial  ===>             (if not cataloged)

Alternative pre-allocated DD name:
   DD name ===>                    (use ddname(member) for members)
```

Figure 4-35 DDL execution confirmation message

We can check that the GLWSAMP.GLWSEPA table space was created via the DB2 Administration Tool, as shown in Figure 4-36. We can type T in front of it to check the related tables.

```
DB2 Admin ------------------ DBOA Table Spaces --------------- Row 1 to 9 of 9
Command ===>                                                  Scroll ===> PAGE

Commands: GRANT  MIG  DIS  STA  STO  ALL
Line commands:
 T - Tables  D - Database  A - Auth  G - Storage group  ICS - Image copy statu
 DIS - Display table space  STA - Start table space  STO - Stop table space
 ? - Show all line commands

Select Name     DB Name   Parts Bpool  L E S I C Tables  Act. pages  Segsz T L
       *        *             * *      * * * * *      *           *      * * *
------ -------- -------- ------ ------ - - - - - ------ ----------- ------ - -
       GLWSEPA  ADMR4         1 BP15   A N A N Y      1        7236     64 G Y
       GLWSEPA  DB2R7T        1 BP15   A N A N Y      1         396     64 G Y
       GLWSEPA  DB2R7TST      1 BP15   A N A N Y      1          36     64 G Y
       GLWSEPA  GLWOA         1 BP15   A N A N Y      1        4858     64 G Y
       GLWSEPA  GLWNEW        1 BP15   A N P N Y      1        9756     64 G Y
       GLWSEPA  GLWSOO2       6 BP15   A N S N Y      1     5577824     64 G Y
       GLWSEPA  GLWSOO3       6 BP15   A N T N Y      0          -1     64 G Y
T      GLWSEPA  GLWSAMP       2 BP15   A N A N Y      1          -1     64 G Y
       GLWSEPA  GLWTEST       1 BP15   A N A N Y      1          36     64 G Y
```

Figure 4-36 GLWSAMP.GLWSEPA on the catalog

We can see that table GLWTEPA was also created, as shown in Figure 4-37. We can type X in front of the table, to see the related indexes on the next panel.

```
DB2 Admin --------------- DBOA Tables, Views, and Aliases ---- Row 1 to 1 of 1
Command ===>                                                  Scroll ===> PAGE

Commands: GRANT  MIG  ALL
Line commands:
 C - Columns  A - Auth  L - List  X - Indexes  S - Table space  D - Database
 V - Views  T - Tables  P - Plans  Y - Synonyms  SEL - Select prototyping
 ? - Show all line commands

Sel   Name                Schema   T DB Name  TS Name    Cols        Rows Chks C
      *                   *        * *        *             *           *    * *
----- ------------------ -------- - -------- -------- ------ ----------- ---- -
X     GLWTEPA             GLWSAMP  T GLWSAMP  GLWSEPA      10          -1    0
```

Figure 4-37 GLWTEPA table on catalog

The indexes were also successfully created, as shown in Figure 4-38 on page 99.

```
DB2 Admin -------------------- DBOA Indexes ----------------- Row 1 to 2 of 2
Command ===>                                                  Scroll ===> PAGE

Commands: DIS  STA  STO  ALL XSPACE
Line commands:
 T - Tables  D - Database  G - Storage group  P - Plans  C - Columns
 DIS - Display index space  STA - Start index space  STO - Stop index space
 ? - Show all line commands

                         Index                        Table             C C C C
Select Index Name        Schema   Table Name          Schema   U   Cols G D L M
       *                 *        *                   *        *      * * * * *
------ ----------------- -------- ------------------ -------- - ------ - - - -
       GLWXEPA1          GLWSAMP  GLWTEPA             GLWSAMP  P      3 Y Y Y N
       GLWXEPA2          GLWSAMP  GLWTEPA             GLWSAMP  D      2 N N Y N
```

Figure 4-38 Indexes on the catalog

Since all related objects were created, we can work on the LOAD job. First, we analyze the SYSPUNCH data set that was created by the DB2 High Performance Unload job. The LOAD statements were created by using PART as shown in Figure 4-39, but since our table space is PBG, no parallelism is accepted for LOAD, at least until DB2 version 10.

```
TEMPLATE U0000001
DSN('DBOAS.GLWSAMP.GLWSEPA.P&PART.')
DISP(OLD,KEEP,KEEP)

LOAD DATA
 SORTKEYS  3226868 RESUME NO  SHRLEVEL NONE  REPLACE  LOG NO
FORMAT INTERNAL

INTO TABLE GLWSAMP.GLWTEPA
PART (1)
INDDN U0000001


INTO TABLE GLWSAMP.GLWTEPA
PART (2)
INDDN U0000001


INTO TABLE GLWSAMP.GLWTEPA
PART (3)
INDDN U0000001


INTO TABLE GLWSAMP.GLWTEPA
PART (4)
INDDN U0000001
```

Figure 4-39 SYSPUNCH generated by DB2 High Performance Unload job

This way, we did not use the SYSPUNCH data set as SYSIN and only copied the control cards to our load batch job, as shown in Figure 4-40.

```
//*********************************************************************
//* STEP LOAD: LOAD TABLE FROM USER SPECIFIED SOURCE
//*********************************************************************
//LOAD EXEC DSNUPROC,SYSTEM=DBOA,
//             LIB='DBOAT.SDSNLOAD',
//             LIB1='DBOAT.SDSNEXIT',
//             UID='CARLOS'
//DSNUPROC.SYSREC DD DISP=SHR,
//             DSN=DBOAS.GLWSAMP.GLWSEPA.P00001
//                DD DISP=SHR,
//             DSN=DBOAS.GLWSAMP.GLWSEPA.P00002
//DSNUPROC.SYSUT1 DD DSN=DBOAS.TEST.SYSUT1,
//             DISP=(MOD,DELETE,CATLG),
//             SPACE=(CYL,(100,5),RLSE),
//             UNIT=SYSDA
//DSNUPROC.SORTOUT DD DSN=DBOAS.TEST.SORTOU,
//             DISP=(MOD,DELETE,CATLG),
//             SPACE=(CYL,(100,5),RLSE),
//             UNIT=SYSDA
//DSNUPROC.SYSMAP DD DSN=DBOAS.TEST.SYSMAP,
//             DISP=(MOD,DELETE,CATLG),
//             SPACE=(CYL,(100,5),RLSE),
//             UNIT=SYSDA
//DSNUPROC.SYSIN DD *
LOAD DATA
 SORTKEYS  3227256 RESUME NO  SHRLEVEL NONE  REPLACE  LOG NO
FORMAT INTERNAL
INTO TABLE GLWSAMP.GLWTEPA
//*
```

Figure 4-40 Load job with control cards generated by DB2 High Performance Unload

The job is submitted and completes with rc=04. Analyzing the job output, we noticed that the load worked correctly, as shown in Figure 4-41.

```
- (RE)LOAD PHASE STATISTICS - TOTAL NUMBER OF RECORDS LOADED=806717 FOR

RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS PROCESSED=806717
RE)LOAD PHASE COMPLETE, ELAPSED TIME=00:00:03
 - SORTBLD PHASE STATISTICS - NUMBER OF KEYS=806717 FOR INDEX GLWSAMP.GLWXEPA2
 - SORTBLD PHASE STATISTICS - NUMBER OF KEYS=806717 FOR INDEX GLWSAMP.GLWXEPA1
```

Figure 4-41 Output messages

We can confirm the load records count by issuing the count command in front of table GLWSAMP.GLWTEPA via the DB2 Administration Tool, as shown in Figure 4-42 on page 101.

```
DB2 Admin --------------- DBOA Tables, Views, and Aliases ---- Row 1 to 1 of 1
Command ===>                                                  Scroll ===> PAGE

Commands: GRANT  MIG  ALL
Line commands:
 C - Columns  A - Auth  L - List  X - Indexes  S - Table space  D - Database
 V - Views  T - Tables  P - Plans  Y - Synonyms  SEL - Select prototyping
 ? - Show all line commands

Sel   Name               Schema   T DB Name  TS Name    Cols        Rows Chks C
      *                  *        * *        *             *           *    * *
----- ------------------ -------- - -------- -------- ------ ----------- ---- -
COUNT GLWTEPA            GLWSAMP  T GLWSAMP  GLWSEPA      10          -1    0
```

Figure 4-42 Count on GLWSAMP.GLWTEPA

The count result of 806,717 rows, as shown in Figure 4-43, is exactly what was unloaded and loaded after the table space changes.

```
DB2 Admin -- Browse Result of SQL Select               ------- Row 1 to 1 of 1
Command ===>                                                  Scroll ===> PAGE

L OWNER   NAME           ROWS
  *       *                 *
- ------- ------- -----------
  GLWSAMP GLWTEPA      806717
```

Figure 4-43 Count results

4.4 DB2 High Performance Unload from FlashCopy

In DB2 10 with FlashCopy support, the CPU and elapsed time cost of copy is driven down to virtually zero. FlashCopy provides significant elapsed time and CPU savings. Before running our scenario, we look at some FlashCopy considerations:

- Ensure that data resides on FlashCopy-enabled DASD to avoid slow DFSMSdss copy. A slow copy would affect the duration of switch phase for SHRLEVEL CHANGE REORG and adversely affect elapsed time.
- No incremental copy is permitted after a FlashCopy image copy (FCIC). Incremental image copies are only permitted after a standard image copy.
- Do not create transaction-consistent image copies in DB2 10 unnecessarily. There is a slightly higher setup cost so you may see increased elapsed time for very small data sets.
- DB2 10 consistent ICs can be taken without blocking the access of applications to the tables. This is very important for availability.
- During a consistent FCIC, the copy process does not wait for blockers, so the benefits are no application outage and no QUIESCE.
- All objects must be storage management subsystem (SMS)-managed residing on FlashCopy 2 volumes. There are two versions of FlashCopy: 1 and 2. FlashCopy 1 is older and backs up data by volume level. FlashCopy 2 is newer and can back up by both volume and data set level. FlashCopy 2 is also much faster.

- The Copy utility uses RTS CHANGELIMIT for improved data set performance. The CHANGELIMIT option is not available for COPY INDEXSPACE. COPY does not support incremental image copy for an index space.
- CHANGELIMIT is not supported with FlashCopy Image Copy. There is no concept of an incremental FlashCopy to only copy changed pages.

4.4.1 Running DB2 High Performance Unload from a FlashCopy

Next, we show a scenario that illustrates how to run DB2 High Performance Unload from a FlashCopy image copy.

The first step is to run the FlashCopy JCL, as shown in Figure 4-44. The SHAALOAD and SFECLOAD libraries are in STEPLIB because jobs were generated with IBM DB2 Automation Tool for z/OS.

```
//STEPLIB  DD DSN=DBTLSP.SHAALOAD,DISP=SHR
//         DD DSN=DBTLSP.SFECLOAD,DISP=SHR
//         DD DSN=DBOAT.SDSNEXIT,DISP=SHR
//         DD DSN=DBOAT.SDSNLOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//*
//SYSIN    DD *
   TEMPLATE C1FLO001
            UNIT     SYSDA
            DSN              'DBOAS.&DB..&SN..T&TIME..P&DSNUM..IC'
            DISP     (NEW,CATLG,DELETE)

   LISTDEF CPY001U1
           INCLUDE TABLESPACE GLWSAMP.GLWNEWTS
           INCLUDE TABLESPACE GLWSAMP.GLWSDPT
           INCLUDE TABLESPACE GLWSAMP.GLWSEMP
           INCLUDE TABLESPACE GLWSAMP.GLWSEPA
           INCLUDE TABLESPACE GLWSAMP.GLWSPJA
           INCLUDE TABLESPACE GLWSAMP.GLWSPRJ
           INCLUDE TABLESPACE GLWSAMP.GLWSSPL
           INCLUDE TABLESPACE GLWSAMP.GLWS001
           INCLUDE TABLESPACE GLWSAMP.GLWS002
           INCLUDE TABLESPACE GLWSAMP.GLWS003
           INCLUDE TABLESPACE GLWSAMP.XGLW0000

   COPY LIST CPY001U1
        FULL        YES

        SHRLEVEL       REFERENCE
        SCOPE          ALL
        FLASHCOPY YES
        FCCOPYDDN (C1FLO001)

/*
//*
```

Figure 4-44 FlashCopy JCL

We submitted the job and it completed with rc=0, as shown in Figure 4-45.

```
13.49.24 JOB21159 ---- THURSDAY,  14 MAR 2013 ----
13.49.24 JOB21159  IRR010I  USERID ADMR3    IS ASSIGNED TO THIS JOB.
13.49.25 JOB21159  ICH70001I ADMR3    LAST ACCESS AT 13:48:15 ON THURSDAY,
MARCH
13.49.25 JOB21159  $HASP373 ADMR3LD  STARTED - INIT 1    - CLASS A - SYS SC63
13.49.25 JOB21159  IEF403I ADMR3LD - STARTED - TIME=13.49.25 - ASID=0020 - SC63
13.49.32 JOB21159  -                                          --TIMINGS
(MINS.)--
13.49.32 JOB21159  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB
CLOCK
13.49.32 JOB21159  -ADMR3LD           IMC00102    00   8103    .00    .00
.12
13.49.32 JOB21159  IEF404I ADMR3LD - ENDED - TIME=13.49.32 - ASID=0020 - SC63
13.49.32 JOB21159  -ADMR3LD  ENDED.  NAME-RESDENT              TOTAL CPU TIME=
13.49.32 JOB21159  $HASP395 ADMR3LD  ENDED
```

Figure 4-45 FlashCopy output

On the job output, we also can find the generated VSAM FlashCopy data sets, as shown in Figure 4-46.

```
ADR454I (001)-DDDS (01), THE FOLLOWING DATA SETS WERE SUCCESSFULLY PROCESSED
PAGE 0004     5695-DF175  DFSMSDSS V1R13.0 DATA SET SERVICES     2013.073 13:49
                          DBOAX.DSNDBC.GLWSAMP.GLWNEWTS.I0001.A001
                          DBOAX.DSNDBC.GLWSAMP.GLWNEWTS.I0001.A002
                          DBOAD.DSNDBC.GLWSAMP.GLWSDPT.I0001.A001
                          DBOAD.DSNDBC.GLWSAMP.GLWSEMP.I0001.A001
                          DBOAD.DSNDBC.GLWSAMP.GLWSEMP.I0001.A002
                          DBOAD.DSNDBC.GLWSAMP.GLWSEMP.I0001.A003
                          DBOAD.DSNDBC.GLWSAMP.GLWSEMP.I0001.A004
                          DBOAX.DSNDBC.GLWSAMP.GLWSEPA.I0001.A001
                          DBOAX.DSNDBC.GLWSAMP.GLWSEPA.I0001.A002
                          DBOAD.DSNDBC.GLWSAMP.GLWSPJA.I0001.A001
                          DBOAD.DSNDBC.GLWSAMP.GLWSPRJ.I0001.A001
                          DBOAD.DSNDBC.GLWSAMP.GLWSSPL.J0001.A001
                          DBOAD.DSNDBC.GLWSAMP.GLWSSPL.J0001.A002
                          DBOAD.DSNDBC.GLWSAMP.GLWSSPL.J0001.A003
                          DBOAD.DSNDBC.GLWSAMP.GLWSSPL.J0001.A004
                          DBOAD.DSNDBC.GLWSAMP.GLWS001.J0001.A001
                          DBOAD.DSNDBC.GLWSAMP.GLWS002.J0001.A001
                          DBOAD.DSNDBC.GLWSAMP.GLWS003.J0001.A001
                          DBOAD.DSNDBC.GLWSAMP.XGLW0000.J0001.A001
                          DBOAD.DSNDBC.GLWSAMP.XGLW0000.J0001.A002
                          DBOAD.DSNDBC.GLWSAMP.XGLW0000.J0001.A003
                          DBOAD.DSNDBC.GLWSAMP.XGLW0000.J0001.A004
ADR006I (001)-STEND(02), 2013.073 13:49:32 EXECUTION ENDS
ADR013I (001)-CLTSK(01), 2013.073 13:49:32 TASK COMPLETED WITH RETURN CODE 0000
```

Figure 4-46 FlashCopy generated data sets

The next step is to run the DB2 High Performance Unload job. We selected table space GLWSAMP.GLWSPJA only in our example. Since the related last image copy is the

FlashCopy that is shown in Figure 4-46 on page 103, we used COPYDDN LAST_IC, as shown in Figure 4-47.

```
//INZHPUCL JOB (999,POK),'RE',
//         REGION=0M,NOTIFY=&SYSUID,
//         MSGCLASS=X,CLASS=T
//PROCLIB JCLLIB ORDER=DBOAM.PROCLIB
/*JOBPARM SYSAFF=SC63
//*************************************************************
//*                    H P U   P R O D U C T                  *
//*************************************************************
//*                                                           *
//* UNLOAD GENERATED BY ADMR3     ON   13/02/25  13:08        *
//*                                                           *
//*************************************************************
//UNLOAD   EXEC PGM=INZUTILB,PARM='DBOA,TESTE'
//STEPLIB  DD DSN=DBTLSP.SINZLINK,DISP=SHR
//         DD DSN=DBOAT.SDSNEXIT,DISP=SHR
//         DD DSN=DBOAT.SDSNLOAD,DISP=SHR
//SYSPUNCH DD DSN=DBOAS.GLWSAMP.GLWSEPA.SYSPUNCH.NEW,
//            SPACE=(TRK,(5535,5535)),RECFM=VB,LRECL=109,
//            BLKSIZE=0,
//            UNIT=SYSDA,
//            DISP=(NEW,CATLG,DELETE)
//UTPRINT  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
GLOBAL
OPTIONS LOADINDDN NO;
TEMPLATE OUTFILE DSN DBOAS.&DB..&TS..UNLOAD UNIT WORK BUFNO 30
UNLOAD TABLESPACE GLWSAMP.GLWSPJA DB2 NO LOCK NO QUIESCE NO
COPYDDN LAST_IC
SELECT * FROM GLWSAMP.GLWTPJA
 OUTDDN (OUTFILE)
 FORMAT INTERNAL LOADDDN SYSPUNCH
 LOADOPT(SORTKEYS &SORTKEYS RESUME NO SHRLEVEL NONE REPLACE LOG NO)
/*
```

Figure 4-47 DB2 High Performance Unload JCL

We submitted the job and it completed with rc=0, using the FlashCopy as input, as shown in Figure 4-48 on page 105.

```
J E S 2  J O B  L O G  --  S Y S T E M  S C 6 3  --  N O D E  W T S C P L X 2

--- FRIDAY,    15 MAR 2013 ----
IRR010I  USERID ADMR3    IS ASSIGNED TO THIS JOB.
ICH70001I ADMR3    LAST ACCESS AT 13:55:44 ON FRIDAY, MARCH 15, 2013
$HASP373 INZHPUCL STARTED - INIT TWS  - CLASS T - SYS SC63
IEF403I INZHPUCL - STARTED - TIME=13.57.29 - ASID=0047 - SC63
IGD01008I STORCLAS SET TO
IGD01010I SG ACS GETS CONTROL &ACSENVIR=ALLOC
IGD01010I &STORCLAS = DBOASEQ
IGD01010I &STORGRP = DBOASEQ
INZX081 GLWSPJA  FLASHCOPY IS BEING READ
INZX006 GLWSPJA  TABLESPACE UNLOAD PHASE STARTED
INZX089 SYS00001 : 269262 RECORDS WRITTEN IN 00:00:00, UNLOAD DONE
INZU222I SYS00001, TOTAL NUMBER OF RECORDS WRITTEN 269262
-                                        --TIMINGS (MINS.)--
----PAG
-JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK   SERV  PG   PAGE
-INZHPUCL          UNLOAD      00   2638    .00    .00    .01   7887   0      0
IEF404I INZHPUCL - ENDED - TIME=13.57.30 - ASID=0047 - SC63
-INZHPUCL ENDED.  NAME-RE                  TOTAL CPU TIME=   .00  TOTAL
ELAPSED
$HASP395 INZHPUCL ENDED
```

Figure 4-48 DB2 High Performance Unload job output

We press PF8 on the job output to show more details, as shown in Figure 4-49 on page 106.

```
INZU281I - UNLOAD STARTING AT POS(4, 1)

INZU277I - PROCESSING UNLOAD 00001 FROM TABLESPACE GLWSAMP.GLWSPJA
INZU291I   TABLESPACE UNLOADED FROM LAST IMAGE COPY
INZU279I - SELECT STATEMENTS USING SINGLE TABLE SPECIFICATION
INZU280I - SELECT 00001 STARTING AT POS(6, 1)
INZI328I - DATASET ALLOCATED. TEMPLATE=OUTFILE
           DDNAME=SYS00001
           DSN=DBOAS.GLWSAMP.GLWSPJA.UNLOAD
INZU180I UTPRINT DD CARD IN JCL IS NOT USED WHEN VUXO2O/SORTCLAS IS SPECIFIED
IN
               OR WHEN SORTCLASS IS SPECIFIED IN SYSIN.

- TABLESPACE   GLWSAMP.GLWSPJA - DB2 HIGH PERFORMANCE UNLOAD - STATISTICS -
03/
* CREATOR.TABLE               *      OBID *   ROWS READ   *   ROWS KEPT   * TS
P
*----------------------------*-----------*--------------*--------------*----
-
* GLWSAMP.GLWTPJA             *        34 *        269262 *        269262 *
*----------------------------*-----------*--------------*--------------*----
-
* TOTAL UNLOAD STATISTICS ....*           *        269262 *        269262 *
* INVALID ROWS................*                         0
* NUMBER OF PAGES IN ERROR....*                         0

INZU260I GENERATING LOAD STATEMENT FOR SELECT STARTING AT POS(6, 1)
INZU222I SYS00001, TOTAL NUMBER OF RECORDS WRITTEN 269262
INZU376I SELECT 1  NUMBER OF RECORDS WRITTEN 269262
```

Figure 4-49 DB2 High Performance Unload job messages

4.5 Generating DB2 High Performance Unload jobs from DB2 Automation Tool

DB2 Automation Tool V4.1 APAR PM70461 (PTF UK82220) has added support for DB2 High Performance Unload V4.2 via the Stand Alone Utilities menu option.

DB2 Automation Tool V4.1 is customized by using the Tools Customizer. After applying the APAR, you start Tools Customizer and then specify the product parameters to customize the product parameters. See Figure 4-50 on page 107.

```
CCQPPRD                  Product Parameters                          15:53:01
Command ===>                                                 Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks
and steps are preselected. Ensure that all parameters are specified for each
selected step within a task. Press End to save and exit.

Commands: SAVE - Save parameter values
Line Commands: / - Select

 Product to Customize
   Product metadata library . : PDWITT.TCZ.SHAADENU    >  LPAR  . . : RS22
   Product name . . . . . . . : IBM DB2 Automation Tool >  Version . : 4.1.0

Product customization library .: CSJENN.HL2.$RS22$.HAA410
                                                                  More:   - +

  / HPU Support
    DB2 HPU load libraries . . . . . . . . .                    >     Add...
    DB2 HPU ver/rel/mod  . . . . . . . . . . 420 (420)

/ Create/Update DB2 Objects
```

Figure 4-50 Customize the product parameters

High Performance Unload support parameters have been added under the Update Control File section.

Enter the DB2 High Performance Unload load libraries and DB2 High Performance Unload version/release/modification fields. If the High Performance Unload Support step is selected, these fields are both required.

When you generate customization jobs, the High Performance Unload support job is generated, as shown Figure 4-51 on page 108. This job updates the DB2 control file used by DB2 Automation Tool.

```
CCQPCST                Finish Product Customization          Row 1 to 6 of 10
Command ===>                                                 Scroll ===> CSR

Submit the members in the order in which they apply to all DB2 entries. To
submit the job, browse the member and issue the TSO SUBMIT command, or browse
the customized library and submit the jobs from there.

Product to Customize
  Product metadata library . : PDWITT.TCZ.SHAADENU       > LPAR  . . : RS22
  Product name . . . . . . . : IBM DB2 Automation Tool   > Version . : 4.1.0

Line Commands: E - Edit  B - Browse

    Product customization library .: CSJENN.HL2.$RS22$.HAA410

 Cmd Member   SSID GrpAttch Template Date       Description
 -   -------- ---- ----     -------- ---------- ------------------------------>
     AOV41     --   --      HAAV41   2013/05/13 Configure Startup CLIST 1
     A1V41C    --   --      HAAV41C  2013/05/13 Configure Startup CLIST 2
     A2EXECS   --   --      HAAEXECS 2013/05/13 HAA Required Execs
     A3CNTFL   --   --      HAACNTFL 2013/05/13 Create Control File
     A4CF2UAA DA1A  --      HAACF2UP 2013/05/13 Update Control File SSID specif
     A6CF1U2   --   --      HAACF1U2 2013/05/13 HPU Support
```

Figure 4-51 High Performance Unload support job

After submitting this job, High Performance Unload functionality is available in DB2 Automation Tool.

You can change High Performance Unload support parameters by using option 0.5 from the DB2 Automation Tool Main Menu after starting the DB2 Automation Tool V4.1 CLIST. See Figure 4-52.

```
HAAPNL5  V4R1 ------------------ HPU Utility Parms ------ 2013/05/13  11:06:58
Command ==>
-------------------------------------------------------------------------------
 HPU Utility Information:

 HPU Loadlib1  . . . . . . VENDOR.HPU42.SINZLINK
 HPU Loadlib2  . . . . . .
 HPU Loadlib3  . . . . . .
 HPU Version . . . . . . . 420 (420)
```

Figure 4-52 Changing High Performance Unload support parameters

After selecting the DB2 subsystem on the DB2 Automation Tool Main Menu, from the main panel for Stand Alone Utilities (Figure 4-53 on page 109), you can now additionally select option 4.

```
AUTOTOOL V4R1 ---------- Stand Alone Utilities ---------- 2013/04/09  15:31:56
Option  ===>

  Options: 1 - Backup System   2 - Restore System   3 - Unload Utility
           4 - High Performance Unload

  DB2 Subsystem ID: DA1A     Current SQLID: PDDUDE       User: PDDUDE  - RHO
```

Figure 4-53 High Performance Unload option

You then select the tables to unload, as shown in Figure 4-54.

```
AUTOTOOL V4R1   ------ HPU Tablespace Selection ------    2013/04/09  15:37:46
 Option  ===>                                                Scroll ===> CSR

--------------------------------------------------------------------------------
 Line Commands: S  - Select Table space                  DB2 Subsystem ID: DA1A
                                                        No rows to display
   Database   Like *
   Tablespace Like *
   Creator    Like *        >

--------------------------------------------------------------------------------
 Cmd  Dbname   Tsname   Part  Creator   OBID   Created Timestamp
```

Figure 4-54 Selecting the criteria for the objects

You specify selection criteria and press Enter. See Figure 4-55 on page 110.

```
AUTOTOOL V4R1 ---------- HPU Options ---------- 2013/04/09  15:40:03
 Option   ===>

--------------------------------------------------------------------------------
  Database   Name: A1681CPD                                    User: PDDUDE
  Tablespace Name: ABPCP1TS                       DB2 Subsystem ID: DA1A
  Creator    Name: PDKILI    >

--------------------------------------------------------------------------------

  Build Unload Job  . . . . N (Yes/No)
  Utility ID  . . . . . . .                  (16 Characters)
                           Include         Update
  Select  Table and Columns N (Yes/No)  . . N (Yes/No)
  UNLDDN  Options . . . . . N (Yes/No)  . . N (Yes/No)
  COPYDDN Options . . . . . N (Yes/No)  . . N (Yes/No)
  Options Block . . . . . . . . . . . . . . N (Yes/No)
  DB2 . . . . . .   (blank, Y - Yes, N - No, F - Force)
  LOCK  . . . . . .   (blank, Y - Yes, N - No)
  QUIESCE . . . . . .   (blank, Y - Yes, N - No)
  QUIESCECAT  . . . . . .   (blank, Y - Yes, N - No)
  UNLMAXROWS  . . . . . .           (blank, 1 - 2147483647)
  UNLFREQROWS . . . . . .           (blank, 1 - 2147483647)
```

Figure 4-55 The selected table spaces for unloading

The help panel for Figure 4-55 shows a detailed explanation of the fields in this panel.

The High Performance Unload Options panel lets you set options and generate JCL for High Performance Unload. Data can be unloaded in two ways:

- All rows and columns from a selected table space are unloaded. This type of unload is called a physical unload. All data is unloaded to one output DD (UNLDDN).
- You can select or filter specific rows or columns from tables in a selected table space. This type of unload is called a *logical unload*.

  SQL SELECT statements are used to retrieve the data and to specify the output format. Each SELECT statement receives its own output DD.

There are the field descriptions:

Database Name The database name of the table space.

Tablespace Name The selected table space.

Creator Name The creator of the table space.

User The current user ID.

DB2 Subsystem ID The DB2 SSID.

Build Unload Job Type
Type Y in this field to generate High Performance Unload JCL.

Utility ID Type a 1 - 16 character utility ID to be used to uniquely identify the utility to DB2. The ID must begin with a letter. The remainder can be alphanumeric or a special character; see the user guide for a list of valid special characters.

Select Table and Columns

Use this field to select or filter specific table rows or columns from the table space. Type Y in the Include field and Y in the corresponding Update field.

UNLDDN Options Use this field to include and modify options for the UNLDDN DD. This DD is required for physical unloads of entire table spaces. Type Y in the Include field and Y in the corresponding Update field.

COPYDDN Options Use this field to include and modify options for the COPYDDN DD. This DD is required if you want to unload data from a full or incremental image copy of the specified table space. Type Y in the Include field and Y in the corresponding Update field.

Options Block Use this field to specify default conversions to transform the data during the unload. The effect of the options that are specified in the Options block depends on the value that is specified for the DB2 HPU VUU057/OPALLFMT PARMLIB parameter.

See the *IBM DB2 High Performance Unload for z/OS Version 4 Release 1 User's Guide,* SC19-3169, for more information.

DB2 Type Y to allow DB2 to process SELECT statements that are not supported by High Performance Unload.

Type N to reject SELECT statements that are not supported by High Performance Unload.

Type F to force reading of the table using DB2 SQL access.

LOCK Type Type Y to prevent other processes from accessing the table space when the job is running.

QUIESCE Type Type Y to issue a QUIESCE command against the table space before unloading it.

QUIESCECAT Type Y to issue a QUIESCE command against the DB2 catalog table spaces before unloading data.

UNLMAXROWS For a physical unload, you can limit the number of rows that are unloaded by specifying a value in this field.

Valid values are 1 - 2147483647.

UNLFREQROWS For a physical unload, you can specify an unload frequency. This value causes one row to be unloaded every *n* rows, where *n* is the value that you enter in this field.

Valid values are 1 - 2147483647.

Figure 4-56 on page 112 shows the main High Performance Unload options.

```
AUTOTOOL V4R1 ----------     HPU Options Block ---------- 2013/04/09  15:45:02
Option   ===>                                                 Scroll ===> CSR
                                       DB2 Subsystem ID: DA1A User: PDDUDE


   NULL Off . . . . . .       (blank, 0 - Off)
    or
   when NULL  . . . . . 6F    (blank, Single character, or Hex)
   and when not NULL    00    (Single character or Hex)
   NULLID . . . . . . . Y     (blank, Y - Yes, N - No)
   NULLPOS  . . . . . . A     (blank, A - After, B - Before)
   PIC sign . . . . . . -     (blank, +, -, P)
       position . . . . L     (blank, L - Lead, T - Trail)
       decimal  . . . . .     (blank, ",", or ".")
       mask . . . . . . 00.0  (enter ? for the list of values)
   DATE . . . . . . . . DATE_DB2      (enter ? for the list of values)
   DATEDELIM  . . . . .       (blank, single character, or hexadecimal value)
   TIME . . . . . . . . TIME_DB2      (enter ? for the list of values)
   TIMEDELIM  . . . . .       (blank, single character, or hexadecimal value)
   TIMESTAMP  . . . . . TIMESTAMP_B   (enter ? for the list of values)
   LENGTHBYTE . . . . . Y     (blank, Y - Yes, N - No)
   LENGTH . . . . . . . R     (blank, R - Real, M - Max)
```

Figure 4-56 High Performance Unload options

Example 4-1 shows an example of the generated JCL.

Example 4-1 Sample unload job

```
//*                                                                    *
//UNLOAD    EXEC  PGM=INZUTILB,REGION=1024M,COND=(4,LT),
//          PARM=(DA1A,)
//*
//STEPLIB  DD DSN=HAA.WRK41RH0.LOADLIB,DISP=SHR
//         DD DSN=HAA.MNT0410.LOADLIB,DISP=SHR
//         DD DSN=HAA.PRD0410.LOADLIB,DISP=SHR
//         DD DSN=FEC.WRK0130.LOADLIB,DISP=SHR
//         DD DSN=FEC.MNT0130.LOADLIB,DISP=SHR
//         DD DSN=RSRTE.EMC.LINKLIB,DISP=SHR
//         DD DSN=RSRTE.EMC.LINKLIB,DISP=SHR
//         DD DSN=DA1A.SDSNEXIT,DISP=SHR
//         DD DSN=DSN.VA10.SDSNLOAD,DISP=SHR
//         DD DSN=VENDOR.HPU42.SINZLINK,DISP=SHR
//UTPRINT  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
TEMPLATE  UNLDD
           DSN      'PDDUDE.&DB..&SN..T&TIME.'
           UNIT     SYSDA
           SPACE    TRK
           DISP     (MOD,CATLG,CATLG)


 UNLOAD TABLESPACE A1681CPD.ABPCP1TS
```

```
OPTIONS
  NULL X'6F' X'00'
  NULLID YES
  NULLPOS AFTER
  DATE DATE_DB2
  TIME TIME_DB2
  TIMESTAMP TIMESTAMP_B
  PIC ( '-' , LEAD , '.' , '00.0' )
  LENGTHBYTE YES
  LENGTH REAL
UNLDDN UNLDD
```

5

IBM DB2 Sort for z/OS

DB2 Sort for z/OS V1.3 (DB2 Sort) is a DB2 Tool that provides high-speed utility sort processing for data that is stored in DB2 for z/OS databases.

DB2 Sort also takes advantage of Tools Customizer to use its panels during product installation.

This chapter contains the following sections:

- Benefits of DB2 Sort
- Sort capacity exceeded reduction
- DB2 Sort OPTMODE parameter
- The environment
- Unload and Load executions with DB2 Sort enabled and disabled

5.1 Benefits of DB2 Sort

DB2 Sort improves the sort performance of many of the DB2 utilities in the IBM DB2 Utilities Suite and of several other DB2 management tools. DB2 Sort improves sort performance through enhanced sort technology and by communicating with DB2 utilities and tools and then adjusting system resources to ensure optimal sorting. This approach to sorting can result in significantly reduced sort CPU time. It also improves zIIP eligibility.

Both DB2 utilities and DB2 Tools can benefit from improved sorting performance by using DB2 Sort:

- CHECK DATA
- CHECK INDEX
- CHECK LOB
- LOAD
- REBUILD INDEX
- REORG INDEX if in-line statistics are collected for a data-partitioned secondary index (DPSI)
- REORG TABLESPACE
- RUNSTATS
- DB2 Utilities Enhancement Tool for z/OS V2.2 and subsequent releases
- DB2 Log Analysis Tool for z/OS V3.3 and subsequent releases
- DB2 Recovery Expert for z/OS V3.1 and subsequent releases
- DB2 Change Accumulation Tool for z/OS V3.1 and subsequent releases
- DB2 High Performance Unload for z/OS V4.2 and subsequent releases

Use of DB2 Sort 1.3 with DB2 utilities, as compared with running DB2 utilities alone, might result in the reduction of Sort CPU usage in the following manner:

- Up to 74% reduction on machines with zIIP engines
- Up to 43% reduction on machines without zIIP engines

Use of DB2 Sort 1.3 with DB2 utilities, as compared with running DB2 utilities alone, might result in the reduction of utility CPU usage in the following manner:

- Up to 49% reduction on machines with zIIP engines
- Up to 25% reduction on machines without zIIP engines

Use of DB2 Sort 1.3 with DB2 utilities, as compared with running DB2 utilities alone, might result in the reduction of Utility Elapsed Time in the following manner:

- Up to 50% reduction on machines with zIIP engines
- Up to 49% reduction on machines without zIIP engines

5.2 Sort capacity exceeded reduction

Utilities that fail with "`Sort Capacity Exceeded`" messages during utility execution waste CPU and DBA time. Typically, you might have to resolve space problems and possibly restart or rerun the job.

DB2 Sort's sophisticated disk allocation algorithms reduce these errors. These "`Sort capacity exceeded`" errors are typically caused by large data volumes and inaccurate real-time or old RUNSTATS SPACE statistics. They cause problems operationally since the utility fails after using large amounts of CPU. The DBA must resize objects and restart utilities (frequently in the middle of the night), disturbing a DBA's sleep.

Estimates from outdated RUNSTATS or bad real-time statistics (RTS) can over or under allocate sort work.

The DB2 Sort dynamic allocation algorithms are more dynamic than existing technology used by utilities. With the existing model, if the number of DYNALLOC sortworks is set to 6, only six are obtained in a static fashion.

With DB2 Sort, sort work data sets can be dynamically allocated on an as needed basis, not statically done at the start of the process. This prevents space from being tied up when not needed. By default, the maximum number of JCL and dynamically allocated sortworks is 32. This maximum can be increased to 255 either by a runtime parameter or installation parameter.

As a result, the DB2 Sort product is more resilient in cases of poor file size estimates.

The DYNAMIC nature of how the space is obtained as needed during the processing tends to have a smaller footprint in the sort work pool. When there are a number of applications running at the same time, this reduces the amount of space that can be held by all applications at any one point in time, leaving the cylinders available for other applications.

Space release algorithms at the end of input processing return unused space to the sort work pool at the earliest time.

The DYNALLOC facility in DB2 Sort has the ability to RETRY a failed DYNALLOC request. It waits for a specified number of minutes and retries the DYNALLOC to see whether space has become available at a later point. The number of minutes to wait and the number of retries can be set at installation time or overridden at execution.

Another way that DB2 Sort helps reduce "`SORT CAPACITY EXCEEDED`" errors is through more intelligent use of secondary allocation space. Before DB2 Sort issues an SVC 37 to get additional space on an existing sort work data set (either JCL allocated or DYNALLOCed), it first checks to see how large the largest contiguous piece of storage is on the DASD volume. If the secondary allocation space request is larger than this, we reduce the secondary allocation request to fit on to the pack.

This can be significant. The primary space for a data set can be obtained in up to five pieces. The secondary space may only be obtained in one piece. If the secondary allocation size is several hundred cylinders and the pack has lots of space but the largest piece is 150 cylinders, the secondary allocation will fail unless it is modified. In this case, the sort can lose out on thousands of cylinders worth of sort workspace that it could not use simply because the secondary allocation quantity is too large. This is a case where the DASD space is really available, but the sort could end with a "`SORT CAPACITY EXCEEDED`" message.

5.3 DB2 Sort OPTMODE parameter

DB2 Sort has several parameters that can influence sort behavior on the system, and one of the most important parameters is OPTMODE.

There are three options available through the OPTMODE parameter, OPTMODE=BALANCE, CPU, or ELAP:

- BALANCE gives the best mix of CPU and elapsed time performance by using memory objects in moderation. This is our current specification. The default is ELAP.
- CPU uses the least amount of central storage, allocating storage only for DATA SPACEs. No storage is allocated for memory object sorting, which provides the best CPU time.
- ELAP maximizes the use of memory object space to achieve the best elapsed time performance.

All of these options use the current logic to limit central storage use if auxiliary storage availability is low. The OPTMODE parameter is available both as a documented PARM and a CNKOPTNS macro installation default parameter.

DB2 SORT decides how much storage to use based on the OPTMODE parameter setting:

- OPTMODE=ELAP: DB2 SORT uses up to 90% of current available storage for central storage and memory objects.
- OPTMODE=BALANCE: DB2 SORT uses up to 50% of current available storage for central storage and memory objects.
- OPTMODE=CPU: DB2 SORT uses up to 30% of current available storage for central storage and does not use any memory objects.

The use of memory objects favors elapsed time reduction. OPTMODE=CPU can bring better CPU performance for some systems because there are no memory objects used. *CPU is especially beneficial in cases where there are no zIIPs.*

With memory objects, the movement of data from buffers to memory objects is CPU intensive. This means more CPU may be used for OPTMODE ELAP. However, if zIIP processors are available, more CPU can be offloaded to zIIPs, so OPTMODE ELAP might be preferred.

For those systems constrained by memory, OPTMODE ELAP might be the best choice. OPTMODE=CPU is not beneficial for constrained systems. For a system with limited region sizes, OPTMODE=ELAP makes DB2 SORT use more storage than BALANCE and CPU. If the file size is large for the utilities, OPTMODE BALANCE cannot provide enough storage to get the best elapsed performance. If the file size for a REORG job is large (for example, 50 GB or 100 GB), OPTMODE=ELAP helps the elapsed performance, too.

DB2 Sort APAR/PTF (PM80144/UK91133) is going to help these cases. The message "`CNK418I DATASPACE(S) AND/OR ZSPACE USED`" informs us that a memory object was used.

5.4 The environment

DB2 Sort was used in our environment. To verify its usefulness, we evaluated the difference between DB2 Sort and DFSORT for a set of DB2 utilities in our environment. We used the installation verification procedure (IVP), which is a part of the Tools Customizer customization.

The environment for our tests is summarized in Table 5-1 on page 119.

Table 5-1 Environment

Machine type	z196 (2817-716)
Operating system level	1.13
DB2 level	DB2 10
Number of regular processors	2 CPs
Number of zIIP processors	2
Amount of real memory	8 GB

There are different table spaces (tables) of 8, 10, 25, and 40 GB of data. They are partitioned with five indexes.

The tests consist of running three DB2 utilities (Load, Rebuild, and Reorg) with DFSORT and DB2 Sort and collecting Sort CPU time, Step CPU time, and elapsed time for the comparison.

Important: The regular processors and specialty engines in our test logical partition (LPAR) are both shared with other LPARs and this setup is not ideal for performance measurements. There are roughly 34 other LPARs on the same system, with a combination of z/OS, IBM z/VM®, and CF. The disk units are shared by roughly 45 LPARs. In such a dynamic environment, CPU and elapsed time measurements can be adversely affected.

For this configuration, we expect the relative percentage of performance improvement, the absolute numbers for CPU time reported, and the zIIP reroute, to be close to a dedicated environment. The elapsed times show elongations depending on contention with other activity and consequently a reduction of overall percentage improvement.

5.4.1 DB2 Sort Spreadsheet Reporters

DB2 Sort product provides some IVP jobs and two XLS spreadsheet report programs. See the program in the SCNKSAMP and SCNKBENU files. Here, we describe how to set up and collect the data for the two spreadsheets.

Setting up the spreadsheet and running the IVP jobs

Before proceeding with the IVP work, some setup is required before we can collect the results of the IVP by using DB2 Sort Reporter Spreadsheets.

We executed the following steps:

1. Download the CNKIVXLS from the SCNKBENU library to our workstation as `CNKIVXLS.XLS`. This is the Excel spreadsheet. *Excel 2003 or higher is required to produce the reports.*
2. Turn on SMF tracing to collect SMF type 30(4) and DB2 IFCD 25 SMF record 102. Issue the following command:

   ```
   -DBOA start trace(p) dest(smf) class(30) ifcid(25) tdata(cor,cpu)
   ```

3. Verify SMF collection by issuing a DB2 Trace command:

   ```
   -DBOA DIS TRACE
   ```

   The display output is similar to Figure 5-1 on page 120.

```
RESPONSE=SC63
 DSNW127I  -DBOA CURRENT TRACE ACTIVITY IS -
 TNO TYPE   CLASS        DEST QUAL IFCID
 01  STAT   01,03,04,05, SMF  NO
 01         06
 02  ACCTG  *            SMF  NO
 03  PERFM  30           SMF  NO   025
 *********END OF DISPLAY TRACE SUMMARY DATA*********
```

Figure 5-1 Display trace output

4. Customize the two IVP jobs, CNKIVP1 and CNKIVP2, from SCNKSAMP. CNKIVP1 creates a database with two table spaces/tables. The first table has 40 billion records and the second table has 10 billion records. The LOAD, REBUILD INDEX, and REORG utilities are then invoked against these table spaces by first using DFSORT and then using DB2 SORT. CNKIVP2 is a similar job to CNKIVP2, this time, creating 25 billion and 8 billion record tables.
5. Run the IVP jobs. You need some disk space for SORTWORK areas and so on. Read the instructions within the JCL for details of space requirements. *Also, if you are using a storage group that already exists, you need to comment out the "DROP STORAGE GROUP" statement that is at the end of the IVP jobs.* You do not want early retirement.

   It took several hours for these jobs to complete, so be patient.

   Important: If you are using an existing storage group, you need to comment out the "DROP STORAGE GROUP" statement that is at the end of the IVP jobs.

6. After both jobs completed, we extracted the SMF records. We used the IEASMFDP job in Figure 5-2 on page 121. Because we are running DB2 10 with DB2 SMF compressed records, we need to decompress the DB2 SMF records.

```
//S01 EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//*N DD DISP=SHR,DSN=SYS1.SC63.BIG.MAN1
//IN DD DISP=SHR,DSN=SMFDATA.DB2RECS
//OUT DD DISP=(,CATLG),UNIT=SYSDA,DSN=ADMR2.DUSAR.SMFIN,
// SPACE=(CYL,(500,50),RLSE)
 INDD(IN,OPTIONS(DUMP))
 OUTDD(OUT,TYPE(30(4),102))
 DATE(2012272,2012272)
 START(0000),END(2400)
//*****************************************************
//*        RUN SMF DECOMPRESSION PROGRAM
//*****************************************************
//S03       EXEC PGM=DSNTSMFD,COND=(4,LT)
//STEPLIB   DD DSN=DBOAM.RUNLIB.LOAD,DISP=SHR
//SYSPRINT  DD SYSOUT=*
//SYSUDUMP  DD SYSOUT=*
//SMFINDD   DD DSN=ADMR2.DUSAR.SMFIN,DISP=SHR
//SMFOUTDD  DD DSN=ADMR2.DUSAR.SMFOUT,
//             DCB=ADMR2.DUSAR.SMFIN,
//             DISP=(,CATLG),
//             UNIT=SYSDA,
//             SPACE=(TRK,(1200,200),RLSE)
```

Figure 5-2 JCL to extract the SMF records

7. We then ran the customized SCNKSAMP(CNKIVRPT) job against our SMF file `ADMR2.DUSAR.SMFOUT`. This produces a text file `&SYSUID.CNKDATA.TXT`. This file was downloaded to our workstation into the `C:\TEMP` directory as `CNKDATA.TXT`. *The spreadsheet requires this file name in this directory.*
8. From the workstation, we loaded up the spreadsheet, enabled the macros, and imported our `TXT` file.

The IVP report

Figure 5-3 on page 122 shows the summary report after we ran our test cases with two enabled zIIP engines and we loaded the spreadsheet.

Percent Improvement Using DB2 Sort

Object Size (Billion Bytes)	DB2 Utility	Sort CPU Time	Step CPU Time	Step Elapsed Time	DB2 Sort zIIP Offload %
8	LOAD	56.9%	23.5%	8.7%	37.6%
8	REBUILD	53.3%	31.8%	14.2%	35.2%
8	REORG	36.7%	33.3%	6.8%	19.6%
10	LOAD	56.7%	23.9%	23.8%	36.8%
10	REBUILD	53.1%	31.5%	14.9%	34.5%
10	REORG	36.3%	33.0%	-6.9%	19.1%
25	LOAD	61.4%	32.6%	11.3%	31.4%
25	REBUILD	59.5%	40.1%	24.4%	30.2%
25	REORG	48.8%	45.1%	25.1%	21.3%
40	LOAD	61.9%	34.1%	29.8%	31.0%
40	REBUILD	61.3%	42.6%	37.3%	30.2%
40	REORG	50.6%	46.9%	43.2%	21.3%
	Averages	**55.0%**	**38.8%**	**30.6%**	**29.0%**

Environment

LPAR Name	A08
Machine Type	2817 716
Operating System Level	1.13
DB2 Level	10
Number of Regular Processors	2
Number of zIIP Processors	2
Amount of Memory	8 G
"Above the Line" Region	1,460M

Average DB2 Sort % Improvement by Utility

DB2 Utility	Sort CPU Time	Step CPU Time	Step Elapsed Time	Average % zIIP Offload
LOAD	61.1%	32.0%	22.7%	34.2%
REBUILD	59.8%	40.4%	31.2%	32.5%
REORG	48.1%	44.4%	34.9%	20.3%

Figure 5-3 Comparison of DB2 Sort against DFSORT with two zIIP engines

The spreadsheet produces several types of summary and detail reports but it was clear with our simple test that DB2 Sort has the potential to provide sizeable performance improvements in elapsed and CPU times. The IVP spreadsheet provides you with a direct comparison between DB2 Sort and non-DB2 Sort utility runs in its reports and graphs. Furthermore, the numbers show that DB2 Sort can take advantage of the existing specialty engines with further savings in the cost of execution.

We expect the elapsed time to be even better in a dedicated I/O environment.

The full spreadsheet report is available as additional material and can be downloaded as described in Appendix C, “Additional material” on page 551.

5.4.2 DB2 Utilities Sort Analysis Reporter

The DB2 Utilities Sort Analysis Reporter (DUSAR) is available from the web[1] and independent from the existence of DB2 Sort in your environment. The DB2 Utilities Sort Analysis Reporter (DUSAR) is a tool to assist you in assessing the CPU time spent on utilities sorting to determine the potential impact and value that DB2 Sort for z/OS might provide for your specific environment. It provides a report on the historical sort activity for a number of utility executions and sort CPU time within utilities over a specific period of time. You can use this report to evaluate your current environment.

[1] http://www14.software.ibm.com/webapp/download/preconfig.jsp?id=2011-10-04+11%3A12%3A18.796304R&S_TACT=&S_CMP=

Important: You cannot directly use the TXT output from the IVP pre-processing job CNKIVRPT as input to the DUSAR spreadsheet.

How to run DUSAR

The setup is similar to the IVP spreadsheet reporter. You download the XLS spreadsheet from the website or the SCNKBENU(CNKDUXLS) library as CNKDUXLS.XLS. You collect SMF data with record type 30, subclass 4, and type 102 IFCID(25). This spreadsheet specifically excludes SMF records that have a step name #CNK. These records are associated with the IVP records and considered not to be part of a customer assessment.

When we ran the DUSAR spreadsheet, we included the IVP SMF records by modifying the SORT control statements in the sample job. We changed all occurrences of #CNK to #VNK in SCNKSAMP(CNKDUSAR). We ran the customized JCL and produced the DUSARDAT.TXT. This was downloaded to the workstation to C:\TEMP\DUSARDAT.TXT. This is a requirement of the spreadsheet; the file must be in the directory C:\TEMP as file DUSARDAT.TXT. Opening the CNKDUXLS.XLS Excel spreadsheet, we imported the TXT file and produced some reports for DFSORT usage.

DUSAR Reporting: DFSORT or DB2 Sort

The DUSAR reporter, by default, only extracts SMF sort records related to DFSORT type 'T' records. To get the same information for DB2 Sort, we modified the CNKDUSAR. Search for the following line:

```
INCLUDE=(5,1,CH,EQ,C'T',&,35,4,CH,NE,C'#CNK')
```

Modify it to this line:

```
INCLUDE=(5,1,CH,EQ,C'S',&,35,4,CH,NE,C'#CNK')
```

There should be two occurrences.

We executed two runs of DUSAR, using the SMF output from our IVP runs as a data source.

The two spreadsheets with details can be downloaded as described in Appendix C, “Additional material” on page 551. See DFSORT.XLS and DB2SORT.XLS.

DUSAR Reporting: DFSORT sample

The first run extracted DB2 sort records and produced an XLS spreadsheet from DFSORT utilities workload executions. Figure 5-4 on page 124 shows the Utilities Executions report.

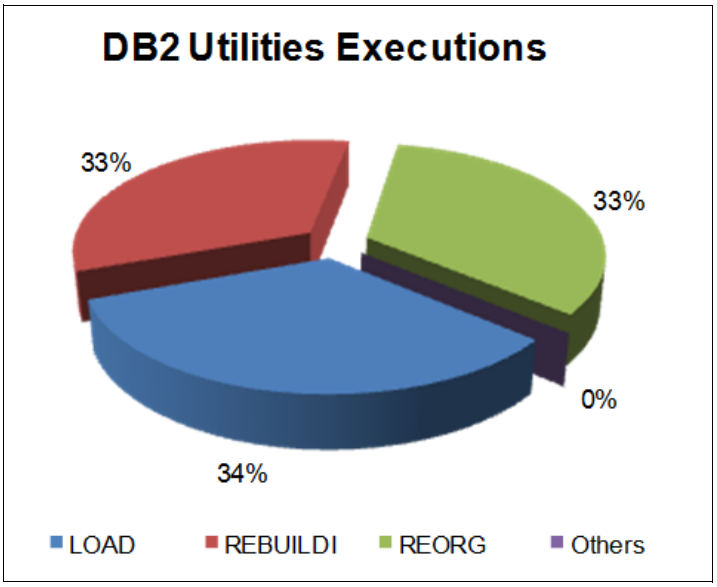

Figure 5-4 DB2 Utilities Executions report - DFSort

Figure 5-5 shows the DB2 utilities overall CPU usage for sorting.

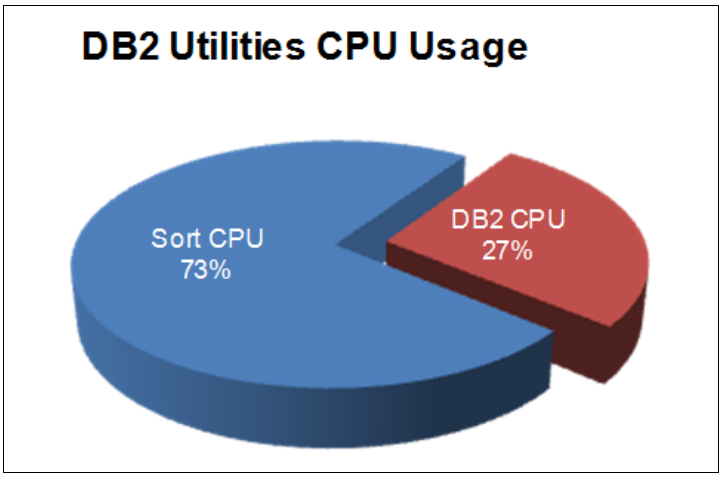

Figure 5-5 Utilities workload CPU breakout - DFSort

You can see that the sorting activity is a major contributor to CPU utilization of the utilities workload.

DUSAR Reporting: DB2 Sort

The second run of CNKDUXLS extracted DB2 Sort records from the imported SMF data. We modified the CNKDUSAR job to change from type `'T'` to type `'S'`. The output produced charts related to DB2 Sort characteristics. Figure 5-6 on page 125 shows the CPU breakout for the utilities workload executions for DB2 Sort.

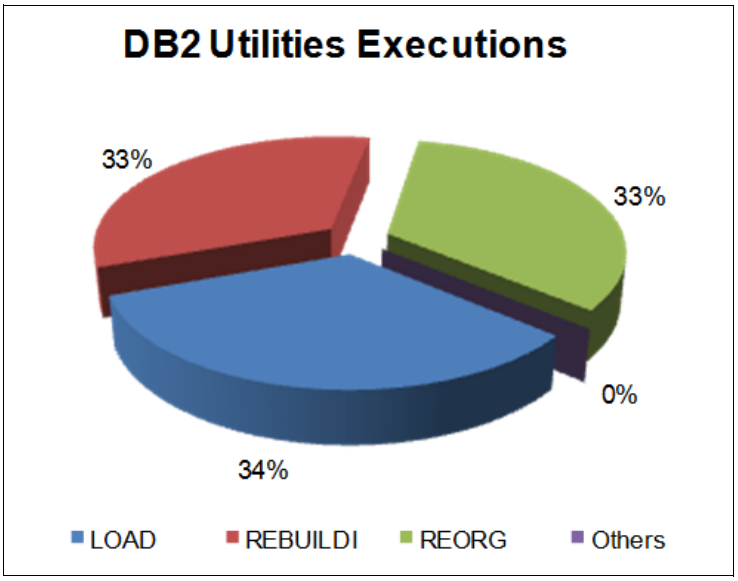

Figure 5-6 DB2 Utilities Executions report - DB2 Sort

Figure 5-7 shows the DB2 utilities overall CPU usage for sorting.

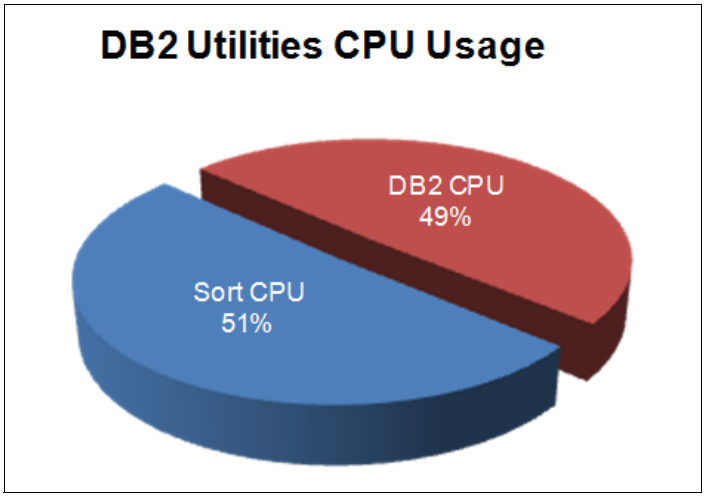

Figure 5-7 Utilities workload CPU break out - DB2 Sort

CPU% due to sorting is reduced from 73% to 51%.

5.4.3 DB2 Sort performance

For this quick value assessment, we used DB2 Sort more or less straight out of the box in our test environment and without considering the tuning or performance aspects that can be associated with this product and with our environment. DB2 Sort has some additional performance options, such as the LOAD PRESORT and PRESORTED options, which work with Utilities Enhancement Tool. These options can reduce LOAD times and avoid the sort for the clustering index.

The PRESORT option sorts the data before the load phase. Through a policy, Utilities Enhancement Tool can change the PRESORT YES to the PRESORTED option, which tells the utility that the data is already in the correct sort order. As a result, the LOAD utility, knowing the data is already in clustering order, performs significantly better. If the data is not in sorted order, the utility still completes, but without the performance advantage it has with the data in sorted order.

5.5 Unload and Load executions with DB2 Sort enabled and disabled

In the following chapters, we show a scenario comparing a DB2 Unload using DB2 High Performance Unload and after a Load with DB2 Sort enabled and disabled.

We use 312 million rows in a table with three indexes in this scenario and DB2 Administration Tool.

6

IBM DB2 Automation Tool for z/OS

Do you still create utility jobs manually to maintain several objects? Do you think that your maintenance jobs need to be run on a predefined frequency basis? IBM DB2 Automation Tool for z/OS V4.1 (DB2 Automation Tool) helps you with these challenges. Industry specialists know that data grows extremely fast. Databases, such as DB2 for z/OS, can be very useful to store all the data that requires data availability and security.

The current economic environment and budget reductions are also important industry challenges. Pressure is constant to analyze the IT budget to lower IT costs and invest only in technologies that provide a good return on investment.

DB2 Automation Tool can help IT staff reduce IT think time to repetitive tasks and also to analyze the environment in order to run only what is needed when it is needed, reducing the CPU utilization for maintenance jobs that do not really need to run in a defined maintenance window.

Combining Object, Utility, and Job Profiles, DB2 Automation Tool can reduce and facilitate manual routine tasks and focus on more complex job responsibilities that add more value to your company. It can automate common DB2 maintenance tasks, as well as generate JCL for more complex tasks on one or more objects. We describe profiles in more detail in the next sections.

Using Exception Profiles and DB2 Automation Tool, you can define in a Utility Profile when to run a utility against an object in an Object Profile. You select the conditions from a statistics list in the Exception Profile. For example, you can specify that the value in the PERCINDREF column in SYSTABLEPART for a table must be greater than a specified value in order to trigger a REORG, or you can set DB2 Automation Tool to run an image copy (IC) in case an object has changed.

DB2 Automation Tool also has a feature that exploits DB2 10 autonomic statistics in order to analyze when a Runstats needs to run. DB2 Automation Tool's statistics monitoring profiles enable you to define what objects to monitor and identify the criteria that define out-of-date statistics.

Integration with other DB2 Tools is also important in an IT environment. DB2 Automation Tool has an ISPF interface to generate utilities JCL, including an interface with High Performance Unload to generate the necessary job.

We describe the following topics by using scenarios that we set up to run at our test environment:

- DB2 Automation Tool: Profiles
- Automation of REORG and backups using exception reporting
- Automation of RUNSTATS by using autonomic statistics
- DB2 administrative task scheduler
- Interfacing DSNACCOX with DB2 Automation Tool
- DB2 Automation Tool and CHECK DATA

6.1 DB2 Automation Tool: Profiles

By using DB2 Automation Tool, DBAs can easily define sets of profiles to run maintenance jobs against a set of objects by using the ISPF interface. One of the most important advantages of this approach is that all profiles can be reusable and combined to generate the necessary maintenance jobs according to your business needs.

Senior DBAs can set the maintenance environment for the first time and let less experienced DBAs monitor the job execution. Senior DBAs can then be free to dedicate their time to other more complex challenges and tasks.

Utility JCL can be generated and run only when the defined conditions are met. This approach saves processor time and I/O processing when compared to the standard practice of running the utilities against all objects in an application during a maintenance window.

We describe the following DB2 Automation Tool profiles and show you how to use each one:

- Object Profiles
- Utility Profiles
- Exception Profiles
- Job Profiles

6.1.1 DB2 Automation Tool Object Profiles

Object Profiles allow you to create reusable lists of objects. You can group related objects into one profile, such as all objects for a particular application. In an Object Profile, you can include objects on which you want to run utilities, as well as exclude objects that you want the utilities to ignore. Object Profiles are similar to DB2 TEMPLATEs. They allow table spaces and index spaces to be chosen for processing in much the same way.

You can select one or more of these objects to include in an Object Profile:

- Entire table spaces
- Selected partitions of a table space
- Entire indexes
- Selected partitions of an index
- Table spaces, index spaces, or both on a particular volume or set of volumes
- Wildcards that can automatically include table spaces or index spaces that match a specified mask

From the DB2 Automation Tool main menu (see Figure 6-1 on page 130), we can access the profile sections. We can build the Object Profile, which identifies which objects are the targets of the utility job. Profiles can be used freely; they are not associated with only one job or utility. In our scenario, we build an Object Profile to be associated with an Image Copy Utility Profile and a REORG Utility Profile.

We select option 1, Object Profiles.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2012/04/28  23:26:41
Option  ===>
--------------------------------------------------------------------------------

       Options: 0 - Setup                   8 - Dataset Manager
                1 - Object Profiles         9 - Data Page Display
                2 - Utility Profiles       10 - Disaster Recovery
                3 - Exception Profiles     11 - Stand Alone Utilities
                4 - Job Profiles           12 - DB2 Admin Scheduler
                5 - Quick Build            13 - DB2 Autonomic Statistics
                6 - Execution Reports       X - Exit
                7 - DB2 Command Processor

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DBOA         (1-4 Character Subsystem ID or ? for list)
 Current SQLID:    ADMR2        User: ADMR2   - HAA
```

Figure 6-1 The DB2 Automation Tool main menu panel - Building an Object Profile

A search panel is shown where we can search for Object Profiles that we have already built. You can use two search criteria; a profile name (Like *) and a creator name. We select Profile Like * and Creator Like ADMR2. See Figure 6-2.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2012/04/28  23:34:24
Option  ===> 1
--------------------------------------------------------------------------------

       Options: 0 - Setup                   8 - Dataset Manager
                1 - Object Profiles         9 - Data Page Display
                2 - Utility Profiles       10 - Disaster Recovery
                3 - Exception Profiles     11 - Stand Alone Utilities
             Esssssss Enter Objects Profile Like to Display sssssssN
             e                                                    e ics
             e Profile Like *                                     e
             e Creator Like ADMR2                                 e
             e  F1=HELP      F2=SPLIT     F3=END       F4=RETURN   e
------------ e  F5=IFIND     F6=RCHANGE   F7=UP        F8=DOWN     e----------
 DB2 Subsyst DssssssssssssssssssssssssssssssssssssssssssssssssssssssssM list)
 Current SQLID:    ADMR2        User: ADMR2   - HAA
--------------------------------------------------------------------------------
```

Figure 6-2 DB2 Automation Tool object selection panel

In our case, we have already set up Object Profiles DSN00037 and GLWSAMP under the creator name ADMR2. Pressing Enter displays the object files that we created. See Figure 6-3 on page 131.

```
AUTOTOOL V4R1   -------- Objects Profile Display -------- 2012/04/28  23:38:04
Option  ===>                                                 Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import
               Q - Quick   V - View    U - Update  J - Jobs    R - Rename
Profile Like  *                                        DB2 Subsystem: DBOA
Creator Like  ADMR2                                  Row 1 of 2             >
--------------------------------------------------------------------------------

Cmd  Name                           Creator   Updt
     DSN00037                       ADMR2      U
     GLWSAMP                        ADMR2      U
***************************** Bottom of Data*********************************
```

Figure 6-3 Objects Profile Display

If we view one of these profiles, we can see the details that make up the Object Profile. In our case, we select the GLWSAMP database and we see that the profile defines five table spaces. See Figure 6-4.

If IX is Y, all indexes that are related to the tables in the table space are included in the list. And, if referential integrity (RI) is Y, all referentially related table spaces are automatically included.

In our case, we did not choose these options.

```
AUTOTOOL V4R1   ---- View Object Profile Display      --- 2012/04/28  23:41:43
Option  ===>                                                 Scroll ===> CSR

--------------------------------------------------------------------------------
Commands: Explode - View all objects.

Creator: ADMR2        Profile: GLWSAMP                        User: ADMR2
Description: OBJECT PROFILE FOR GLWSAMP
Share Option U  (U - Update, V - View, N - No)        Row 1 of 5             >
--------------------------------------------------------------------------------
                                                    Volume /
         Wild ---- Process --- Inc/ IX DB Name/ IX Crtr/ IX Name/
Cmd Type Card IX RI Clone Util Exc  TS Crtr     DB Name  TS Name
    TS   N    N  N  N     N    INC  DB2R2       GLWSAMP  GLWSDPT
    TS   N    N  N  N     N    INC  DB2R2       GLWSAMP  GLWSEMP
    TS   N    N  N  N     N    INC  DB2R2       GLWSAMP  GLWSEPA
    TS   N    N  N  N     N    INC  DB2R2       GLWSAMP  GLWSSPL
    TS   N    N  N  N     N    INC  GLWSAMP     GLWSAMP  XGLW0000
***************************** Bottom of Data ********************************
```

Figure 6-4 Viewing objects that make up an Object Profile

We have the option to update the existing profiles. For example, we can include or exclude objects. Pressing PF3 returns us to our original search list.

Creating an Object Profile is straightforward. We use the C line command to create a new profile. See Figure 6-5 on page 132.

```
AUTOTOOL V4R1   -------- Objects Profile Display -------- 2012/04/28  23:45:41
Option  ===>                                                 Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import
               Q - Quick   V - View    U - Update  J - Jobs    R - Rename
Profile Like  *                                        DB2 Subsystem: DBOA
Creator Like  ADMR2                                  Row 1 of 2             >
--------------------------------------------------------------------------------
          Esssssssssssssssss Enter New Objects Profile Data sssssssssssssssssN
Cmd  Name e                                                                    e
 C   DSNO e                                                                    e
     GLWS e    Creator . . . . ADMR2                                           e
********* e    Profile Name  .                                                 e
          e    Description . .                                                 e
          e    Update Option . U  (U - Update, V - View only, N - No access)   e
          e  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=IFIND     e
          e  F6=RCHANGE   F7=UP        F8=DOWN      F9=SWAP     F10=LEFT      e
          DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-5 Entering details for a new Object Profile

We can now provide a Profile Name and Description. Entering some details and pressing Enter brings up the object selection panel. See Figure 6-6. In this panel, we can select the type of objects that we want to include: table spaces, indexes, and objects by volume.

```
AUTOTOOL V4R1   -------- Objects Profile Display -------- 2012/04/28  23:45:41
Option  ===>                                                 Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import
               Q - Quick   V - View    U - Update  J - Jobs    R - Rename
Profile Like  *                                        DB2 Subsystem: DBOA
Creator Like  ADMR2                                  Row 1 of 2             >
--------------------------------------------------------------------------------
             Es Add Objects to the Object Profile sN
Cmd  Name    e                                     e
 C   DSN0003 e    Add Tablespaces . . N (Yes/No)   e
     GLWSAMP e                                     e
************ e    Add Indexes . . . . N (Yes/No)   e **************************
             e                                     e
             e    Add Volumes . . . . N (Yes/No)   e
             e  F1=HELP     F2=SPLIT    F3=END     e
             e  F4=RETURN   F5=IFIND    F6=RCHANGE e
             DsssssssssssssssssssssssssssssssssssssM
```

Figure 6-6 Selecting the object types that make up the Object Profile

We select table spaces for our example and press Enter. The next selection panel (Figure 6-7 on page 133) enables us to drill deeper into the selection process and to retrieve table spaces based on certain selection criteria.

```
 AUTOTOOL V4R1   ---- Update Object Profile Display    --- 2012/04/28  23:57:14
 Option  ===>                                                Scroll ===> CSR

 ------------------------------------------------------------------------------
 Commands: Explode - View all objects.
 Line Commands: A - Add  D - Delete  E - Explode  U - Update  R - Repeat
 Creator: ADMR2        Profile: DSN00038                      User: ADMR2
 Description: OBJECT PROFILE FOR DSN00038
Esssssssssssssssssss Enter Tablespaces Like to DisplayssssssssssssssssssssssssssssssN
e                                                                               e
e Database Like. . ABP*     Wildcard   y (Yes/No) e
e Tablespace Like. *        Exclude    I (E - Exclude, I - Include) e
e Creator Like . . *        > e
e e
e Process Dependent Indexes . . . . . . . . . .  N (Yes/No) e
e Process Referentially Dependent Tablespaces.  N (Y - Yes, N - No, e
e                                                  B - Build time Expansion, e
e                                                  R - Run time Expansion) e
e Process Cloned Tables. . . . . . . . . . . . .  N (Yes/No) e
e e
e  F1=HELP     F2=SPLIT    F3=END      F4=RETURN   F5=IFIND    F6=RCHANGE e
e  F7=UP       F8=DOWN     F9=SWAP     F10=LEFT    F11=RIGHT e
DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-7 Using the wildcard option when building Object Profiles

We select all databases that begin with ABP and indicate that this is a wildcard selection. If we do not select Y in the Wildcard field, we are presented with the list of all table spaces that are associated with databases that begin with ABP.

Pressing Enter displays the object entry that we added to our Object Profile. See Figure 6-8. We can make further changes to the selection later by using the Update line command. In our scenario, we make no further changes.

```
AUTOTOOL V4R1   ---- Update Object Profile Display    --- 2012/04/29  00:03:39
Option  ===>                                                Scroll ===> CSR

------------------------------------------------------------------------------
Commands: Explode - View all objects.
Line Commands: A - Add  D - Delete  E - Explode  U - Update  R - Repeat
Creator: ADMR2        Profile: DSN00038                      User: ADMR2
Description: OBJECT PROFILE FOR DSN00038
Share Option U  (U - Update, V - View, N - No)        Row 1 of 1             >
------------------------------------------------------------------------------
                                                  Volume /
         Wild ---- Process --- Inc/ IX DB Name/ IX Crtr/ IX Name/
Cmd Type Card IX RI Clone Util Exc  TS Crtr     DB Name  TS Name
    TS   Y    N  N  N     N    INC  *          ABP*     *
****************************** Bottom of Data ********************************
```

Figure 6-8 An example of a wildcard entry in an Object Profile

We press PF3 to save the entry into our Object Profile. We see that it is listed in the Objects Profile Display panel. See Figure 6-9 on page 134.

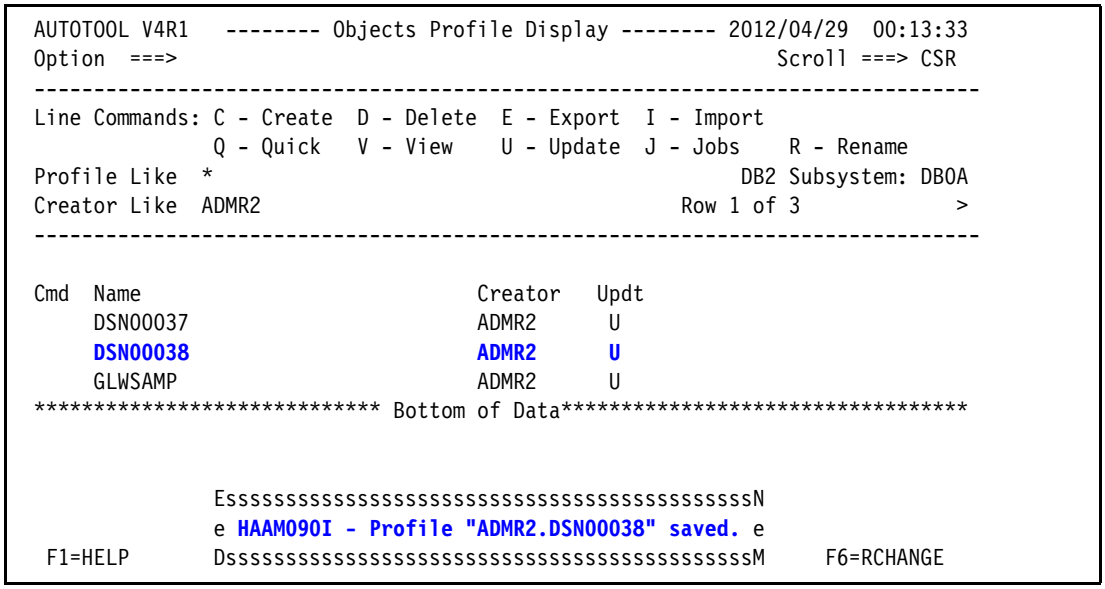

Figure 6-9 Saving the Object Profile

In our example, we created an Object Profile that reflects the databases that are prefixed by ABP. We called our profile DSN00038. It is a wildcard-type entry, which means all table spaces that match the wildcard pattern are included. Next, we build the Utility Profile.

The Object Profile Display panel gives us a selection of line commands. Table 6-1 shows a summary of the line command functions.

Table 6-1 Line commands

Abbreviation	Command	Action
C	Create	Create a new Object Profile
D	Delete	Delete an Object Profile
E	Export	Export an Object Profile to another SSID or data set
I	Import	Import an Object Profile from a data set
J	Jobs	View all the Job Profiles in which an Object Profile is included
Q	Quick	Perform a quick build using the Object Profile
R	Rename	Rename an Object Profile (not accessible from Job Profiles)
V	View	View an Object Profile (no changes are allowed)
U	Update	Update an Object Profile
S	Select	Select an Object Profile for inclusion in a Job Profile

6.1.2 DB2 Automation Tool Utility Profiles

A *Utility Profile* is a collection of specified utilities and the options to use for those utilities. Using a similar technique to creating Object Profiles, we now create a Utility Profile.

The following list shows the utilities that are available with DB2 Automation Tool:

- Recover
- Copy
- DB2 Recovery Expert Image Copy
- Copy to Copy
- Reinstates
- TS REORG
- IX REORG
- Quiesce
- Modify
- Repair
- Check Data
- Rebind

In our sample, we perform an image copy.

From the DB2 Automation Tool main panel we select option 2, Utilities Profiles, as shown in Figure 6-10.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2013/02/20  19:53:10
Option  ===>
--------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Disaster Recovery
                3 - Exception Profiles      11 - Stand Alone Utilities
                4 - Job Profiles            12 - DB2 Admin Scheduler
                5 - Quick Build             13 - DB2 Autonomic Statistics
                6 - Execution Reports        X - Exit
                7 - DB2 Command Processor

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DBOA         (1-4 Character Subsystem ID or ? for list)
 Current SQLID:                 User: ADMR3   - HAA
```

Figure 6-10 DB2 Automation Tool Utility Profiles selection

In this case, we enter * for both profile name and creator, as shown in Figure 6-11 on page 136.

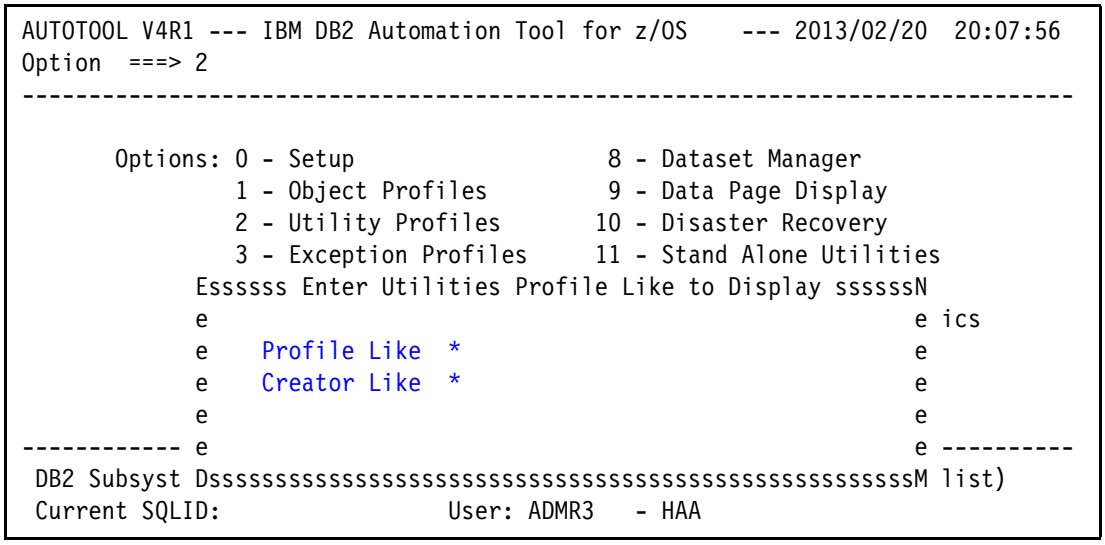

Figure 6-11 DB2 Automation Tool Profile criteria

This lists all the existing Utility Profiles, including the samples that are shipped with the product. See Figure 6-12 on page 137.

We have already built an Image Copy (with FlashCopy) profile, which is called IMAGE_COPY_FC. It has our creator name (normally, the TSO user ID), ADMR2, and it is available to anyone to update, as indicated by the U in the Updt field.

Profiles can have one of three use options:

U	Allow others to update
V	Allow others to view
N	Hide from other users

The samples that are distributed with the product are only available to view. They cannot be altered.

```
AUTOTOOL V4R1   ------- Utilities Profile Display ------- 2012/04/29  00:19:48
Option  ===>                                                 Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import
               Q - Quick   U - Update  V - View    J - Jobs    R - Rename
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                   Row 1 of 18            >
--------------------------------------------------------------------------------

Cmd  Name                           Creator   Updt
     CHECK DATA                     DB2AUTH    V
     COPY TO COPY                   DB2AUTH    V
     COPY TO DASD                   DB2AUTH    V
     COPY TO TAPE                   DB2AUTH    V
     IMAGE_COPY_FC                  ADMR2      U
     MODIFY RECOVERY                DB2AUTH    V
     ONLINE IX REORG                DB2AUTH    V
     ONLINE TS REORG                DB2AUTH    V
     QUIESCE                        DB2AUTH    V
     RECOVER                        DB2AUTH    V
     REPAIR SET NOCOPYPEND          DB2AUTH    V
     REPOSITORY MAINTENANCE         DB2AUTH    V
     RESIZE                         DB2AUTH    V
     RUNSTATS CATALOG               DB2AUTH    V
     RUNSTATS REPOSITORY            DB2AUTH    V
     STANDARD IX REORG              DB2AUTH    V
     STANDARD TS REORG              DB2AUTH    V
     VERIFY                         DB2AUTH    V
**************************** Bottom of Data ********************************
```

Figure 6-12 A list of all the Utility Profile entries, including the shipped samples

We select V against our Image Copy profile to view the details.

The view in Figure 6-13 on page 138 displays the utilities that are associated with the Utility Profile. From the Utility Profile Options panel, we can see that the Utility Profile IMAGE_COPY_FC is associated with the Image Copy utility.

```
AUTOTOOL V4R1 --------- Utility Profile Options --------- 2012/04/29  00:26:09
Option   ===>

  Creator: ADMR2      Profile: IMAGE_COPY_FC                    User: ADMR2
  Description  IMAGE COPY USING FLASHCOPY
  Share Option U (U - Update, V - View, N - No)


                                Include      View
                                Utility      Utility
Data Page Verification Reporting N (Yes/No)   N (Yes/No)
Reallocation . . . . . . . . . . N (Yes/No)   N (Yes/No)
Recover  . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Image Copy . . . . . . . . . . . Y (Yes/No)   N (Yes/No)
Recovery Expert Image Copy . . . N (Yes/No)   N (Yes/No)
Copy to Copy . . . . . . . . . . N (Yes/No)   N (Yes/No)
Runstats . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
TS Reorg . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
IX Reorg . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Quiesce  . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Modify . . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Repair . . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Check Data . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Rebind . . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
```

Figure 6-13 Options for Utility Profile

Selecting Y in the View Utility column displays the Image Copy options that are associated with this Image Copy Utility Profile.

We can use U for update instead of V for view when selecting this Utility Profile. This allows us to modify the Utility Profile, including any of the utility options that are associated with this profile. The panel in Figure 6-14 on page 139 shows the options that are associated with the Image Copy Utility Profile.

```
AUTOTOOL V4R1 ------------ Image Copy options ----------- 2012/04/29  00:29:32
Option   ===>                                                Scroll ===> CSR
  Creator: ADMR2      Name: IMAGE_COPY_FC                   User: ADMR2

   Exception Rule . . . . . . . . . A (A - Accepted, R - Rejected, B - Both)
   Image Copy Utility mode  . . . . D (D - DB2, S - Symmetrix, E - Ess)
     View EMC Symm/IBM ESS Optns    N (Yes/No)
   View Image Copy DSN specs  . . . Y (Yes/No) <---------
   Utility ID . . . . . . . . . . . UTILICFC         (16 characters)

   Parallel . . . . . . . . . . . . N (Yes/No)
     Number of objects  . . . . . .    (0 - 99)
     Number of tape units . . . . .    (0 - 99)
   Filter DDname  . . . . . . . . .          (8 character DD name)
   Sharelevel . . . . . . . . . . . R (R - Reference, C - Change)
   Full Image Copy  . . . . . . . . Y (Yes/No)
   Check Page . . . . . . . . . . . N (Yes/No)
   Concurrent . . . . . . . . . . . N (Yes/No)
   Change Limit
     First Percent  . . . . . . . .      (% value)
     Second Percent . . . . . . . .      (% value)
     Report only  . . . . . . . . . N (Yes/No)
   Max Tape Volume/DASD Unit Cnt    5   (1-255 volumes)
   Stack Copy Control Cards . . . . Y (Yes/No)
   Scope  . . . . . . . . . . . . . A (A - All, P - Pending)

 Optional Skeletals                -BEFORE-     -AFTER-
   JCL Skeletal . . . . . . . . .          . .          (8 Character Name)
   Control Cards Skeletal . . . .          . .          (8 Character Name)
   Step End Skeletal  . . . . . .          . .          (8 Character Name)
```

Figure 6-14 Image Copy options associated with the Image Copy Utility Profile

We highlighted the fields that we entered when we created this Utility Profile. We selected a full image copy and the share level (Sharelevel) reference option. For the Image Copy Utility mode, we used DB2. We supplied a Utility ID of UTILICFC. We select the next level of options, particularly, the data set name specifications. We select Y for the *View Image Copy DSN specs*. The panel in Figure 6-15 is displayed.

```
AUTOTOOL V4R1 ------------ Image Copy Options ----------- 2012/04/29  00:34:02
Option   ===>

  Creator: ADMR2      Name: IMAGE_COPY_FC                   User: ADMR2

  Enter the Image Copy options to associate with this utility profile

                               Take Image Copy    View Options

   Local Primary  . . . . . . . N (Yes/No) . . . . N (Yes/No)
   Local Backup . . . . . . . . N (Yes/No) . . . . N (Yes/No)
   Recovery Site Primary  . . . N (Yes/No) . . . . N (Yes/No)
   Recovery Site Backup . . . . N (Yes/No) . . . . N (Yes/No)
   FlashCopy  . . . . . . . . . Y (Yes/No) . . . . Y (Yes/No)
```

Figure 6-15 Image Copy type options

In this panel, we select one of the types of image copy that we can perform: Local, Recovery, or FlashCopy. We selected a FlashCopy type and selected the entry by selecting Y in the relevant field. To view the options associated with this Image Copy Type, we select Y in the View field. The next panel is displayed to select the Image Copy options that are specific to FlashCopy. See Figure 6-16.

```
AUTOTOOL V4R1 ------------ FlashCopy Options ------------ 2012/04/29  00:39:41
Option   ===>
Image Copy FlashCopy
  Creator: ADMR2      Name: IMAGE_COPY_FC                    User: ADMR2

View DSN create spec  . . . Y (Yes/No)
CONSISTENT  . . . . . . . . N  (Yes/No)
Unit Type . . . . . . . . . SYSDA     (SYSDA - DISK - etc.)
Catalog Options
DISP=Status . . . . . . . .    (M - MOD, N - NEW, O - OLD, S - SHR)

     Normal Termination . .    (C - CATLG, D - DEL, K - KEEP, U - UNCATLG)

     Abnormal Termination      (C - CATLG, D - DEL, K - KEEP, U - UNCATLG)

Data Class  . . . . . . . .           (8 character class)
Storage Class . . . . . . .           (8 character class)
Management Class  . . . . .           (8 character class)
Expiration date *or*    . .           (YYYYDDD - YYDDD)
Retention period  . . . . .           (4 digit number)
```

Figure 6-16 FlashCopy options for Image Copy

This panel shows additional information, such as whether we want the image copy to be consistent and whether the image copy is written to disk or tape, for example. By selecting Y in the “View DSN create spec” field, we can display the data set name generation panel where we decide the format of our FlashCopy default subsystem name (DSN). All FlashCopy Image Copies are VSAM files.

See Figure 6-17 on page 141 for an example of the DSN Generation panel. We use this panel to determine the format of the IC data set. We can use a variety of codes to help format the DSN specification. Because we are defining a FlashCopy Image Copy, we need to ensure that the variable &DSNUM is included in the data set specification so that the partition number is included in the data set name.

```
AUTOTOOL V4R1 ------- IC Flash Copy DSN Generation ------ 2012/04/29  00:43:13
Option   ===>
  Creator: ADMR2      Name: IMAGE_COPY_FC                    User: ADMR2
Image Copy FlashCopy
  Qualifier code      Free form literal              Show DSN  N
  Current dataset name generation qualifier string:
ADMR2.&DB..&SN..T&TIME..P&DSNUM..IC
Valid dataset name generation codes are:

(* marked items are not supported in IC dynamic dataset generation.)
   1. Database               11. Date (YYYYDDD)    21. Unique
   2. Space Name             12. Year (YYYY)       27. Utility Name
   3. Partition              13. Month (MM)        28. Job Name
*  7. Vcatname               14. Day (DD)          29. Step Name
   8. Subsystem ID           15. Julian Day (DDD)  30. Substring Qualifier
*  9. User ID                16. Hours (HH)        31. Use freeform literal
  10. Time (HHMMSS)          17. Minutes (MM)      32. Dsnum (required)º
                             18. Seconds (SS)
                          *  19. Timestamp
                          *  20. Random Number
```

Figure 6-17 Image copy DSN Generation panel

We have completed the view of the IMAGE_COPY_IC Utility Profile that we have built. We press PF3 to return to the main menu to continue to build the Exception Profile.

6.1.3 DB2 Automation Tool Exception Profiles

An *Exception Profile* contains the conditions under which users want to run utilities. When combined with Object Profiles and Utility Profiles, the Exception Profiles act as a filter against the objects specified in the Object Profile. The utilities specified in the Utility Profiles are only generated on the objects in the Object Profiles that satisfy the “conditions (exceptions)” specified in the Exception Profiles.

From the main DB2 Automation Tool menu (see Figure 6-1 on page 130), select Option 3, Exception Profiles. There are 184 available selection criteria that we can use to select candidate objects. Also, we can provide our own criteria through a user exit interface. This can be REXX CLIST, a program, or a stored procedure with pre, current, and post exit points. From this panel, we select the exception that we tested against each object. Multiple criteria can be selected using boolean AND or OR operations.

For this scenario, we select objects that have no previous image copy. In the Exception Profiles list, with a search criteria of asterisks, we can see that a supplied Exception Profile called NEVER_COPIED exists. We use that sample Exception Profile because it incorporates the exception that we want, which is to select only table spaces that have no full image copy. See Figure 6-18 on page 142.

```
AUTOTOOL V4R1   ------- Exceptions Profile Display ------ 2012/04/29  00:49:59
Option  ===>                                                 Scroll ===> CSR
Line Commands: C - Create  D - Delete  E - Export  I - Import
               U - Update  V - View    J - Jobs    R - Rename
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DB0A
 Creator Like  *                                  Row 1 of 7             >
--------------------------------------------------------------------------------
 Display Statistics Reports and/or Perform Statistics Maintenance? N (Yes/No)
--------------------------------------------------------------------------------

Cmd  Name                           Creator   Updt
     CLUSTERRATIO VALUE RANGE       DB2AUTH    V
     COPY CONDITIONS                DB2AUTH    V
     LARGE OBJECTS                  DB2AUTH    V
     NEVER COPIED                   DB2AUTH    V
     REORG CONDITIONS               DB2AUTH    V
     SATURDAY ONLY                  DB2AUTH    V
     SMALL OBJECTS                  DB2AUTH    V
***************************** Bottom of Data ***********************************
```

Figure 6-18 Exceptions Profile Display

We view the Exception Profile by selecting the V line command. Again, we cannot update this particular Exception Profile because it is part of the samples that are distributed with the product. It is only available to view. But for our scenario, view is sufficient. When we view the exception criteria, we are presented with the panel that is shown in Figure 6-19 on page 143.

```
AUTOTOOL V4R1   ---- View Exceptions Profile Display   --- 2012/04/29 00:53:13
Option  ===>                                                Scroll ===> CSR

                                                    "*" indicates a DAT stat
--------------------------------------------------------------------------------
Creator: DB2AUTH     Profile: NEVER COPIED                    User: ADMR2
--------------------------------------------------------------------------------
Share Option V (U - Update, V - View,  N - No)
Description  SPACES WITHOUT A FULL COPY          Scroll Right for Column Help
Use Stats From: C (R - Repository,         View Runstats Options:   N (Yes/No)
                   C - Catalog,        Save Triggers in Repository: N (Yes/No)
                   U - Runstats,   WTO number of triggered Objects: N (Yes/No)
                   S - Shadow,   Combine IX/TS Exceptions when
                   H - History)   evaluating an IX triggering a TS: N (Yes/No)
------------------------------------------------------ Row 1 of 206        +>
S Statistics Type--- *Column------------ Cond ---------Exception Value---------
  DAY OF WEEK         MONDAY
                      TUESDAY
                      WEDNESDAY
                      THURSDAY
                      FRIDAY
                      SATURDAY
                      SUNDAY
  DAY OF MONTH        NTH_MONDAY
                      NTH_TUESDAY
                      NTH_WEDNESDAY
                      NTH_THURSDAY
                      NTH_FRIDAY
                      NTH_SATURDAY
                      NTH_SUNDAY
                      NTH_DAY
                      LAST_DAY
                      DAY_MONTH                 /    DD/MM
  TIME OF DAY         TIME_FROM                 :      M
                      TIME_TO                   :      M
--------------------------------------------------------------------------------
  OBJECT              EXCLUDE|ONLY
                      LOB
                      PGSIZE_32K
                      PBG_TS
                      AND|OR
 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=IFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 6-19 Viewing the Exception Profile

We need to page down to see the exception that is associated with this exception panel. See Figure 6-20 on page 144.

```
AUTOTOOL V4R1   ---- View Exceptions Profile Display   --- 2012/04/29 00:55:16
Option  ===>                                                Scroll ===> CSR

                                                  "*" indicates a DAT stat
--------------------------------------------------------------------------------
Creator: DB2AUTH     Profile: NEVER COPIED                    User: ADMR2
--------------------------------------------------------------------------------
Share Option V (U - Update, V - View,  N - No)
Description  SPACES WITHOUT A FULL COPY           Scroll Right for Column Help
Use Stats From: C (R - Repository,         View Runstats Options:   N (Yes/No)
                   C - Catalog,        Save Triggers in Repository: N (Yes/No)
                   U - Runstats,   WTO number of triggered Objects: N (Yes/No)
                   S - Shadow,   Combine IX/TS Exceptions when
                   H - History)   evaluating an IX triggering a TS: N (Yes/No)
---------------------------------------------------- Row 101 of 206     -+>
S Statistics Type--- *Column------------ Cond ---------Exception Value---------
                     *PERCENT_MAXALLOC
                     *PQTY
                     *SQTY
O SYSCOPY             ICTYPE              NE  F
                      DAYS
                     *CHGD_SINCE_LAST_IC
--------------------------------------------------------------------------------
  DB2 DISPLAY STATUS  TRIGGER_IF_1_MATCH
                      STATUS_ARBDP
                      STATUS_AREO*
                      STATUS_AREOR
                      STATUS_AREST
                      STATUS_AUXW
                      STATUS_CHKP
                      STATUS_COPY
                      STATUS_GRECP
                      STATUS_ICOPY
                      STATUS_LPL
                      STATUS_LSTOP
                      STATUS_PSRBD
                      STATUS_RBDP
                      STATUS_RBDP*
                      STATUS_RECP
                      STATUS_REFP
```

Figure 6-20 Scrolling the view of the Exception Profile

Scrolling through the display, we can see where the exception is specified. From this display (Figure 6-20), we can see that this profile was defined with the following selection criteria: Select this table space if the catalog SYSCOPY IC column is not F (Full).

This is defined within the panel as O for OR, SYSCOPY, ICTYPE =(NE F) - NOT EQUAL FULL. This criteria is tested against each object selected in our Object Profiles that we associate with the Job Profile. If a Full Image Copy does not exist for the object, we perform a full image copy.

We have 184 exception criteria (with 206 rows available) from which we can select, plus the capability to have our own user criteria. If you press PF11 to scroll to the right, you get more details about the selection criteria. See Figure 6-21 on page 145.

```
AUTOTOOL V4R1   ---- View Exceptions Profile Display   --- 2012/04/29 01:04:40
Option  ===>                                                Scroll ===> CSR

                                                  "*" indicates a DAT stat
--------------------------------------------------------------------------------
Creator: DB2AUTH     Profile: NEVER COPIED                    User: ADMR2
--------------------------------------------------------------------------------
Share Option V (U - Update, V - View,  N - No)
Description  SPACES WITHOUT A FULL COPY           Scroll Right for Column Help
Use Stats From: C (R - Repository,         View Runstats Options:   N (Yes/No)
                   C - Catalog,        Save Triggers in Repository: N (Yes/No)
                   U - Runstats,   WTO number of triggered Objects: N (Yes/No)
                   S - Shadow,   Combine IX/TS Exceptions when
                   H - History)   evaluating an IX triggering a TS: N (Yes/No)
--------------------------------------------------- Row 101 of 206    <-+>
S Statistics Type--- *Column------------ ----------Column Description----------
  MVS CATALOG        *PERCENT_MAXALLOC   % of max allocation for each individua
                     *PQTY               Primary space allocation in units of 4
                     *SQTY               Secondary space allocation in units of
O SYSCOPY             ICTYPE             Oper type: A-Alter, B-Rebuild IX, F|I-
                      DAYS               Trigger Exception if specified Utility
                     *CHGD_SINCE_LAST_IC Trigger Exception if changes have been
--------------------------------------------------------------------------------
  DB2 DISPLAY STATUS  TRIGGER_IF_1_MATCH When ANDing Conditions, Select this fi
                      STATUS_ARBDP       Trigger Exception if Index is in (EQ)/
                      STATUS_AREO*       Trigger Exception if object is in (EQ)
                      STATUS_AREOR       Trigger Exception if object is in (EQ)
                      STATUS_AREST       Trigger Exception if object is in (EQ)
                      STATUS_AUXW        Trigger Exception if object is in (EQ)
                      STATUS_CHKP        Trigger Exception if object is in (EQ)
                      STATUS_COPY        Trigger Exception if Tablespace is in
                      STATUS_GRECP       Trigger Exception if object is GBP-dep
                      STATUS_ICOPY       Trigger Exception if Index is in (EQ)/
                      STATUS_LPL         Trigger Exception if object has (EQ)/d
                      STATUS_LSTOP       Trigger Exception if Logical Partition
                      STATUS_PSRBD       Trigger Exception if entire NPI space
                      STATUS_RBDP        Trigger Exception if physical or logic
                      STATUS_RBDP*       Trigger Exception if logical NPI is in
                      STATUS_RECP        Trigger Exception if object is in (EQ)
                      STATUS_REFP        Trigger Exception if object is in (EQ)
                      STATUS_REORP       Trigger Exception if object is in (EQ)
```

Figure 6-21 View Exception Profile details

You can see the full details of all the exception criteria by pressing PF1 (Help).

We have selected our Exception Profile. We are ready to build the final profile, which is the Job Profile. We press PF3 to return to the main menu to complete the scenario and to run our job.

6.1.4 DB2 Automation Tool Job Profiles

Job Profiles are used to connect profiles. A Job Profile is the master profile. It is associated with one or more Utility Profiles, Object Profiles, and Exception Profiles. The combined profiles, which are headed by the Job Profile, form the basis of a DB2 Automation Tool task. We can submit this task manually or schedule it by using the DB2 administration task

scheduler. Select Option 4, Job Profiles, from the main DB2 Automation Tool menu (Figure 6-1 on page 130). Here, we can list or create Job Profiles.

For this scenario, we keep it simple and manually submit our image copy job. But first, we have to build the profile and associate it with the other profiles that we have built. Similar to the other profile options, Option 4 presents us with a Job Profile list after we provide the selection criteria. In this case, we opted to use an asterisk for both the name and creator fields, which lists all Job Profiles. See Figure 6-22.

We have the installed sample profiles and the already created image copy Job Profile. From the selection list, we select the Job Profile that is created by ADMR2, which is called IMAGE_COPY.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2012/04/29  01:12:22
Option  ===>                                                Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                   Row 1 of 11            >
--------------------------------------------------------------------------------

Cmd  Name                            Creator   Updt
     COPY CATALOG AND DIRECTORY      DB2AUTH    V
     COPY LARGE SPACES TO TAPE       DB2AUTH    V
     COPY SMALL SPACES TO DASD       DB2AUTH    V
     COPY TO DASD                    DB2AUTH    V
     IMAGE_COPY                      ADMR2      U
     ONLINE REORG                    DB2AUTH    V
     RECALL SP                       DB2AUTH    V
     RECOVER CATALOG AND DIRECTORY   DB2AUTH    V
     REPOSITORY STATS                DB2AUTH    V
     RUNSTATS CATALOG                DB2AUTH    V
     STANDARD REORG                  DB2AUTH    V
***************************** Bottom of Data ********************************
```

Figure 6-22 Jobs Profile Display

We select V to view the Job Profile. The display shows that we associated the Exception, Utility, and Object Profiles that we built previously with this particular Job Profile. See Figure 6-23 on page 147.

Our job will perform an image copy against two Object Profiles, DSN00037 and GLWSAMP. The Utility Profile defines our image copy options: a Full Image copy using FlashCopy with SHRLEVEL reference.

The Exception Profile tests each object that is selected by the Object Profiles, in this case, table spaces within our database objects that are prefixed with ABP and the five table spaces in Object Profile GLWSAMP. An image copy will be performed against any table space that does not have a full image copy already.

```
AUTOTOOL V4R1   ------ View Jobs Profile Display   ------ 2012/04/29  01:14:35
Option  ===>                                                  Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: V - View
--------------------------------------------------------------------------------
Creator: ADMR2        Profile: IMAGE_COPY                    User: ADMR2
--------------------------------------------------------------------------------
Share Option U  (U - Update,     Description TEST AND IMAGE COPY
                 V - View,
                 N - No)
View Job Generation Options   N  (Yes/No)  Row 1 of 4             >
--------------------------------------------------------------------------------
                                                              <-------
Cmd  Type  Order  Name                              Creator   Userid
     OBJS     1   DSN00037                         ADMR2     ADMR2
     OBJS     2   GLWSAMP                          ADMR2     ADMR2
     UTIL     1   IMAGE_COPY_FC                    ADMR2     ADMR2
     EXCP     1   NEVER COPIED                     DB2AUTH   ADMR2
***************************** Bottom of Data ********************************
```

Figure 6-23 Object, Utility, and Exception Profiles associated with our Job Profile

We press PF3 to return to the Jobs Profile Display list so that we can build the JCL that represents our Job Profile. In Figure 6-22 on page 146, we select the line command B against the Job Profile IMAGE_COPY to start the build process. The panel in Figure 6-24 is displayed.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2012/04/29  01:19:50
Option  ===>                                                  Scroll ===> CSR
--------------------------------------------------------------------------------
L Esssssssssssssssssss Build Job for ADMR2.IMAGE_COPY sssssssssssssssssssN
  e                                                                       e
- e Build Online or Batch. . O (O - Online, B - Batch)                    e -
  e                                                                       e
  e Edit Generated Job . . . Y (Yes/No)                                   e
- e Schedule Job . . . . . . N (Yes/No)   Update options . . N (Yes/No)    e -
  e                                                                       e
C e Build job in Dataset . . ADMR2.AUTO.JCL                               e
  e               Member . . TEST                                         e
  e                                                                       e
  e    Job Cards:                                                         e
  e ==> //ADMR2S JOB (999,POK),'DB2 UTILITY',                            e
  e ==> // REGION=0M,NOTIFY=&SYSUID,MSGCLASS=X,CLASS=A                    e
  e ==> //PROCLIB JCLLIB ORDER=DBOAM.PROCLIB                              e
  e ==> /*JOBPARM SYSAFF=SC63                                             e
  e  F1=HELP     F2=SPLIT    F3=END      F4=RETURN   F5=IFIND    F6=RCHANGE e
  e  F7=UP       F8=DOWN     F9=SWAP    F10=LEFT    F11=RIGHT              e
  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     STANDARD REORG                 DB2AUTH    V
```

Figure 6-24 Jobs Profile Display

Here, we build the job that performs the image copies.

The build is a a two-step process:

- The first step tests the exception against the selected objects and determines whether any objects are candidates for the utility.
- If there are candidates, in the second step, utility jobs are built to create the image copies against the objects that triggered the exception criteria.

If no objects are selected, we are notified and get a return code 6.

We have two options for the first step. We can either run it as a batch job, or run it online in our TSO session. If we run in batch mode and no objects are selected, we get a return code 6. If there is no Exception Profile, all objects that are defined by the Object Profile trigger an image copy.

The second step is where we build an Image Copy job, invoke the utility, and use the generated control statements and templates to select the target objects that meet our criteria.

In this scenario, we build our utility JCL in batch. We press Enter, and the next Jobs Profile Display is shown. See Figure 6-25.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2012/04/29  01:19:50
Option  ===>                                                 Scroll ===> CSR
--------------------------------------------------------------------------------
L Esssssssssssssssssss Build Job for ADMR2.IMAGE_COPY sssssssssssssssssssN
  e                                                                       e
- e You have selected your job to be built in a batch mode.  The          e -
  e batch generation JCL will be stored in dataset                        e
  e ADMR2.AUTO.JCL(TEST).                                                 e
- e Please specify in the dataset and member below where you want the     e -
  e JCL built by the batch module to be placed.                           e
C e                                                                       e
  e Schedule Job  . . . . . N (Yes/No)   Update options . . N (Yes/No)    e
  e                                                                       e
  e Build job in Dataset. . ADMR2.AUTO.JCL                                e
  e              Member . . IMAGECP
  e                                                                       e
  e Jobcard data to be used on the generated job:                         e
  e ==> //ADMR2S JOB (999,POK),'DB2 UTILITY',                            e
  e ==> // REGION=0M,NOTIFY=&SYSUID,MSGCLASS=X,CLASS=A                    e
  e ==> //PROCLIB JCLLIB ORDER=DBOAM.PROCLIB                              e
  e ==> /*JOBPARM SYSAFF=SC63                                             e
  e  F1=HELP     F2=SPLIT    F3=END      F4=RETURN   F5=IFIND    F6=RCHANGE e
* e  F7=UP       F8=DOWN     F9=SWAP    F10=LEFT    F11=RIGHT               e *
  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-25 Jobs Profile Display

With this panel (Figure 6-25), we can choose to schedule the job via the administrative task scheduler or to save the batch build job in the JCL data set ADMR2.AUTO.JCL with member name TEST. Also, we have to supply a second member that is named IMAGECP for the resultant utility JCL job. This job contains the JCL that actually runs the utility. If no objects are selected, this becomes an IEFBR14 job. We enter the member name for the build job as TEST and member name IMAGECP for the utility JCL.

Pressing Enter enables us to submit the build job, member TEST, from our ISPF session. See Figure 6-26 on page 149.

```
File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
sssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
EDIT       ADMR2.AUTO.JCL(TEST) - 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.
000001 //ADMR2S JOB (999,POK),'DB2 UTILITY',
000002 // REGION=0M,NOTIFY=&SYSUID,MSGCLASS=X,CLASS=A
000003 //PROCLIB JCLLIB ORDER=DBOAM.PROCLIB
000004 /*JOBPARM SYSAFF=SC63
000005 //*
000006 //** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * **
000007 //* *
000008 //*  Job Generated by IBM DB2 Automation Tool V4R1.01 *
000009 //* *
000010 //*  DB2 SSID: DBOA *
000011 //*  SQLID:    ADMR2 *
000012 //*  Profile:  ADMR2.IMAGE_COPY *
000013 //*  Desc:     TEST AND IMAGE COPY *
000014 //*  User:     ADMR2 *
000015 //*  Date:     Sunday April 29, 2012 *
000016 //*  Time:     01:32:29.98 *
000017 //* *
000018 //** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * **
000019 //* *
000020 //** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * **
000021 //* *
000022 //*  Step:     HAA@BULD *
000023 //* *
000024 //*  Desc:     This job will generate the JCL for jobs profile *
000025 //*            ADMR2.IMAGE_COPY in a batch mode. *
000026 //*            The generated job will be placed in dataset *
000027 //*            ADMR2.AUTO.JCL(IMAGECP). *
000028 //* *
000029 //*  Return Codes: *
000030 //* *
000031 //*  (00) - Job was built successfully with no warnings or errors *
000032 //* *
000033 //*  (04) - Job was built with warning messages and the Build Job on *
000034 //*         Errors indicator was a "Y" or "W" *
```

Figure 6-26 Job containing the JCL to run the utility

After running the job, we get a condition code 0. If we look at the output from System Display and Search Facility (SDSF) for the job we have just run, we can see that three table spaces are included. These table spaces have met the criteria of the Exception Profile: they do not have a full image copy. See Figure 6-27 on page 150.

```
Control card stream processed by IBM Shared Profile Support...

GENERATE_UTILITY_JOB ( DB2_SUBSYSTEM DBOA USER_INDICATOR HAA DB2_SQLID ADMR2 PRO
PROFILE_DESCRIPTION 'TEST AND IMAGE COPY' EXECUTION_LIB_2 HAA.SHAALOAD EXECUTION
DEBUG_MODE OFF GEN_TO_MEMBER TESTB JOB_CARD_1_1 '//ADMR2S JOB (999,POK),"DB2 UTI
'// REGION=0M,NOTIFY=&SYSUID,MSGCLASS=X,C' JOB_CARD_2_2 'LASS=A' JOB_CARD_3_1 '/
'/*JOBPARM SYSAFF=SC63' )
IBM Shared Profile Support messages follow...

Using JOBS Profile    ADMR2.IMAGE_COPY                          that includes..
      EXCP Profile  DB2AUTH.NEVER COPIED
      OBJS Profile    ADMR2.DSN00037
      OBJS Profile    ADMR2.GLWSAMP
      UTIL Profile    ADMR2.IMAGE_COPY_FC

THE FOLLOWING EXCEPTIONS WERE DETECTED IN THE EXCEPTION PROFILE(S):
      SYSCOPY

SYSCOPY STATISTICS ARE BEING RETRIEVED

Following are the Objects included in generated JCL based on the Utility Profile the
ACCEPTED count includes Objects that met both Exception Profile and Realloca though
there is no EXCEPTION RULE for a Reallocation Utility.  If there is no Ex
All Objects default to ACCEPTED Objects.  PART Level statistics are displayed fo
Number of Accepted Objects.....................................3
Number of Rejected Objects.....................................0
Total Number of Objects Included in Generated JCL..............3
******************************** BOTTOM OF DATA********************************
```

Figure 6-27 Output from phase 1 build job in DB2 Automation Tool

If we had built the job online, the same informational messages are shown, although they are on the panel in the batch run of the TEST job. So, the results from our Exception Profile have identified three table spaces that have no full image copy.

Also, if we now look into the ADMR2.AUTO.JCL, we find the member IMAGECP. This job has been built to create an Image Copy of the three table spaces that have been identified by the exception criteria.

The IMAGECP job has two steps:

- The first step registers this particular utility run with the DB2 Automation Tool job tracking started task, which is, by default, called HAAPROC.
- The second step performs the image copies. We run the IMAGECP job and get return code 0 for both steps.

We can now check that the image copies have been created. In ISPF option 3.4, we can see that a VSAM file was created based on the DSN format that we provided in the Utility Profile. See Figure 6-28 on page 151.

We also know that these files are FlashCopy files because they are VSAM files, and we can see one FlashCopy data set file for each partition within the three table spaces. For this reason, it is important to include the partition number as part of the DSN name.

```
Menu  Options  View  Utilities  Compilers  Help
ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
DSLIST - Data Sets Matching ADMR2.GLWSAMP.**                        Row 1 of 24
Command ===>                                                  Scroll ===> CSR

Command - Enter "/" to select action                 Message           Volume
------------------------------------------------------------------------------
         ADMR2.GLWSAMP.GLWSEMP.T010949.P00001.IC                        *VSAM*
         ADMR2.GLWSAMP.GLWSEMP.T010949.P00001.IC.D                      SBOX8N
         ADMR2.GLWSAMP.GLWSEMP.T010949.P00002.IC                        *VSAM*
         ADMR2.GLWSAMP.GLWSEMP.T010949.P00002.IC.D                      SBOX8O
         ADMR2.GLWSAMP.GLWSEMP.T010949.P00003.IC                        *VSAM*
         ADMR2.GLWSAMP.GLWSEMP.T010949.P00003.IC.D                      SBOX8M
         ADMR2.GLWSAMP.GLWSEMP.T010949.P00004.IC                        *VSAM*
         ADMR2.GLWSAMP.GLWSEMP.T010949.P00004.IC.D                      SBOX8O
         ADMR2.GLWSAMP.GLWSSPL.T010949.P00001.IC                        *VSAM*
         ADMR2.GLWSAMP.GLWSSPL.T010949.P00001.IC.D                      SBOX8O
         ADMR2.GLWSAMP.GLWSSPL.T010949.P00002.IC                        *VSAM*
         ADMR2.GLWSAMP.GLWSSPL.T010949.P00002.IC.D                      SBOX8P
         ADMR2.GLWSAMP.GLWSSPL.T010949.P00003.IC                        *VSAM*
         ADMR2.GLWSAMP.GLWSSPL.T010949.P00003.IC.D                      SBOX8P
         ADMR2.GLWSAMP.GLWSSPL.T010949.P00004.IC                        *VSAM*
         ADMR2.GLWSAMP.GLWSSPL.T010949.P00004.IC.D                      SBOX8M
         ADMR2.GLWSAMP.XGLWOOOO.T010949.P00001.IC                       *VSAM*
         ADMR2.GLWSAMP.XGLWOOOO.T010949.P00001.IC.D                     SBOX8P
         ADMR2.GLWSAMP.XGLWOOOO.T010949.P00002.IC                       *VSAM*
         ADMR2.GLWSAMP.XGLWOOOO.T010949.P00002.IC.D                     SBOX8O
         ADMR2.GLWSAMP.XGLWOOOO.T010949.P00003.IC                       *VSAM*
         ADMR2.GLWSAMP.XGLWOOOO.T010949.P00003.IC.D                     SBOX8O
         ADMR2.GLWSAMP.XGLWOOOO.T010949.P00004.IC                       *VSAM*
         ADMR2.GLWSAMP.XGLWOOOO.T010949.P00004.IC.D                     SBOX8P
```

Figure 6-28 ISPF 3.4 list showing FlashCopy data sets that are created by Image Copy

These are FlashCopy image copies because they are VSAM files. We can see that we have three table spaces and individual VSAM files for each partition of the table space. Also, if we look at the control member that was built by the first job, we can see the utility control statements that identify the three table spaces. See Figure 6-29 on page 152.

```
 Menu  Utilities  Compilers  Help
 sssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
 BROWSE    ADMR2.AUTO.CNTL(IMC00102) - 01.00          Line 00000000 Col 001 080
 Command ===>                                              Scroll ===> CSR
******************************** Top of Data**********************************
   TEMPLATE C1FL0001
            UNIT     SYSDA
            DSN             'ADMR2.&DB..&SN..T&TIME..P&DSNUM..IC'

   LISTDEF CPY001U1
          INCLUDE TABLESPACE GLWSAMP.GLWSEMP
          INCLUDE TABLESPACE GLWSAMP.GLWSSPL
          INCLUDE TABLESPACE GLWSAMP.XGLW0000

   COPY LIST CPY001U1
        FULL        YES

        SHRLEVEL       REFERENCE
        SCOPE          ALL
        FLASHCOPY YES
        FCCOPYDDN (C1FL0001)
```

Figure 6-29 Utility control statements that are built by DB2 Automation Tool for image copy with FlashCopy

We use the DB2 Administration Tool line command to interpret one of the SYSCOPY entries representing one of our FlashCopy Image Copies, as shown in Figure 6-30 on page 153.

```
DB2 Admin --------- DBOA Interpretation of an Object in SYSCOPY --------- 12:30
Option ===>

                                                                More:     +
Recovery details for table space: GLWSEMP In database  : GLWSAMP

Data set number within TS . . . : 0
Recovery status change reason . : Full image copy
Status change date (yymmdd) . . : 120612   Time (hhmmss) : 122638
Status change timestamp . . . . : 2012-06-12-12.26.38.835706
Log RBA at latest status change : 00125CDD0399 (Hex)
Share Level . . . . . . . . . . : C - Change
Data Sharing Member Name  . . . : Not in share mode at time of operation
Image copy data set name  . . . : GLWSAMP.GLWSEMP
Image copy data set device type :
Type of image copy              : FC
Type of media for image copy  . : (cataloged)
Special recovery considerations : STYPE=T
Point in current RBA or LRSN  . : 00125CDD03CD (hex)
Object of recovery info . . . . : T - Table Space
Lowest partition in range . . . : 1
Highest partition in range  . . : 4
Pages written to copy data set  : -1.000000000000000E+00
Pages in TS or I at copy time . : -1.000000000000000E+00
Total number of changed pages . : -1.000000000000000E+00
Job name of utility . . . . . . : ADMR2S
Authorization ID of utility . . : ADMR2

Oldest data version number  . . : 0
Logical partition number  . . . : 0
 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=IFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 6-30 DB2 Administration Tool display of SYSCOPY entry for an image copy

Finally, we can view the history and output of the image copy job by selecting option 6, the Execution Reports option, from the main DB2 Automation Tool menu. This option displays a list of execution reports. See Figure 6-31 on page 154.

We can check the return codes for each step from the jobs that we ran. The purpose of the DB2 Automation Tool HAAPROC started task is to capture and record this information.

```
AUTOTOOL V4R1  ---- Execution Reports Job Display    ---- 2012/04/29  02:00:36
Option  ===>                                                 Scroll ===> CSR
--------------------------------------------------------------------------------

Line Commands: B - Build  D - Delete  O - Objects  R - Restart  S - Steps
 Profile    Like  *                                      DB2 Subsystem: DBOA
 Creator    Like  *        Database Like  *           Row 1 of 1             >
 Space Type =     A  (I - Index, T - Tablespace, A - Any)
 Space      Like  *              Date From  04/26/2012  to  04/30/2012
 Jobname    Like  *              Time From  14:21:21    to  14:21:21
 Jobnum     Like  *              View Type  J  (J - Jobs, O - Objects)
--------------------------------------------------------------------------------

Cmd  Jobname   Jobnum    Completion  Reason Code    Creator   Profile ID
     ADMR2A    JOB05741       R0000     00000000    ADMR2     C97D73B1158F9335
***************************** Bottom of Data**********************************
```

Figure 6-31 List of execution reports

If we select the S line command for Steps, we can see the return codes from each step. See Figure 6-32.

```
AUTOTOOL V4R1   ------------ Job Step Display ----------- 2012/04/29  02:02:45
Option  ===>                                                 Scroll ===> CSR
--------------------------------------------------------------------------------
     Jobname: ADMR2A                                      DB2 Subsystem: DBOA
     Jobnum : JOB05741                                  Row 1 of 2             >
--------------------------------------------------------------------------------

Step# Stepname    Completion  Reason Code    Start Time
   1  REG00101         R0000     00000000    2012-04-28-21.09.49.560000
   2  IMC00102         R0000     00000000    2012-04-28-21.09.49.660000
***************************** Bottom of Data**********************************
```

Figure 6-32 Job step with reason codes

Also, we can rebuild or restart the job if it failed by using the B and R options.

6.2 Automation of REORG and backups using exception reporting

The best utility is one you do not have to run. To determine whether to run a REORG, we need statistics about our objects. We have used Runstats to collect these statistics for some time but at a cost. However, with real-time statistics, we can get an instant picture of the state of our objects and take any necessary action dynamically and automatically.

DB2 Automation Tool enables you to determine which objects or groups of objects need what maintenance using exception criteria. There are 184 different exception criteria within DB2 Automation Tool that we can use. By using the DB2 Administrative Scheduler, we can run exception reports at regular intervals to evaluate objects that require maintenance. After we identify the objects that need maintenance, we can automatically execute a utility job, therefore eliminating unnecessary DBA intervention. The same approach can be used to check whether an Image Copy really needs to run.

In the past, users defined maintenance windows where they run all the REORGs, image copies, Runstats, and other related utilities even if they were not needed. By using DB2 Automation Tool, you can apply filter criteria against the objects that you define to selectively generate utility JCL for only those objects that need maintenance. The result is that processor resources are saved.

6.2.1 Automation of REORG using exception reporting

In this scenario, we run a REORG against our database only if certain criteria are met. The criteria that we use test the real-time statistic (RTS) fields that are found in the RTS SYSTABLESPACESTATS table space. Some of these fields are new in DB2 10.

- REORGCLUSTERSENS

  The number of times that data has been read by SQL statements that are sensitive to the clustering sequence of the data since the last REORG or LOAD REPLACE, or since the object was created.

- REORGSCANACCESS

  The number of times that data is accessed for SELECT, FETCH, searched UPDATE, or searched DELETE since the last CREATE, LOAD REPLACE, or REORG, or since the object was created. A null value indicates that the number of times data is accessed is unknown.

- REORGINDEXACCESS

  The number of times the index was used for SELECT, FETCH, searched UPDATE, searched DELETE, or used to enforce referential integrity constraints, or since the object was created. For hash overflow indexes, this is the number of times DB2 has used the hash overflow index. A null value indicates that the number of times the index has been used is unknown.

We need to build some profiles in DB2 Automation Tool in order to define the objects and the criteria. First, we build the Object Profiles to define the objects in which we are interested. See 6.1, “DB2 Automation Tool: Profiles” on page 129 if you are unfamiliar with building Object Profiles. In our list, we select GLWSAMP as the target for this REORG job. From the View Object Profile Display panel, we can see that GLWSAMP has five table spaces. See Figure 6-33 on page 156.

```
AUTOTOOL V4R1   ---- View Object Profile Display      --- 2012/04/29  15:03:36
Option  ===>                                                 Scroll ===> CSR

--------------------------------------------------------------------------------
Commands: Explode - View all objects.

Creator: ADMR2        Profile: GLWSAMP                        User: ADMR2
Description: OBJECT PROFILE FOR GLWSAMP
Share Option U  (U - Update, V - View, N - No)        Row 1 of 5             >
--------------------------------------------------------------------------------
                                                    Volume /
         Wild ---- Process --- Inc/ IX DB Name/ IX Crtr/ IX Name/
Cmd Type Card IX RI Clone Util Exc  TS Crtr     DB Name  TS Name
    TS   N    N  N  N     N    INC  DB2R2       GLWSAMP  GLWSDPT
    TS   N    N  N  N     N    INC  DB2R2       GLWSAMP  GLWSEMP
    TS   N    N  N  N     N    INC  DB2R2       GLWSAMP  GLWSEPA
    TS   N    N  N  N     N    INC  DB2R2       GLWSAMP  GLWSSPL
    TS   N    N  N  N     N    INC  GLWSAMP     GLWSAMP  XGLWOOOO
****************************** Bottom of Data ********************************
```

Figure 6-33 Viewing Object Profiles in DB2 Automation Tool

Next, we need to select our Utility Profile. Again, if you are unfamiliar with building or selecting the Utility Profile, see 6.1, “DB2 Automation Tool: Profiles” on page 129. We select the standard REORG profile for this run. Figure 6-34 on page 157 shows a view of some of the many options associated with REORG that you can define in the DB2 Automation Tool Utility Profile. We have used the defaults that are provided, but of course, you can change these defaults according to your requirements. For now, a simple REORG is all that is required to get us started.

```
AUTOTOOL V4R1 ------ Reorg Utility Profile Options ------ 2012/04/29  15:11:43
Option   ===>                                                Scroll ===> CSR

  Creator: ADMR2      Name: REORG_V10                       User: ADMR2
                                                                 More:     +
                              ------Include------  ------View-------
   Online reorg  . . . . . . . N (Yes/No) . . . . . N (Yes/No)
   Copy options  . . . . . . . N (Yes/No) . . . . . N (Yes/No)
   Statistics options  . . . . N (Yes/No) . . . . . N (Yes/No)
   Discard . . . . . . . . . . N (Yes/No) . . . . . N (Yes/No)
    View Discard DSN options  . . . . . . . . . . . N (Yes/No)
    Nopad  . . . . . . . . . . N (Yes/No)
   View Sysrec DSN options  . . . . . . . . . . . . N (Yes/No)
   View Syspunch DSN options  . . . . . . . . . . . N (Yes/No)
   Exception Rule  . . . . . . A (A - Accepted, R - Rejected, B - Both)
   Utility ID  . . . . . . . .                  (16 characters)
   Parallel  . . . . . . . . .  (Yes/No/Blank)
   Reuse . . . . . . . . . . . N (Yes/No)
   Log . . . . . . . . . . . . Y (Yes/No)
   Fastswitch  . . . . . . . . N (Yes/No)
   Sortdata  . . . . . . . . . N (Yes/No)
   Scope . . . . . . . . . . . A (A - All, P - Pending)
   Rebalance . . . . . . . . . N (Yes/No)
   Keep Dictionary . . . . . . N (Yes/No)
   Sort Device Type  . . . . . (CART/DISK/etc.)
   Sort Number . . . . . . . . (Number)
   Nosysrec  . . . . . . . . . N (Yes/No)
   Unload Data . . . . . . . . C (C - Continue, E - External,
                                  O - Only, P - Pause)
   Preformat . . . . . . . . . N (Yes/No)
   Rowformat . . . . . . . . .  (B - BRF, R - RRF, Blank)
   Sortkeys  . . . . . . . . . N (Yes/No)
   Offposlimit . . . . . . . .       (Number)
   Indreflimit . . . . . . . .       (Number)
   Report Only . . . . . . . . N (Yes/No)
   Max Concurrent Idx Builds   05   (0-99)
   Max Tape Vols/DASD Units    5    (1-255)
   Decompress alloc multiplier 03   (1-99)
   Group Partitions by . . . . N (J - Job, S - Step, N - None)
   Auto Estimate Hash Space    Y (Yes/No)
   Perform LOB Dependency checks Y (Yes/No)
```

Figure 6-34 The REORG Utility Profile of DB2 Automation Tool

We now want to set up the Exception Profile. DB2 Automation Tool becomes essential because the Exception Profile enables us to tailor the REORG. We specify the exceptions we want to test, as shown in Figure 6-35 on page 158. If you are unfamiliar with building Exception Profiles, see 6.1.3, “DB2 Automation Tool Exception Profiles” on page 141.

```
AUTOTOOL V4R1   ---- View Exceptions Profile Display   --- 2012/04/29 15:20:16
Option  ===>                                                Scroll ===> CSR

                                                    "*" indicates a DAT stat
--------------------------------------------------------------------------------
Creator: ADMR2       Profile: AVOID REORG                      User: ADMR2
--------------------------------------------------------------------------------
Share Option U (U - Update, V - View,  N - No)
Description  CHECK CLUSTERING, INDEX, SCAN        Scroll Right for Column Help
Use Stats From: R (R - Repository,         View Runstats Options:   N (Yes/No)
                   C - Catalog,        Save Triggers in Repository: N (Yes/No)
                   U - Runstats,   WTO number of triggered Objects: N (Yes/No)
                   S - Shadow,   Combine IX/TS Exceptions when
                   H - History)   evaluating an IX triggering a TS: N (Yes/No)
------------------------------------------------------- Row 66 of 206      -+>
S Statistics Type--- *Column------------ Cond ---------Exception Value---------
                     UNCLUST_INS
                     UNCLUST_INS_PCT
                     DISORGED_LOBS
  REALTIME REORG TS  DISORGED_LOBS_PCT
                     RELOCATED_ROWS
                     RELOCATED_ROWS_PCT
                     MASS_DELETES
0                    CLUSTERSENS          >   500
                     HASHACCESS
A                    SCANACCESS           >   1000
  REALTIME REORG IX    STAT
                     DAYS_SINCE_LAST
                     INS_DEL
                     INS_DEL_PCT
                     APPENDED_INS
                     APPENDED_INS_PCT
                     PSEUDO_DEL
                     PSEUDO_DEL_PCT
                     LEAFFAR_SPLITS_PCT
                     NLEAF_SPLITS
                     NLEAF_SPLITS_PCT
                     NUMLEVELS_UPDATED
                     MASS_DELETES
0                    INDEXACCESS          >   1000
  REALTIME RUNSTATS    STAT
 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=IFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP      F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 6-35 Building the REORG Exception Profile by using RTS criteria

We use the following boolean logic for the criteria:

- CLUSTERSENS > 500 AND SCANACCESS > 1000 for a **Table Space** REORG
- INDEXACCESS > 1000 for an **Index** REORG

Of course, some of the criteria might not be applicable to every object in the Object Profile we have selected, but that is acceptable. We can generalize rather than build complex or specific Exception Profiles for every object. It is one of the features of DB2 Automation Tool. Profiles are interchangeable and can be used according to the job or task you are building. Apart from including boolean logic of “AND” “OR” when selecting exception criteria, you can also select more than one Exception Profile when building your Job Profile. The fact that there are no indexes in this database is irrelevant. The next Object Profile that we associated with this

Exception Profile might have indexes or some DB2 10 hash tables. Because this is a general Exception Profile, we leave in the Index Criteria.

One point to make about using multiple Exception Profiles is that if an exception criteria selects or triggers an object for execution by the associated Utility Profile, subsequent Exception Profiles will not “untrigger” the selection.

For example, objects GLWSAMP.ATABLE are defined by an Object Profile OBJS. Selecting objects from this profile are two Exception Profiles, EXA and EXB. If EXA selects or triggers an object, it will not be subjected to the criteria defined in Exception Profile EXB. If you require all the conditions to be true, do not include them in separate Exception Profiles. They should all be included in one Exception Profile by using the 'And' condition.

Finally, we can build the Job Profile that combines the Exception, Utility, and Object Profiles. Again, see 6.1.4, “DB2 Automation Tool Job Profiles” on page 145 if you are unfamiliar with building this type of profile. Figure 6-36 gives us a view of our Job Profile.

```
AUTOTOOL V4R1   ------ View Jobs Profile Display   ------ 2012/04/29  15:30:25
Option  ===>                                                 Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: V - View
--------------------------------------------------------------------------------
Creator: ADMR2        Profile: REORG                          User: ADMR2
--------------------------------------------------------------------------------
Share Option U  (U - Update,     Description USE EXCEPTION TO REDUCE REORG
                 V - View,
                 N - No)
View Job Generation Options   N  (Yes/No)  Row 1 of 3             >
--------------------------------------------------------------------------------
                                                              <-------
Cmd  Type  Order  Name                             Creator   Userid
     OBJS     1   GLWSAMP                          ADMR2     ADMR2
     UTIL     1   REORG_V10                       ADMR2     ADMR2
     EXCP     1   AVOID REORG                     ADMR2     ADMR2
****************************** Bottom of Data ********************************
```

Figure 6-36 View Jobs Profile Display showing profiles for REORG job

We return to the Jobs Profile Display main panel, where we have several options. We want to build the GLWSAMP_REORG job, so we select the B line command against the Job Profile. See Figure 6-37 on page 160.

This line command begins the build process.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2012/06/02  08:54:53
Option  ===>                                                  Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  ADMR2                                 Row 1 of 11           >
--------------------------------------------------------------------------------

Cmd  Name                           Creator   Updt
     DB2SORT_IVP_IC                 ADMR2      U
     DB2SORT_IVP_IC_FC              ADMR2      U
     GLWSAMP_IC                     ADMR2      U
     GLWSAMP_IC_FC_EX               ADMR2      U
     GLWSAMP_IC_FC_NX               ADMR2      U
 b   GLWSAMP_REORG                  ADMR2      U
     GLWSAMP_REORG_EX               ADMR2      U
     GLWSAMP_REORG_EX_FC            ADMR2      U
     GLWSAMP_RUNSTATS               ADMR2      U
     IMAGE_COPY_RX_SLB              ADMR2      U
     RECOVER GLWSAMP                ADMR2      U
****************************** Bottom of Data ********************************
```

Figure 6-37 Building a job from the Jobs Profile Display panel

The Jobs Profile Display panel is shown in Figure 6-38. From this panel, we can decide whether we want to perform the build phase online or as a batch job.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2012/04/29  15:47:56
Option  ===>                                                  Scroll ===> CSR
--------------------------------------------------------------------------------
L Esssssssssssssssssssss Build Job for ADMR2.REORG sssssssssssssssssssssN
  e                                                                      e
- e Build Online or Batch. . O (O - Online, B - Batch)                   e-
  e                                                                      e
  e Edit Generated Job . . . Y (Yes/No)                                  e
- e Schedule Job . . . . . . N (Yes/No)   Update options . . N (Yes/No)  e-
  e                                                                      e
C e Build job in Dataset . . ADMR2.AUTO.JCL                              e
  e               Member . . REORGT                                        e
  e                                                                      e
* e    Job Cards:                                                       e*
  e ==> //ADMR2S JOB (999,POK),'DB2 UTILITY',                           e
  e ==> // REGION=0M,NOTIFY=&SYSUID,MSGCLASS=X,CLASS=A                  e
  e ==> //PROCLIB JCLLIB ORDER=DBOAM.PROCLIB                            e
  e ==> /*JOBPARM SYSAFF=SC63                                           e
  e  F1=HELP     F2=SPLIT    F3=END      F4=RETURN   F5=IFIND    F6=RCHANGE e
  e  F7=UP       F8=DOWN     F9=SWAP    F10=LEFT    F11=RIGHT              e
  DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-38 Jobs Profile Display

We choose the online build and build the job, which executes the REORG. This job is placed in member ADMR2.AUTO.JCL(REORGT). This job is executed if, during the build phase, there are objects (table spaces in our case) that meet the criteria of the Object and Exception Profiles that we have just built. If we get a condition code 6 from the phase 1 build job, there are no qualifying objects and consequently, the job in member REORGT is not built.

The online build has run. Now, we can look at the message log. See Figure 6-39.

We view the Build Process Message Display panel. When we scroll through the messages, we can see that four table spaces out of our five table spaces have been triggered for a REORG. This panel shows the power of using real-time statistics (RTS) as a selection criteria. Although we selected 80% of our sample database table spaces, we nevertheless saved 20%. If we were running this against a database of several hundred or thousand table spaces, the savings can amount to significant time and resource.

```
AUTOTOOL V4R1 ------ Build Process Message Display ------ 2012/04/29  15:53:46
Option  ===>                                                  Scroll ===> CSR
-------------------------------------------------------------------------------
DB2 Subsystem ID: DBOA     Current SQLID:    ADMR2     User:              ADMR2
                                                          Row 34 of 41  - >
      36 Informational        1 Warning        0 Error Messages
                            Order messages by S (S - Sequence, T - Type)
-------------------------------------------------------------------------------
         IX Creator  IndexSpc
 Msg ID  TS Database TableSpc Part# Message
HAAB533I TS GLWSAMP  GLWSDPT  00000 AUTOESTSPACE keyword not supported on non-H
HAAB533I TS GLWSAMP  GLWSEMP   ALL  AUTOESTSPACE keyword not supported on non-H
HAAB533I TS GLWSAMP  GLWSEPA   ALL  AUTOESTSPACE keyword not supported on non-H
HAAB533I TS GLWSAMP  GLWSSPL   ALL  AUTOESTSPACE keyword not supported on non-H
HAAB533I TS GLWSAMP  XGLW0000  ALL  AUTOESTSPACE keyword not supported on non-H
HAAB007I 00004 Objects were triggered by Exception Processing
****************************** Bottom of Data ********************************
```

Figure 6-39 Online build output

So, we can now run the REORG job that has been built for us. We know that four out of the five objects that are selected have been triggered for REORG.

If we look in the SYSTABLESPACESTATS, we can see that the DB2 Automation Tool has selected the correct objects based on the criteria that we defined in the Exception Profile AVOID_REORG. See Figure 6-40 on page 162.

```
 DB2 Admin -- Browse Result of SQL Select              ----- Row 1 to 17 of 17
Command ===>                                                 Scroll ===> PAGE

L NAME         REORGCLUSTERSENS      REORGSCANACCESS PARTITION
  *                           *                    *         *
- -------- -------------------- -------------------- ---------
  GLWSDPT              46428087             46560462         0
  GLWSEMP                 78714                94510         1
  GLWSEMP                 77781               104189         2
  GLWSEMP                 77949                92709         3
  GLWSEMP                 83409               106018         4
  XGLW0000                    0                  723         1
  XGLW0000                    0                 2207         2
  XGLW0000                    0                  360         3
  XGLW0000                    0                  444         4
  GLWSEPA              83413858             83413858         1
  GLWSSPL                     0                    0         1
  GLWSSPL                     0                    0         2
  GLWSSPL                     0                    0         3
  GLWSSPL                  1009                 1009         4
  GLWSEPA              58685068             58685068         2
  GLWSEPA               9595463              9595463         3
  GLWSEPA                     0                    0         4
****************************** END OF DB2 DATA*******************************
```

Figure 6-40 SYSTABLESPACESTATS listing of triggered objects

In the REORGT JCL job built by DB2 Automation Tool, we can see that the following LISTDEF control statements have been generated to select the trigger objects. See Figure 6-41. For partitioned tables spaces, ALL partitions have been triggered for selection in this REORG job. We could have selected individual partitions, if required. This selection criteria is controlled within the Object Profile definition.

```
File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
sssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
EDIT       ADMR2.AUTO.CNTL(RTS00102) - 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 changed to STATS ON (from STATS EXT) because
==MSG>           extended ISPF statistics do not exist for this member.
000001    LISTDEF REO11002
000002            INCLUDE TABLESPACE GLWSAMP.GLWSDPT
000003            INCLUDE TABLESPACE GLWSAMP.GLWSEMP
000004            INCLUDE TABLESPACE GLWSAMP.GLWSEPA
000005            INCLUDE TABLESPACE GLWSAMP.GLWSSPL
000006
000007    REORG TABLESPACE LIST REO11002
000008          SCOPE        ALL
000009          LOG          YES
000010          SORTDATA     NO
000011          SHRLEVEL     NONE
000012          UNLOAD       CONTINUE
000013
****** **************************** Bottom of Data****************************
```

Figure 6-41 Control statements generated by a REORG Utility Profile

We now submit the REORG. *It failed immediately with a JCL error.* Checking the JCL that has been created, we can see that we have some very large space allocations. See Figure 6-42. This is not what we want. Something has gone wrong with our DB2 Automation Tool setup.

```
SDSF EDIT    ADMR2A   (JOB05902) JCLEDIT                  Columns 00001 00072
Command ===>                                               Scroll ===> CSR
000054 //REG#OBJT DD DISP=SHR,DSNAME=ADMR2.AUTO.CNTL(REG00101)
000055 /*
000056 //*
000057 //** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
000058 //*                                                                    *
000059 //*  Step:     RTS00102                                               *
000060 //*                                                                    *
000061 //*  Desc:     This step will invoke the IBM Reorg Tablespace Utility  *
000062 //*                                                                    *
000063 //** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
000064 //*                                                                    *
000065 //RTS00102 EXEC  PGM=DSNUTILB,REGION=1024M,COND=(4,LT),
000066 //          PARM=(DBOA,)
000067 //*
000068 //STEPLIB  DD DSN=HAA.SHAALOAD,DISP=SHR
000069 //         DD DSN=FEC.SFECLOAD,DISP=SHR
000070 //         DD DSN=DBOAT.SDSNEXIT,DISP=SHR
000071 //         DD DSN=DBOAT.SDSNLOAD,DISP=SHR
000072 //SYSOUT   DD SYSOUT=*
000073 //UTPRINT  DD SYSOUT=*
000074 //SYSPRINT DD SYSOUT=*
000075 //*
000076 //SYSREC   DD DSN=ADMR2.ADMR2A.RTS00102.REORG.SYSREC,
000077 //            DISP=(MOD,DELETE,CATLG),
000078 //            UNIT=(SYSALLDA,5),SPACE=(CYL,(21480,10740),,,ROUND)
000079 //SYSUT1   DD DSN=ADMR2.ADMR2A.RTS00102.REORG.SYSUT1,
000080 //            DISP=(MOD,DELETE,CATLG),
000081 //            UNIT=(SYSALLDA,5),SPACE=(CYL,(21480,10740),,,ROUND)
000082 //SORTOUT  DD DSN=ADMR2.ADMR2A.RTS00102.REORG.SORTOUT,
000083 //            DISP=(MOD,DELETE,CATLG),
000084 //            UNIT=(SYSALLDA,5),SPACE=(CYL,(21480,10740),,,ROUND)
000085 //SORTWK01 DD UNIT=(SYSALLDA,5),SPACE=(CYL,(00050,00050),,,ROUND)
000086 //SORTWK02 DD UNIT=(SYSALLDA,5),SPACE=(CYL,(00050,00050),,,ROUND)
000087 //SORTWK03 DD UNIT=(SYSALLDA,5),SPACE=(CYL,(00050,00050),,,ROUND)
000088 //SORTWK04 DD UNIT=(SYSALLDA,5),SPACE=(CYL,(00050,00050),,,ROUND)
000089 //SORTWK05 DD UNIT=(SYSALLDA,5),SPACE=(CYL,(00050,00050),,,ROUND)
000090 //SYSIN    DD DISP=SHR,DSNAME=ADMR2.AUTO.CNTL(RTS00102)
000091 /*
000092 //*
```

Figure 6-42 Incorrect setup parameters

We discover the problem in the DB2 Automation Tool setup. We have incorrectly defined the primary and secondary space allocations for our environment. We update the Setup Menu, rebuild the job, and resubmit. See Figure 6-43 on page 164.

```
AUTOTOOL V4R1 ---- Shared Profile Parameters for DBOA --- 2012/04/29  17:23:48
Command ==>                                                Scroll ===> CSR

Enter or Update Specific Shared Profile Support Parameters   User id: HAA
 Catalog/History PackageList . . HAAC41OC
 Shadow Catalog PackageList  . . HAAC41OS
 Repository PackageList  . . . . HAAC41O
 Work File Unit Device . . . . . SYSALLDA (SYSDA, DISK, etc.)
 Sort Work File Unit Device  . . SYSALLDA (SYSDA, DISK, etc.)
 Build Informational Message DD  HAAERROR (8 character name)
 Build Warning       Message DD  HAAERROR (8 character name)
 Build Error         Message DD  HAAERROR (8 character name)
 Job Tracking Subsystem Name . . DBA      (Automation Tool Tracking STC SSID)
 Max Primary Space Allocation    002000   (1-999999) C (Trks/Cyls/Mbytes)
 Secondary Allocation Percent    050      (1-999) % of Primary Allocation
 Utility REGION Size . . . . . . 1024     (0-2047)    M (Megabytes)
 DB2 Fetch Buffer Size . . . . . 0004     (1-256)     M (Megabytes)
 Parallel MVS Catalog Locates    10       (1-99)
 Terminate Utility if an ABEND   N        (Y/N)
 Generate STEPLIB DDs  . . . . . Y        (Y/N)
 Gen Image Copy DSNs in GMT  . . Y        (Y/N)
 Enable Admin Scheduler Support  Y        (Y/N)
   Admin Scheduler Max History   0010     (1-9999)

Entering the following fields will override the calculated amount of Sort
Work DD's space quantities and/or the number of DD's generated in the job
 Primary Sort Work Space . . . . 00350    (1-99999)  C (Cyls)
 Secondary Sort Work Space . . . 00150    (1-99999)  C (Cyls)
 Number of Sort Work DDs . . . . 05       (1-99)
```

Figure 6-43 Correct setup parameters for the space allocation

The job seems to be running correctly, until we suddenly get an abend SB37:

```
B37-04,IFG0554A,ADMR2A,RTS00102,SYSREC,8313,SBOX10,041A041D,ADMR2.ADMR2
```

We look into the Exception Report menu, which is option 6 from the main DB2 Automation Tool panel, and we can see that the Job Tracking Subsystem has captured our job. See Figure 6-44 on page 165.

The first step of any job built by DB2 Automation Tool is a registration step. This step enables DB2 to trap output of the job being submitted, making the job details available to view through the Exception Report option. We look at the output and determine that there is insufficient space for SYSREC on DASD. This detail is kept in the Utility Profile REORG_V10.

```
AUTOTOOL V4R1  ---- Execution Reports Job Display    ---- 2012/04/29  16:58:50
Option  ===>                                                 Scroll ===> CSR
--------------------------------------------------------------------------------

Line Commands: B - Build  D - Delete  O - Objects  R - Restart  S - Steps
 Profile    Like  *                                      DB2 Subsystem: DBOA
 Creator    Like  ADMR2    Database Like  *           Row 1 of 4             >
 Space Type =     A  (I - Index, T - Tablespace, A - Any)
 Space      Like  *               Date From  04/26/2012  to  04/30/2012
 Jobname    Like  *               Time From  14:21:21    to  14:21:21
 Jobnum     Like  *               View Type  J  (J - Jobs, O - Objects)
--------------------------------------------------------------------------------

Cmd  Jobname   Jobnum    Completion  Reason Code    Creator   Profile ID
     ADMR2A    JOB05741       R0000     00000000    ADMR2     C97D73B1158F9335
     ADMR2A    JOB05902       R0000     00000000    ADMR2     C97E51845B310B35
     ADMR2A    JOB05904       R0000     00000000    ADMR2     C97E51845B310B35
     ADMR2A    JOB05906       S0B37     00000004    ADMR2     C97E51845B310B35
****************************** Bottom of Data ********************************
```

Figure 6-44 Exception Reports Job Display - captured jobs from DB2 Automation Tool

We have modified our Utility Profile so that the DD SYSREC has additional DASD space. We restart the job and see what happens. We use the line command B to rebuild the job online. Again, we look at the messages from the online build. At the bottom, we see that there are now only two table spaces that meet the criteria. So, our previous job has reorganized two of the triggered objects. We check that by looking at the SYSTABLESPACESTATS. We open the DB2 Administration Tool and display the table. See Figure 6-45.

```
 DB2 Admin -- Browse Result of SQL Select              ----- Row 1 to 17 of 17
Command ===>                                                 Scroll ===> CSR

L NAME         REORGCLUSTERSENS      REORGSCANACCESS PARTITION
  *                           *                    *         *
- -------- -------------------- -------------------- ---------
  GLWSDPT                    0                    0         0
  GLWSEMP                    0                    0         1
  GLWSEMP                    0                    0         2
  GLWSEMP                    0                    0         3
  GLWSEMP                    0                    0         4
  XGLWOOOO                   0                  723         1
  XGLWOOOO                   0                 2207         2
  XGLWOOOO                   0                  360         3
  XGLWOOOO                   0                  444         4
  GLWSEPA             83413858             83413858         1
  GLWSSPL                    0                    0         1
  GLWSSPL                    0                    0         2
  GLWSSPL                    0                    0         3
  GLWSSPL                 1009                 1009         4
  GLWSEPA             58685068             58685068         2
  GLWSEPA              9595463              9595463         3
  GLWSEPA                    0                    0         4
******************************* END OF DB2 DATA ******************************
```

Figure 6-45 Updated SYSTABLESPACESTATS list showing objects after REORG

As we can see, the first two table spaces now have CLUSTERSENS or SCANACCESS values less than the criteria defined in our Exception Profile. The previous REORG has optimized these two table spaces. Of course, we still have to resubmit the rebuilt REORG to hopefully finish the job. This time, the job finishes successfully. So, we experienced a few setup glitches but a successful REORG.

In the previous scenarios, we have scheduled the jobs manually. DB2 Automation Tool also allows you to schedule the jobs through the DB2 administrative task scheduler. This is controlled through the Schedule Jobs option on the Jobs Profile Display. See Figure 6-38 on page 160. Using the Schedule Jobs option we can schedule to run the job at a predetermined time, such as during the batch maintenance window. When we select to run under the scheduler, we are presented with the standard time frame panel that DB2 Automation Tool uses.

```
AUTOTOOL V4R1   -------- Schedule DB2 Admin Task -------- 2012/06/02  13:23:16
Option   ===>                                          Scroll ===> CSR

Task Name  . . . . AUTOTOOL UTIL: &JOBNAME                          >
Task Description                                                    >

Begin Timestamp  . . &CURRENT                   (DB2 Timestamp)
End Timestamp  . . . &CURRENT + 5 MINUTES       (DB2 Timestamp)
Max Invocations  . . 1          (Integer, Blank)
SSID . . . . . . . . DBOA       (Blank for any datasharing member)
Job Wait . . . . . . Y          (Y - Yes, N - No, P - Purge)
Execution Threads. . 001        (Integer)

Invocation Options:
  Interval (minutes)            (Integer, Blank)
          -Or-
  Trigger:
    Task Name . . .                                                 >
    Cond  . . . . .             (GT,GE,EQ,LT,LE,NE)
    Code  . . . . .             (Integer, Blank)
          -Or-
  Point in Time . .                                                 >
```

Figure 6-46 DB2 Automation Tool schedule panel

6.2.2 Controlling backups by using DB2 Automation Tool

In a previous section, we built a Utility Profile to run FlashCopy image copies and an Exception Profile to make the Utility Profile run only for objects that do not have a Full Image Copy. In this chapter, we build a complete backup scenario that will ensure the following copies for our GLWSAMP database objects:

- A full image copy is taken every Sunday.
- Incremental copies are taken daily if the percent changes of is greater than 10% according to real-time statistics.
- A full image copy is taken on new objects regardless of when they are created.
- A full image copy is taken for all objects in copy-pending.

6.2.3 Daily and weekly Utility Profiles

Since we have already created a full image copy Utility Profile, on the next pages, we show you how to create the incremental image copy profile.

To create the new profile, from the DB2 Automation Tool panel, we select option 2, as shown in Figure 6-47.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2013/03/06  14:37:27
Option  ===>
--------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Disaster Recovery
                3 - Exception Profiles      11 - Stand Alone Utilities
                4 - Job Profiles            12 - DB2 Admin Scheduler
                5 - Quick Build             13 - DB2 Autonomic Statistics
                6 - Execution Reports        X - Exit
                7 - DB2 Command Processor

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DBOA          (1-4 Character Subsystem ID or ? for list)
 Current SQLID:                  User: ADMR3    - HAA
```

Figure 6-47 Selecting Utility Profiles

In the next panel, we selected the asterisk (*) for both the Creator and Profile Like options to see all the profiles that are defined, as shown in Figure 6-48.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2013/03/06  14:38:50
Option  ===> 2
--------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Disaster Recovery
                3 - Exception Profiles      11 - Stand Alone Utilities
              Essssss Enter Utilities Profile Like to Display ssssssN
              e                                                    e ics
              e    Profile Like  *                                 e
              e    Creator Like  *                                 e
              e                                                    e
------------ e                                                    e ----------
 DB2 Subsyst DssssssssssssssssssssssssssssssssssssssssssssssssssssM list)
 Current SQLID:                  User: ADMR3    - HAA
```

Figure 6-48 Utility Profiles selection criteria

This action shows all the defined profiles, as shown in Figure 6-49 on page 168.

```
AUTOTOOL V4R1   ------- Utilities Profile Display ------- 2013/02/27  12:25:22
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import
               Q - Quick   U - Update  V - View    J - Jobs    R - Rename
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                    Row 1 of 24            >
--------------------------------------------------------------------------------

Cmd  Name                            Creator   Updt
     CHECK DATA                      DB2AUTH    V
     COPY TO COPY                    DB2AUTH    V
     COPY TO DASD                    DB2AUTH    V
     COPY TO TAPE                    DB2AUTH    V
     IC TO DISK                      ADMR4      U
     IC&REORG                        ADMR3      U
     IMAGE_COPY                      ADMR2      U
     IMAGE_COPY_FC                   ADMR2      U
     MODIFY RECOVERY                 DB2AUTH    V
     MODIFY_DELETE                   ADMR2      U
     ONLINE IX REORG                 DB2AUTH    V
     ONLINE TS REORG                 DB2AUTH    V
     QUIESCE                         DB2AUTH    V
     RECOVER                         DB2AUTH    V
     REORG V10                       ADMR3      U
     REPAIR SET NOCOPYPEND           DB2AUTH    V
     REPOSITORY MAINTENANCE          DB2AUTH    V
     RESIZE                          DB2AUTH    V
     RUNSTATS                        ADMR4      U
     RUNSTATS CATALOG                DB2AUTH    V
     RUNSTATS REPOSITORY             DB2AUTH    V
     STANDARD IX REORG               DB2AUTH    V
     STANDARD TS REORG               DB2AUTH    V
     VERIFY                          DB2AUTH    V
****************************** Bottom of Data ********************************
```

Figure 6-49 Utility Profiles already defined

We typed C in front of any defined profile to create a new profile, as shown in Figure 6-50 on page 169.

```
AUTOTOOL V4R1   ------- Utilities Profile Display ------- 2013/02/27  12:25:22
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import
               Q - Quick   U - Update  V - View    J - Jobs    R - Rename
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                    Row 1 of 24            >
--------------------------------------------------------------------------------

Cmd  Name                              Creator   Updt
 C   CHECK DATA                        DB2AUTH    V
     COPY TO COPY                      DB2AUTH    V
     COPY TO DASD                      DB2AUTH    V
     COPY TO TAPE                      DB2AUTH    V
     IC TO DISK                        ADMR4      U
     IC&REORG                          ADMR3      U
     IMAGE_COPY                        ADMR2      U
     IMAGE_COPY_FC                     ADMR2      U
     MODIFY RECOVERY                   DB2AUTH    V
     MODIFY_DELETE                     ADMR2      U
     ONLINE IX REORG                   DB2AUTH    V
     ONLINE TS REORG                   DB2AUTH    V
     QUIESCE                           DB2AUTH    V
     RECOVER                           DB2AUTH    V
     REORG V10                         ADMR3      U
     REPAIR SET NOCOPYPEND             DB2AUTH    V
     REPOSITORY MAINTENANCE            DB2AUTH    V
     RESIZE                            DB2AUTH    V
     RUNSTATS                          ADMR4      U
     RUNSTATS CATALOG                  DB2AUTH    V
     RUNSTATS REPOSITORY               DB2AUTH    V
     STANDARD IX REORG                 DB2AUTH    V
     STANDARD TS REORG                 DB2AUTH    V
     VERIFY                            DB2AUTH    V
****************************** Bottom of Data ********************************
```

Figure 6-50 Creating a new Utility Profile

On the next panel, we entered the profile name, description, and also the update option. If we choose U, the Utility Profile can be updated by other users. If we choose V, updates will not be allowed for other users. If we choose N, even views will not be allowed for other users.

We selected U, as shown in Figure 6-51 on page 170.

```
AUTOTOOL V4R1   ------- Utilities Profile Display ------- 2013/02/27  12:25:22
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import
               Q - Quick   U - Update  V - View    J - Jobs    R - Rename
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                    Row 1 of 24           >
------------ Essssssssssssss Enter New Utilities Profile Data sssssssssssssN
             e                                                              e
Cmd  Name    e  Creator       ADMR3                                         e
 C   CHECK D e                                                              e
     COPY TO e  Profile Name  IC_NEW                                        e
     COPY TO e                                                              e
     COPY TO e  Description   INCREMENTAL IC                                e
     IC TO D e                                                              e
     IC&REOR e  Update Option U  (U - Update, V - View only, N - No access) e
     IMAGE_C e                                                              e
     IMAGE_C e                                                              e
     MODIFY  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     MODIFY_DELETE                  ADMR2      U
     ONLINE IX REORG                DB2AUTH    V
     ONLINE TS REORG                DB2AUTH    V
     QUIESCE                        DB2AUTH    V
     RECOVER                        DB2AUTH    V
     REORG V10                      ADMR3      U
     REPAIR SET NOCOPYPEND          DB2AUTH    V
     REPOSITORY MAINTENANCE         DB2AUTH    V
     RESIZE                         DB2AUTH    V
     RUNSTATS                       ADMR4      U
     RUNSTATS CATALOG               DB2AUTH    V
     RUNSTATS REPOSITORY            DB2AUTH    V
     STANDARD IX REORG              DB2AUTH    V
     STANDARD TS REORG              DB2AUTH    V
     VERIFY                         DB2AUTH    V
****************************** Bottom of Data ********************************
```

Figure 6-51 Utility Profile information

On the next panel, we select which utilities will be within the profile and whether they can be updated. In our case, we selected Image Copy with Yes on Update Utility, as shown in Figure 6-52 on page 171.

```
AUTOTOOL V4R1 --------- Utility Profile Options --------- 2013/02/27  12:49:05
Option   ===>

  Creator: ADMR3      Profile: IC_NEW                          User: ADMR3
  Description  INCREMENTAL IC
  Share Option U (U - Update, V - View, N - No)


                                Include      Update
                                Utility      Utility
Data Page Verification Reporting N (Yes/No)   N (Yes/No)
Reallocation . . . . . . . . . . N (Yes/No)   N (Yes/No)
Recover  . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Image Copy . . . . . . . . . . . Y (Yes/No)   Y (Yes/No)
Recovery Expert Image Copy . . . N (Yes/No)   N (Yes/No)
Copy to Copy . . . . . . . . . . N (Yes/No)   N (Yes/No)
Runstats . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
TS Reorg . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
IX Reorg . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Quiesce  . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Modify . . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Repair . . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Check Data . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Rebind . . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
```

Figure 6-52 Image Copy Utility selection

On the next panel, we enter the image copy options. Since we only want incremental image copies, we typed N for the Full Image Copy option. In order for the Utility Profile to accept the rules that will be created in the Exception Profile next, we selected A for Exception Rule, as shown in Figure 6-53 on page 172.

```
AUTOTOOL V4R1 ------------ Image Copy options ----------- 2013/02/27  12:56:16
Option   ===>                                                Scroll ===> PAGE
  Creator: ADMR3      Name: IC_NEW                          User: ADMR3

   Exception Rule . . . . . . . . . . A (A - Accepted, R - Rejected, B - Both)
   Image Copy Utility mode  . . . . D (D - DB2, S - Symmetrix, E - Ess)
     Alter EMC Symm/IBM ESS Optns   N (Yes/No)
   Alter Image Copy DSN specs . . . Y (Yes/No)
   Utility ID . . . . . . . . . . .                  (16 characters)

   Parallel . . . . . . . . . . . . N (Yes/No)
     Number of objects  . . . . . .    (0 - 99)
     Number of tape units . . . . .    (0 - 99)
   Filter DDname  . . . . . . . . .          (8 character DD name)
   Sharelevel . . . . . . . . . . . R (R - Reference, C - Change)
   Full Image Copy  . . . . . . . . N (Yes/No)
   Check Page . . . . . . . . . . . N (Yes/No)
   Concurrent . . . . . . . . . . . N (Yes/No)
   Change Limit
     First Percent  . . . . . . . .      (% value)
     Second Percent . . . . . . . .      (% value)
     Report only  . . . . . . . . . N (Yes/No)
   Max Tape Volume/DASD Unit Cnt    5   (1-255 volumes)
   Stack Copy Control Cards . . . . Y (Yes/No)
   Scope  . . . . . . . . . . . . . A (A - All, P - Pending)

 Optional Skeletals                -BEFORE-     -AFTER-
   JCL Skeletal . . . . . . . . .          . .          (8 Character Name)
   Control Cards Skeletal . . . .          . .          (8 Character Name)
   Step End Skeletal  . . . . . .          . .          (8 Character Name)
```

Figure 6-53 Image Copy options

Since it is only an example to illustrate the feature, on the next panel, we chose the Local Primary option, as shown in Figure 6-54.

```
AUTOTOOL V4R1 ------------ Image Copy Options ----------- 2013/02/27  13:01:59
Option   ===>

  Creator: ADMR3      Name: IC_NEW                          User: ADMR3

  Enter the Image Copy options to associate with this utility profile

                               Take Image Copy    Update Options

   Local Primary  . . . . . . . Y (Yes/No) . . . . Y (Yes/No)
   Local Backup . . . . . . . . N (Yes/No) . . . . N (Yes/No)
   Recovery Site Primary  . . . N (Yes/No) . . . . N (Yes/No)
   Recovery Site Backup . . . . N (Yes/No) . . . . N (Yes/No)
   FlashCopy  . . . . . . . . . N (Yes/No) . . . . N (Yes/No)
```

Figure 6-54 Local Primary Image Copy selection

The next panel has the image copy data set options. You can select SMS options as Data Class or Storage Class that satisfy your environment's rules. However, for our example, we only selected CART for Unit Type. Then, the data set will be created on tape. We selected 120 for the Retention period, as shown in Figure 6-55.

```
AUTOTOOL V4R1 ------------ Image Copy Options ----------- 2013/02/27  13:07:43
Option   ===>
 Creator: ADMR3      Name: IC_NEW                            User: ADMR3

 Image Copy options for Image Copy Local Primary
 Use Threshold Unit if allocated space exceeds x Meg/Gig/Trks/Cyls  Optional
                                                    Quantity M|G|T|C
                           Std Unit       Threshold Unit
 Update DSN create spec . . Y  . . . . . . N  (Yes/No)
 Unit Type  . . . . . . . . CART     . . .           (CART - DISK - etc.)
 Catalog Options
 DISP=Status  . . . . . . . M  . . . . . . M  (M - MOD, N - NEW,
                                               O - OLD, S - SHR)
      Normal Termination    C  . . . . . . C  (C - CATLG, D - DEL,
                                               K - KEEP, U - UNCATLG)
      Abnormal Termination  C  . . . . . . C  (C - CATLG, D - DEL,
                                               K - KEEP, U - UNCATLG)
 Data Class . . . . . . . .          . . .           (8 character class)
 Storage Class  . . . . . .          . . .           (8 character class)
 Management Class . . . . .          . . .           (8 character class)
   Parameters Only required if Unit Type is a Tape device:
 Expiration date *or* . . .          . . .           (YYYYDDD - YYDDD)
 Retention period . . . . . 120  . . . . .           (4 digit number)
```

Figure 6-55 Local Primary Image Copy Options display

The next panel allows us to determine the format of the IC data set. We can use a variety of codes to help format the DSN specification. We selected DB0AS.&DB..&SN..T&TIME..IC, as shown in Figure 6-56 on page 174.

```
AUTOTOOL V4R1 ------- LP Image Copy DSN Generation ------ 2013/02/27  13:14:14
Option   ===>
 Creator: ADMR3      Name: IC_NEW                          User: ADMR3
Image Copy Local Primary Non-Threshold
 Qualifier code      Free form literal             Show DSN  N
 GDG Limit . . .     (1-255)
 Current dataset name generation qualifier string:
DBOAS.&DB..&SN..T&TIME..IC
Valid dataset name generation codes are
(* marked items are not supported in IC dynamic dataset generation.)
   1. Database               13. Month (MM)        25. Primary/Backup (P/B)
   2. Space Name             14. Day (DD)          26. ICTYPE
   3. Partition/DSNUM        15. Julian Day (DDD)  27. Utility Name
*  4. Volser                 16. Hours (HH)        28. Job Name
*  5. Partition/DSNUM        17. Minutes (MM)      29. Step Name
      only when partitioned  18. Seconds (SS)      30. Utility ID
*  7. Vcatname            *  19. Timestamp         31. Listdef
   8. Subsystem ID        *  20. Random Number     32. Sequence
*  9. User ID                21. Unique            33. Substring Qualifier
  10. Time (HHMMSS)       *  22. GDG (+1)..(+n)    34. Use freeform literal
  11. Date (YYYYDDD)         23. ICBACKUP (#24.#25)
  12. Year (YYYY)            24. Local/Recovery (L/R)
```

Figure 6-56 Image Copy Dataset Name options

That completes the build of the Image Copy Utility Profile. We press PF3 to return to the main menu to start building the Exception Profiles.

For the weekly full image copy scenario, we will use the IMAGE_COPY_FC Utility Profile that was created in 6.1.2, “DB2 Automation Tool Utility Profiles” on page 134.

6.2.4 Image copy Exception Profile

The next step is to set an Exception Profile to check whether the updated pages % occurred, according to real-time statistics.

We start on the DB2 Automation Tool panel, where we select option 3, Exception Profiles, as shown in Figure 6-57 on page 175.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2013/02/27  13:57:34
Option  ===> 3
--------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Disaster Recovery
                3 - Exception Profiles      11 - Stand Alone Utilities
                4 - Job Profiles            12 - DB2 Admin Scheduler
                5 - Quick Build             13 - DB2 Autonomic Statistics
                6 - Execution Reports        X - Exit
                7 - DB2 Command Processor

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DBOA         (1-4 Character Subsystem ID or ? for list)
 Current SQLID:                 User: ADMR3   - HAA
```

Figure 6-57 Exception Profiles main menu selection

On the next panel, we use the asterisk (*) for the Profile and Creator Like options to get all profiles that were set on the environment and press Enter, as shown in Figure 6-58.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2013/02/27  13:57:34
Option  ===> 3
--------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Disaster Recovery
                3 - Exception Profiles      11 - Stand Alone Utilities
             Esssss Enter Exceptions Profile Like to Display ssssssN
             e                                                    e ics
             e    Profile Like  *                                 e
             e    Creator Like  *                                 e
             e                                                    e
------------ e                                                    e ----------
 DB2 Subsyst DssssssssssssssssssssssssssssssssssssssssssssssssssssM list)
 Current SQLID:                 User: ADMR3   - HAA
--------------------------------------------------------------------------------
```

Figure 6-58 Exception Profile selection

The next panel shows all defined Exception Profiles and the options that are available to manage these profiles or to create a new one. We typed C in front of one of these profiles to create a new one, as shown in Figure 6-59 on page 176.

```
AUTOTOOL V4R1   ------- Exceptions Profile Display ------ 2013/02/27  14:07:51
Option  ===>                                                Scroll ===> PAGE
Line Commands: C - Create  D - Delete  E - Export  I - Import
               U - Update  V - View    J - Jobs    R - Rename
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                   Row 1 of 10           >
--------------------------------------------------------------------------------
 Display Statistics Reports and/or Perform Statistics Maintenance? N (Yes/No)
--------------------------------------------------------------------------------

Cmd  Name                            Creator   Updt
 C   AVOID REORG                     ADMR2      U
     CLUSTERRATIO VALUE RANGE        DB2AUTH    V
     COPY CONDITIONS                 DB2AUTH    V
     LARGE OBJECTS                   DB2AUTH    V
     MISSING STATS                   ADMR4      U
     NEVER COPIED                    DB2AUTH    V
     REORG CONDITIONS                DB2AUTH    V
     SATURDAY ONLY                   DB2AUTH    V
     SMALL OBJECTS                   DB2AUTH    V
     TEST_AUTO                       ADMR2      U
****************************** Bottom of Data ********************************
```

Figure 6-59 Creating an Exception Profile

The next panel requires information that is related to the new Exception Profile, as shown in Figure 6-60.

```
AUTOTOOL V4R1   ------- Exceptions Profile Display ------ 2013/02/27  14:07:51
Option  ===>                                                Scroll ===> PAGE
Line Commands: C - Create  D - Delete  E - Export  I - Import
               U - Update  V - View    J - Jobs    R - Rename
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                   Row 1 of 10           >
--------------------------------------------------------------------------------
 Display St Esssssssssssssss Enter New Exceptions Profile Data sssssssssssssssN
---------- e                                                                   e
           e    Creator . . . ADMR3                                            e
Cmd  Name  e                                                                   e
 C   AVOID  e    Profile Name  AVOID_IC                                        e
     CLUSTE e                                                                  e
     COPY C e    Description                                                   e
     LARGE  e                                                                  e
     MISSIN e    Update Option U  (U - Update, V - View only, N - No access)   e
     NEVER  e                                                                  e
     REORG  e                                                                  e
     SATURD DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     SMALL OBJECTS                   DB2AUTH    V
     TEST_AUTO                       ADMR2      U
****************************** Bottom of Data ********************************
```

Figure 6-60 Exception Profile description

The next panel shows all the exception options that DB2 Automation Tool provides. There are several options that can be chosen according to user needs, as shown in Figure 6-61. We selected all the days of the week, except Sunday, because in our scenario, full image copy will run on Sunday.

```
Option  ===>                                                  Scroll ===> PAGE
Line Commands: A - And  O - Or  S - Select  D - Deselect  R - Repeat
   CONDitions: LT|<|LE|<=|EQ|=|GT|>|GE|>=|NE|¬=|<> "*" indicates a DAT stat
------------------------------------------------------------------------------
Creator: ADMR3       Profile: AVOID_IC                       User: ADMR3
------------------------------------------------------------------------------
Share Option U (U - Update, V - View,  N - No)
Description                                      Scroll Right for Column Help
Use Stats From: R (R - Repository,         Update Runstats Options: N (Yes/No)
                   C - Catalog,        Save Triggers in Repository: N (Yes/No)
                   U - Runstats,   WTO number of triggered Objects: N (Yes/No)
                   S - Shadow,   Combine IX/TS Exceptions when
                   H - History)   evaluating an IX triggering a TS: N (Yes/No)
------------------------------------------------------- Row 1 of 206        +>
S Statistics Type--- *Column------------ Cond ---------Exception Value---------
S DAY OF WEEK         MONDAY         And   EQ
S                     TUESDAY         Or   EQ
S                     WEDNESDAY       Or   EQ
S                     THURSDAY        Or   EQ
S                     FRIDAY          Or   EQ
S                     SATURDAY        Or   EQ
                      SUNDAY
  DAY OF MONTH        NTH_MONDAY
                      NTH_TUESDAY
                      NTH_WEDNESDAY
                      NTH_THURSDAY
                      NTH_FRIDAY
                      NTH_SATURDAY
                      NTH_SUNDAY
                      NTH_DAY
                      LAST_DAY
                      DAY_MONTH                  /    DD/MM
  TIME OF DAY         TIME_FROM                  :      M
                      TIME_TO                    :      M
------------------------------------------------------------------------------
  OBJECT              EXCLUDE|ONLY
                      LOB
                      PGSIZE_32K
                      PBG_TS
                      AND|OR
```

Figure 6-61 Exception options

We press PF8 until we find the other desired option, which is UPDATED_PAGES_PCT in our scenario. We created the Exception Profile to make the Image Copy run only if the updated pages % (UPDATED_PAGES_PCT) is greater than 10, as shown in Figure 6-62 on page 178.

```
AUTOTOOL V4R1   ---- Update Exceptions Profile Display --- 2013/03/04 14:36:08
Option  ===>                                                Scroll ===> PAGE
Line Commands: A - And  O - Or  S - Select  D - Deselect  R - Repeat
   CONDitions: LT|<|LE|<=|EQ|=|GT|>|GE|>=|NE|¬=|<> "*" indicates a DAT stat
------------------------------------------------------------------------------
Creator: ADMR3       Profile: AVOID_IC                       User: ADMR3
------------------------------------------------------------------------------
Share Option U (U - Update, V - View,  N - No)
Description                                        Scroll Right for Column Help
Use Stats From: R (R - Repository,         Update Runstats Options: N (Yes/No)
                   C - Catalog,        Save Triggers in Repository: N (Yes/No)
                   U - Runstats,   WTO number of triggered Objects: N (Yes/No)
                   S - Shadow,   Combine IX/TS Exceptions when
                   H - History)   evaluating an IX triggering a TS: N (Yes/No)
------------------------------------------------------- Row 51 of 206      -+>
S Statistics Type--- *Column------------ Cond ---------Exception Value---------
  INDEXSPACESTATS     NLEAF
  REALTIME ICOPY        STAT
                      REORG_OR_LOAD
                      REORG_LOAD_STATS
                      DAYS_SINCE_LAST
                      UPDATED_PAGES
A                     UPDATED_PAGES_PCT   GT  10
                      COPY_CHANGES
                      COPY_CHANGES_PCT
  REALTIME REORG TS     STAT
                      DAYS_SINCE_LAST
                      DAYS_SINCE_HASH
                      DATAISMORETHANHASH
                      INS_UPD_DEL
                      INS_UPD_DEL_PCT
                      UNCLUST_INS
                      UNCLUST_INS_PCT
                      DISORGED_LOBS
                      DISORGED_LOBS_PCT
                      RELOCATED_ROWS
                      RELOCATED_ROWS_PCT
                      MASS_DELETES
                      CLUSTERSENS
                      HASHACCESS
                      SCANACCESS
```

Figure 6-62 Defining Exception Profile criteria

We press PF3 and receive a confirmation message, as shown in Figure 6-63 on page 179.

```
AUTOTOOL V4R1   ------- Exceptions Profile Display ------ 2013/02/27  14:24:36
Option  ===>                                                Scroll ===> PAGE
Line Comma EssssssssssssssssssssssssssssssssssssssssssssssssN ort
           e HAAM09OI - Profile "ADMR3.AVOID_IC" saved. e ame
---------- DsssssssssssssssssssssssssssssssssssssssssssssssM ---------------------
 Profile Like  *                                             DB2 Subsystem: DBOA
 Creator Like  *                                      Row 1 of 11            >
------------------------------------------------------------------------------
 Display Statistics Reports and/or Perform Statistics Maintenance? N (Yes/No)
------------------------------------------------------------------------------

Cmd  Name                            Creator   Updt
     AVOID REORG                     ADMR2      U
     AVOID_IC                        ADMR3      U
     CLUSTERRATIO VALUE RANGE        DB2AUTH    V
     COPY CONDITIONS                 DB2AUTH    V
     LARGE OBJECTS                   DB2AUTH    V
     MISSING STATS                   ADMR4      U
     NEVER COPIED                    DB2AUTH    V
     REORG CONDITIONS                DB2AUTH    V
     SATURDAY ONLY                   DB2AUTH    V
     SMALL OBJECTS                   DB2AUTH    V
     TEST_AUTO                       ADMR2      U
***************************** Bottom of Data ********************************
```

Figure 6-63 Profile creation confirmation message

The next step is to build the Exception Profile that triggers the IMAGE_COPY_FC Utility Profile to run every Sunday.

On the Exceptions Profile Display panel, we typed C in front of a profile in order to create a new profile, as shown in Figure 6-64 on page 180.

```
AUTOTOOL V4R1   ------- Exceptions Profile Display ------ 2013/03/05  13:45:35
Option  ===>                                                Scroll ===> PAGE
Line Commands: C - Create  D - Delete  E - Export  I - Import
               U - Update  V - View    J - Jobs    R - Rename
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                    Row 1 of 11            >
--------------------------------------------------------------------------------
 Display Statistics Reports and/or Perform Statistics Maintenance? N (Yes/No)
--------------------------------------------------------------------------------

Cmd  Name                            Creator   Updt
     AVOID REORG                     ADMR2      U
 C   AVOID_IC                        ADMR3      U
     CLUSTERRATIO VALUE RANGE        DB2AUTH    V
     COPY CONDITIONS                 DB2AUTH    V
     LARGE OBJECTS                   DB2AUTH    V
     MISSING STATS                   ADMR4      U
     NEVER COPIED                    DB2AUTH    V
     REORG CONDITIONS                DB2AUTH    V
     SATURDAY ONLY                   DB2AUTH    V
     SMALL OBJECTS                   DB2AUTH    V
     TEST_AUTO                       ADMR2      U
```

Figure 6-64 Creating the new Exception Profile

On the next panel, we entered the Profile Name and the Update Option, as shown in Figure 6-65.

```
AUTOTOOL V4R1   ------- Exceptions Profile Display ------ 2013/03/05  13:45:35
Option  ===>                                                Scroll ===> PAGE
Line Commands: C - Create  D - Delete  E - Export  I - Import
               U - Update  V - View    J - Jobs    R - Rename
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                    Row 1 of 11            >
--------------------------------------------------------------------------------
 Display St Esssssssssssssss Enter New Exceptions Profile Data sssssssssssssssN
---------- e                                                                   e
           e    Creator . . . ADMR3                                            e
Cmd  Name  e                                                                   e
     AVOID  e    Profile Name  FULLIC_WEEKLY                                   e
 C   AVOID_ e                                                                   e
     CLUSTE e    Description                                                    e
     COPY C e                                                                   e
     LARGE  e    Update Option U  (U - Update, V - View only, N - No access)   e
     MISSIN e                                                                   e
     NEVER  e                                                                   e
     REORG  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     SATURDAY ONLY                   DB2AUTH    V
     SMALL OBJECTS                   DB2AUTH    V
     TEST_AUTO                       ADMR2      U
```

Figure 6-65 FULLIC_WEEKLY Exception Profile

On the Update Exceptions Profile Display, we chose Sunday, as shown in Figure 6-66.

```
AUTOTOOL V4R1   ---- Update Exceptions Profile Display --- 2013/03/05 13:58:24
Option  ===>                                                 Scroll ===> PAGE
Line Commands: A - And  O - Or  S - Select  D - Deselect  R - Repeat
   CONDitions: LT|<|LE|<=|EQ|=|GT|>|GE|>=|NE|¬=|<> "*" indicates a DAT stat
------------------------------------------------------------------------------
Creator: ADMR3       Profile: FULLIC_WEEKLY                   User: ADMR3
------------------------------------------------------------------------------
Share Option U (U - Update, V - View,  N - No)
Description                                        Scroll Right for Column Help
Use Stats From: R (R - Repository,         Update Runstats Options: N (Yes/No)
                   C - Catalog,        Save Triggers in Repository: N (Yes/No)
                   U - Runstats,   WTO number of triggered Objects: N (Yes/No)
                   S - Shadow,   Combine IX/TS Exceptions when
                   H - History)   evaluating an IX triggering a TS: N (Yes/No)
------------------------------------------------------- Row 1 of 206        +>
S Statistics Type--- *Column------------ Cond ---------Exception Value---------
  DAY OF WEEK         MONDAY
                      TUESDAY
                      WEDNESDAY
                      THURSDAY
                      FRIDAY
                      SATURDAY
S                     SUNDAY
  DAY OF MONTH        NTH_MONDAY
                      NTH_TUESDAY
                      NTH_WEDNESDAY
                      NTH_THURSDAY
                      NTH_FRIDAY
                      NTH_SATURDAY
                      NTH_SUNDAY
                      NTH_DAY
                      LAST_DAY
                      DAY_MONTH                  /    DD/MM
  TIME OF DAY         TIME_FROM                  :      M
                      TIME_TO                    :      M
------------------------------------------------------------------------------
  OBJECT              EXCLUDE|ONLY
                      LOB
                      PGSIZE_32K
                      PBG_TS
                      AND|OR
```

Figure 6-66 FULLIC_WEEKLY options

We press PF3 and a confirmation message is displayed, as shown in Figure 6-67 on page 182.

```
AUTOTOOL V4R1   ------- Exceptions Profile Display ------ 2013/03/05  14:05:06
Option  ===>                                                 Scroll ===> PAGE
Line Comma EsssssssssssssssssssssssssssssssssssssssssssssssssssssssN
           e HAAM09OI - Profile "ADMR3.FULLIC_WEEKLY" saved. e
---------- DssssssssssssssssssssssssssssssssssssssssssssssssssssssssM ----------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                    Row 1 of 12            >
--------------------------------------------------------------------------------
 Display Statistics Reports and/or Perform Statistics Maintenance? N (Yes/No)
--------------------------------------------------------------------------------

Cmd  Name                            Creator   Updt
     AVOID REORG                     ADMR2      U
     AVOID_IC                        ADMR3      U
     CLUSTERRATIO VALUE RANGE        DB2AUTH    V
     COPY CONDITIONS                 DB2AUTH    V
     FULLIC_WEEKLY                   ADMR3      U
     LARGE OBJECTS                   DB2AUTH    V
     MISSING STATS                   ADMR4      U
     NEVER COPIED                    DB2AUTH    V
     REORG CONDITIONS                DB2AUTH    V
     SATURDAY ONLY                   DB2AUTH    V
     SMALL OBJECTS                   DB2AUTH    V
     TEST_AUTO                       ADMR2      U
```

Figure 6-67 FULLIC_WEEKLY confirmation message

6.2.5 Running the profiles

The next step in our process is to create a Job Profile.

We start at the DB2 Automation Tool for z/OS panel, where we selected option 4, Job Profiles, as shown in Figure 6-68.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2013/02/27  14:52:09
Option  ===> 4
--------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Disaster Recovery
                3 - Exception Profiles      11 - Stand Alone Utilities
                4 - Job Profiles            12 - DB2 Admin Scheduler
                5 - Quick Build             13 - DB2 Autonomic Statistics
                6 - Execution Reports        X - Exit
                7 - DB2 Command Processor

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DBOA         (1-4 Character Subsystem ID or ? for list)
 Current SQLID:                 User: ADMR3   - HAA
--------------------------------------------------------------------------------
```

Figure 6-68 Job Profiles selection

On the next panel, we used the asterisk (*) for both the Profile and Creator Like options, as shown in Figure 6-69.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2013/02/27  14:52:09
Option  ===> 4
--------------------------------------------------------------------------------

       Options: 0 - Setup                   8 - Dataset Manager
                1 - Object Profiles         9 - Data Page Display
                2 - Utility Profiles       10 - Disaster Recovery
                3 - Exception Profiles     11 - Stand Alone Utilities
             Essssssss Enter Jobs Like to Display sssssssssN uler
             e                                              e tatistics
             e Profile Like *                               e
             e Creator Like *                               e
             e                                              e
------------ e                                              e ----------------
 DB2 Subsyst DssssssssssssssssssssssssssssssssssssssssssssssssM ? for list)
 Current SQLID:                 User: ADMR3   - HAA
--------------------------------------------------------------------------------
```

Figure 6-69 Job Profiles selection criteria

The next panel displays all available Object Profiles and the options to manage them. We typed C in front of one of them to create our new profile, as shown in Figure 6-70.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/02/27  14:59:14
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                    Row 1 of 13           >
--------------------------------------------------------------------------------

Cmd  Name                            Creator   Updt
 c   AVOID REORG                     ADMR3      U
     COPY CATALOG AND DIRECTORY      DB2AUTH    V
     COPY LARGE SPACES TO TAPE       DB2AUTH    V
     COPY SMALL SPACES TO DASD       DB2AUTH    V
     COPY TO DASD                    DB2AUTH    V
     GLWSAMP                         ADMR3      U
     IMAGE_COPY_ABP                  ADMR2      U
     ONLINE REORG                    DB2AUTH    V
     RECALL SP                       DB2AUTH    V
     RECOVER CATALOG AND DIRECTORY   DB2AUTH    V
     REPOSITORY STATS                DB2AUTH    V
     RUNSTATS CATALOG                DB2AUTH    V
     STANDARD REORG                  DB2AUTH    V
****************************** Bottom of Data ********************************
```

Figure 6-70 Creating a new Job Profile

On the next panel, we entered the Profile Name and Update Option and pressed Enter, as shown in Figure 6-71.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/02/27  14:59:14
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                    Row 1 of 13           >
--------- Essssssssssssssssss Enter New Jobs Profile Data sssssssssssssssssN
          e                                                                 e
Cmd  Name e    Creator . . . . ADMR3                                        e
 C   AVOI e    Profile Name  . AVOID IC                                     e
     COPY e    Description . .                                              e
     COPY e    Update Option . U  (U - Update, V - View only, N - No access)  e
     COPY e                                                                 e
     COPY e                                                                 e
     GLWS e                                                                 e
     IMAG e                                                                 e
     ONLI e                                                                 e
     RECA e                                                                 e
     RECO e                                                                 e
     REPO e                                                                 e
     RUNS DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     STANDARD REORG                  DB2AUTH    V
****************************** Bottom of Data ********************************
```

Figure 6-71 Job Profile description

On the next panel, we only changed the Recall Migrated Spaces option to Y, as shown in Figure 6-72 on page 185.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/02/27  14:59:14
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
Esssssssssssssssss Generation Options for ADMR3.AVOID IC sssssssssssssssssN
e                                                                             e
e Option  ===>                                                                e
e                                                                             e
e                                                                More:     +  e
e Update Setup Override Options  . . . . . N (Yes/No)                         e
e Update Template/Listdef/Option parms . . N (Yes/No)                         e
e Update Job Break Down Options  . . . . . N (Yes/No)                         e
e Automatically Gen GDG Base . . . . . . . 000 (0-255 Limit)                  e
e Load Balance jobs by . . . . . . . . . . N (T - Time, D - Dasd, N - None)   e
e Capture run times for Load Balancing . . N (Yes/No)                         e
e Start spaces in Utility/Read Only  . . . N (N - No, U - Utility,            e
e                                             R - Read only)                  e
e Prefix Utility ID with jobname . . . . . N (J - Job, S - Step, B - Both,    e
e                                             N - No)                         e
e Set JCL member equal to jobname  . . . . N (Yes/No)                         e
e Generate Job when Errors encountered . . Y (Y - Yes, N - No, W - Warnings)  e
e Preview Exception Report . . . . . . . . N (Y - Yes, N - No)                e
e Evaluate Multiple Exception Profiles . . A (A - All, O - One at a time)     e
e Recall Migrated Spaces . . . . . . . . . Y (Yes/No)                         e
e Use DSNACCOR Exception Table . . . . . . N (Yes/No)                         e
e Include Job Registration Step. . . . . . Y (Yes/No)                         e
e Utility work dataset high level  . . . .          Optional                  e
DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-72 Generation Options

On the next panel, we select the type of profiles to add to the Job Profile. In our case, we typed Y for all types, as shown in Figure 6-73 on page 186.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/02/27  18:05:25
Option  ===>                                                  Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                       Row 1 of 14            >
------------ Essssssss Adding Profiles to the Job Profile sssssssssN ---------
             e                                                              e
Cmd  Name    e    Add Objects Profiles       Y    (Yes/No)                  e
 U   AVOID I e                                                              e
     AVOID R e    Add Utilities Profiles     Y    (Yes/No)                  e
     COPY CA e                                                              e
     COPY LA e    Add Exception Profiles     Y    (Yes/No)                  e
     COPY SM e                                                              e
     COPY TO e                                                              e
     GLWSAMP e                                                              e
     IMAGE_C e                                                              e
     ONLINE  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     RECALL SP                        DB2AUTH    V
     RECOVER CATALOG AND DIRECTORY    DB2AUTH    V
     REPOSITORY STATS                 DB2AUTH    V
     RUNSTATS CATALOG                 DB2AUTH    V
     STANDARD REORG                   DB2AUTH    V
****************************** Bottom of Data ***********************************
```

Figure 6-73 Adding Profiles to the Job Profile

On the Enter Objects Profile Like to Display selection criteria panel, we typed GLWSAMP on the Profile Like option to use the profile that was created in the previous chapter, as shown in Figure 6-74 on page 187.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/02/27  18:05:25
Option  ===>                                                  Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                       Row 1 of 14           >
------------ Essssssss Enter Objects Profile Like to Display sssssssN ----------
             e                                                      e
Cmd  Name    e Profile Like GLWSAMP                                 e
 U   AVOID I e Creator Like *                                       e
     AVOID R e                                                      e
     COPY CA e                                                      e
     COPY LA DssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     COPY SMALL SPACES TO DASD       DB2AUTH    V
     COPY TO DASD                    DB2AUTH    V
     GLWSAMP                         ADMR3      U
     IMAGE_COPY_ABP                  ADMR2      U
     ONLINE REORG                    DB2AUTH    V
     RECALL SP                       DB2AUTH    V
     RECOVER CATALOG AND DIRECTORY   DB2AUTH    V
     REPOSITORY STATS                DB2AUTH    V
     RUNSTATS CATALOG                DB2AUTH    V
     STANDARD REORG                  DB2AUTH    V
****************************** Bottom of Data ***********************************
```

Figure 6-74 Enter Objects Profile Like to Display selection criteria

We selected GLWSAMP, as shown in Figure 6-75.

```
AUTOTOOL V4R1   -------- Objects Profile Display -------- 2013/02/27  18:11:28
Option  ===>                                                  Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import  S - Select
               Q - Quick   V - View    U - Update
Profile Like  GLWSAMP                                    DB2 Subsystem: DBOA
Creator Like  *                                       Row 1 of 2            >
--------------------------------------------------------------------------------

Cmd  Name                           Creator   Updt
 S   GLWSAMP                        ADMR2      U
```

Figure 6-75 Object Profile selection

We press PF3 and the next panel is displayed for the Utilities Profile, as shown in Figure 6-76 on page 188. We selected the asterisk (*) for both the Creator and Profile Like options.

```
AUTOTOOL V4R1   ------ Update Jobs Profile Display ------ 2013/03/05  14:48:38
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: V - View  A - Add  D - Delete  U - Update
--------------------------------------------------------------------------------
Creator: ADMR3        Profile: AVOID IC                      User: ADMR3
--------------------------------------------------------------------------------
Share Option U  (U - Update,     Description
             Essssss Enter Utilities Profile Like to Display ssssssN
             e                                                   e
Update Job G e    Profile Like  *                                e
------------ e    Creator Like  *                                e ----------
             e                                                   e
Cmd  Type  O e                                                   e
     OBJS    DsssssssssssssssssssssssssssssssssssssssssssssssssssssssM
 A   UTIL     1   IC_NEW                          ADMR3     ADMR3
     EXCP     1   AVOID_IC                        ADMR3     ADMR3
****************************** Bottom of Data ***********************************
```

Figure 6-76 Utilities Profiles selection to add to the Job Profile

We selected the Utility Profile IC_NEW, as shown in Figure 6-77 on page 189.

```
AUTOTOOL V4R1   ------- Utilities Profile Display ------- 2013/03/05  17:28:10
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import  S - Select
               Q - Quick   U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                   Row 1 of 25            >
--------------------------------------------------------------------------------

Cmd  Name                              Creator   Updt
     CHECK DATA                        DB2AUTH    V
     COPY TO COPY                      DB2AUTH    V
     COPY TO DASD                      DB2AUTH    V
     COPY TO TAPE                      DB2AUTH    V
     IC TO DISK                        ADMR4      U
     IC&REORG                          ADMR3      U
 S   IC_NEW                            ADMR3      U
     IMAGE_COPY                        ADMR2      U
     IMAGE_COPY_FC                     ADMR2      U
     MODIFY RECOVERY                   DB2AUTH    V
     MODIFY_DELETE                     ADMR2      U
     ONLINE IX REORG                   DB2AUTH    V
     ONLINE TS REORG                   DB2AUTH    V
     QUIESCE                           DB2AUTH    V
     RECOVER                           DB2AUTH    V
     REORG V10                         ADMR3      U
     REPAIR SET NOCOPYPEND             DB2AUTH    V
     REPOSITORY MAINTENANCE            DB2AUTH    V
     RESIZE                            DB2AUTH    V
     RUNSTATS                          ADMR4      U
     RUNSTATS CATALOG                  DB2AUTH    V
     RUNSTATS REPOSITORY               DB2AUTH    V
     STANDARD IX REORG                 DB2AUTH    V
     STANDARD TS REORG                 DB2AUTH    V
     VERIFY                            DB2AUTH    V
```

Figure 6-77 Utility Profile IC_NEW selection to Job Profile

We press PF3 and the next panel is displayed. Then, we typed ADMR3 for the Creator Like option, as shown in Figure 6-78 on page 190.

```
AUTOTOOL V4R1   ------- Utilities Profile Display ------- 2013/02/27  18:29:31
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import  S - Select
               Q - Quick   U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  ADMR3                                Row 1 of 3             >
------------ Esssss Enter Exceptions Profile Like to Display ssssssN ----------
             e                                                      e
Cmd  Name    e    Profile Like  *                                   e
     IC&REOR e    Creator Like  ADMR3                              e
     IC_NEW  e                                                     e
     REORG V e                                                     e
************ DssssssssssssssssssssssssssssssssssssssssssssssssssssM **********
```

Figure 6-78 Utility Profile to be added to Job Profile

We selected the AVOID_IC Exception Profile, as shown in Figure 6-79.

```
AUTOTOOL V4R1   ------- Exceptions Profile Display ------ 2013/03/05  17:29:59
Option  ===>                                                Scroll ===> PAGE
Line Commands: C - Create  D - Delete  E - Export  I - Import  S - Select
               U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  ADMR3                                  Row 1 of 2             >
--------------------------------------------------------------------------------
 Display Statistics Reports and/or Perform Statistics Maintenance? N (Yes/No)
--------------------------------------------------------------------------------

Cmd  Name                          Creator   Updt
 S   AVOID_IC                      ADMR3      U
     FULLIC_WEEKLY                 ADMR3      U
```

Figure 6-79 AVOID_IC and FULLIC_WEEKLY selection

We press PF3 and a panel with all the profiles related to the new Job Profile is displayed, as shown in Figure 6-80 on page 191.

```
AUTOTOOL V4R1   ------ Update Jobs Profile Display ------ 2013/03/05  17:30:58
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: V - View  A - Add  D - Delete  U - Update
--------------------------------------------------------------------------------
Creator: ADMR3        Profile: AVOID IC                      User: ADMR3
--------------------------------------------------------------------------------
Share Option U  (U - Update,     Description
                 V - View,
                 N - No)
Update Job Generation Options N  (Yes/No)  Row 1 of 3             >
--------------------------------------------------------------------------------
                                                          <-------
Cmd  Type  Order  Name                              Creator   Userid
     EXCP      1   AVOID_IC                         ADMR3     ADMR3
     OBJS      1   GLWSAMP                          ADMR4     ADMR3
     UTIL      1   IC_NEW                           ADMR3     ADMR3
```

Figure 6-80 Update Jobs Profile Display

We press PF3 and on the Jobs Profile Display, we type B in front of the AVOID IC profile to build the jobs, as shown in Figure 6-81.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/02/27  19:17:31
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  GLWSAMP                                  DB2 Subsystem: DBOA
 Creator Like  *                                     Row 1 of 14            >
--------------------------------------------------------------------------------

Cmd  Name                            Creator   Updt
 B   AVOID IC                        ADMR3      U
     AVOID REORG                     ADMR3      U
     COPY CATALOG AND DIRECTORY      DB2AUTH    V
     COPY LARGE SPACES TO TAPE       DB2AUTH    V
     COPY SMALL SPACES TO DASD       DB2AUTH    V
     COPY TO DASD                    DB2AUTH    V
     GLWSAMP                         ADMR3      U
     IMAGE_COPY_ABP                  ADMR2      U
     ONLINE REORG                    DB2AUTH    V
     RECALL SP                       DB2AUTH    V
     RECOVER CATALOG AND DIRECTORY   DB2AUTH    V
     REPOSITORY STATS                DB2AUTH    V
     RUNSTATS CATALOG                DB2AUTH    V
     STANDARD REORG                  DB2AUTH    V
***************************** Bottom of Data ********************************
```

Figure 6-81 AVOID IC build

On the next panel, we selected to build the job in Batch mode. The built job JCL will be stored on the data set ADMR3.HAA.JCL, as shown in Figure 6-82 on page 192.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/02/28  13:01:39
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
L Esssssssssssssssssss Build Job for ADMR3.AVOID IC sssssssssssssssssssN
  e                                                                    e
- e Build Online or Batch. . B (O - Online, B - Batch)                 e -
  e                                                                    e
  e Edit Generated Job . . . Y (Yes/No)                                e
- e Schedule Job . . . . . . N (Yes/No)   Update options . . N (Yes/No)  e -
  e                                                                    e
C e Build job in Data set. . ADMR3.HAA.JCL                             e
  e               Member . . AVOIDICB                                  e
  e                                                                    e
  e    Job Cards:                                                      e
  e ==> //ADMR3LD JOB (ADMR3),'RESDENT',                              e
  e ==> //*       RESTART=STEPNAME, <== FOR R                          e
  e ==> //        REGION=0M,NOTIFY=&SYSUID,                            e
  e ==> //        MSGCLASS=H,CLASS=A                                   e
  e                                                                    e
  e                                                                    e
  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     RECOVER CATALOG AND DIRECTORY   DB2AUTH    V
     REPOSITORY STATS                DB2AUTH    V
     RUNSTATS CATALOG                DB2AUTH    V
     STANDARD REORG                  DB2AUTH    V
****************************** Bottom of Data ***********************************
```

Figure 6-82 Build options

The image copy job that will be generated as the result of the profile will be stored on ADMR3.HAA.JCL, member name AVOIDJOB, as shown in Figure 6-83 on page 193.

```
AUTOTOOL V4R1   -------- Schedule DB2 Admin Task -------- 2013/02/28  13:23:52
Option   ===> CAN                                      Scroll ===> PAGE

Ta Esssssssssssssssssss Build Job for ADMR3.AVOID IC sssssssssssssssssssN
Ta e                                                                    e
   e You have selected your job to be built in a batch mode.  The       e
Be e batch generation JCL will be stored in data set                    e
En e ADMR3.HAA.JCL(AVOIDICB).                                           e
Ma e Please specify in the data set and member below where you want the e
SS e JCL built by the batch module to be placed.                        e
Jo e                                                                    e
Ex e Schedule Job  . . . . . N (Yes/No)   Update options . . N (Yes/No) e
   e                                                                    e
In e Build job in Data set . ADMR3.HAA.JCL                              e
   e              Member . . AVOIDJOB                                   e
   e                                                                    e
   e Jobcard data to be used on the generated job:                      e
   e ==> //ADMR3LD JOB (ADMR3),'RESDENT',                              e
   e ==> //*       RESTART=STEPNAME, <== FOR R                          e
   e ==> //        REGION=0M,NOTIFY=&SYSUID,                            e
   e ==> //        MSGCLASS=H,CLASS=A                                   e
   e                                                                    e
   e                                                                    e
   DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-83 Generated job data set store

The JCL from the build job will be displayed. We only copied the first lines as shown in Figure 6-84 on page 194. This job creates the JCL for the image copy on the objects that met the exception criteria (10% updated pages).

```
//ADMR3LD JOB (ADMR3),'RESDENT',
//*       RESTART=STEPNAME, <== FOR R
//        REGION=0M,NOTIFY=&SYSUID,
//        MSGCLASS=H,CLASS=A
//*
/*JOBPARM SYSAFF=SC63
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                         *
//*  Job Generated by IBM DB2 Automation Tool V4R1.01                       *
//*                                                                         *
//*  DB2 SSID: DBOA                                                         *
//*  SQLID:                                                                 *
//*  Profile:  ADMR3.AVOID IC                                               *
//*  Desc:                                                                  *
//*  User:     ADMR3                                                        *
//*  Date:     Thursday February 28, 2013                                   *
//*  Time:     13:43:41.49                                                  *
//*                                                                         *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                         *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                         *
//*  Step:     HAA@BULD                                                     *
//*                                                                         *
//*  Desc:     This job will generate the JCL for jobs profile              *
//*            ADMR3.AVOID IC in a batch mode.                              *
//*            The generated job will be placed in data set                 *
//*            ADMR3.HAA.JCL(AVOIDJOB).                                     *
//*                                                                         *
//*  Return Codes:                                                          *
//*                                                                         *
//*  (00) - Job was built successfully with no warnings or errors           *
//*                                                                         *
//*  (04) - Job was built with warning messages and the Build Job on        *
//*         Errors indicator was a "Y" or "W"                               *
//*                                                                         *
//*  (06) - Job was not built - Exception processing did not flag           *
```

Figure 6-84 Build job JCL

We submitted the job and received a return code 6, which means that no objects met our exception criteria, as shown in Figure 6-85 on page 195.

```
HAAB007W No Objects were triggered by Exception Processing
HAAB025I Build JCL will be written to ADMR3.HAA.JCL.
HAAB026I Build JCL member "AVOIDJOB" successfully written.
HAAB002I HAA$BULD Returning with RC=00000006
HAAB002I HAA#BULD Returning with RC=00000006
HAAB341I GPR  0-3  00000000_24E92F80 00000000_24EB1710 00000000_24E90000
0000000
HAAB341I GPR  4-7  00000000_24E7CC34 00000000_24E1D068 00000000_24E1D018
0000000
HAAB341I GPR  8-11 00000000_24DA90E8 00000000_24E1B000 00000000_24E18000
0000000
HAAB341I GPR 12-15 00000000_24EB2588 00000000_24EB1588 00000000_24EAF241
0000000
1IBM Shared Profile Support -- Print Exception Conditions --  V04.10 Run Date
20

 Use Statistics from the DAT REPOSITORY

 Combine IX/TS Exceptions when evaluating an IX triggering a TS Condition: N

 And|Or Catalog Table----- Column------------ Cond
-------------Value-----------
   OR   REALTIME ICOPY     UPDATED_PAGES_PCT   GT  10
1IBM Shared Profile Support -- Print Exception Triggers --  V04.10 Run Date
2013

 0 Triggers created...

READY
  PROFILE  NOPREFIX
READY
  ISPSTART PGM(HAA@BULD)
ISPD118
The initially invoked module ended with a return code = 6
Control card stream processed by IBM Shared Profile Support...

GENERATE_UTILITY_JOB ( DB2_SUBSYSTEM DBOA USER_INDICATOR HAA PROFILE_NAME
'AVOID
EXECUTION_LIB_2 DBTLSP.SHAALOAD EXECUTION_LIB_4 DBTLSP.SFECLOAD DEBUG_MODE OFF
P
GEN_TO_MEMBER AVOIDJOB JOB_CARD_1_1 '//ADMR3LD JOB (ADMR3),"RESDENT",'
JOB_CARD_
JOB_CARD_3_1 '//        REGION=0M,NOTIFY=&SYSUID,' JOB_CARD_4_1 '//
MSGCL
IBM Shared Profile Support messages follow...

Using JOBS Profile    ADMR3.AVOID IC                            that
includes..
      EXCP Profile    ADMR3.AVOID_IC
      OBJS Profile    ADMR2.GLWSAMP
```

Figure 6-85 Build job messages

When we browse ADMR3.HAA.JCL(AVOIDJOB), we see that no image copy statement was generated, as shown in Figure 6-86.

```
//ADMR3LD JOB (ADMR3),'RESDENT',
//*       RESTART=STEPNAME, <== FOR R
//        REGION=0M,NOTIFY=&SYSUID,
//        MSGCLASS=H,CLASS=A
//*
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                       *
//*  Job Generated by IBM DB2 Automation Tool V4R1.01                     *
//*                                                                       *
//*  DB2 SSID: DBOA                                                       *
//*  SQLID:                                                               *
//*  Profile:  ADMR3.AVOID IC                                             *
//*  Desc:                                                                *
//*  User:     ADMR3                                                      *
//*  Date:     Thursday February 28, 2013                                 *
//*  Time:     13:57:30.84                                                *
//*                                                                       *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                       *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                       *
//*  Step:     IBM#DLC                                                    *
//*                                                                       *
//*  Desc:     No exception processing occurred.                          *
//*                                                                       *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                       *
//IBM#DLC  EXEC  PGM=IEFBR14
//SYSPRINT DD    SYSOUT=*
*************************** Bottom of Data ****************************
```

Figure 6-86 Image Copy-generated job JCL

In a production environment, with defined daily image copies scheduled, Automation Tool can analyze whether the image copy is really needed. In our scenario, for example, this analysis avoids our running unnecessary image copy jobs, therefore, helping us to reduce CPU usage and I/O costs.

Another traditional approach in several production environments is to take at least one weekly full image copy for all objects. On the next pages, we show how Automation Tool can easily help in this task by using the profiles that we have already created on previous pages.

On the Jobs Profile Display panel, we typed C in front of an Object Profile to create a new one, as shown in Figure 6-87 on page 197.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/03/05  17:33:16
Option  ===>                                                  Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                       Row 1 of 16            >
--------------------------------------------------------------------------------

Cmd  Name                              Creator   Updt
 C   AVOID IC                          ADMR3      U
     AVOID REORG                       ADMR3      U
     COPY CATALOG AND DIRECTORY        DB2AUTH    V
     COPY LARGE SPACES TO TAPE         DB2AUTH    V
     COPY SMALL SPACES TO DASD         DB2AUTH    V
     COPY TO DASD                      DB2AUTH    V
     GLWSAMP                           ADMR3      U
     IC GLW                            ADMR4      U
     IMAGE_COPY_ABP                    ADMR2      U
     ONLINE REORG                      DB2AUTH    V
     RECALL SP                         DB2AUTH    V
     RECOVER CATALOG AND DIRECTORY     DB2AUTH    V
     REPOSITORY STATS                  DB2AUTH    V
     RUNSTATS CATALOG                  DB2AUTH    V
     RUNSTATS GLWNEW                   ADMR4      U
     STANDARD REORG                    DB2AUTH    V
```

Figure 6-87 FULLIC_WEEKLY creation

The new Job Profile will be called FULLIC_WEEKLY. We selected the Update option, as shown in Figure 6-88 on page 198.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/03/05  17:33:16
Option  ===>                                                  Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                       Row 1 of 16            >
--------- Essssssssssssssssss Enter New Jobs Profile Data sssssssssssssssssssN
          e                                                                    e
Cmd  Name e    Creator . . . . ADMR3                                          e
 C   AVOI e    Profile Name  . FULLIC_WEEKLY                                  e
     AVOI e    Description . .                                                e
     COPY e    Update Option . U  (U - Update, V - View only, N - No access)  e
     COPY e                                                                   e
     COPY e                                                                   e
     COPY e                                                                   e
     GLWS e                                                                   e
     IC G e                                                                   e
     IMAG e                                                                   e
     ONLI e                                                                   e
     RECA e                                                                   e
     RECO DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     REPOSITORY STATS               DB2AUTH    V
     RUNSTATS CATALOG               DB2AUTH    V
     RUNSTATS GLWNEW                ADMR4      U
     STANDARD REORG                 DB2AUTH    V
```

Figure 6-88 FULLIC_WEEKLY

On the next panel, we only changed the Recall Migrated Spaces option to Y, as shown in Figure 6-89 on page 199.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/03/05  17:33:16
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
Essssssssssssssss Generation Options for ADMR3.FULLIC_WEEKLY sssssssssssssssN
e                                                                             e
e Option  ===>                                                                e
e                                                                             e
e                                                              More:     +    e
e Update Setup Override Options  . . . . . N (Yes/No)                         e
e Update Template/Listdef/Option parms . . N (Yes/No)                         e
e Update Job Break Down Options  . . . . . N (Yes/No)                         e
e Automatically Gen GDG Base . . . . . . . 000 (0-255 Limit)                  e
e Load Balance jobs by . . . . . . . . . . N (T - Time, D - Dasd, N - None)   e
e Capture run times for Load Balancing . . N (Yes/No)                         e
e Start spaces in Utility/Read Only  . . . N (N - No, U - Utility,            e
e                                             R - Read only)                  e
e Prefix Utility ID with jobname . . . . . N (J - Job, S - Step, B - Both,    e
e                                             N - No)                         e
e Set JCL member equal to jobname  . . . . N (Yes/No)                         e
e Generate Job when Errors encountered . . Y (Y - Yes, N - No, W - Warnings)  e
e Preview Exception Report . . . . . . . . N (Y - Yes, N - No)                e
e Evaluate Multiple Exception Profiles . . A (A - All, O - One at a time)     e
e Recall Migrated Spaces . . . . . . . . . Y (Yes/No)                         e
e Use DSNACCOR Exception Table . . . . . . N (Yes/No)                         e
e Include Job Registration Step. . . . . . Y (Yes/No)                         e
e Utility work dataset high level  . . . .          Optional                  e
DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-89 FULLIC_WEEKLY options

We selected Y on all options on the next panel in order to add Objects, Utilities, and Exception Profiles, as shown in Figure 6-90 on page 200.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/03/05  17:43:44
Option  ===>                                                  Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                       Row 1 of 17            >
------------ Essssssssss Adding Profiles to the Job Profile sssssssssN ---------
             e                                                          e
Cmd  Name    e    Add Objects Profiles       Y    (Yes/No)           e
     AVOID I e                                                       e
     AVOID R e    Add Utilities Profiles     Y    (Yes/No)           e
     COPY CA e                                                       e
     COPY LA e    Add Exception Profiles     Y    (Yes/No)           e
     COPY SM e                                                       e
     COPY TO e                                                       e
 U   FULLIC_ e                                                       e
     GLWSAMP e                                                       e
     IC GLW  DssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     IMAGE_COPY_ABP                  ADMR2      U
     ONLINE REORG                    DB2AUTH    V
     RECALL SP                       DB2AUTH    V
     RECOVER CATALOG AND DIRECTORY   DB2AUTH    V
     REPOSITORY STATS                DB2AUTH    V
     RUNSTATS CATALOG                DB2AUTH    V
     RUNSTATS GLWNEW                 ADMR4      U
     STANDARD REORG                  DB2AUTH    V
```

Figure 6-90 Profiles that will be related to FULLIC_WEEKLY

For the Object Profile, we selected the same GLWSAMP that was used before, as shown in Figure 6-93 on page 203.

```
AUTOTOOL V4R1   -------- Objects Profile Display -------- 2013/03/05  17:46:10
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import  S - Select
               Q - Quick   V - View    U - Update
Profile Like  *                                          DB2 Subsystem: DBOA
Creator Like  *                                      Row 1 of 17           >
--------------------------------------------------------------------------------

Cmd  Name                              Creator   Updt
     ABP                               ADMR2      U
     AUTOMATION TOOL REPOSITORY        DB2AUTH    V
     DB2 CAT/DIR DSN%                  DB2AUTH    V
     DB2 CATALOG                       DB2AUTH    V
     DB2 CATALOG AND DIRECTORY         DB2AUTH    V
     DB2 DIRECTORY                     DB2AUTH    V
     DB2SORT_IVP                       ADMR2      U
     DSN00037                          ADMR2      U
     GLSSAMP                           ADMR3      U
 S   GLWSAMP                           ADMR2      U
     GLWSAMP                           ADMR4      U
     GLWS003                           ADMR3      U
     SAMPLE DATABASE                   DB2AUTH    V
     SAMPLE DATABASE BY PARTITION      DB2AUTH    V
     SAMPLE DATABASE WITH EXCLUDE      DB2AUTH    V
     SAPR3                             DB2AUTH    V
     TEST                              GLWSAMP    U
```

Figure 6-91 FULLIC_WEEKLY - Object Profile related

For the Utility Profile, we selected IMAGE_COPY_FC, as shown in Figure 6-92 on page 202, which was used in the previous chapter and is a full image copy utility.

```
AUTOTOOL V4R1   ------- Utilities Profile Display ------- 2013/03/05  17:57:53
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import  S - Select
               Q - Quick   U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                  Row 1 of 25            >
--------------------------------------------------------------------------------

Cmd  Name                              Creator   Updt
     CHECK DATA                        DB2AUTH    V
     COPY TO COPY                      DB2AUTH    V
     COPY TO DASD                      DB2AUTH    V
     COPY TO TAPE                      DB2AUTH    V
     IC TO DISK                        ADMR4      U
     IC&REORG                          ADMR3      U
     IC_NEW                            ADMR3      U
     IMAGE_COPY                        ADMR2      U
 S   IMAGE_COPY_FC                     ADMR2      U
     MODIFY RECOVERY                   DB2AUTH    V
     MODIFY_DELETE                     ADMR2      U
     ONLINE IX REORG                   DB2AUTH    V
     ONLINE TS REORG                   DB2AUTH    V
     QUIESCE                           DB2AUTH    V
     RECOVER                           DB2AUTH    V
     REORG V10                         ADMR3      U
     REPAIR SET NOCOPYPEND             DB2AUTH    V
     REPOSITORY MAINTENANCE            DB2AUTH    V
     RESIZE                            DB2AUTH    V
     RUNSTATS                          ADMR4      U
     RUNSTATS CATALOG                  DB2AUTH    V
     RUNSTATS REPOSITORY               DB2AUTH    V
     STANDARD IX REORG                 DB2AUTH    V
     STANDARD TS REORG                 DB2AUTH    V
     VERIFY                            DB2AUTH    V
```

Figure 6-92 FULLIC_WEEKLY - Utility Profile selection

For the Exception Profile, we selected FULLIC_WEEKLY, as shown in Figure 6-93 on page 203.

```
AUTOTOOL V4R1   ------- Exceptions Profile Display ------ 2013/03/05  18:06:58
Option  ===>                                                Scroll ===> PAGE
Line Commands: C - Create  D - Delete  E - Export  I - Import  S - Select
               U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                    Row 1 of 13            >
--------------------------------------------------------------------------------
 Display Statistics Reports and/or Perform Statistics Maintenance? N (Yes/No)
--------------------------------------------------------------------------------

Cmd  Name                              Creator   Updt
     AVOID REORG                       ADMR2      U
     AVOID_IC                          ADMR3      U
     CLUSTERRATIO VALUE RANGE          DB2AUTH    V
     COPY CONDITIONS                   DB2AUTH    V
 S   FULLIC_WEEKLY                     ADMR3      U
     LARGE OBJECTS                     DB2AUTH    V
     MISSING STATS                     ADMR4      U
     NEVER COPIED                      DB2AUTH    V
     REORG CONDITIONS                  DB2AUTH    V
     SATURDAY ONLY                     DB2AUTH    V
     SMALL OBJECTS                     DB2AUTH    V
     TEST_AUTO                         ADMR2      U
     TEST1                             ADMR5      U
```

Figure 6-93 FULLIC_WEEKLY - Exception Profile selection

The next panel shows the Job Profile that we created and the related profiles within it, as shown in Figure 6-94.

```
AUTOTOOL V4R1   ------ Update Jobs Profile Display ------ 2013/03/05  18:08:30
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: V - View  A - Add  D - Delete  U - Update
--------------------------------------------------------------------------------
Creator: ADMR3        Profile: FULLIC_WEEKLY                 User: ADMR3
--------------------------------------------------------------------------------
Share Option U  (U - Update,     Description
                 V - View,
                 N - No)
Update Job Generation Options N  (Yes/No)  Row 1 of 3            >
--------------------------------------------------------------------------------
                                                              <-------
Cmd  Type  Order  Name                              Creator   Userid
     EXCP      1   FULLIC_WEEKLY                   ADMR3     ADMR3
     OBJS      1   GLWSAMP                         ADMR2     ADMR3
     UTIL      1   IMAGE_COPY_FC                   ADMR2     ADMR2
```

Figure 6-94 FULLIC_WEEKLY - All related profiles

We press PF3 and receive a confirmation message. On the Jobs Profile Display panel, we type B in front of FULLIC_WEEKLY to build the Job Profile, as shown in Figure 6-95 on page 204.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/03/05  18:09:56
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                      Row 1 of 17            >
--------------------------------------------------------------------------------

Cmd  Name                              Creator   Updt
     AVOID IC                          ADMR3      U
     AVOID REORG                       ADMR3      U
     COPY CATALOG AND DIRECTORY        DB2AUTH    V
     COPY LARGE SPACES TO TAPE         DB2AUTH    V
     COPY SMALL SPACES TO DASD         DB2AUTH    V
     COPY TO DASD                      DB2AUTH    V
 B   FULLIC_WEEKLY                     ADMR3      U
     GLWSAMP                           ADMR3      U
     IC GLW                            ADMR4      U
     IMAGE_COPY_ABP                    ADMR2      U
     ONLINE REORG                      DB2AUTH    V
     RECALL SP                         DB2AUTH    V
     RECOVER CATALOG AND DIRECTORY     DB2AUTH    V
     REPOSITORY STATS                  DB2AUTH    V
     RUNSTATS CATALOG                  DB2AUTH    V
     RUNSTATS GLWNEW                   ADMR4      U
     STANDARD REORG                    DB2AUTH    V
```

Figure 6-95 FULLIC_WEEKLY build

On the next panel, we choose to build the profile in batch, as shown in Figure 6-96 on page 205.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/03/05  18:09:56
Option  ===>                                                  Scroll ===> PAGE
--------------------------------------------------------------------------------
L Esssssssssssssssss Build Job for ADMR3.FULLIC_WEEKLY sssssssssssssssssN
  e                                                                      e
- e Build Online or Batch. . B (O - Online, B - Batch)                   e -
  e                                                                      e
  e Edit Generated Job . . . Y (Yes/No)                                  e
- e Schedule Job . . . . . . N (Yes/No)   Update options . . N (Yes/No)  e -
  e                                                                      e
C e Build job in Data set. . ADMR3.HAA.JCL                               e
  e               Member . . FULLIC                                      e
  e                                                                      e
  e    Job Cards:                                                        e
  e ==> //ADMR3LD JOB (ADMR3),'RESDENT',                                 e
  e ==> //*       RESTART=STEPNAME, <== FOR R                            e
  e ==> //        REGION=0M,NOTIFY=&SYSUID,                              e
  e ==> //        MSGCLASS=H,CLASS=A                                     e
  e                                                                      e
  e                                                                      e
  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     ONLINE REORG                    DB2AUTH    V
     RECALL SP                       DB2AUTH    V
     RECOVER CATALOG AND DIRECTORY   DB2AUTH    V
     REPOSITORY STATS                DB2AUTH    V
     RUNSTATS CATALOG                DB2AUTH    V
     RUNSTATS GLWNEW                 ADMR4      U
     STANDARD REORG                  DB2AUTH    V
```

Figure 6-96 FULLIC_WEEKLY build options

On the next panel, we choose where to store the JCL, as shown in Figure 6-97 on page 206.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/03/05  18:16:40
Option  ===>                                                  Scroll ===> PAGE
--------------------------------------------------------------------------------
L Esssssssssssssssss Build Job for ADMR3.FULLIC_WEEKLY sssssssssssssssssN
  e                                                                      e
- e You have selected your job to be built in a batch mode.  The         e -
  e batch generation JCL will be stored in data set                      e
  e ADMR3.HAA.JCL(FULLIC).                                               e
- e Please specify in the data set and member below where you want the   e -
  e JCL built by the batch module to be placed.                          e
C e                                                                      e
  e Schedule Job  . . . . . N (Yes/No)   Update options . . N (Yes/No)   e
  e                                                                      e
  e Build job in Data set . ADMR3.HAA.JCL                                e
  e              Member . . FULLICJ                                      e
  e                                                                      e
  e Jobcard data to be used on the generated job:                        e
  e ==> //ADMR3LD JOB (ADMR3),'RESDENT',                                e
  e ==> //*       RESTART=STEPNAME, <== FOR R                            e
  e ==> //        REGION=0M,NOTIFY=&SYSUID,                              e
  e ==> //        MSGCLASS=H,CLASS=A                                     e
  e                                                                      e
  e                                                                      e
  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     REPOSITORY STATS               DB2AUTH    V
     RUNSTATS CATALOG               DB2AUTH    V
     RUNSTATS GLWNEW                ADMR4      U
     STANDARD REORG                 DB2AUTH    V
```

Figure 6-97 FULLIC_WEEKLY - Image copy build data set information

On the next panel, the build job is displayed, as shown in Figure 6-98 on page 207.

```
//ADMR3LD JOB (ADMR3),'RESDENT',
//*       RESTART=STEPNAME, <== FOR R
//        REGION=0M,NOTIFY=&SYSUID,
//        MSGCLASS=H,CLASS=A
//*
/*JOBPARM SYSAFF=SC63
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                         *
//*  Job Generated by IBM DB2 Automation Tool V4R1.01                       *
//*                                                                         *
//*  DB2 SSID: DBOA                                                         *
//*  SQLID:                                                                 *
//*  Profile:  ADMR3.FULLIC_WEEKLY                                          *
//*  Desc:                                                                  *
//*  User:     ADMR3                                                        *
//*  Date:     Tuesday March 05, 2013                                       *
//*  Time:     18:18:08.81                                                  *
//*                                                                         *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                         *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                         *
//*  Step:     HAA@BULD                                                     *
//*                                                                         *
//*  Desc:     This job will generate the JCL for jobs profile              *
//*            ADMR3.FULLIC_WEEKLY in a batch mode.                         *
//*            The generated job will be placed in data set                 *
//*            ADMR3.HAA.JCL(FULLICJ).                                      *
//*                                                                         *
//*  Return Codes:                                                          *
//*                                                                         *
//*  (00) - Job was built successfully with no warnings or errors           *
//*                                                                         *
//*  (04) - Job was built with warning messages and the Build Job on        *
//*         Errors indicator was a "Y" or "W"                               *
//*                                                                         *
//*  (06) - Job was not built - Exception processing did not flag           *
```

Figure 6-98 FULLIC_WEEKLY - Build job JCL

We submitted the job, and it completed with rc=0. Analyzing the output, we found the objects that would be included in the image copy job, as shown in Figure 6-99 on page 208.

```
 Total Number of Objects Included in Generated JCL..............10
0         TS IndexSpc               Index-TS DS- DS- --Current- --Current- Cur
M
 Jobname- or TableSpc Database Part --Name-- Ext Num Allocation ---Used--- Usd
-
          IX                                          ----KB---- ----KB---- -%-
 ADMR3LD  TS GLWSDPT  GLWSAMP     0            2   1       1488       1084  72
0         TS GLWSEMP  GLWSAMP     1            3   1       3744       2648  70
0         TS GLWSEMP  GLWSAMP     2            3   1       3744       2604  69
0         TS GLWSEMP  GLWSAMP     3            3   1       3744       2624  70
0         TS GLWSEMP  GLWSAMP     4            3   1       3744       2272  60
0         TS GLWSEPA  GLWSAMP     1            5   1      64944      57640  88
0
0         TS GLWSEPA  GLWSAMP     2            1   1          0          8   0
0
0         TS GLWSEPA  GLWSAMP   ALL            6   2      64944      57648  44
0
0         TS GLWSPJA  GLWSAMP     0            7   1      25344      25344 100
0         TS GLWSPRJ  GLWSAMP     0            6   1      14544      14544 100
0         TS GLWSSPL  GLWSAMP     1            1   1        720          8   1
0         TS GLWSSPL  GLWSAMP     2            1   1        720          8   1
0         TS GLWSSPL  GLWSAMP     3            1   1        720          8   1
0         TS GLWSSPL  GLWSAMP     4            1   1        720        136  18
0         TS GLWS001  GLWSAMP     0            1   1        240        188  78
0         TS GLWS002  GLWSAMP     0            2   1       1488         72   4
0         TS GLWS003  GLWSAMP     0            2   1       1488          8   0
0         TS XGLW0000 GLWSAMP     1            4   1       7200       7200 100
0         TS XGLW0000 GLWSAMP     2            3   1       7200       7200 100
0         TS XGLW0000 GLWSAMP     3            2   1       7200       7200 100
0         TS XGLW0000 GLWSAMP     4            4   1       7200       7200 100
```

Figure 6-99 FULLIC_WEEKLY - Build job output

We look at the generated full image copy job, as shown in Figure 6-100 on page 209.

```
//IMCO0102 EXEC  PGM=DSNUTILB,REGION=1024M,COND=(4,LT),
//          PARM=(DBOA,'UTILITYFC')
//*
//STEPLIB  DD DSN=DBTLSP.SHAALOAD,DISP=SHR
//         DD DSN=DBTLSP.SFECLOAD,DISP=SHR
//         DD DSN=DBOAT.SDSNEXIT,DISP=SHR
//         DD DSN=DBOAT.SDSNLOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//*
//SYSIN    DD *
   TEMPLATE C1FL0001
            UNIT     SYSDA
            DSN              'ADMR2.&DB..&SN..T&TIME..P&DSNUM..IC'
            DISP     (NEW,CATLG,DELETE)

   LISTDEF CPY001U1
          INCLUDE TABLESPACE GLWSAMP.GLWSDPT
          INCLUDE TABLESPACE GLWSAMP.GLWSEMP
          INCLUDE TABLESPACE GLWSAMP.GLWSEPA
          INCLUDE TABLESPACE GLWSAMP.GLWSPJA
          INCLUDE TABLESPACE GLWSAMP.GLWSPRJ
          INCLUDE TABLESPACE GLWSAMP.GLWSSPL
          INCLUDE TABLESPACE GLWSAMP.GLWS001
          INCLUDE TABLESPACE GLWSAMP.GLWS002
          INCLUDE TABLESPACE GLWSAMP.GLWS003
          INCLUDE TABLESPACE GLWSAMP.XGLW0000

   COPY LIST CPY001U1
        FULL        YES

        SHRLEVEL       REFERENCE
        SCOPE          ALL
        FLASHCOPY YES
        FCCOPYDDN (C1FL0001)

/*
```

Figure 6-100 FULLIC-WEEKLY - Generated batch job

To complete this scenario, we ensure that a full image copy will be triggered weekly. The previous scenario ensured that image copies were taken if the percentage of updated pages was greater than 10%. But, if a new table space with new tables is created on Monday, for example, it does not have a full image copy generated until Sunday. This is not a good approach. Next, we work on a profile to check this situation, also.

Another important DB2 Automation Tool Exception Profile type that can be used for backup purposes is DB2 DISPLAY STATUS, where exceptions are based on the display status. DB2 Automation Tool runs the display database command to retrieve this information. We can check whether objects are in copy-pending status and generate image copy jobs, for example. We include the DB2 DISPLAY STATUS triggers in the next profile.

To illustrate this scenario better, we created a table space GLWNEWTS on the GLWSAMP database. We also placed table space GLWS001 on the GLWSAMP database in copy pending status.

We start with the Automation Tool Exceptions Profile Display menu. We typed C in front of any Utility Profile to create a new one, as shown in Figure 6-101.

```
AUTOTOOL V4R1   ------- Exceptions Profile Display ------ 2013/03/06  19:06:11
Option  ===>                                                Scroll ===> PAGE
Line Commands: C - Create  D - Delete  E - Export  I - Import
               U - Update  V - View    J - Jobs    R - Rename
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                    Row 1 of 13            >
--------------------------------------------------------------------------------
 Display Statistics Reports and/or Perform Statistics Maintenance? N (Yes/No)
--------------------------------------------------------------------------------

Cmd  Name                           Creator   Updt
 C   AVOID REORG                    ADMR2      U
     AVOID_IC                       ADMR3      U
     CLUSTERRATIO VALUE RANGE       DB2AUTH    V
     COPY CONDITIONS                DB2AUTH    V
     FULLIC_WEEKLY                  ADMR3      U
     LARGE OBJECTS                  DB2AUTH    V
     MISSING STATS                  ADMR4      U
     NEVER COPIED                   DB2AUTH    V
     REORG CONDITIONS               DB2AUTH    V
     SATURDAY ONLY                  DB2AUTH    V
     SMALL OBJECTS                  DB2AUTH    V
     TEST_AUTO                      ADMR2      U
     TEST1                          ADMR5      U
```

Figure 6-101 Creating the new Exception Profile

On the next panel, we enter the Profile Name and Update option, as shown in Figure 6-102 on page 211.

```
AUTOTOOL V4R1   ------- Exceptions Profile Display ------ 2013/03/06  19:06:11
Option  ===>                                                Scroll ===> PAGE
Line Commands: C - Create  D - Delete  E - Export  I - Import
               U - Update  V - View    J - Jobs    R - Rename
------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                   Row 1 of 13            >
------------------------------------------------------------------------------
 Display St Essssssssssssss Enter New Exceptions Profile Data sssssssssssssN
---------- e                                                                 e
           e    Creator . . . ADMR3                                          e
Cmd  Name  e                                                                 e
 C   AVOID  e    Profile Name  CPPENDING & NEW OBJECTS                       e
     AVOID_ e                                                                 e
     CLUSTE e    Description                                                  e
     COPY C e                                                                 e
     FULLIC e    Update Option U  (U - Update, V - View only, N - No access)  e
     LARGE  e                                                                 e
     MISSIN e                                                                 e
     NEVER  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     REORG CONDITIONS               DB2AUTH    V
     SATURDAY ONLY                  DB2AUTH    V
     SMALL OBJECTS                  DB2AUTH    V
     TEST_AUTO                      ADMR2      U
     TEST1                          ADMR5      U
```

Figure 6-102 New Exception Profile name

We press Enter, and on the next panel, we can see the exception options, as shown in Figure 6-103 on page 212.

```
AUTOTOOL V4R1   ---- Update Exceptions Profile Display --- 2013/03/06 19:09:07
Option  ===>                                                Scroll ===> PAGE
Line Commands: A - And  O - Or  S - Select  D - Deselect  R - Repeat
   CONDitions: LT|<|LE|<=|EQ|=|GT|>|GE|>=|NE|¬=|<> "*" indicates a DAT stat
--------------------------------------------------------------------------------
Creator: ADMR3       Profile: CPPENDING & NEW OBJECTS          User: ADMR3
--------------------------------------------------------------------------------
Share Option U (U - Update, V - View,  N - No)
Description                                        Scroll Right for Column Help
Use Stats From: R (R - Repository,         Update Runstats Options: N (Yes/No)
                   C - Catalog,        Save Triggers in Repository: N (Yes/No)
                   U - Runstats,   WTO number of triggered Objects: N (Yes/No)
                   S - Shadow,   Combine IX/TS Exceptions when
                   H - History)   evaluating an IX triggering a TS: N (Yes/No)
------------------------------------------------------- Row 1 of 206        +>
S Statistics Type--- *Column------------ Cond ---------Exception Value---------
  DAY OF WEEK         MONDAY
                      TUESDAY
                      WEDNESDAY
                      THURSDAY
                      FRIDAY
                      SATURDAY
                      SUNDAY
  DAY OF MONTH        NTH_MONDAY
                      NTH_TUESDAY
                      NTH_WEDNESDAY
                      NTH_THURSDAY
                      NTH_FRIDAY
                      NTH_SATURDAY
                      NTH_SUNDAY
                      NTH_DAY
                      LAST_DAY
                      DAY_MONTH                 /    DD/MM
  TIME OF DAY         TIME_FROM                 :      M
                      TIME_TO                   :      M
--------------------------------------------------------------------------------
  OBJECT              EXCLUDE|ONLY
                      LOB
                      PGSIZE_32K
                      PBG_TS
                      AND|OR
```

Figure 6-103 Exception Profile first options

We press PF8 until we find the SYSCOPY and DB2 display options. We typed O in front of the SYSCOPY ICTYPE option, and also typed “NE F” in the Cond column. We also typed O in front of STATUS_COPY and the cond for it is “EQ”, as shown in Figure 6-104 on page 213. With these settings, a full image copy will be generated for GLWSAMP objects if one or both of the following conditions are satisfied:

- No full image copies were taken previously
- The table space is in copy-pending status

```
AUTOTOOL V4R1   ---- Update Exceptions Profile Display --- 2013/03/07 12:01:20
Option  ===>                                                Scroll ===> PAGE
Line Commands: A - And  O - Or  S - Select  D - Deselect  R - Repeat
   CONDitions: LT|<|LE|<=|EQ|=|GT|>|GE|>=|NE|¬=|<> "*" indicates a DAT stat
--------------------------------------------------------------------------------
Creator: ADMR3       Profile: CPPENDING & NEW OBJECTS          User: ADMR3
--------------------------------------------------------------------------------
Share Option U (U - Update, V - View,  N - No)
Description                                        Scroll Right for Column Help
Use Stats From: R (R - Repository,         Update Runstats Options: N (Yes/No)
                   C - Catalog,        Save Triggers in Repository: N (Yes/No)
                   U - Runstats,   WTO number of triggered Objects: N (Yes/No)
                   S - Shadow,   Combine IX/TS Exceptions when
                   H - History)   evaluating an IX triggering a TS: N (Yes/No)
------------------------------------------------------ Row 101 of 206     -+>
S Statistics Type--- *Column------------ Cond ---------Exception Value---------
  MVS CATALOG        *PERCENT_MAXALLOC
                     *PQTY
                     *SQTY
O SYSCOPY             ICTYPE              NE  F
                      DAYS
                     *CHGD_SINCE_LAST_IC
--------------------------------------------------------------------------------
  DB2 DISPLAY STATUS  TRIGGER_IF_1_MATCH
                      STATUS_ARBDP
                      STATUS_AREO*
                      STATUS_AREOR
                      STATUS_AREST
                      STATUS_AUXW
                      STATUS_CHKP
O                     STATUS_COPY         EQ
                      STATUS_GRECP
                      STATUS_ICOPY
                      STATUS_LPL
                      STATUS_LSTOP
                      STATUS_PSRBD
                      STATUS_RBDP
                      STATUS_RBDP*
                      STATUS_RECP
                      STATUS_REFP
                      STATUS_REORP
```

Figure 6-104 SYSCOPY and DB2 DISPLAY STATUS selection

We press PF3 and receive a confirmation message.

The next step is to create a new Job Profile to work on this scenario. On the Jobs Profile Display panel, we type C in front of any Job Profile to create a new one, as shown in Figure 6-105 on page 214.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/03/07  12:29:52
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                    Row 1 of 18            >
--------------------------------------------------------------------------------

Cmd  Name                              Creator   Updt
 C   AVOID IC                          ADMR3      U
     AVOID REORG                       ADMR3      U
     COPY CATALOG AND DIRECTORY        DB2AUTH    V
     COPY LARGE SPACES TO TAPE         DB2AUTH    V
     COPY SMALL SPACES TO DASD         DB2AUTH    V
     COPY TO DASD                      DB2AUTH    V
     FULLIC_WEEKLY                     ADMR3      U
     GLWSAMP                           ADMR3      U
     IC GLW                            ADMR4      U
     IMAGE_COPY_ABP                    ADMR2      U
     ONLINE REORG                      DB2AUTH    V
     RECALL SP                         DB2AUTH    V
     RECOVER CATALOG AND DIRECTORY     DB2AUTH    V
     REPOSITORY STATS                  DB2AUTH    V
     RUNSTATS CATALOG                  DB2AUTH    V
     RUNSTATS GLWNEW                   ADMR4      U
     STANDARD REORG                    DB2AUTH    V
     TEST55                            ADMR3      U
```

Figure 6-105 Creating the monitor profile

On the next panel, we enter the Profile Name and Update Option, as shown in Figure 6-106 on page 215.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/03/07  12:29:52
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                       Row 1 of 18           >
--------- Esssssssssssssssss Enter New Jobs Profile Data sssssssssssssssssN
          e                                                                 e
Cmd  Name e    Creator . . . . ADMR3                                        e
 C   AVOI e    Profile Name  . MONITOR PROFILE                              e
     AVOI e    Description . .                                              e
     COPY e    Update Option . U  (U - Update, V - View only, N - No access)  e
     COPY e                                                                 e
     COPY e                                                                 e
     COPY e                                                                 e
     FULL e                                                                 e
     GLWS e                                                                 e
     IC G e                                                                 e
     IMAG e                                                                 e
     ONLI e                                                                 e
     RECA DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
     RECOVER CATALOG AND DIRECTORY   DB2AUTH    V
     REPOSITORY STATS                DB2AUTH    V
     RUNSTATS CATALOG                DB2AUTH    V
     RUNSTATS GLWNEW                 ADMR4      U
     STANDARD REORG                  DB2AUTH    V
     TEST55                          ADMR3      U
```

Figure 6-106 New job Profile Name and Update Option

On the Generation Options for ADMR3.MONITOR PROFILE panel, we did not change any options, as shown in Figure 6-107 on page 216.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/03/07  12:29:52
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
Esssssssssssssss Generation Options for ADMR3.MONITOR PROFILE sssssssssssssssN
e                                                                              e
e Option  ===>                                                                 e
e                                                                              e
e                                                                More:     +  e
e Update Setup Override Options  . . . . . N (Yes/No)                          e
e Update Template/Listdef/Option parms . . N (Yes/No)                          e
e Update Job Break Down Options  . . . . . N (Yes/No)                          e
e Automatically Gen GDG Base . . . . . . . 000 (0-255 Limit)                   e
e Load Balance jobs by . . . . . . . . . . N (T - Time, D - Dasd, N - None)    e
e Capture run times for Load Balancing . . N (Yes/No)                          e
e Start spaces in Utility/Read Only  . . . N (N - No, U - Utility,             e
e                                             R - Read only)                   e
e Prefix Utility ID with jobname . . . . . N (J - Job, S - Step, B - Both,     e
e                                             N - No)                          e
e Set JCL member equal to jobname  . . . . N (Yes/No)                          e
e Generate Job when Errors encountered . . Y (Y - Yes, N - No, W - Warnings)   e
e Preview Exception Report . . . . . . . . N (Y - Yes, N - No)                 e
e Evaluate Multiple Exception Profiles . . A (A - All, O - One at a time)      e
e Recall Migrated Spaces . . . . . . . . . N (Yes/No)                          e
e Use DSNACCOR Exception Table . . . . . . N (Yes/No)                          e
e Include Job Registration Step. . . . . . Y (Yes/No)                          e
e Utility work dataset high level  . . . .          Optional                   e
DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-107 Monitor profile options

We selected Y to add the Objects, Utilities, and Exception Profiles, as shown in Figure 6-108 on page 217.

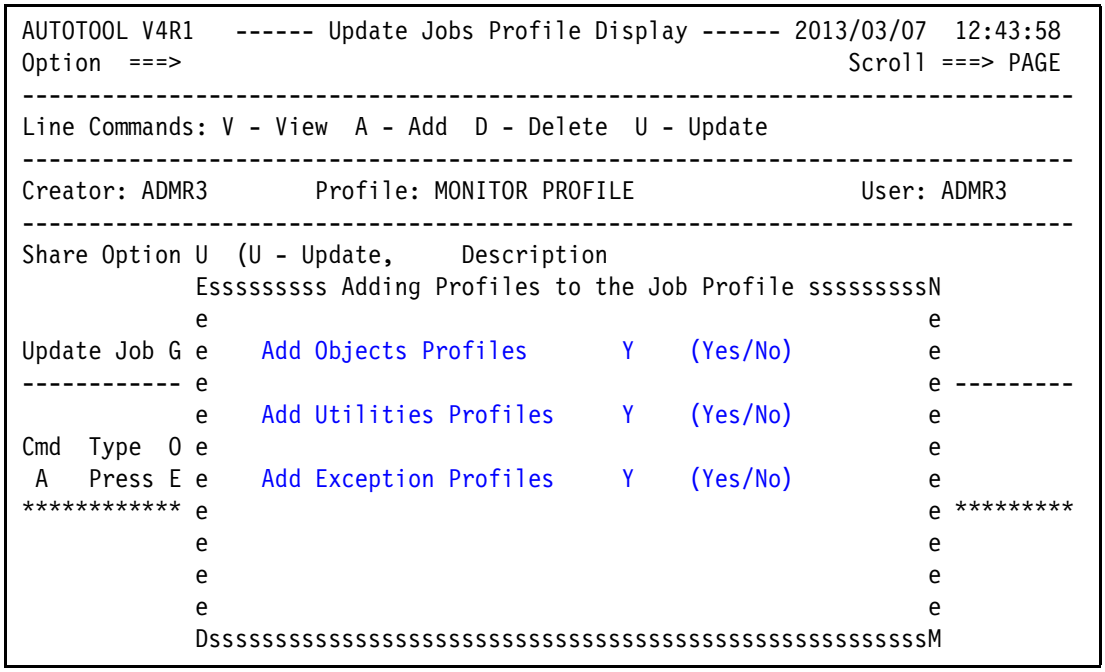

Figure 6-108 Monitor profile - General profiles selection

For Object Profiles, we selected GLWSAMP (which we used previously), as shown in Figure 6-109 on page 218.

```
AUTOTOOL V4R1   -------- Objects Profile Display -------- 2013/03/07  12:44:50
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import  S - Select
               Q - Quick   V - View    U - Update
Profile Like  *                                          DB2 Subsystem: DBOA
Creator Like  *                                      Row 1 of 18            >
--------------------------------------------------------------------------------

Cmd  Name                              Creator   Updt
     ABP                               ADMR2      U
     ACT                               ADMR4      U
     AUTOMATION TOOL REPOSITORY        DB2AUTH    V
     DB2 CAT/DIR DSN%                  DB2AUTH    V
     DB2 CATALOG                       DB2AUTH    V
     DB2 CATALOG AND DIRECTORY         DB2AUTH    V
     DB2 DIRECTORY                     DB2AUTH    V
     DB2SORT_IVP                       ADMR2      U
     DSN00037                          ADMR2      U
     GLSSAMP                           ADMR3      U
 S   GLWSAMP                           ADMR2      U
     GLWSAMP                           ADMR4      U
     GLWS003                           ADMR3      U
     SAMPLE DATABASE                   DB2AUTH    V
     SAMPLE DATABASE BY PARTITION      DB2AUTH    V
     SAMPLE DATABASE WITH EXCLUDE      DB2AUTH    V
     SAPR3                             DB2AUTH    V
     TEST                              GLWSAMP    U
```

Figure 6-109 Monitor profile - Object Profile selection

We pressed PF3 and received a confirmation message. For Utilities Profiles, we selected IMAGE_COPY_FC (which we used previously), as shown in Figure 6-110 on page 219.

```
AUTOTOOL V4R1   ------- Utilities Profile Display ------- 2013/03/07  12:55:15
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import  S - Select
               Q - Quick   U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                  Row 1 of 26            >
--------------------------------------------------------------------------------

Cmd  Name                              Creator   Updt
     CHECK DATA                        DB2AUTH    V
     COPY TO COPY                      DB2AUTH    V
     COPY TO DASD                      DB2AUTH    V
     COPY TO TAPE                      DB2AUTH    V
     IC TO DISK                        ADMR4      U
     IC&REORG                          ADMR3      U
     IC_NEW                            ADMR3      U
     IMAGE_COPY                        ADMR2      U
 S   IMAGE_COPY_FC                     ADMR2      U
     MODIFY RECOVERY                   DB2AUTH    V
     MODIFY_DELETE                     ADMR2      U
     ONLINE IX REORG                   DB2AUTH    V
     ONLINE TS REORG                   DB2AUTH    V
     QUIESCE                           DB2AUTH    V
     RECOVER                           DB2AUTH    V
     REORG V10                         ADMR3      U
     REPAIR SET NOCOPYPEND             DB2AUTH    V
     REPOSITORY MAINTENANCE            DB2AUTH    V
     RESIZE                            DB2AUTH    V
     RUNSTATS                          ADMR4      U
     RUNSTATS CATALOG                  DB2AUTH    V
     RUNSTATS REPOSITORY               DB2AUTH    V
     STANDARD IX REORG                 DB2AUTH    V
     STANDARD TS REORG                 DB2AUTH    V
     UNLOAD                            ADMR4      U
     VERIFY                            DB2AUTH    V
```

Figure 6-110 Monitor profile - Utility Profile selection

We pressed PF3 and received a confirmation message. For the Exception Profile, we selected the Exception Profile that we created in the previous pages, CPPENDING & NEW OBJECTS, as shown in Figure 6-111 on page 220.

```
AUTOTOOL V4R1   ------- Exceptions Profile Display ------ 2013/03/07  12:57:40
Option  ===>                                                Scroll ===> PAGE
Line Commands: C - Create  D - Delete  E - Export  I - Import  S - Select
               U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  *                                    Row 1 of 14            >
--------------------------------------------------------------------------------
 Display Statistics Reports and/or Perform Statistics Maintenance? N (Yes/No)
--------------------------------------------------------------------------------

Cmd  Name                            Creator   Updt
     AVOID REORG                     ADMR2      U
     AVOID_IC                        ADMR3      U
     CLUSTERRATIO VALUE RANGE        DB2AUTH    V
     COPY CONDITIONS                 DB2AUTH    V
 S   CPPENDING & NEW OBJECTS         ADMR3      U
     FULLIC_WEEKLY                   ADMR3      U
     LARGE OBJECTS                   DB2AUTH    V
     MISSING STATS                   ADMR4      U
     NEVER COPIED                    DB2AUTH    V
     REORG CONDITIONS                DB2AUTH    V
     SATURDAY ONLY                   DB2AUTH    V
     SMALL OBJECTS                   DB2AUTH    V
     TEST_AUTO                       ADMR2      U
     TEST1                           ADMR5      U
```

Figure 6-111 Monitor profile - Exception Profile selection

We pressed PF3 and received a confirmation message. On the next panel, we see all the profiles that are associated with MONITOR PROFILE, as shown in Figure 6-112.

```
AUTOTOOL V4R1   ------ Update Jobs Profile Display ------ 2013/03/07  13:00:02
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: V - View  A - Add  D - Delete  U - Update
--------------------------------------------------------------------------------
Creator: ADMR3        Profile: MONITOR PROFILE                User: ADMR3
--------------------------------------------------------------------------------
Share Option U  (U - Update,     Description
                 V - View,
                 N - No)
Update Job Generation Options N  (Yes/No)  Row 1 of 3             >
--------------------------------------------------------------------------------
                                                              <-------
Cmd  Type  Order  Name                            Creator   Userid
     EXCP      1   CPPENDING & NEW OBJECTS        ADMR3     ADMR3
     OBJS      1   GLWSAMP                        ADMR2     ADMR2
     UTIL      1   IMAGE_COPY_FC                  ADMR2     ADMR2
```

Figure 6-112 Monitor profile - All selected profiles

We pressed PF3 and received a confirmation message. The next step is to build the Job Profile.

On the Jobs Profile Display panel, we type B in front of MONITOR PROFILE to build it, as shown in Figure 6-113.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/03/07  13:03:37
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                    Row 1 of 19            >
--------------------------------------------------------------------------------

Cmd  Name                             Creator   Updt
     AVOID IC                         ADMR3      U
     AVOID REORG                      ADMR3      U
     COPY CATALOG AND DIRECTORY       DB2AUTH    V
     COPY LARGE SPACES TO TAPE        DB2AUTH    V
     COPY SMALL SPACES TO DASD        DB2AUTH    V
     COPY TO DASD                     DB2AUTH    V
     FULLIC_WEEKLY                    ADMR3      U
     GLWSAMP                          ADMR3      U
     IC GLW                           ADMR4      U
     IMAGE_COPY_ABP                   ADMR2      U
 B   MONITOR PROFILE                  ADMR3      U
     ONLINE REORG                     DB2AUTH    V
     RECALL SP                        DB2AUTH    V
     RECOVER CATALOG AND DIRECTORY    DB2AUTH    V
     REPOSITORY STATS                 DB2AUTH    V
     RUNSTATS CATALOG                 DB2AUTH    V
     RUNSTATS GLWNEW                  ADMR4      U
     STANDARD REORG                   DB2AUTH    V
     TEST55                           ADMR3      U
****************************** Bottom of Data ********************************
```

Figure 6-113 Monitor profile build

The Job Profile is built in batch, as shown in Figure 6-114 on page 222.

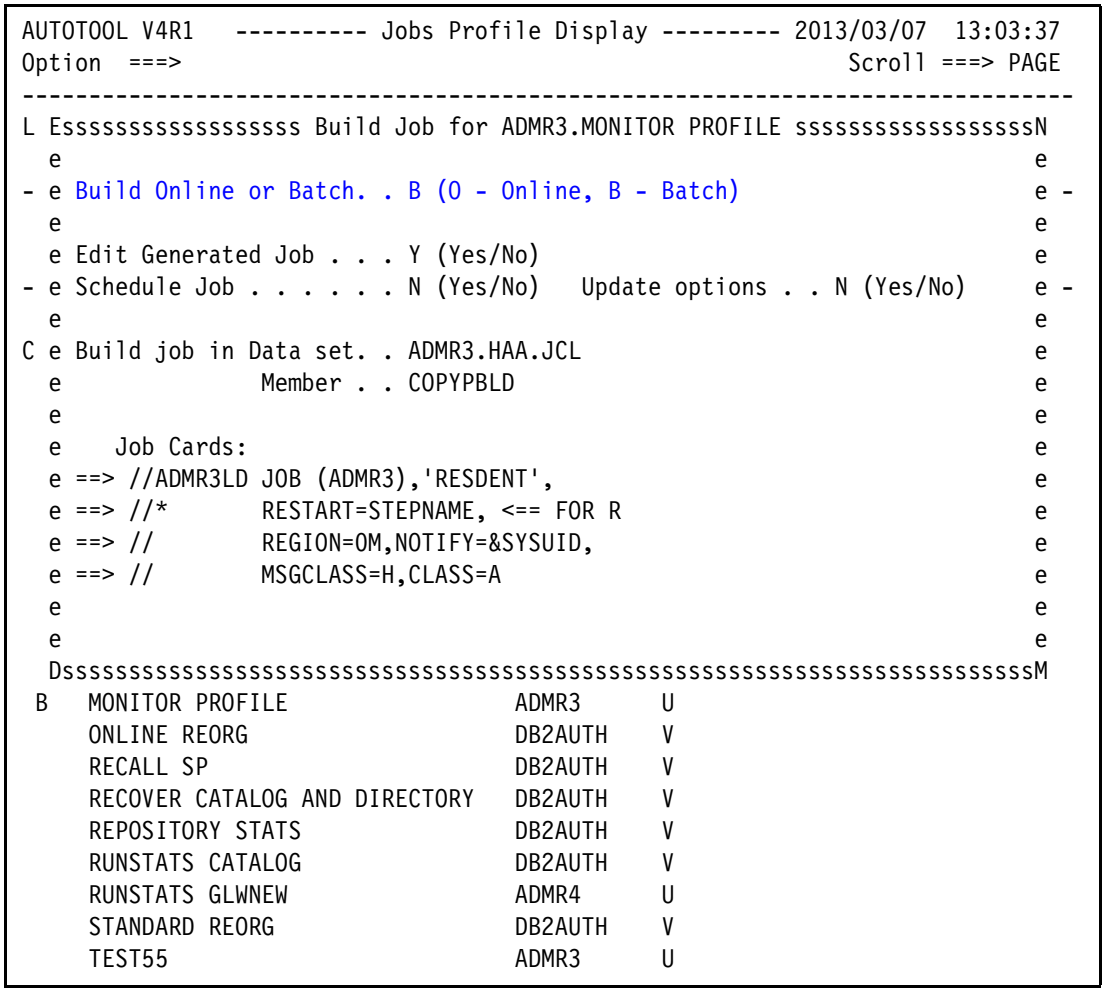

Figure 6-114 Monitor profile build job options

The full image copy-generated JCL will be stored on ADMR3.HAA.JCL(MONITOR), as shown in Figure 6-115 on page 223.

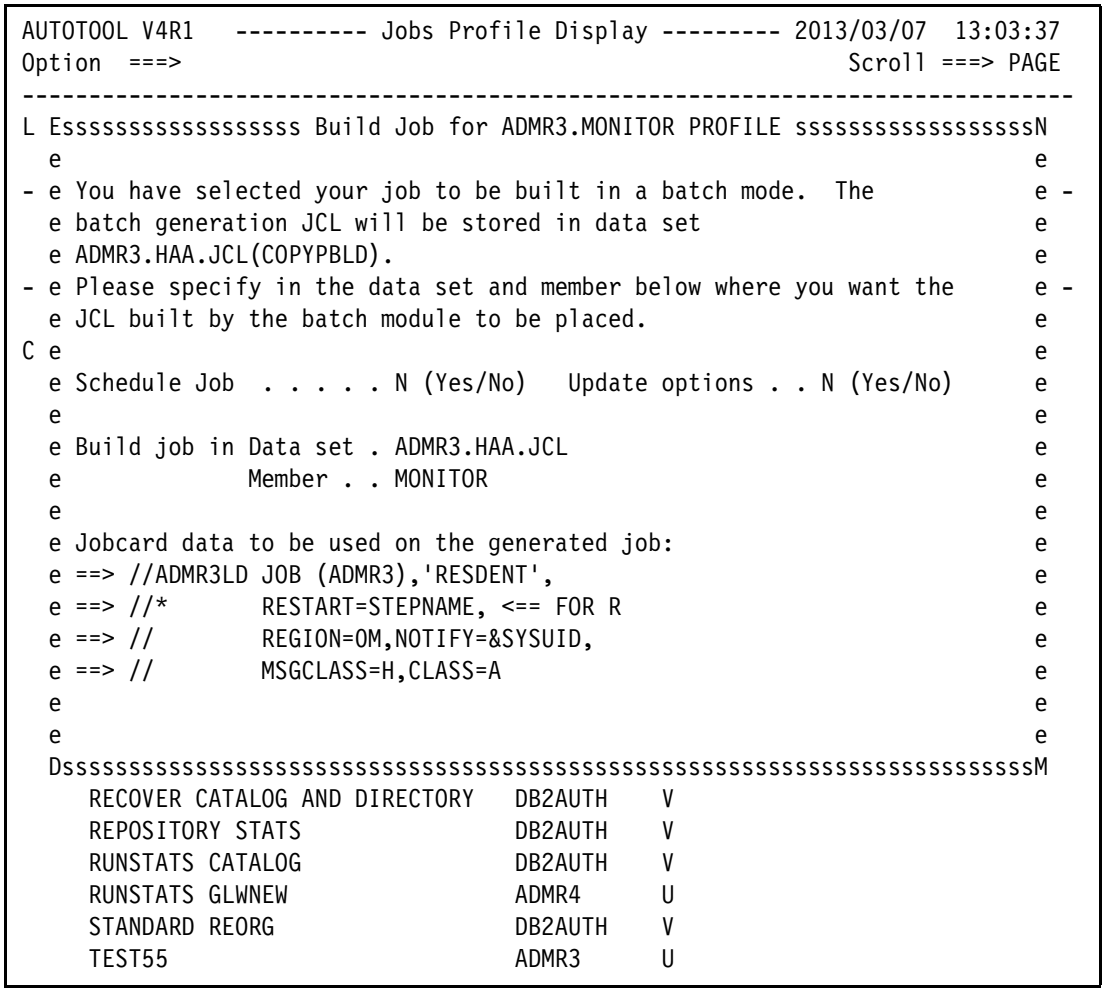

Figure 6-115 Monitor profile - Image copy job store options

We pressed Enter and the build job JCL is displayed on the next panel, as shown in Figure 6-116 on page 224.

```
//ADMR3LD JOB (ADMR3),'RESDENT',
//*       RESTART=STEPNAME, <== FOR R
//        REGION=0M,NOTIFY=&SYSUID,
//        MSGCLASS=H,CLASS=A
//*
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                         *
//*  Job Generated by IBM DB2 Automation Tool V4R1.01                       *
//*                                                                         *
//*  DB2 SSID: DBOA                                                         *
//*  SQLID:                                                                 *
//*  Profile:  ADMR3.MONITOR PROFILE                                        *
//*  Desc:                                                                  *
//*  User:     ADMR3                                                        *
//*  Date:     Thursday March 07, 2013                                      *
//*  Time:     13:06:25.78                                                  *
//*                                                                         *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                         *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                         *
//*  Step:     HAA@BULD                                                     *
//*                                                                         *
//*  Desc:     This job will generate the JCL for jobs profile              *
//*            ADMR3.MONITOR PROFILE in a batch mode.                       *
//*            The generated job will be placed in data set                 *
//*            ADMR3.HAA.JCL(MONITOR).                                      *
//*                                                                         *
//*  Return Codes:                                                          *
//*                                                                         *
//*  (00) - Job was built successfully with no warnings or errors           *
//*                                                                         *
//*  (04) - Job was built with warning messages and the Build Job on        *
//*         Errors indicator was a "Y" or "W"                               *
//*                                                                         *
//*  (06) - Job was not built - Exception processing did not flag           *
//*         any objects to process.                                         *
```

Figure 6-116 Monitor profile - Build job JCL

We submitted the job and it completed with rc=0. We checked the job output and noticed that both GLWNEWTS and GLWS001 met the exception criteria, as shown in Example 6-1.

Example 6-1 Monitor profile - Build job output messages

```
Using JOBS Profile    ADMR3.MONITOR PROFILE                    that includes..
       EXCP Profile    ADMR3.CPPENDING & NEW OBJECTS
       OBJS Profile    ADMR2.GLWSAMP
       UTIL Profile    ADMR2.IMAGE_COPY_FC

 THE FOLLOWING EXCEPTIONS WERE DETECTED IN THE EXCEPTION PROFILE(S):
       SYSCOPY
       DB2 DISPLAY STATUS

 SYSCOPY STATISTICS ARE BEING RETRIEVED
 DB2 DISPLAY STATISTICS ARE BEING RETRIEVED

 Following are the Objects included in generated JCL based on the Utility Profile EXCEPTION RULE.  Note that
 the ACCEPTED count includes Objects that met both Exception Profile and Reallocation Utility Criteria even
 though there is no EXCEPTION RULE for a Reallocation Utility.  If there is no Exception Profile in the Job,
```

```
 All Objects default to ACCEPTED Objects.  PART Level statistics are displayed for all Objects.
 Number of Accepted Objects.....................................2
 Number of Rejected Objects.....................................0
 Total Number of Objects Included in Generated JCL..............2
0         TS IndexSpc              Index-TS DS- DS- --Current- --Current- Cur Max --Max- New-Prim New-Secn New--DB2 New--DB2 Acc Re
 Jobname- or TableSpc Database Part --Name-- Ext Num Allocation ---Used--- Usd -GB -Alloc Allocatn Allocatn Allocatn Allocatn Utl Ut
          IX                                    ----KB---- ----KB---- -%-     ---%-- ---KB--- ---KB--- Prim--KB Secn--KB Cnt Cn
 ADMR3LD  TS GLWNEWTS GLWSAMP     1           2   1       1584       1584 100 000   0.00        0        0        0        0   1
0         TS GLWNEWTS GLWSAMP     2           1   1          0        144   0 000   0.00        0        0        0        0   1
0         TS GLWNEWTS GLWSAMP   ALL           3   2       1584       1728  50 000   0.00        0        0        0        0   2
0         TS GLWSO01  GLWSAMP     0           1   1        240        152  63  64   0.00        0        0        0        0   1
0******** ******** END OF OBJECTS ********
```

On ADMR3.HAA.JCL(MONITOR), we can see that the full image copy JCL was generated for both objects, as shown in Figure 6-117.

```
//IMCO0102 EXEC  PGM=DSNUTILB,REGION=1024M,COND=(4,LT),
//          PARM=(DBOA,'UTILITYFC')
//*
//STEPLIB  DD DSN=DBTLSP.SHAALOAD,DISP=SHR
//         DD DSN=DBTLSP.SFECLOAD,DISP=SHR
//         DD DSN=DBOAT.SDSNEXIT,DISP=SHR
//         DD DSN=DBOAT.SDSNLOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//*
//SYSIN    DD *
   TEMPLATE C1FLO001
            UNIT     SYSDA
            DSN             'ADMR2.&DB..&SN..T&TIME..P&DSNUM..IC'
            DISP     (NEW,CATLG,DELETE)

   LISTDEF CPYO01U1
          INCLUDE TABLESPACE GLWSAMP.GLWNEWTS
          INCLUDE TABLESPACE GLWSAMP.GLWSO01

   COPY LIST CPYO01U1
        FULL        YES

        SHRLEVEL       REFERENCE
        SCOPE          ALL
        FLASHCOPY YES
        FCCOPYDDN (C1FLO001)

/*
//*
```

Figure 6-117 Monitor profile - Generated image copy job

6.3 Automation of RUNSTATS by using autonomic statistics

DB2 10 provides a collection of stored procedures that are called *autonomic statistics procedures*. You can use these procedures to collect and keep current object statistics without the need to run the RUNSTATS utility batch jobs. First, we review why we need statistics in the first place.

Historically, we ran the RUNSTATS batch utility to collect statistics about our objects that DB2 stores in the catalog. Those statistics are used for two purposes:

- To access path statistics that are used by BIND/PREPARE in the optimization of access paths
- To provide space usage that is used by the DBA to monitor space usage, to determine when to run REORG, and to assist in capacity planning

If we did not collect these statistics and take the necessary action, such as rebind a plan or package, eventually, our database suffers with unnecessary I/O and CPU consumption.

Determining when to run RUNSTATS and on what objects is the tricky part. Do I run RUNSTATS on every object? Do I run it once a week? Well, "*no*" is the plain answer to these questions. We only need to focus on database objects that change dynamically. To do this, we can employ several methods to help us to determine when to identify those objects and then submit a RUNSTATS job. Of course, best practices also suggest that it is pointless to run RUNSTATS on static objects whose characteristics seldom change.

Of course, performing all this investigative work to determine which objects qualify can be a lengthy job for a DBA and can tie up significant resources. What we need is an automated process that can monitor objects, and when the criteria are met, generate an alert. That alert can then be used as a trigger for a dynamic invocation of RUNSTATS against the objects that caused the alert to be raised. Well, that is what DB2 10 autonomic monitoring effectively gives us. However, to use this monitoring process, we have to set up various profiles and that is where DB2 Automation Tool can help.

IBM provides two DB2 stored procedures to deal with autonomic statistics:

- Monitor stored procedure: ADMIN_UTL_MONITOR

  The MONITOR collects statistics about objects that we have defined via a profile. This procedure will generate an alert if the statistics exceed a criteria, placing the alert into a table.

- Execution procedure: ADMIN_UTL_EXECUTE

  The EXECUTE stored procedure actually performs the RUNSTATS.

The two procedures interact, and one of the first things that the MONITOR procedure does is to "kick off" the EXECUTE stored procedure, causing it to examine the alerts table and start any necessary RUNSTATS jobs. The EXECUTE procedure is restricted through the use of a predefined maintenance window where we define when the EXECUTE procedure can run. The maintenance window is determined by a timing profile. When the maintenance window is closed, the EXECUTE stored procedure stops any further RUNSTATS operations and reschedules itself for the next maintenance window. Both stored procedures run under the control of the DB2 Administrative Task Scheduler.

The following tables are used by the two procedures:

- SYSIBM.SYSTABLES.PROFILES
- SYSIBM.SYSAUTOTIMEWINDOWS
- SYSIBM.SYSAUTOALERTS
- SYSIBM.SYSAUTORUNS_HIST

At our test environment, we ensured that we had set up the DB2 administrative task scheduler and had the relevant Workload Manager (WLM) application environments ready to go. To test our WLM environment, we used the DB2 installation verification procedure (IVP) jobs to execute the DSNWZP stored procedure, therefore, ensuring that everything was in place.

The DB2 administrative task scheduler task is defined in DSNZPARM as it is started when DB2 first starts up. See Figure 6-118 for the overall architecture. The DB2 administrative task scheduler is described in 6.4, “DB2 administrative task scheduler” on page 237.

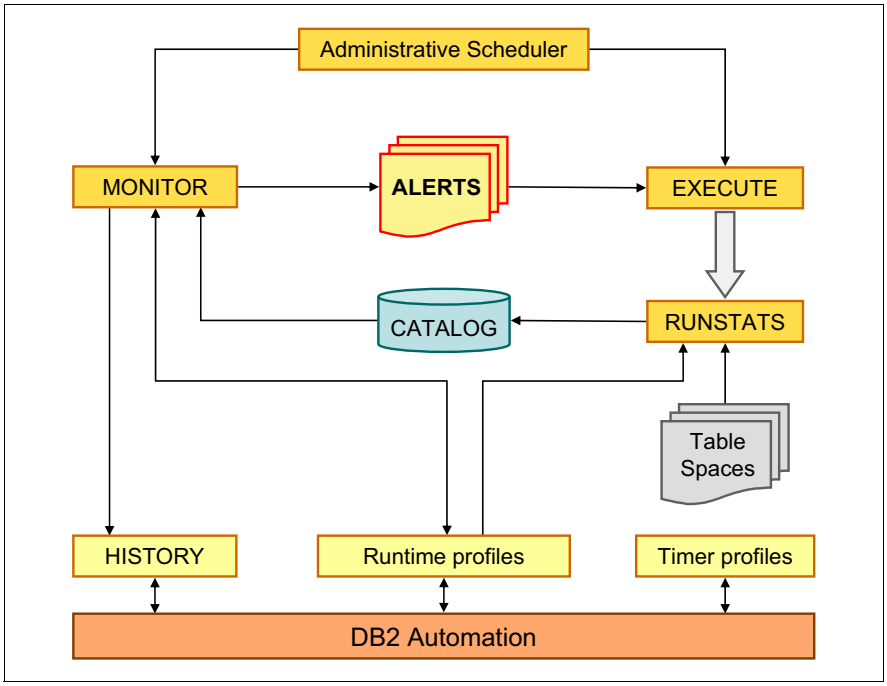

Figure 6-118 Maintaining DB2 10 autonomic profiles with DB2 Automation Tool

Where does DB2 Automation Tool enter this process? Well, DB2 Automation Tool provides an ISPF interface and its own stored procedure to manage and build the relevant profiles that are required to run the two IBM procedures. This includes identifying which objects, selection criteria, and maintenance window time profiles and so on. DB2 Automation Tool also provides for a view facility to show the history of the generated alerts and the RUNSTATS jobs that have been run.

By using these features that are provided by DB2 Automation Tool, we can easily set up, control, and monitor the new autonomic Runstats function of DB2. Setting up autonomic Runstats without DB2 Automation Tool requires a DBA to manage the profiles through SQL and does not provide an effective way to report on which profiles triggered a Runstats or to view the Runstats output.

For more information about DB2 10 autonomics, see Chapter 33. “Maintaining Statistics in a catalog” in *DB2 10 for z/OS Managing Performance*, SC19-2978.

Example of setting up monitoring profiles with DB2 Automation Tool

In this scenario, we demonstrate how to set up the automation profiles to monitor our GLWSAMP database. First, we need to access the Autonomics Statistics Panel of DB2 Automation Tool. We select option 13 from the primary panel, as shown in Figure 6-119 on page 228.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2012/04/28  15:25:07
Option  ===>
--------------------------------------------------------------------------------

       Options: 0 - Setup                   8 - Dataset Manager
                1 - Object Profiles         9 - Data Page Display
                2 - Utility Profiles       10 - Disaster Recovery
                3 - Exception Profiles     11 - Stand Alone Utilities
                4 - Job Profiles           12 - DB2 Admin Scheduler
                5 - Quick Build            13 - DB2 Autonomic Statistics
                6 - Execution Reports       X - Exit
                7 - DB2 Command Processor

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DBOA         (1-4 Character Subsystem ID or ? for list)
 Current SQLID:    DB2AUTH      User: ADMR2   - HAA
--------------------------------------------------------------------------------
```

Figure 6-119 Primary Option panel - Selecting DB2 Autonomic Statistics

This takes us to the next panel, as shown in Figure 6-120, where we can begin to set up the relevant profiles. We select option 1 - Statistics Monitor Profiles.

```
AUTOTOOL V4R1 --------- DB2 Autonomic Statistics -------- 2012/04/28  15:31:02
Option  ===>
--------------------------------------------------------------------------------

       Options: 1 - Statistics Monitor Profiles
                2 - Runstats Maintenance Windows
                3 - Execution History
                4 - View Alerts

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DBOA         User: ADMR2   - HAA
--------------------------------------------------------------------------------
```

Figure 6-120 DB2 Autonomic Statistics options panel - Defining the profiles

Selecting option 1 takes us to the profile generation panel (Figure 6-121 on page 229). Press Enter to generate the necessary “Create” line entry.

```
AUTOTOOL V4R1   ------ Statistics Monitor Profiles ------ 2012/04/28  15:32:41
Option  ===>                                                  Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: C - Create   D - Delete  E - Export   H - Execution History
               I - Import   U - Update  V - View     X - Execute
               S - Schedule R - Rename
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                   Row 1 of 1          >
--------------------------------------------------------------------------------

Cmd  Name                            Creator   Updt
 C   Line Cmds: (Create,Import)
***************************** Bottom of Data *********************************
```

Figure 6-121 Creating a Statistics Monitor Profile

Press Enter again, note the C line command, and enter details in the “Enter New Stats Monitor Profile Data” panel, as shown in Figure 6-122. Press Enter to save the details.

```
 S - Schedule R - Rename
--------------------------------------------------------------------------
  Like  *                                          DB2 Subsystem: DBOA
   Esssssssssssss Enter New Stats Monitor Profile Data sssssssssssssssN
   e                                                                    e
   e    Creator . . . . DB2AUTH                                         e
   e    Profile Name  . GLWSAMP_MON1                                    e
   e    Description . . Monitor GLW Application                         e
** e    Update Option . U  (U - Update, V - View only, N - No access)   e
   e                                                                    e
   e                                                                    e
   e                                                                    e
   e                                                                    e
   e                                                                    e
   e                                                                    e
   e  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=IFIND     e
   e  F6=RCHANGE   F7=UP        F8=DOWN      F9=SWAP      F10=LEFT     e
   DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-122 Entering details for a Statistics Monitor Profile

Now, we can enter our criteria: both the objects that we want to monitor and the criteria that will generate an alert. The objects are identified in the Restrict Tablespace field. If we specify Y to Test Restrict Tablespace, we see a list of all of the qualifying table spaces that are based on our LIKE selection. The Test Restrict Tablespace feature is useful to display the list of table spaces that the Runstats profile will monitor. See Figure 6-123 on page 230.

```
AUTOTOOL V4R1   ----- Update Stats Monitor Profile    "RIGHT   " is not active
Option  ===>                                                  Scroll ===> CSR
--------------------------------------------------------------------------------
Creator: DB2AUTH      Profile: GLWSAMP_MON1                    User: ADMR2
--------------------------------------------------------------------------------
Description  . . . . . MONITOR GLW APPLICATION
Share Option . . . . . U  (U - Update, V - View, N - No)
--------------------------------------------------------------------------------
Stand Alone  . . . . . N   (Y - Yes, N - No)
Statistics Scope . . . B   (B - Basic, P - Profile, C - Profile Consistency)
Sampling Rate  . . . .     (1-100, Blank)
Sampling Threshold . .            (Integer, Blank)
Number of Changes  . .            (Integer, Blank)
Number of Mass Deletes            (Integer, Blank)
Percent Changes  . . . 20         (0.0 - 100.0, Blank)
Update Inconsistency Thresholds N (Y - Yes, N - No)

Restrict Tablespace    DBNAME LIKE 'GLWSAMP%'
  Test Restrict Tablespace      N (Y - Yes, N - No)
```

Figure 6-123 Entering the statistics monitoring criteria

With this profile, if our database GLWSAMP has more than 20% hits to its table space Insert, Update, or Delete activity since the last RUNSTATS, an alert record is written to the SYSIBM.SYSAUTOALERTS table.

We press PF3 to save the monitor profile.

We can specify various other criteria that control our RUNSTATS alert trigger, such as a number of Insert, Update, or Delete changes, or the number of Mass Deletes. The Help facility of the Update Stats Monitor provides detailed information about how to specify the criteria and details about all other options. See Figure 6-124.

```
AUTOTOOL V4R1   ------ Statistics Monitor Profiles ------ 2012/04/28  16:14:49
Option  ===>                                                  Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: C - Create   D - Delete  E - Export   H - Execution History
               I - Import   U - Update  V - View     X - Execute
               S - Schedule R - Rename
--------------------------------------------------------------------------------
 Profile Like  *                                        DB2 Subsystem: DBOA
 Creator Like  *                                   Row 1 of 1          >
--------------------------------------------------------------------------------

Cmd  Name                           Creator   Updt
     GLWSAMP_MON1                   DB2AUTH    U
***************************** Bottom of Data ********************************
```

Figure 6-124 Statistics Monitor Profiles list

At this point, we have a choice. We can continue to build a monitor schedule profile so that the statistics monitoring profile is executed later by the Administrative Task Scheduler, or we can execute this profile now by selecting the X line command against the profile.

We choose to execute the profile to see whether we generate any alerts based on our 20% criteria. Watching the SYSLOG, we can see that the WLM has run the MONITOR procedure. We can check the output by viewing it from the Execution Output panel in DB2 Automation Tool. Selecting option 3 - Execution History from the Autonomic Statistics Panel (see Figure 6-120 on page 228) takes us to the Execution Output View.

We can see in Example 6-2 that the storage procedure SYSPROC.ADMIN_UTL_MONITOR has run our profile and returned some output. At the bottom of the output, we see that the profile generated three alerts for table spaces within the GLWSAMP database. The three alerts are written to SYSIBM.SYSAUTOALERTS and will be ready for SYSPROC.ADMIN_UTL_EXECUTE to process when it is scheduled by the administrative task scheduler. You can also see that ADMIN_UTL_MONITOR also kicked off ADMIN_UTL_EXECUTE. It is very important to remember that each alert is a record signifying that a RUNSTATS utility will be run against the object listed in the alert.

Example 6-2 Output from testing the Statistics Monitor Profile

```
AUTOTOOL V4R1  ------- AutoStats Execution Output ------- 2012/04/28  16:16:34
Option  ===>                                                Scroll ===> CSR
                                          Row 75 of 99               -
--------------------------------------------------------------------------------
2012-04-28 16:10:54.111972> ----------------------->>out-of-date / missing / inconsistent statistics (only first
1000)<<-----------------------
2012-04-28 16:10:54.115692> TABLESPACE GLWSAMP.GLWSEPA PARTITION(2) REASON(no recent statistics found)
2012-04-28 16:10:54.115714> TABLESPACE GLWSAMP.GLWSEPA PARTITION(3) REASON(no recent statistics found)
2012-04-28 16:10:54.115725> TABLESPACE GLWSAMP.GLWSEPA PARTITION(4) REASON(no recent statistics found)
2012-04-28 16:10:54.116625> ALERT written on GLWSAMP.GLWSEPA partition(2) with options (TABLE("GLWSAMP"."GLWTEPA") USE PROFILE )
2012-04-28 16:10:54.116699> ALERT written on GLWSAMP.GLWSEPA partition(3) with options (TABLE("GLWSAMP"."GLWTEPA") USE PROFILE )
2012-04-28 16:10:54.117689> ALERT written on GLWSAMP.GLWSEPA partition(4) with options (TABLE("GLWSAMP"."GLWTEPA") USE PROFILE )
2012-04-28 16:10:54.118164> ----------------------->>Schedule ADMIN_UTL_EXECUTE <<-----------------------
2012-04-28 16:10:54.460738> ADMIN_UTL_EXECUTE successfully scheduled...
2012-04-28 16:10:54.460841> ----------------------->>execution summary<<-----------------------
2012-04-28 16:10:54.460851> Number of table spaces checked: 1
2012-04-28 16:10:54.460856> Number of tables checked: 1
2012-04-28 16:10:54.460860> Number of alerts written: 3
2012-04-28 16:10:54.460872> stored procedure ends at 2012-04-28 16:10:54.460865
                            2012-04-28 16:10:54.460876> Exiting...
```

If we now look in SYSIBM.SYSAUTOALERTS by using the DB2 Administration Tool, we see some alert entries for GLWSAMP, as shown in Figure 6-125 on page 232.

```
DB2 Admin -- DBOA LIST SYSIBM.SYSAUTOALERTS            - Row 528 to 561 of 561
Command ===>                                                 Scroll ===> PAGE
"RIGHT   " is not active
L             ALERT_ID     HISTORY_ENTRY_ID ACTION   TARGET_QUALIFIER TARGET_OBJECT
                     *                    * *        *                *
- -------------------- -------------------- -------- ---------------- ---------
                   552                    4 RUNSTATS TESTDB6          TESTTS6
                   553                    4 RUNSTATS TRADE            ACCOUNTE
                   554                    4 RUNSTATS TRADE            ACCOUNTP
                   555                    4 RUNSTATS TRADE            HOLDINGE
                   556                    4 RUNSTATS TRADE            KEYGENEJ
                   557                    4 RUNSTATS TRADE            ORDEREJB
                   558                    4 RUNSTATS TRADE            QUOTEEJB
                   517                    4 RUNSTATS RACFDB2          IRRDBU00
                   518                    4 RUNSTATS RACFDB2          IRRDBU00
                   519                    4 RUNSTATS SYSIBMTA         TSDEFS
                   520                    4 RUNSTATS SYSIBMTA         TSTXTS
                   521                    4 RUNSTATS SYSTOOLS         ADHTSCP
                   522                    4 RUNSTATS SYSTOOLS         ADHTSCPS
                   523                    4 RUNSTATS SYSTOOLS         ADHTSCX
                   524                    4 RUNSTATS SYSTOOLS         ADHTSDB
                   525                    4 RUNSTATS SYSTOOLS         ADHTSDMY
                   526                    4 RUNSTATS SYSTOOLS         ADHTSENT
                   527                    4 RUNSTATS SYSTOOLS         ADHTSETP
                   528                    4 RUNSTATS SYSTOOLS         ADHTSGM
                   529                    4 RUNSTATS SYSTOOLS         ADHTSGPR
                   530                    4 RUNSTATS SYSTOOLS         ADHTSMET
                   531                    4 RUNSTATS SYSTOOLS         ADHTSMI
                   532                    4 RUNSTATS SYSTOOLS         ADHTSPER
                   533                    4 RUNSTATS SYSTOOLS         ADHTSPRO
                   534                    4 RUNSTATS SYSTOOLS         ADHTSRE
                   535                    4 RUNSTATS SYSTOOLS         ADHTSRS
                   536                    4 RUNSTATS SYSTOOLS         ADHTSRT
                   537                    4 RUNSTATS SYSTOOLS         ADHTSRUL
                   538                    4 RUNSTATS SYSTOOLS         ADHTSSM
                   539                    4 RUNSTATS SYSTOOLS         ADHTSUM
                   540                    4 RUNSTATS SYSTOOLS         ADHTSUPR
                   541                    4 RUNSTATS SYSTOOLS         ADHTSXG
                   560                    7 RUNSTATS GLWSAMP          GLWSEPA
                   561                    7 RUNSTATS GLWSAMP          GLWSEPA
```

Figure 6-125 RUNSTATS alerts in the SYSIBM.SYSAUTOALERTS table space

The monitoring schedule controls when and how often the profile is analyzed to determine whether a Runstats utility (or alert) will be generated. For example, it is a good idea to monitor a statistics profile once a week after a weekly batch run that performs a significant insert or update activity against an application. To set up the monitoring schedule entry, we use the S line command against our monitoring profile. This schedules the execution of the monitor profile by using the administrative task scheduler. Entering option S displays the panel that is shown in Figure 6-126 on page 233.

```
AUTOTOOL V4R1   -------- Schedule DB2 Admin Task -------- 2012/04/28  16:30:02
Option   ===>                                          Scroll ===> CSR

Task Name  . . . . AUTOTOOL STATMONITOR: GLWSAMP_MON1                >
Task Description                                                     >

Begin Timestamp  . . &CURRENT                   (DB2 Timestamp)
End Timestamp  . . . &CURRENT + 5 MINUTES       (DB2 Timestamp)
Max Invocations  . . 1          (Integer, Blank)
SSID . . . . . . . . DBOA       (Blank for any datasharing member)

Invocation Options:
  Interval (minutes)            (Integer, Blank)
          -Or-
  Trigger:
    Task Name . . .                                                  >
    Cond  . . . . .             (GT,GE,EQ,LT,LE,NE)
    Code  . . . . .             (Integer, Blank)
         -Or-
  Point in Time . .                                                  >
```

Figure 6-126 Setting up the monitoring scheduler profile

This uses the standard DB2 Automation Tool time entry format panel, which is consistent throughout the DB2 Automation Tool product.

For help with this panel, use the PF1 panels; this help provides comprehensive information about how to define the Timestamp and Point In Time fields.

In this case, we set up a monitoring scheduler to start anytime between CURRENT TIME through to CURRENT time plus 5 minutes. This will test that the monitoring is scheduled correctly. We should see the EXECUTE stored procedure start. In a real scheduling situation, we will probably use the Point in Time (PIT) time stamp to schedule the monitoring on a daily basis. The PIT time stamp will not add any additional triggers because we have already collected those from the previous run.

We have already executed our monitoring profile through the previous EXECUTE option, and we now have to run some RUNSTATS. A maintenance window defines when to process the alerts. Processing an alert means running Runstats against the object designated by the alert. Maintenance windows should be defined during off-peak application times or off-peak system processing times to avoid causing I/O contention against the DB2 objects while Runstats is being run.

Next, we set up the RUNSTATS maintenance window profile so all the alerts that we collected are processed and our database GLWSAMP is optimized with some valid statistics saved in the catalog.

Press PF3 until you return to the main window for DB2 Autonomic Statistics, and select option 2 - Runstats Maintenance Windows, as shown in Figure 6-127 on page 234.

```
AUTOTOOL V4R1 --------- DB2 Autonomic Statistics -------- 2012/04/28  16:37:18
Option  ===>
--------------------------------------------------------------------------------

       Options: 1 - Statistics Monitor Profiles
                2 - Runstats Maintenance Windows
                3 - Execution History
                4 - View Alerts

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DBOA         User: ADMR2   - HAA
--------------------------------------------------------------------------------
```

Figure 6-127 Building the RUNSTATS Maintenance Window Profile

From the RUNSTATS Maintenance Window panel, we press Enter again. Note the C on the command line in Figure 6-128.

```
AUTOTOOL V4R1   ------ Runstats Maintenance Window ------ 2012/04/28  16:41:10
Option  ===>                                                 Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  R - Repeat  S - Schedule Runstats
               U - Update
--------------------------------------------------------------------------------
 DB2 Subsystem: DBOA                                  Row 1 of 1
--------------------------------------------------------------------------------
Cmd  Window ID SSID  Month/Week  Month  Day  Time From  Time To   Action
 C              Error Messages                                     ç
****************************** Bottom of Data ********************************
```

Figure 6-128 Creating a RUNSTATS maintenance window

We can create and build a maintenance window profile. The C line command displays the Create Runstats Execution Window panel that is shown in Figure 6-129.

```
--------------------------------------------------------------------------------
 Esssssssssssssssssss Create Runstats Execution Window sssssssssssssssssssN
 e                                                                         e
 e   Window ID . . .                                                      e
 e   DB2 Subsystem .      (SSID, blank)                                   e
 e   Month or Week . M (M - Month, W - Week)                              e
 e   Month . . . . .    (1-12, blank)                                     e
 e   Day . . . . . .    (1-31, blank)                                     e
 e   Time From . . . 00:00:00 (HH:MM:SS)                                  e
 e   Time To . . . . 00:00:00 (HH:MM:SS)                                  e
 e   Action  . . . . RUNSTATS                                 >           e
 e   Max Tasks . . .        (Integer)                                     e
 e                                                                         e
 e  F1=HELP     F2=SPLIT    F3=END      F4=RETURN   F5=IFIND    F6=RCHANGE    e
 e  F7=UP       F8=DOWN     F9=SWAP     F10=LEFT    F11=RIGHT                 e
 DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-129 Creating a RUNSTATS execution window

We can build as many maintenance windows as we need to run our RUNSTATS alerts. The maintenance windows can be defined for one or all subsystems of a data sharing group. Selecting a time frame is flexible and it is defined in this panel.

Entering an M for month means that the Month value (1 - 12) depicts which month this profile will run. Blank means every month. Depict the day of the month by entering a number in the Day field. Leaving the day field blank will run this RUNSTATS profile run every day of the month. If using Week as your selection criteria, the Day field relates to a day of the week (1 - 7) or blank if every day. If you select Week as your criteria, the Month field should be left blank, which makes sense. PF1 provides comprehensive help if you need it.

We can set the time that we want to start and stop this particular RUNSTATS maintenance window. We can also specify how many RUNSTATS tasks we want to run during that time. In our example, we want RUNSTATS to run every day between 17:10:00 and 17:20:00, and a maximum of 20 tasks can run concurrently during the maintenance slot. Our maintenance window is shown in Figure 6-130.

```
--------------------------------------------------------------------------------
 Esssssssssssssssssss Update Runstats Execution Window sssssssssssssssssssN
 e e
 e   Window ID . . . 21 e
 e   DB2 Subsystem . DBOA (SSID, blank)
 e   Month or Week . W (M - Month, W - Week) e
 e   Month . . . . .    (1-12, blank) e
 e   Day . . . . . .    (1-31, blank) e
 e   Time From . . . 17:10:00 (HH:MM:SS) e
 e   Time To . . . . 17:20:00 (HH:MM:SS) e
 e   Action  . . . . RUNSTATS                                >
 e   Max Tasks . . . 20     (Integer)
 e
 e  F1=HELP     F2=SPLIT    F3=END      F4=RETURN   F5=IFIND    F6=RCHANGE
 e  F7=UP       F8=DOWN     F9=SWAP    F10=LEFT    F11=RIGHT
 DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-130 Defining a maintenance window for RUNSTATS

We save our entry by pressing PF3, which returns us to the Runstats Maintenance Window. This panel now shows one entry. See Figure 6-131 on page 236.

```
AUTOTOOL V4R1   ------ Runstats Maintenance Window ------ 2012/04/28  17:02:16
Option  ===>                                                  Scroll ===> CSR
-------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  R - Repeat  S - Schedule Runstats
               U - Update
-------------------------------------------------------------------------------
  DB2 Subsystem: DBOA                                  Row 1 of 1
-------------------------------------------------------------------------------
 Cmd  Window ID SSID  Month/Week  Month  Day  Time From  Time To   Action
      21        DBOA  W                     17:10:00    17:20:00  RUNSTATS
***************************** Bottom of Data ********************************




  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=IFIND     F6=RCHANGE
  F7=UP        F8=DOWN      F9=SWAP      F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 6-131 Viewing Runstats maintenance entries

At 17.10, the EXECUTE stored procedure starts and the alerts that are generated by our monitoring profile will execute. We can see the results of the RUNSTATS by viewing the output from the Execution History panel. Select option 3 from the DB2 Autonomic Statistics main panel (Figure 6-120 on page 228).

This option shows you the history table list, where we can select a monitor or execution run. We want to look at the run that ran between 17:10:00 and 17:20:00. At the bottom of the output in Example 6-3, we can see that the maintenance window was not long enough to resolve all our alerts.

Example 6-3 Output from the execution of RUNSTATS under Administration Scheduler

```
AUTOTOOL V4R1  ------- AutoStats Execution Output ------- 2012/04/28  17:25:48
Option  ===>                                                Scroll ===> CSR
                                        Row 223 of 252              -
-------------------------------------------------------------------------------
2012-04-28 17:20:13.355069> RUNSTATS alert x000000000000020E solved with return code 0
2012-04-28 17:20:13.356467> RUNSTATS alert x0000000000000210 solved with return code 0
2012-04-28 17:20:13.357800> RUNSTATS alert x000000000000022B solved with return code 0
2012-04-28 17:20:13.359439> RUNSTATS alert x000000000000022D solved with return code 0
2012-04-28 17:20:13.361632> Thread(RUNSTATS) disconnected from DBOA
2012-04-28 17:20:15.361745> ----------------------->>Step 1: detect new alerts< <-----------------------
2012-04-28 17:20:15.362205> 32 new alerts with action RUNSTATS detected
2012-04-28 17:20:15.362310> ----------------------->>Step 2: get maintenance windows<<-----------------------
2012-04-28 17:20:15.362487> DSNA649I  DSNX7EXE RUNSTATS ALERTS COULD NOT BE RES OLVED BY PROCEDURE. REASON 1
2012-04-28 17:20:15.362520> ----------------------->>Reschedule SYSPROC.ADMIN_UTL_EXECUTE<<-----------------------
2012-04-28 17:20:15.386563> plan next execution of task 'DB2 AUTO PROCEDURE EXECUTE' at 2012-04-29 17:10:00.000000
2012-04-28 17:20:15.412243> Updating DB2 scheduler task 'DB2 AUTO PROCEDURE EXECUTE' returned return_code=0: THE TASK
DB2 AUTO PROCEDURE EXECUTE WAS UPDATED SUCCESSFULLY.
2012-04-28 17:20:15.412361> ----------------------->>Execution summary<<------- ----------------
2012-04-28 17:20:15.412368> RUNSTATS alerts solved: 67
2012-04-28 17:20:15.412375> RUNSTATS duplicated alerts completed: 9
2012-04-28 17:20:15.412379> RUNSTATS remaining alerts: 32
2012-04-28 17:20:15.412388> stored procedure ends at 2012-04-28 17:20:15.412382
```

We solved 67 alerts, of which nine were duplicate and 32 remain. These will be resolved during the next batch maintenance window.

With DB2 Tools, we demonstrated how we can set up monitoring and scheduling profiles to use the Autonomic features that are delivered with DB2 V10. This reduces the amount of maintenance work that we perform, therefore, saving costs and improving throughput through our limited maintenance window. We can also target and set the criteria to select only the objects that require RUNSTATS. We will not waste valuable time and resource running RUNSTATS against static objects. Reducing the amount of time that we spend on RUNSTATS provides us with more time to concentrate on REORGs.

DB2 Automation Tool provides a powerful user interface to ease the setup and management of autonomic Runstats. DBAs can effectively manage the conditions under which Runstats should be run, and when those Runstats utilities should be executed. The powerful reporting provided by DB2 Automation Tool allows DBAs to see easily how effective each statistics profile is by allowing them to view the history of the alerts that are generated by each profile.

6.4 DB2 administrative task scheduler

The administrative task scheduler is a started task that can be seen as an additional DB2 address space, even if it is in a separate process. The administrative task scheduler is accessed through an SQL API and stores the scheduled tasks in two redundant task lists.

The administrative task scheduler is part of DB2 for z/OS. When properly configured, it is available and operable when DB2 first starts. The administrative task scheduler starts as a task on the z/OS system during DB2 startup. The administrative task scheduler has its own address space, which is named after the started task name.

Each DB2 subsystem has its own distinct administrative task scheduler connected to it. DB2 is aware of the administrative task scheduler whose name is defined in the subsystem parameter ADMTPROC. The administrative task scheduler is aware of DB2 by the subsystem name that is defined in the DB2SSID parameter of the started task.

The administrative task scheduler has an SQL interface, consisting of stored procedures (ADMIN_TASK_ADD and ADMIN_TASK_REMOVE) and user-defined table functions (ADMIN_TASK_LIST and ADMIN_TASK_STATUS) defined in DB2. This SQL interface allows you to remotely add or remove administrative tasks, and to list those tasks and their execution status.

The administrative task scheduler executes the tasks according to their defined schedules. The status of the last execution is stored in the task lists as well, and you can access it through the SQL interface.

The DB2 administrative task scheduler allows jobs to be scheduled for one or more executions based on a set of user-provided parameters. The user may define a window of time in which to execute the job. Additional interval or trigger parameters may be specified to indicate when a job is executed within the defined window.

6.4.1 DB2 administrative task scheduler integration with DB2 Automation Tool

DB2 Automation Tool provides the capability to add batch builds and utility execution jobs to the administrative task scheduler. In addition, DB2 Automation Tool provides an interface that allows the following functionality:

- Create new tasks
- Update and view existing tasks
- Delete tasks

- View the status and output of executed tasks

The DB2 administrative task scheduler interface can be used to schedule Job Profiles for both batch or online generations. It can also be used when running stand-alone utilities through the Automation Tool IPSF interface. Another DB2 Automation Tool feature that can use DB2 administrative task scheduler is running Runstats when needed by using DB2 autonomic statistics, as shown in the previous chapter.

6.4.2 Using DB2 administrative task scheduler

On the next pages, we illustrate a simple scenario to schedule a DB2 Automation Tool Job Profile to run by using the DB2 administrative task scheduler interface.

On the DB2 Automation Tool main panel, we choose option 4 - Job Profiles, as shown in Figure 6-132.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2013/03/12  09:54:30
Option  ===> 4
--------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Disaster Recovery
                3 - Exception Profiles      11 - Stand Alone Utilities
                4 - Job Profiles            12 - DB2 Admin Scheduler
                5 - Quick Build             13 - DB2 Autonomic Statistics
                6 - Execution Reports        X - Exit
                7 - DB2 Command Processor

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DBOA         (1-4 Character Subsystem ID or ? for list)
 Current SQLID:                 User: ADMR3   - HAA
```

Figure 6-132 Job Profiles choice on main panel

On the next panel, we choose ADMR3 for the Creator Like option, as shown in Figure 6-133 on page 239.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2013/03/12  09:54:30
Option  ===> 4
--------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Disaster Recovery
                3 - Exception Profiles      11 - Stand Alone Utilities
              Essssssss Enter Jobs Like to Display sssssssssN uler
              e                                              e tatistics
              e Profile Like *                               e
              e Creator Like ADMR3                           e
              e                                              e
------------ e                                              e ----------------
 DB2 Subsyst DssssssssssssssssssssssssssssssssssssssssssssssssM ? for list)
 Current SQLID:                 User: ADMR3   - HAA
--------------------------------------------------------------------------------
```

Figure 6-133 Creator search criteria

We type B in front of FULLIC_WEEKLY to build the profile, as shown in Figure 6-134.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/03/12  10:11:35
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  ADMR3                                  Row 1 of 7            >
--------------------------------------------------------------------------------

Cmd  Name                            Creator   Updt
     AVOID IC                        ADMR3      U
     AVOID REORG                     ADMR3      U
 B   FULLIC_WEEKLY                   ADMR3      U
     GLWSAMP                         ADMR3      U
     MONITOR PROFILE                 ADMR3      U
     TEST                            ADMR3      U
     TEST55                          ADMR3      U
***************************** Bottom of Data ********************************
```

Figure 6-134 Building the new Job Profile

On the next panel, we choose to build the profile in batch and also to schedule the job, as shown in Figure 6-135 on page 240.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2013/03/12  13:00:45
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
L Esssssssssssssssss Build Job for ADMR3.FULLIC_WEEKLY sssssssssssssssssN
  e                                                                     e
- e Build Online or Batch. . B (O - Online, B - Batch)                  e -
  e                                                                     e
  e Edit Generated Job . . . Y (Yes/No)                                 e
- e Schedule Job . . . . . . Y (Yes/No)   Update options . . N (Yes/No)  e -
  e                                                                     e
C e Build job in Data set. . ADMR3.HAA.JCL                              e
  e               Member . . ICBLD                                      e
  e                                                                     e
  e    Job Cards:                                                       e
  e ==> //ADMR3LD JOB (ADMR3),'RESDENT',                               e
  e ==> //*       RESTART=STEPNAME, <== FOR R                           e
  e ==> //        REGION=0M,NOTIFY=&SYSUID,                             e
  e ==> //        MSGCLASS=H,CLASS=A                                    e
* e                                                                     e *
  e                                                                     e
  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-135 New Job Profile options

On the next panel, we changed the “Begin Timestamp” to the task starts in five minutes from the current time, as shown in Figure 6-136 on page 241.

```
AUTOTOOL V4R1 ---------- Update DB2 Admin Task  --------- 2013/03/12  10:53:44
Option   ===>                                                Scroll ===> PAGE

Task Name  . . . . AUTOTOOL BUILD: &PROFILE                                  >
Task Description                                                             >

Begin Timestamp  . . &CURRENT + 5 MINUTES       (DB2 Timestamp)
End Timestamp  . . . &CURRENT + 1 DAY           (DB2 Timestamp)
Max Invocations  . . 1          (Integer, Blank)
SSID . . . . . . . . DBOA       (Blank for any datasharing member)

Invocation Options:
  Interval (minutes)            (Integer, Blank)
          -Or-
  Trigger:
    Task Name . . .                                                          >
    Cond  . . . . .             (GT,GE,EQ,LT,LE,NE)
    Code  . . . . .             (Integer, Blank)
          -Or-
  Point in Time . .                                                          >

Execution Source:
  JCL Data set. . .
  JCL Member  . . .             (If Partitioned Data set)
  Job Wait  . . . . Y           (Y - Yes, N - No, P - Purge)
          -Or-
  Procedure Schema                                                           >
  Procedure Name                                                             >
  Procedure Input                                                            >
```

Figure 6-136 Scheduling the Job Profile

On the next panel, we specify where the JCL will be stored, as shown in Figure 6-137 on page 242.

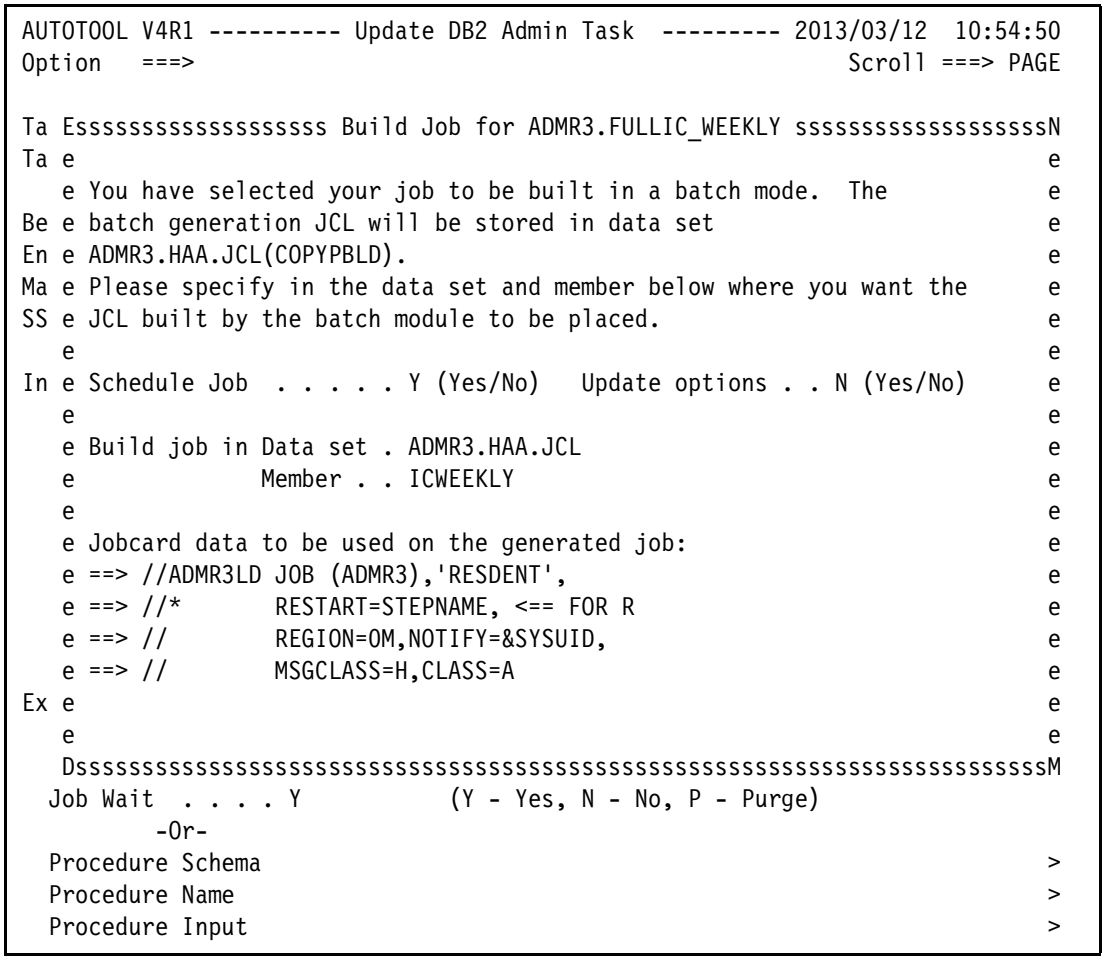

Figure 6-137 ICWEEKLY schedule job data set store

We receive a confirmation message, as shown in Figure 6-138 on page 243.

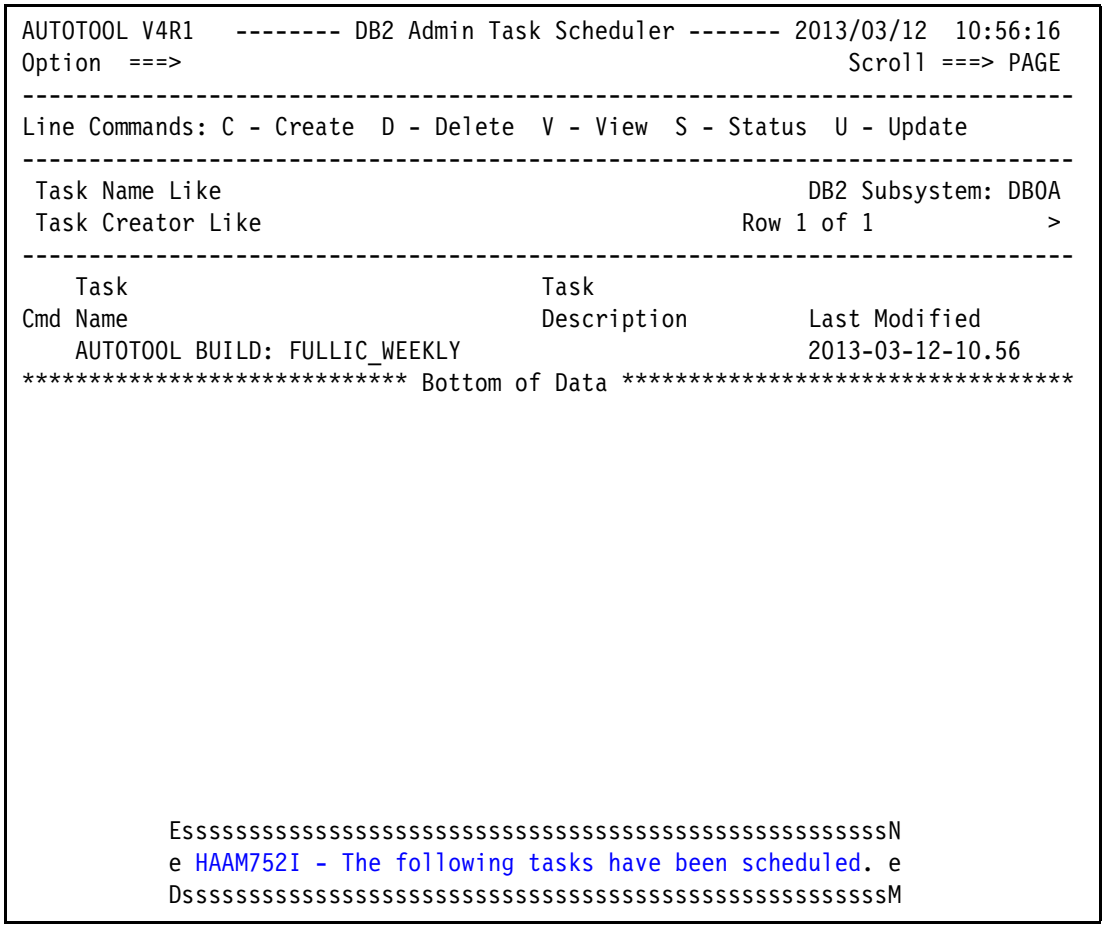

Figure 6-138 Schedule confirmation message

To check the status of the created task, we can use the DB2 administrative task scheduler interface with DB2 Automation Tool.

On the main panel, we choose option 12 - DB2 Admin Scheduler, as shown in Figure 6-139 on page 244.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2013/03/12  13:19:12
Option  ===> 12
--------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Disaster Recovery
                3 - Exception Profiles      11 - Stand Alone Utilities
                4 - Job Profiles            12 - DB2 Admin Scheduler
                5 - Quick Build             13 - DB2 Autonomic Statistics
                6 - Execution Reports        X - Exit
                7 - DB2 Command Processor

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DBOA         (1-4 Character Subsystem ID or ? for list)
 Current SQLID:                 User: ADMR3   - HAA
--------------------------------------------------------------------------------
```

Figure 6-139 DB2 Admin Scheduler choice

On the next panel, we type ADMR3 for Task Creator Like and press Enter, as shown in Figure 6-140.

```
AUTOTOOL V4R1   -------- DB2 Admin Task Scheduler ------- 2013/03/12  13:21:20
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  V - View  S - Status  U - Update
--------------------------------------------------------------------------------
 Task Name Like     *                                   DB2 Subsystem: DBOA
 Task Creator Like  ADMR3                            No rows to display
--------------------------------------------------------------------------------
    Task                              Task
Cmd Name                              Description         Last Modified
```

Figure 6-140 Task search criteria

On the next panel, we can see that task is completed. We type O in front of ADMR3 to see the job output, as shown in 245.

```
AUTOTOOL V4R1   --------- DB2 Admin Task Status --------- 2013/03/12  13:55:39
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: S - Status Detail  O - View Output
--------------------------------------------------------------------------------
 Task Name    AUTOTOOL BUILD: FULLIC_WEEKLY
 Task Creator ADMR3                          DB2 Subsystem: DBOA
 Max History  0010                           Row 1 of 1
--------------------------------------------------------------------------------
Cmd Userid  SSID  Status      Start Timestamp             End Timestamp
 O  ADMR3   DBOA  COMPLETED   2013-03-12-13.49.30.000000  2013-03-12-13.49.32
```

Figure 6-141 Choosing the output view

Checking the output, we can see that build job completed with rc=00, as shown in Figure 6-142 on page 246.

```
 AUTOTOOL V4R1   --------- DB2 Admin Task SYSOUT --------- 2013/03/12  13:56:36
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
 Task Name    AUTOTOOL BUILD: FULLIC_WEEKLY
 Task Creator ADMR3                          DB2 Subsystem: DBOA
 JobID        JOB20920                       Row 1 of 350          +>
--------------------------------------------------------------------------------
                  J E S 2  J O B  L O G  --  S Y S T E M  S C 6 3  --  N O D E

13.49.30 JOB20920 ---- TUESDAY,   12 MAR 2013 ----
13.49.30 JOB20920  IRR010I  USERID ADMR3    IS ASSIGNED TO THIS JOB.
13.49.30 JOB20920  ICH70001I ADMR3    LAST ACCESS AT 13:44:43 ON TUESDAY, MARCH
13.49.30 JOB20920  $HASP373 ADMR3LD  STARTED - INIT 1    - CLASS A - SYS SC63
13.49.30 JOB20920  IEF403I ADMR3LD - STARTED - TIME=13.49.30 - ASID=0020 - SC63
13.49.30 JOB20920  -                                          --TIMINGS (MINS.)-
13.49.30 JOB20920  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOC
13.49.30 JOB20920  -ADMR3LD           PROFILE     00      8    .00    .00    .0
13.49.31 JOB20920  -ADMR3LD           HAA@BULD    00   2068    .00    .00    .0
13.49.31 JOB20920  IEF404I ADMR3LD - ENDED - TIME=13.49.31 - ASID=0020 - SC63
13.49.31 JOB20920  -ADMR3LD  ENDED.  NAME-RESDENT             TOTAL CPU TIME=
13.49.31 JOB20920  $HASP395 ADMR3LD  ENDED
------ JES2 JOB STATISTICS ------
  12 MAR 2013 JOB EXECUTION DATE
          123 CARDS READ
          350 SYSOUT PRINT RECORDS
            0 SYSOUT PUNCH RECORDS
           22 SYSOUT SPOOL KBYTES
         0.02 MINUTES EXECUTION TIME
        1 //ADMR3LD JOB (ADMR3),'RESDENT',
          //*       RESTART=STEPNAME, <== FOR R
          //        REGION=0M,NOTIFY=&SYSUID,
          //        MSGCLASS=H,CLASS=A
          //*
          //** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
          //*
          //*  Job Generated by IBM DB2 Automation Tool V4R1.01
          //*
          //*  DB2 SSID: DBOA
          //*  SQLID:
          //*  Profile:  ADMR3.FULLIC_WEEKLY
          //*  Desc:
```

Figure 6-142 Job output

We press PF8 on the job output panel to see more messages, as shown in Figure 6-143 on page 247.

```
HAAB049I Using JOBS Profile    ADMR3.FULLIC_WEEKLY                          that
HAAB050I       OBJS Profile    ADMR2.GLWSAMP
HAAB050I       UTIL Profile    ADMR2.IMAGE_COPY_FC

HAAB025I Build JCL will be written to ADMR3.HAA.JCL.
HAAB026I Build JCL member "ICWEEKLY" successfully written.
HAAB560I Scheduled Taskname: AUTOTOOL UTIL: ADMR3LD
HAAB560I   Dataset: ADMR3.HAA.JCL(ICWEEKLY)
```

Figure 6-143 Output messages

The image copy job is on ADMR3.HAA.JCL(ICWEEKLY), as shown in Figure 6-143. Figure 6-144 shows the JCL that is generated.

```
//SYSIN    DD *
   TEMPLATE C1FL0001
            UNIT     SYSDA
            DSN              'ADMR3.&DB..&SN..T&TIME..P&DSNUM..IC'
            DISP     (NEW,CATLG,DELETE)

   LISTDEF CPY001U1
           INCLUDE TABLESPACE GLWSAMP.GLWNEWTS
           INCLUDE TABLESPACE GLWSAMP.GLWSDPT
           INCLUDE TABLESPACE GLWSAMP.GLWSEMP
           INCLUDE TABLESPACE GLWSAMP.GLWSEPA
           INCLUDE TABLESPACE GLWSAMP.GLWSPJA
           INCLUDE TABLESPACE GLWSAMP.GLWSPRJ
           INCLUDE TABLESPACE GLWSAMP.GLWSSPL
           INCLUDE TABLESPACE GLWSAMP.GLWS001
           INCLUDE TABLESPACE GLWSAMP.GLWS002
           INCLUDE TABLESPACE GLWSAMP.GLWS003
           INCLUDE TABLESPACE GLWSAMP.XGLW0000

   COPY LIST CPY001U1
        FULL        YES

        SHRLEVEL       REFERENCE
        SCOPE          ALL
        FLASHCOPY YES
        FCCOPYDDN (C1FL0001)

/*
//*
```

Figure 6-144 Generated JCL

Checking DB2 Admin Task Scheduler, we noticed that the task with the ICWEEKLY job was also scheduled. We typed S in front of it to see the status, as shown in Figure 6-145 on page 248.

```
AUTOTOOL V4R1   -------- DB2 Admin Task Scheduler ------- 2013/03/12  14:05:20
Option  ===>                                                Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  V - View  S - Status  U - Update
--------------------------------------------------------------------------------
 Task Name Like     *                                    DB2 Subsystem: DBOA
 Task Creator Like  ADMR3                             Row 1 of 2            >
--------------------------------------------------------------------------------
    Task                                Task
Cmd Name                                Description         Last Modified
    AUTOTOOL BUILD: FULLIC_WEEKLY                           2013-03-12-13.44
 S  AUTOTOOL UTIL: ADMR3LD                                  2013-03-12-13.49
```

Figure 6-145 Built job task on DB2 Admin Scheduler

On the next panel, shown in Figure 6-146, we can see the task details.

```
AUTOTOOL V4R1   --------- DB2 Admin Task Status --------- 2013/03/12  14:08:24
Option  ===>                                                Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: S - Status Detail  O - View Output
--------------------------------------------------------------------------------
 Task Name    AUTOTOOL UTIL: ADMR3LD
 Task Creator ADMR3                              DB2 Subsystem: DBOA
 Max History  0010                               Row 1 of 1
--------------------------------------------------------------------------------
Cmd Userid  SSID  Status      Start Timestamp             End Timestamp
    ADMR3   DBOA  COMPLETED   2013-03-12-13.54.32.000000  2013-03-12-13.54.35
```

Figure 6-146 Built job task details on DB2 Admin Scheduler

6.5 Interfacing DSNACCOX with DB2 Automation Tool

When creating an Exception Profile, we can call the real-time statistics stored procedure DSNACCOX from within DB2 Automation Tool as a selection criteria for utilities execution. DSNACCOX uses a set of criteria to make recommendations to help you maintain DB2.

6.5.1 DSNACCOX recommendations

DSNACCOX validates objects against real-time statistics (RTS) and catalog data and returns recommendations in a result set. A default set of criteria is provided, but these can be overridden. The recommendations are listed:

- Reorg
- Image Copy
- Runstats
- Indicates when a data set has exceeded a number of extents
- Indicates when objects are in a restricted state

For more information about DSNACCOX, see *IBM DB2 10 for z/OS Managing Performance*, SC19-2978.

6.5.2 REXX procedure ADMCOX

We wrote the REXX procedure ADMCOX so that we could test DSNACCOX and see what it returned. This was in preparation to writing the assembler routine, AUTOTEST, which calls DSNACCOX from within DB2 Automation Tool.

ADMCOX calls DSNACCOX passing objects as its criteria. Figure 6-147 shows an example.

```
//ADMR2A JOB (999,POK),'DB2 UTILITY',
//         REGION=0M,NOTIFY=&SYSUID,
//         MSGCLASS=X,CLASS=T
//PROCLIB JCLLIB ORDER=DBOAM.PROCLIB
/*JOBPARM SYSAFF=SC63
//CALLSP   EXEC PGM=IKJEFT01,DYNAMNBR=50,REGION=6M
//STEPLIB  DD   DSN=DBOAT.SDSNLOAD,DISP=SHR
//SYSPROC  DD  DSN=DB2TOOLS.ED.REXX,DISP=SHR
//SYSTSPRT DD  SYSOUT=*
//SYSTSOUT DD  SYSOUT=*
//SYSTSIN  DD  *
 ADMCOX DBOA
//SYSIN    DD   *
QUERYTYPE = "'REORG'"
CRITERIA = "DBNAME LIKE 'GLWSAMP%' AND NAME LIKE '%'"
/*
```

Figure 6-147 Example of REXX procedure ADMCOX

In this example, we set our CRITERIA field to select only databases that begin 'GLWSAMP' and all related table spaces. We have also told DSNACCOX that we are only interested in 'REORG', which is set by the QUERYTYPE field. All of the fields are described in *IBM DB2 10 for z/OS Managing Performance*, SC19-2978, and can be overridden in ADMCOX. The output from this run is shown in Figure 6-148.

The REXX procedure ADMCOX and the assembler routine AUTOTEST are available for download as described in Appendix C, “Additional material” on page 551.

```
READY
 ADMCOX DBOA
CRITERIA = DBNAME LIKE 'GLWSAMP%' AND NAME LIKE '%'
Reorg report for Database:  GLWSAMP
-------------------------------------------------------------------------------
TABLE SPACE  P#   LAST REORG       RRTMASSD. RRTINSERT%  RRTUINSRT%  RRTINDREF%
-------------------------------------------------------------------------------
GLWS001(TS)  P0  12-05-04 16:17           0       0.00        0.00        0.00
GLWSDPT(TS)  P0  **NEVER**                0       0.00        0.00        0.00
GLWSEPA(TS)  P1  **NEVER**                0       0.00        0.00        0.00
GLWSEPA(TS)  P4  **NEVER**                0       0.00        0.00        0.00
GLWSEPA(TS)  P3  **NEVER**                0       0.00        0.00        0.00
GLWSEPA(TS)  P2  **NEVER**                0       0.00        0.00        0.00
GLWSPJA(TS)  P0  **NEVER**                0       0.00        0.00        0.00
GLWSPRJ(TS)  P0  **NEVER**                0       0.00        0.00        0.00
GLWSSPL(TS)  P4  12-05-04 16:17           0      66.65        0.00        0.00
-------------------------------------------------------------------------------

ADMCOX ended rc=0
```

Figure 6-148 Sample output from ADMCOX

We can see that some of the table spaces have never been through REORG. We use this information when testing our assembler program AUTOTEST to see that the same information is returned under DB2 Automation Tool.

6.5.3 Building the DSNACCOX Exception Profile

The panel in Figure 6-149 shows how we have set AUTOTEST as a stored procedure within our Exception Profile TEST_AUTO.

AUTOTEST is an assembler program that calls DSNACCOX and then analyzes the returned result sets (Figure 6-149).

```
AUTOTOOL V4R1   ---- View Exceptions Profile Display   --- 2012/06/13 15:03:14
Option  ===>                                                Scroll ===> CSR

                                                  "*" indicates a DAT stat
------------------------------------------------------------------------------
Creator: ADMR2       Profile: TEST_AUTO                      User: ADMR2
------------------------------------------------------------------------------
Share Option U (U - Update, V - View,  N - No)
Description  TEST CALLING DSNACCOX               Scroll Right for Column Help
Use Stats From: R (R - Repository,         View Runstats Options:   N (Yes/No)
                   C - Catalog,        Save Triggers in Repository: N (Yes/No)
                   U - Runstats,   WTO number of triggered Objects: N (Yes/No)
                   S - Shadow,   Combine IX/TS Exceptions when
                   H - History)   evaluating an IX triggering a TS: N (Yes/No)
------------------------------------------------------- Row 29 of 206      -+>
S Statistics Type--- *Column------------ Cond ---------Exception Value---------
  USER EXIT           LOAD_MODULE
                      CLIST_REXX_EXEC
                      STORED_PROC_PRE
0                     STORED_PROCEDURE    EQ  AUTOTEST
                      STORED_PROC_POST
------------------------------------------------------------------------------
  TABLESPACESTATS       STAT
                      TOTALROWS
                      SPACE
                      EXTENTS
                      NACTIVE
                      NPAGES
                      DATASIZE
                      UNCOMPRSD_DATASIZE
```

Figure 6-149 Exception Profile invoking the AUTOTEST stored procedure

There are three entry points for stored procedures: Before any processing performed by exception processing, during exception processing, and after exception processing is finished. The stored procedure is called once per included object defined by the Object Profiles. You cannot use REXX as the programming language because REXX only allows one INOUT parameter in its parameter list. DB2 Automation Tool passes two INOUT fields in its parameter list, which is the reason why we coded AUTOTEST in assembler.

6.5.4 AUTOTEST stored procedure

The AUTOTEST stored procedure is an assembler program that calls DSNACCOX. AUTOTEST returns either an RC=0 (Skip Object) or RC=4 (Trigger Object) after processing

the result sets that are returned by DSNACCOX. In the sample code, AUTOTEST always sets RC=4. The code was written as a proof of concept only. Further code is required to make it useful. The logic of the program is shown in Figure 6-150.

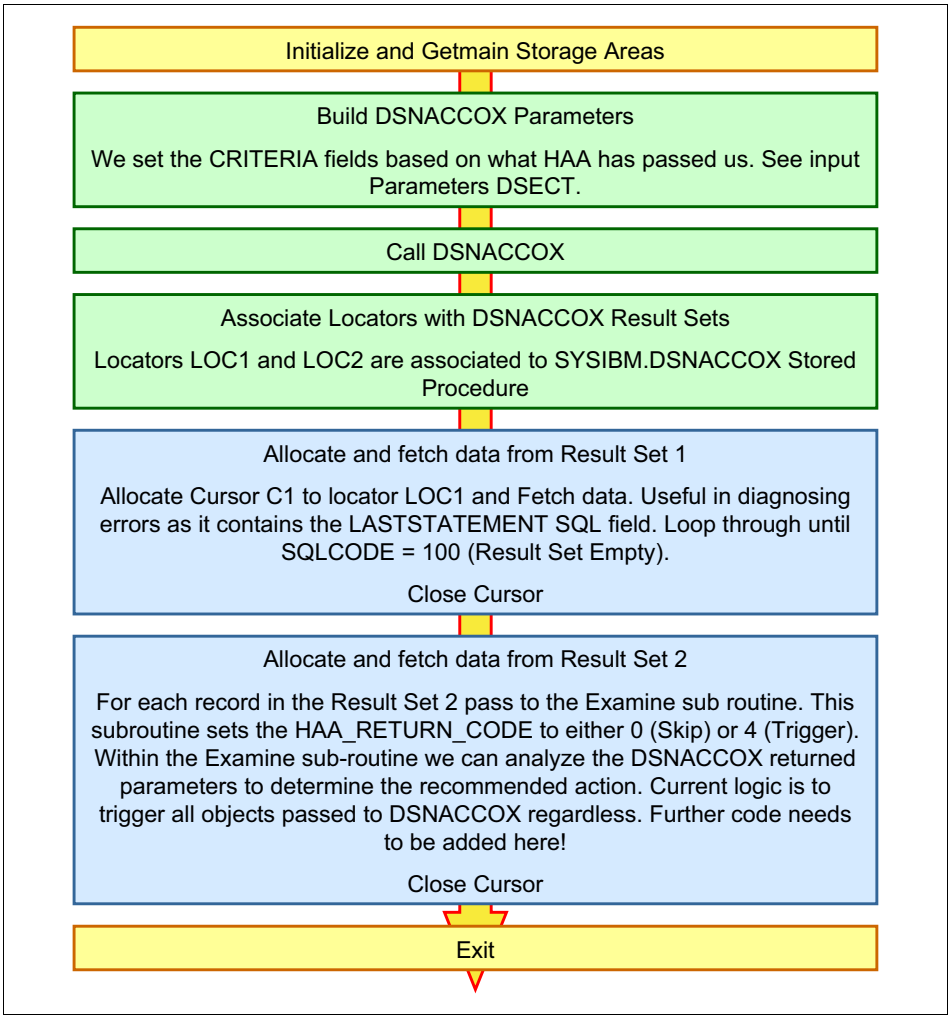

Figure 6-150 AUTOTEST logic when calling DSNACCOX

Assembling and installing AUTOTEST

AUTOTEST must be assembled and bound correctly for it to work successfully with DB2 Automation Tool. The stored procedure AUTOTEST must also be defined with the correct calling parameters. See Example 6-4.

See also the ASMSP and CREATESP sample jobs as described for download in Appendix C, “Additional material” on page 551.

Example 6-4 Stored procedure Autotest definition

```
//ADMR2D JOB (999,POK),'DB2 UTILITY',
//         REGION=0M,NOTIFY=&SYSUID,MSGCLASS=X,CLASS=T
//PROCLIB JCLLIB ORDER=DBOAM.PROCLIB
/*JOBPARM SYSAFF=SC63
//S01 EXEC DSNHASM,MEM=AUTOTEST,
//             PARM.PC='HOST(ASM),STDSQL(NO)',
//             PARM.ASM='NORENT,OBJECT,NODECK',
//             PARM.LKED='CALL,XREF,LIST,LET,RMODE=ANY,AMODE=31,
```

```
//             NORENT',USER=DBOAM
//SYSIN        DD DSN=ADMR2.DB2.ASM(AUTOTEST),DISP=SHR
//ASM.SYSLIB   DD DSN=CEE.SCEEMAC,DISP=SHR
//             DD DSN=SYS1.MACLIB,DISP=SHR
//             DD DSN=SYS1.MODGEN,DISP=SHR
//             DD DSN=DBOAT.SDSNMACS,DISP=SHR
//             DD DSN=DBOAT.SDSNSAMP,DISP=SHR
//             DD DSN=ADMR2.DB2.ASM,DISP=SHR
//LKED.SYSLMOD DD DSN=DBOAT.SDSNEXIT,DISP=SHR
//LKED.SYSLIB  DD DSN=CEE.SCEELKED,
//             DISP=SHR
//             DD DSN=CEE.SCEERUN,
//             DISP=SHR
//             DD DSN=CEE.SCEESPC,
//             DISP=SHR
//             DD DSN=CEE.SCEESPCO,
//             DISP=SHR
//             DD DSN=DBOAT.SDSNLOAD,
//             DISP=SHR
//LKED.SYSIN   DD *
 INCLUDE SYSLIB(DSNRLI)
 SETCODE AC(0)
 NAME AUTOTEST(R)
//S01     EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
//STEPLIB  DD  DISP=SHR,DSN=DBOAT.SDSNLOAD
//DBRMLIB  DD  DISP=SHR,DSN=DBOAM.DBRMLIB.DATA
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
  DSN SYSTEM(DBOA)
  FREE PLAN(AUTOTEST)
  FREE PACKAGE(AUTOTEST.AUTOTEST)
  BIND PACKAGE(AUTOTEST) MEM(AUTOTEST) -
       ACT(REP) ISO(CS) CURRENTDATA(YES) ENCODING(EBCDIC)

  BIND PLAN(AUTOTEST) PKLIST(AUTOTEST.AUTOTEST) -
       ACT(REP) ISO(CS) CURRENTDATA(YES) ENCODING(EBCDIC)

//BIND     EXEC PGM=HAA#BIND,PARM='DBOA'
//*
//STEPLIB  DD   DISP=SHR,DSN=HAA.SHAALOAD
//         DD   DISP=SHR,DSN=FEC.SFECLOAD
//         DD   DISP=SHR,DSN=DBOAT.SDSNEXIT
//         DD   DISP=SHR,DSN=DBOAT.SDSNLOAD
//DBRMLIB  DD   DISP=SHR,DSN=HAA.SHAADBRM
//         DD   DISP=SHR,DSN=DBOAM.DBRMLIB.DATA
//*
//SYSTSPRT DD   SYSOUT=*
//SYSUDUMP DD   SYSOUT=*
//SYSTSIN  DD   *
  DSN SYSTEM(DBOA)
  BIND PACKAGE (HAAC410U) -
       MEMBER (AUTOTEST) -
       QUALIFIER (ADMR2) -
       OWNER (DB2AUTH)  -
       ACTION (REPLACE) -
```

```
     DYNAMICRULES (RUN) -
     ENCODING (EBCDIC) -
     EXPLAIN (NO) -
     SQLERROR(CONTINUE) -
     ISOLATION (CS) -
     VALIDATE (RUN)
BIND PACKAGE (HAAC41OC) -
     MEMBER (AUTOTEST) -
     QUALIFIER (ADMR2) -
     OWNER (DB2AUTH)  -
     ACTION (REPLACE) -
     DYNAMICRULES (RUN) -
     ENCODING (EBCDIC) -
     EXPLAIN (NO) -
     SQLERROR(CONTINUE) -
     ISOLATION (CS) -
     VALIDATE (RUN)
BIND PLAN   (HAAP41O2)      -
     PKLIST (HAAC41O.*    -
             HAAC41OC.*    -
             HAAC41OS.*    -
             HAAC41OU.*    -
            )                       -
             ACTION       (REPLACE)  -
             RETAIN                  -
             DYNAMICRULES (RUN)      -
             EXPLAIN      (NO)       -
             ISOLATION    (CS)       -
             SQLRULES     (DB2)      -
             VALIDATE     (RUN)
END
```

6.6 DB2 Automation Tool and CHECK DATA

DB2 Automation Tool provides the capability to include all table spaces that are related via referential integrity (RI). A utility can maintain referential constraints by executing actions on all referentially related objects at the same time.

We show an example of checking data, including the referentially dependent table spaces. We start from the DB2 Automation Tool main panel shown in Figure 6-151 on page 254.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2012/05/04  16:25:13
Option  ===>
--------------------------------------------------------------------------------

       Options: 0 - Setup                   8 - Dataset Manager
                1 - Object Profiles         9 - Data Page Display
                2 - Utility Profiles       10 - Disaster Recovery
                3 - Exception Profiles     11 - Stand Alone Utilities
                4 - Job Profiles           12 - DB2 Admin Scheduler
                5 - Quick Build            13 - DB2 Autonomic Statistics
                6 - Execution Reports       X - Exit
                7 - DB2 Command Processor

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DBOA         (1-4 Character Subsystem ID or ? for list)
 Current SQLID:    ADMR4        User: ADMR4   - HAA
--------------------------------------------------------------------------------
```

Figure 6-151 DB2 Automation Tool main panel

We select option 1 to create an Object Profile. Figure 6-152 is displayed.

```
AUTOTOOL V4R1   -------- Objects Profile Display -------- 2012/05/04  16:38:58
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import
               Q - Quick   V - View    U - Update  J - Jobs    R - Rename
Profile Like  *                                        DB2 Subsystem: DBOA
Creator Like  ADMR4                                 Row 1 of 5             >
--------------------------------------------------------------------------------

Cmd  Name                            Creator   Updt
 c   CLUSTERSENS                     ADMR4      U
     DBOA BACKUP CATALOG             ADMR4      U
     GLWOA.GLWSO01                   ADMR4      U
     REORG WITH EXCEPTIONS           ADMR4      U
     RI DEPENDANT TABLESPACE         ADMR4      U
***************************** Bottom of Data********************************
```

Figure 6-152 Objects Profile Display

We type C in the Cmd field on any of the rows of the existing profiles, for instance, CLUSTERSENS, and a panel for new profile data is displayed. See Figure 6-153 on page 255.

```
AUTOTOOL V4R1   -------- Objects Profile Display -------- 2012/05/04  16:38:58
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import
               Q - Quick   V - View    U - Update  J - Jobs    R - Rename
Profile Like  *                                        DB2 Subsystem: DBOA
Creator Like  ADMR4                                  Row 1 of 5             >
--------------------------------------------------------------------------------
          Esssssssssssssssss Enter New Objects Profile Data sssssssssssssssssN
Cmd  Name e                                                                    e
 C   CLUS e                                                                    e
     DBOA e    Creator . . . . ADMR4                                           e
     GLWO e    Profile Name  . RI CHECK DATA                                   e
     REOR e    Description . . Check data                                      e
     RI D e    Update Option . U  (U - Update, V - View only, N - No access)   e
********* e                                                                    e
          e                                                                    e
          DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-153 Entering new Object Profile data

We press Enter and a new panel is displayed. See Figure 6-154.

```
AUTOTOOL V4R1   -------- Objects Profile Display -------- 2012/05/04  16:38:58
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import
               Q - Quick   V - View    U - Update  J - Jobs    R - Rename
Profile Like  *                                        DB2 Subsystem: DBOA
Creator Like  ADMR4                                  Row 1 of 5             >
--------------------------------------------------------------------------------
             Es Add Objects to the Object Profile sN
Cmd  Name    e                                    e
 C   CLUSTER e    Add Tablespaces . . Y (Yes/No)   e
     DBOA BA e                                    e
     GLWOA.G e    Add Indexes . . . . N (Yes/No)   e
     REORG W e                                    e
     RI DEPE e    Add Volumes . . . . N (Yes/No)   e
************ e                                    e *************************
             e                                    e
             DsssssssssssssssssssssssssssssssssssM
```

Figure 6-154 Window for profile data

On Figure 6-154, we type Y for Add Tablespaces and press Enter. A new panel is displayed (Figure 6-155 on page 256).

```
AUTOTOOL V4R1   -------- Objects Profile Display -------- 2012/05/04  16:38:58
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import
               Q - Quick   V - View    U - Update  J - Jobs    R - Rename
Profile Like  *                                        DB2 Subsystem: DBOA
Creator Like  ADMR4                                 Row 1 of 5             >
--------------------------------------------------------------------------------
Esssssssssssssssssss Enter Tablespaces Like to Display sssssssssssssssssssssN
e                                                                            e
e Database Like. . glw%     Wildcard   y (Yes/No)                            e
e Tablespace Like. *        Exclude    I (E - Exclude, I - Include)          e
e Creator Like . . *        >                                                e
e                                                                            e
e Process Dependent Indexes . . . . . . . . . .  N (Yes/No)                  e
e Process Referentially Dependent Tablespaces.  Y (Y - Yes, N - No,           e
e                                                  B - Build time Expansion, e
e                                                  R - Run time Expansion)   e
e Process Cloned Tables. . . . . . . . . . . .  N (Yes/No)                   e
e                                                                            e
e                                                                            e
e                                                                            e
DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-155 Objects Profile Display with referential integrity B and R options

On Figure 6-155, we type Y for Process Referentially Dependent Tablespaces, and press Enter. The new update is shown in Figure 6-156.

```
AUTOTOOL V4R1   ---- Update Object Profile Display    --- 2012/05/04  17:10:45
Option  ===> EXPLODE                                        Scroll ===> PAGE

--------------------------------------------------------------------------------
Commands: Explode - View all objects.
Line Commands: A - Add  D - Delete  E - Explode  U - Update  R - Repeat
Creator: ADMR4        Profile: RI CHECK DATA                  User: ADMR4
Description: CHECK DATA
Share Option U  (U - Update, V - View, N - No)        Row 1 of 1             >
--------------------------------------------------------------------------------
                                                  Volume /
         Wild ---- Process --- Inc/ IX DB Name/ IX Crtr/ IX Name/
Cmd Type Card IX RI Clone Util Exc  TS Crtr     DB Name  TS Name
    TS   Y    N  Y  N     N    INC  *          GLW%     *
***************************** Bottom of Data ***********************************
```

Figure 6-156 Update Object Profile Display

On Figure 6-156, we type EXPLODE to view all objects. Figure 6-157 on page 257 lists all objects.

```
AUTOTOOL V4R1   ----- Explode Object Profile Display ---- 2012/05/04  17:16:36
Option  ===>                                                  Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: S - Select to Exclude
Creator: ADMR4        Profile: RI CHECK DATA                  User: ADMR4

                                                          Row 1 of 52           +>
--------------------------------------------------------------------------------
                                                  Volume /
         Wild ---- Process --- Inc/ IX DB Name/ IX Crtr/ IX Name/
Cmd Type Card IX RI Clone Util Exc  TS Crtr     DB Name  TS Name
    TS   N    N  N  N     N    EXC  DB2R2       GLWSAMP  GLWSDPT
    TS   N    N  N  N     N    EXC  DB2R2       GLWSAMP  GLWSEMP
    TS   N    N  N  N     N    EXC  DB2R2       GLWSAMP  GLWSEPA
    TS   N    N  N  N     N    EXC  DB2R2       GLWSAMP  GLWSPJA
    TS   N    N  N  N     N    EXC  DB2R2       GLWSAMP  GLWSPRJ
    TS   N    N  N  N     N    EXC  DB2R2       GLWSAMP  GLWSSPL
    TS   N    N  N  N     N    EXC  DB2R2       GLWSAMP  GLWS001
    TS   N    N  N  N     N    EXC  DB2R2       GLWSAMP  GLWS002
    TS   N    N  N  N     N    EXC  DB2R2       GLWSAMP  GLWS003
    TS   N    N  N  N     N    EXC  GLWSAMP     GLWSAMP  XGLW0000
 s  TS   N    N  N  N     N    EXC  ADMR3       GLWS002  GLWSDPT
 s  TS   N    N  N  N     N    EXC  ADMR3       GLWS002  GLWSEMP
 s  TS   N    N  N  N     N    EXC  ADMR3       GLWS002  GLWSEPA
 s  TS   N    N  N  N     N    EXC  ADMR3       GLWS002  GLWSPJA
 s  TS   N    N  N  N     N    EXC  ADMR3       GLWS002  GLWSPRJ
 s  TS   N    N  N  N     N    EXC  ADMR3       GLWS002  GLWSSPL
 s  TS   N    N  N  N     N    EXC  ADMR3       GLWS002  GLWS001
 s  TS   N    N  N  N     N    EXC  ADMR3       GLWS002  GLWS002
 s  TS   N    N  N  N     N    EXC  ADMR3       GLWS002  GLWS003
```

Figure 6-157 Exploded list of objects

We are only interested in DB=GLWSAMP, so on Figure 6-157, we “Select to Exclude” all other objects with the line command S. Figure 6-158 on page 258 is displayed.

```
AUTOTOOL V4R1   ---- Update Object Profile Display    --- 2012/05/04  18:08:29
Option  ===>                                                Scroll ===> PAGE

--------------------------------------------------------------------------------
Commands: Explode - View all objects.
Line Commands: A - Add  D - Delete  E - Explode  U - Update  R - Repeat
Creator: ADMR4        Profile: RI CHECK DATA                  User: ADMR4
Description: CHECK DATA
Share Option U  (U - Update, V - View, N - No)        Row 1 of 44           +>
--------------------------------------------------------------------------------
                                                    Volume /
         Wild ---- Process --- Inc/ IX DB Name/ IX Crtr/ IX Name/
Cmd Type Card IX RI Clone Util Exc  TS Crtr     DB Name  TS Name
    TS   Y    N  Y  N     N    INC  *           GLW%     *
    TS   N    N  N  N     N    EXC  GLWSAMP     GLWSAMP  XGLWOOOO
    TS   N    N  N  N     N    EXC  ADMR3       GLWSOO2  GLWSDPT
    TS   N    N  N  N     N    EXC  ADMR3       GLWSOO2  GLWSEMP
    TS   N    N  N  N     N    EXC  ADMR3       GLWSOO2  GLWSEPA
    TS   N    N  N  N     N    EXC  ADMR3       GLWSOO2  GLWSPJA
    TS   N    N  N  N     N    EXC  ADMR3       GLWSOO2  GLWSPRJ
    TS   N    N  N  N     N    EXC  ADMR3       GLWSOO2  GLWSSPL
    TS   N    N  N  N     N    EXC  ADMR3       GLWSOO2  GLWSOO1
    TS   N    N  N  N     N    EXC  ADMR3       GLWSOO2  GLWSOO2
    TS   N    N  N  N     N    EXC  ADMR3       GLWSOO2  GLWSOO3
    TS   N    N  N  N     N    EXC  GLWSOO2     GLWSOO2  GLWTEPAE
    TS   N    N  N  N     N    EXC  GLWSOO2     GLWSOO2  XGLWOOOO
    TS   N    N  N  N     N    EXC  ADMR3       GLWSOO3  GLWSDPT
    TS   N    N  N  N     N    EXC  ADMR3       GLWSOO3  GLWSEMP
    TS   N    N  N  N     N    EXC  ADMR3       GLWSOO3  GLWSEPA
    TS   N    N  N  N     N    EXC  ADMR3       GLWSOO3  GLWSPJA
```

Figure 6-158 Update Object Profile Display with GLW% DB name

On Figure 6-158, we press Enter.

A panel is displayed that states that the Object queue has been modified, and we press PF3 to save the Object Profile.

Then, we select option 2 - Utility Profiles on the main panel in Figure 6-151 on page 254. We create a profile called JCL FOR CHECK DATA, as shown in Figure 6-159 on page 259.

```
AUTOTOOL V4R1 --------- Utility Profile Options --------- 2012/05/04  18:44:21
Option   ===>

  Creator: ADMR4      Profile: JCL FOR CHECK DATA              User: ADMR4
  Description  SET OF CMPL.RI SET CHECK DATA
  Share Option U (U - Update, V - View, N - No)


                                Include      Update
                                Utility      Utility
Data Page Verification Reporting N (Yes/No)   N (Yes/No)
Reallocation . . . . . . . . . . N (Yes/No)   N (Yes/No)
Recover  . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Image Copy . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Recovery Expert Image Copy . . . N (Yes/No)   N (Yes/No)
Copy to Copy . . . . . . . . . . N (Yes/No)   N (Yes/No)
Runstats . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
TS Reorg . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
IX Reorg . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Quiesce  . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Modify . . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Repair . . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Check Data . . . . . . . . . . . Y (Yes/No)   Y (Yes/No)
Rebind . . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
```

Figure 6-159 Utility profile creation for CHECK DATA

For Check Data, we type Y for Include Utility Check Data and Y for Update Utility Check Data and press Enter. We see the Check Data Utility Profile panel on Figure 6-160 on page 260.

```
AUTOTOOL V4R1 ---- Check Data Utility Profile Options --- 2012/05/04  19:05:16
Option   ===>                                                Scroll ===> PAGE

  Creator: ADMR4      Name: JCL FOR CHECK DATA              User: ADMR4
                                                                 More:     +
   Exception Rule  . . . . . . A (A - Accepted, R - Rejected, B - Both)
   Include Exception Tables  . Y (Yes/No)    Update Y (Yes/No)
   Utility ID  . . . . . . . . RICHECKDATA      (16 characters)
   Sharelevel  . . . . . . . . C    (R - Reference, C - Change)
   Drain Wait  . . . . . . . .      (blank, 0-1800 seconds)
   Retry . . . . . . . . . . .      (blank, 0-255)
   Retry Delay . . . . . . . .      (blank, 1-1800 seconds)
   Scope . . . . . . . . . . . P    (P - Pending, X - AuxOnly,
                                     A - All, R - RefOnly,
                                     S - XMLSchemaOnly)
   AUXerror  . . . . . . . . . R    (I - Invalidate, R - Report)
   LOBerror  . . . . . . . . . R    (I - Invalidate, R - Report)
   XMLerror  . . . . . . . . . R    (I - Invalidate, R - Report)
   Include ALL XML spaces  . . N    (Yes/No)
   Max Tape Vols/DASD Units. . 5    (blank, 1-255)
   Sort Device Type  . . . . .          (CART/DISK/etc.)
   Sort Number . . . . . . . .      (blank, 2-255)
   Delete  . . . . . . . . . . N    (Yes/No)
   Log . . . . . . . . . . . . N    (Yes/No)
   Exceptions  . . . . . . . . 0        (Number)


 Optional Skeletals:           -BEFORE-     -AFTER-
   JCL Skeletal  . . . . . .          . .          (8 Character Name)
   Control Cards Skeletal  . .        . .          (8 Character Name)
```

Figure 6-160 Check Data Utility Profile Options

On Figure 6-160, we type Y for Include Exception Tables. There are two ways to create Exception Tables: you can create them yourself or you can let DB2 Automation Tool create them for you. We type Update Y to let DB2 Automation Tool create them. We also type a RICHECKDATA for a Utility ID. We type C for Change for Sharelevel. We type P for Pending for Scope. We press Enter.

On the next panel, you type the options for the Exception Tables. See Figure 6-161 on page 261.

```
AUTOTOOL V4R1 ------- Check Data Exception options ------ 2012/05/04  19:14:27
Option   ===>                                                Scroll ===> PAGE

  Creator: ADMR4      Name: JCL FOR CHECK DATA              User: ADMR4

  Enter the options for Exception Tables

   Database . . . . . . . . . . . . DSNDB04  (8 Characters)
   Table space  . . . . . . . . . . EXCEP100 (8 Characters)
   Bufferpool . . . . . . . . . . . BP0      (8 Characters)
   Storage Group  . . . . . . . . . SYSDEFLT (8 Characters)
   Primary Quantity . . . . . . . . 120      (Number in kilobytes)
   Secondary Quantity . . . . . . . 120      (Number in kilobytes)
   Exception Table Creator  . . . . DB2AUTH  (8 Characters)
   Exception Table Suffix . . . . . A_EXCP   (8 Characters)
   Include RID column . . . . . . . Y        (Yes/No)
   Include Timestamp column . . . . Y        (Yes/No)
```

Figure 6-161 Definition of exception tables

On Figure 6-161, we define the options for the exception tables, and then press PF3 three times to save the Utility Profile.

Pressing PF3 again returns us to the primary menu (Figure 6-162). We choose option 4 - Job Profiles to create a Job Profile and press Enter.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2012/05/04  19:34:39
Option  ===> 4
--------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Disaster Recovery
                3 - Exception Profiles      11 - Stand Alone Utilities
                4 - Job Profiles            12 - DB2 Admin Scheduler
                5 - Quick Build             13 - DB2 Autonomic Statistics
                6 - Execution Reports        X - Exit
                7 - DB2 Command Processor

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DB0A         (1-4 Character Subsystem ID or ? for list)
 Current SQLID:    ADMR4        User: ADMR4   - HAA
--------------------------------------------------------------------------------
```

Figure 6-162 Choosing Job Profiles on the primary panel

The Jobs Profile Display panel (Figure 6-163 on page 262) is displayed.

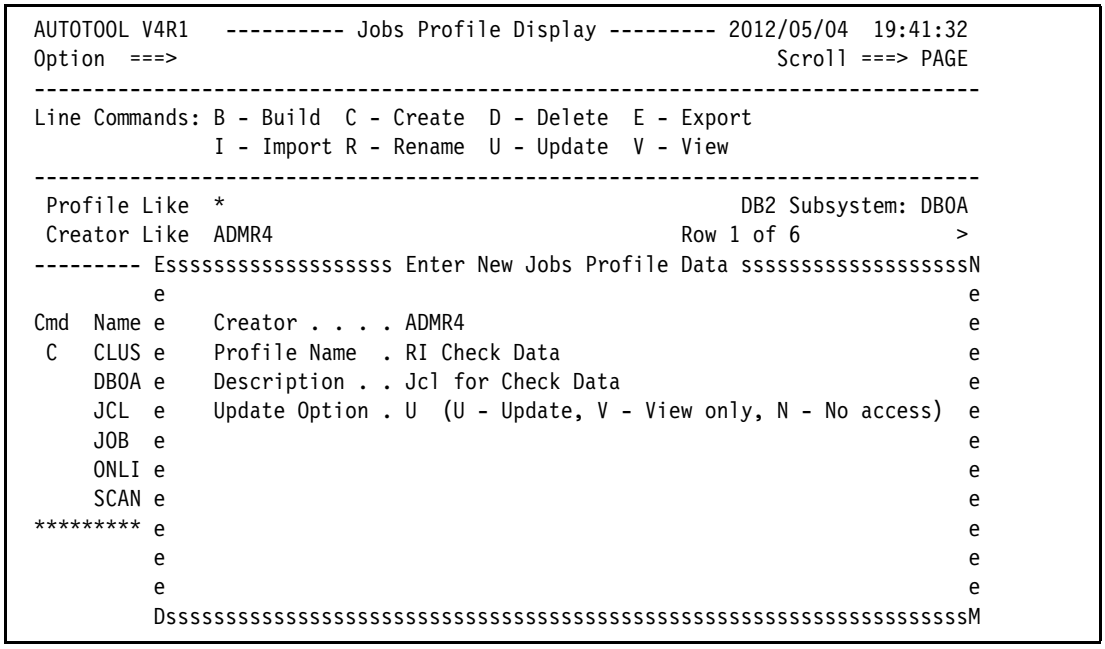

Figure 6-163 Jobs Profile Display

The previously defined job displays. We press Enter to go to the job details listed in Figure 6-164 on page 263. We see the default option profile panel.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2012/05/04  19:52:24
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
Esssssssssssssss Generation Options for ADMR4.RI TS CHECK DATA sssssssssssssssN
e                                                                              e
e Option  ===>                                                                 e
e                                                                              e
e                                                                 More:     +  e
e Update Setup Override Options  . . . . . N (Yes/No)                          e
e Update Template/Listdef/Option parms . . N (Yes/No)                          e
e Update Job Break Down Options  . . . . . N (Yes/No)                          e
e Automatically Gen GDG Base . . . . . . . 000 (0-255 Limit)                   e
e Load Balance jobs by . . . . . . . . . . N (T - Time, D - Dasd, N - None)    e
e Capture run times for Load Balancing . . N (Yes/No)                          e
e Start spaces in Utility/Read Only  . . . N (N - No, U - Utility,             e
e                                             R - Read only)                   e
e Prefix Utility ID with jobname . . . . . N (J - Job, S - Step, B - Both,     e
e                                             N - No)                          e
e Set JCL member equal to jobname  . . . . N (Yes/No)                          e
e Generate Job when Errors encountered . . Y (Y - Yes, N - No, W - Warnings)   e
e Evaluate Multiple Exception Profiles . . A (A - All, O - One at a time)      e
e Recall Migrated Spaces . . . . . . . . . N (Yes/No)                          e
e Use DSNACCOR Exception Table . . . . . . N (Yes/No)                          e
e Include Job Registration Step. . . . . . Y (Yes/No)                          e
e Utility work dataset high level  . . . . ADMR4    Optional                    e
e Pre-Generation User Exit Name. . . . . .         Optional                    e
DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-164 Generation Options for the Job Profile

The page that is shown in Figure 6-164 is the default. We use the default in this example. More information is available in the TSO Help or in the manual.

The option to “Include Job Registration Step” Y (Yes/No) for job tracking needs to have the started task Job Registration already installed. You can use Tools Customizer to do this. We set the “Utility work dataset high level” to our standard (User ID) and press PF3. The next panel is displayed (Figure 6-165 on page 264).

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2012/05/04  19:52:24
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  ADMR4                                  Row 1 of 4             >
------------ Essssssss Adding Profiles to the Job Profile sssssssssN ---------
             e                                                      e
Cmd  Name    e    Add Objects Profiles       Y    (Yes/No)          e
 C   CLUSTER e                                                     e
     DBOA BA e    Add Utilities Profiles     Y    (Yes/No)          e
     ONLIN R e                                                     e
     SCANACE e    Add Exception Profiles     N    (Yes/No)          e
************ e                                                     e *********
             e                                                      e
             e                                                      e
             e                                                      e
             DssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-165 Panel for adding profiles to the Job Profile

On Figure 6-165, we add the Object Profile and the Utility Profile that we have created and then we press Enter. Figure 6-166 is displayed.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2012/05/04  19:52:24
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  ADMR4                                  Row 1 of 4             >
------------ Essssss Enter Objects Profile Like to Display ssssssN ----------
             e                                                    e
Cmd  Name    e Profile Like *                                     e
 C   CLUSTER e Creator Like ADMR4                                 e
     DBOA BA e                                                    e
     ONLIN R e                                                    e
     SCANACE DssssssssssssssssssssssssssssssssssssssssssssssssssssM
***************************** Bottom of Data ********************************
```

Figure 6-166 Enter Objects Profile Like to Display panel

The Objects Profile page is displayed. We press Enter and select the Object Profile. We do the same for the Utility Profile. The panel in Figure 6-167 on page 265 is displayed.

```
AUTOTOOL V4R1   ------ Update Jobs Profile Display ------ 2012/05/04  20:20:38
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: V - View  A - Add  D - Delete  U - Update
--------------------------------------------------------------------------------
Creator: ADMR4        Profile: RI TS CHECK DATA               User: ADMR4
--------------------------------------------------------------------------------
Share Option U  (U - Update,     Description JCL FOR CHECK DATA
                 V - View,
                 N - No)
Update Job Generation Options N  (Yes/No)  Row 1 of 2             >
--------------------------------------------------------------------------------
                                                                <-------
Cmd  Type  Order  Name                              Creator   Userid
     OBJS     1   RI CHECK DATA                     ADMR4     ADMR4
     UTIL     1   JCL FOR CHECK DATA                ADMR4     ADMR4
***************************** Bottom of Data ***********************************
```

Figure 6-167 Jobs Profile Display for the Object and Utility Profiles

We press PF3 and the Job Profile is saved. See Figure 6-168.

On Figure 6-168, we proceed with the next step, which is to build the job.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2012/05/04  20:27:21
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *                                          DB2 Subsystem: DBOA
 Creator Like  ADMR4                                  Row 1 of 5             >
--------------------------------------------------------------------------------

Cmd  Name                            Creator   Updt
     CLUSTERSENS                     ADMR4      U
     DBOA BACKUP CATALOG             ADMR4      U
     ONLIN REORG WITH EXCEPTIONS     ADMR4      U
 B   RI TS CHECK DATA                ADMR4      U
     SCANACESS                       ADMR4      U
***************************** Bottom of Data ***********************************
```

Figure 6-168 Jobs Profile Display for CHECK DATA

We type B next to the job and press Enter. Figure 6-169 on page 266 is displayed.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2012/05/04  20:27:21
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
L Esssssssssssssssss Build Job for ADMR4.RI TS CHECK DATA sssssssssssssssssN
  e                                                                           e
- e Build Online or Batch. . B (O - Online, B - Batch)                        e -
  e                                                                           e
  e Edit Generated Job . . . Y (Yes/No)                                       e
- e Schedule Job . . . . . . N (Yes/No)   Update options . . N (Yes/No)       e -
  e                                                                           e
C e Build job in Dataset . . ADMR4.DATA.JCL                                   e
  e               Member . . CHECKOUT                                         e
  e                                                                           e
  e    Job Cards:                                                             e
  e ==> //ADMR4D JOB (999,POK),'DB2 UTILITY',                                 e
  e ==> //         REGION=0M,NOTIFY=ADMR4,MSGCLASS=X,CLASS=A                  e
* e ==> //*PROCLIB JCLLIB ORDER=DBOAM.PROCLIB                                 e *
  e ==> /*JOBPARM SYSAFF=SC63                                                 e
  e                                                                           e
  e                                                                           e
  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-169 Building the JCL

On Figure 6-169, we choose B for the Batch option and type Y to edit the generated job. We press Enter. The next page (Figure 6-170) is displayed.

```
AUTOTOOL V4R1   ---------- Jobs Profile Display --------- 2012/05/04  20:27:21
Option  ===>                                                 Scroll ===> PAGE
--------------------------------------------------------------------------------
L Esssssssssssssssss Build Job for ADMR4.RI TS CHECK DATA sssssssssssssssssN
  e                                                                           e
- e You have selected your job to be built in a batch mode.  The              e -
  e batch generation JCL will be stored in dataset                            e
  e ADMR4.DATA.JCL(CHECKOUT).                                                 e
- e Please specify in the dataset and member below where you want the         e -
  e JCL built by the batch module to be placed.                               e
C e                                                                           e
  e Schedule Job  . . . . . N (Yes/No)   Update options . . N (Yes/No)       e
  e                                                                           e
  e Build job in Dataset. . ADMR4.DATA.JCL                                    e
  e              Member . . CHECK1OO                                          e
  e                                                                           e
* e Jobcard data to be used on the generated job:                             e *
  e ==> //ADMR4D JOB (999,POK),'DB2 UTILITY',                                 e
  e ==> //         REGION=0M,NOTIFY=ADMR4,MSGCLASS=X,CLASS=A                  e
  e ==> //*PROCLIB JCLLIB ORDER=DBOAM.PROCLIB                                 e
  e ==> /*JOBPARM SYSAFF=SC63                                                 e
  e                                                                           e
  e                                                                           e
  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 6-170 Member definition for job

On Figure 6-170, we enter the member names and press Enter. The job shows in edit mode (Figure 6-171 on page 267).

```
ADMR4.DATA.JCL(CHECKOUT) - 01.00               Columns 00001 00072
 ===>  sub Scroll ===> PAGE
***************************** Top of Data ******************************
//ADMR4D JOB (999,POK),'DB2 UTILITY',
//         REGION=0M,NOTIFY=ADMR4,MSGCLASS=X,CLASS=A
//*PROCLIB JCLLIB ORDER=DBOAM.PROCLIB
/*JOBPARM SYSAFF=SC63
//*
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                      *
//*  Job Generated by IBM DB2 Automation Tool V4R1.01                    *
//*                                                                      *
//*  DB2 SSID: DBOA                                                      *
//*  SQLID:    ADMR4                                                     *
//*  Profile:  ADMR4.RI TS CHECK DATA                                    *
//*  Desc:     JCL FOR CHECK DATA                                        *
//*  User:     ADMR4                                                     *
//*  Date:     Friday May 04, 2012                                       *
//*  Time:     20:45:49.07                                               *
//*                                                                      *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                      *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                      *
//*  Step:     HAAÖBULD                                                  *
//*                                                                      *
//*  Desc:     This job will generate the JCL for jobs profile           *
//*            ADMR4.RI TS CHECK DATA in a batch mode.                   *
//*            The generated job will be placed in dataset               *
//*            ADMR4.DATA.JCL(CHECK100).                                 *
```

Figure 6-171 Job generated by DB2 Automation Tool for review and submission

On Figure 6-171, we review the generated build job, and then Submit the job, split panel, and go to SDSF; H.

Example 6-5 contains the job SYSOUT.

Example 6-5 Job generated with Profile ADMR4.RI TS CHECK DATA

```
ADMR4.DATA.JCL(CHECKOUT) - 01.00               Columns 00001 00072
 ===>  sub Scroll ===> PAGE
***************************** Top of Data ******************************
//ADMR4D JOB (999,POK),'DB2 UTILITY',
//         REGION=0M,NOTIFY=ADMR4,MSGCLASS=X,CLASS=A
//*PROCLIB JCLLIB ORDER=DBOAM.PROCLIB
/*JOBPARM SYSAFF=SC63
//*
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                      *
//*  Job Generated by IBM DB2 Automation Tool V4R1.01                    *
//*                                                                      *
//*  DB2 SSID: DBOA                                                      *
//*  SQLID:    ADMR4                                                     *
//*  Profile:  ADMR4.RI TS CHECK DATA                                    *
//*  Desc:     JCL FOR CHECK DATA                                        *
//*  User:     ADMR4                                                     *
//*  Date:     Friday May 04, 2012                                       *
//*  Time:     20:45:49.07                                               *
//*                                                                      *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                      *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
//*                                                                   *
//*  Step:     HAAÖBULD                                               *
//*                                                                   *
//*  Desc:     This job will generate the JCL for jobs profile        *
//*            ADMR4.RI TS CHECK DATA in a batch mode.                *
//*            The generated job will be placed in dataset            *
//*            ADMR4.DATA.JCL(CHECK1OO).                              *
//*                                                                   *
//*  Return Codes:                                                    *
//*                                                                   *
//*  (OO) - Job was built successfully with no warnings or errors     *
//*                                                                   *
//*  (O4) - Job was built with warning messages and the Build Job on  *
//*         Errors indicator was a "Y" or "W"                         *
//*                                                                   *
//*  (O6) - Job was not built - Exception processing did not flag     *
//*         any objects to process.                                   *
//*                                                                   *
//*  (O8) - Job was built with error messages and the Build Job on    *
//*         Errors indicator was a "Y"                                *
//*                                                                   *
//*  (12) - Job was not built - Errors were detected and the Build Job *
//*         on Errors indicator was not a "Y"                         *
//*                                                                   *
//*  Note:  Build Job on Errors is an option in the Jobs Profile      *
//*         Options screen.  This option has the following values:    *
//*         "Yes" - Build job on Errors or Warnings                   *
//*         "No"  - Do not build job on Errors or Warnings            *
//*         "Warnings" - Build job only if highest severity is a      *
//*                      warning message.                             *
//*                                                                   *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//HAAÖBULD EXEC PGM=IKJEFT1A,REGION=OOOOM
//*
//HAAERROR DD SYSOUT=*
//DLCÅEXCP DD SYSOUT=*
//EXCEPTNS DD SYSOUT=*
//RUNSTATS DD SYSOUT=*
//TRIGGERS DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//STPRINO1 DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//DLCDEBUG DD SYSOUT=*
//DB2PARMS DD DSN=DB2TOOLS.DB2.CONTROL,DISP=SHR
//SYSPROC  DD DSN=DB2TOOLS.HAA41O.CLIST,DISP=SHR
//STEPLIB  DD DSN=HAA.SHAALOAD,DISP=SHR
//         DD DSN=FEC.SFECLOAD,DISP=SHR
//         DD DSN=DBOAT.SDSNEXIT,DISP=SHR
//         DD DSN=DBOAT.SDSNLOAD,DISP=SHR
//ISPLLIB  DD DSN=HAA.SHAALOAD,DISP=SHR
//SYSTSIN  DD *
  PROFILE  NOPREFIX
  ISPSTART PGM(HAAÖBULD)
//*
//HAAÄDATA DD *
  GENERATE_UTILITY_JOB (                                          -
    DB2_SUBSYSTEM      DBOA                                       -
    USER_INDICATOR     HAA                                        -
    DB2_SQLID          ADMR4                                      -
    PROFILE_NAME      'RI TS CHECK DATA'                          -
    PROFILE_CREATOR    ADMR4                                      -
    PROFILE_DESCRIPTION 'JCL FOR CHECK DATA'                      -
    EXECUTION_LIB_2    HAA.SHAALOAD                               -
    EXECUTION_LIB_4    FEC.SFECLOAD                               -
    GEN_TO_DATASET     ADMR4.DATA.JCL                             -
    DEBUG_MODE         OFF                                        -
    GEN_TO_MEMBER      CHECK1OO                                   -
    JOB_CARD_1_1      '//ADMR4D JOB (999,POK),"DB2 UTILITY",'     -
    JOB_CARD_2_1      '//         REGION=OM,NOTIFY=ADMR4,MSGCLA'  -
    JOB_CARD_2_2      'SS=X,CLASS=A'                             -
    JOB_CARD_3_1      '//*PROCLIB JCLLIB ORDER=DBOAM.PROCLIB'     -
    JOB_CARD_4_1      '/*JOBPARM SYSAFF=SC63'                     -
                       )
//*
HAAB048I DB2 SubSystem ID:DBOA; DB2 Version:1O1O; SQLID:ADMR4; ZUSER:ADMR4
HAAB027I Jobs Generation Options follow:
HAAB028I Maximum Number of Jobs.....................1
HAAB029I Maximum Number of Objects per Job.........O
HAAB375I Maximum Number of Objects per Step........99999
HAAB376I Pad Jobs if Max not Exceeded..............Y
HAAB03OI Automatically generate GDG Base...........OOO
HAAB031I Load Balance Jobs by......................N
```

```
HAAB032I Capture Run Times for Load Balancing......N
HAAB033I Process Spaces in Utility (UT) Mode.......N
HAAB034I Prefix Utility ID with Jobname............N
HAAB035I Set JCL Member Name to Jobname............N
HAAB036I Generate Job When Errors Encountered......Y
HAAB106I Evaluate Multiple Exception Profiles......All together
HAAB112I Recall Migrated Spaces....................N
HAAB113I Use DSNACCOR Exception Table..............N
HAAB203I Rebind Dependent Plans / Packages.........N
HAAB037I Utility Dataset High Level Qualifier......ADMR4
HAAB039I Retrieve Jobcard and Comments from Dataset
HAAB040I   Dataset:
HAAB041I   Member:
HAAB042I Jobname Template " - - - - - - - - "
HAAB176I Generate Templates........................Y
HAAB049I Using JOBS Profile    ADMR4.RI TS CHECK DATA                    that includes..
HAAB050I       OBJS Profile    ADMR4.RI CHECK DATA
HAAB050I       UTIL Profile    ADMR4.JCL FOR CHECK DATA

HAAB025I Build JCL will be written to ADMR4.DATA.JCL.
HAAB026I Build JCL member "CHECK100" successfully written.
READY
  PROFILE  NOPREFIX
READY
  ISPSTART PGM(HAAÖBULD)
READY
END
Control card stream processed by IBM Shared Profile Support...

GENERATE_UTILITY_JOB ( DB2_SUBSYSTEM DBOA USER_INDICATOR HAA DB2_SQLID ADMR4 PROFILE_NAME 'RI TS CHECK DATA' PROFILE_CREATOR ADMR4
PROFILE_DESCRIPTION 'JCL FOR CHECK DATA' EXECUTION_LIB_2 HAA.SHAALOAD EXECUTION_LIB_4 FEC.SFECLOAD GEN_TO_DATASET ADMR4.DATA.JCL
DEBUG_MODE OFF GEN_TO_MEMBER CHECK100
JOB_CARD_1_1 '//ADMR4D JOB (999,POK),"DB2 UTILITY",' JOB_CARD_2_1 '//         REGION=0M,NOTIFY=ADMR4,MSGCLA' JOB_CARD_2_2 'SS=X,CLASS=A'
JOB_CARD_3_1 '//*PROCLIB JCLLIB ORDER=DBOAM.PROCLIB' JOB_CARD_4_1 '/*JOBPARM SYSAFF=SC63' )
IBM Shared Profile Support messages follow...

Using JOBS Profile    ADMR4.RI TS CHECK DATA                    that includes..
      OBJS Profile    ADMR4.RI CHECK DATA
Number of Accepted Objects.....................................9
Number of Rejected Objects.....................................0
Total Number of Objects Included in Generated JCL..............9
         TS IndexSpc              Index-TS DS- DS- --Current- --Current- Cur Max --Max- New-Prim New-Secn New--DB2 New--DB2 Acc Rej
Jobname- or TableSpc Database Part --Name-- Ext Num Allocation ---Used--- Usd -GB -Alloc Allocatn Allocatn Allocatn Allocatn Utl Utl
         IX                                    ----KB---- ----KB---- -%-     ---%-- ---KB--- ---KB--- Prim--KB Secn--KB Cnt Cnt
ADMR4D   TS GLWSDPT  GLWSAMP     0            1   1       1488       1488 100  64   0.00        0        0        0        0   1   0
         TS GLWSEMP  GLWSAMP     1            1   1       3744       2552  68   4   0.08        0        0        0        0   1   0
         TS GLWSEMP  GLWSAMP     2            1   1       3744       2516  67   4   0.08        0        0        0        0   1   0
         TS GLWSEMP  GLWSAMP     3            1   1       3744       2528  67   4   0.08        0        0        0        0   1   0
         TS GLWSEMP  GLWSAMP     4            1   1       3744       2188  58   4   0.08        0        0        0        0   1   0
         TS GLWSEPA  GLWSAMP     1            1   1      64944      64944 100  3K   0.00        0        0        0        0   1   0
         TS GLWSEPA  GLWSAMP     2            1   1          0          8   0  3K   0.00        0        0        0        0   1   0
         TS GLWSEPA  GLWSAMP     3            1   1          0          8   0  3K   0.00        0        0        0        0   1   0
         TS GLWSEPA  GLWSAMP     4            1   1          0          8   0  3K   0.00        0        0        0        0   1   0
         TS GLWSEPA  GLWSAMP   ALL            4   4      64944      64968   4  3K   0.00        0        0        0        0   4   0
         TS GLWSPJA  GLWSAMP     0            1   1      25344      25344 100  64   0.03        0        0        0        0   1   0
         TS GLWSPRJ  GLWSAMP     0            1   1      14544      14544 100  64   0.02        0        0        0        0   1   0
         TS GLWSSPL  GLWSAMP     1            1   1        720          8   1   4   0.01        0        0        0        0   1   0
         TS GLWSSPL  GLWSAMP     2            1   1        720          8   1   4   0.01        0        0        0        0   1   0
         TS GLWSSPL  GLWSAMP     3            1   1        720          8   1   4   0.01        0        0        0        0   1   0
         TS GLWSSPL  GLWSAMP     4            2   1       2160       2160 100   4   0.05        0        0        0        0   1   0
         TS GLWS001  GLWSAMP     0            1   1        240        188  78  64   0.00        0        0        0        0   1   0
```

In the job output that is shown in Example 6-5 on page 267, we see the input definitions for the CHECK DATA generation and the list of objects included in the generated JCL.

The job is saved in member ADMR4.DATA.JCL(CHECK100).

Figure 6-172 on page 270 shows the JCL for registration.

```
EDIT       ADMR4.DATA.JCL(CHECK100) - 01.00               Columns 00001 00072
Command ===>                                                Scroll ===> HALF
****** ***************************** Top of Data ******************************
000001 //ADMR4D JOB (999,POK),'DB2 UTILITY',
000002 //         REGION=0M,NOTIFY=ADMR4,MSGCLASS=X,CLASS=A
000003 //*PROCLIB JCLLIB ORDER=DBOAM.PROCLIB
000004 /*JOBPARM SYSAFF=SC63
000005 //*
000006 //** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
000007 //*                                                                     *
000008 //*  Job Generated by IBM DB2 Automation Tool V4R1.01                   *
000009 //*                                                                     *
000010 //*  DB2 SSID: DBOA                                                     *
000011 //*  SQLID:    ADMR4                                                    *
000012 //*  Profile:  ADMR4.RI TS CHECK DATA                                   *
000013 //*  Desc:     JCL FOR CHECK DATA                                       *
000014 //*  User:     ADMR4                                                    *
000015 //*  Date:     Monday May 07, 2012                                      *
000016 //*  Time:     12:07:29.20                                              *
000017 //*                                                                     *
000018 //** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
000019 //*                                                                     *
000020 //** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
000021 //*                                                                     *
000022 //*  Step:     REGISTER                                                 *
000023 //*                                                                     *
000024 //*  Desc:     This step will communicate with the Execution            *
000025 //*            facility and register this job as active.                *
000026 //*                                                                     *
000027 //*  Input:    JOB_TRACK_SSID   - Execution Facility STC Subsystem      *
```

Figure 6-172 Job registration

Figure 6-173 on page 271 shows the member resulting from the analysis job, which we will submit.

```
000070 //** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
000071 //*                                                                   *
000072 //*  Step:     EXC00102                                              *
000073 //*                                                                   *
000074 //*  Desc:     This step will create the check data exception table   *
000075 //*            objects.                                               *
000076 //*                                                                   *
000077 //** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
000078 //*                                                                   *
000079 //EXC00102 EXEC  PGM=HAAÖBSQL,REGION=0000M,COND=(4,LT),
000080 //         PARM=(DBOA,HAAP4101)
000081 //*
000082 //STEPLIB  DD DSN=HAA.SHAALOAD,DISP=SHR
000083 //         DD DSN=FEC.SFECLOAD,DISP=SHR
000084 //         DD DSN=DBOAT.SDSNEXIT,DISP=SHR
000085 //         DD DSN=DBOAT.SDSNLOAD,DISP=SHR
000086 //DB2PARMS DD DSN=DB2TOOLS.DB2.CONTROL,DISP=SHR
000087 //SYSOUT   DD SYSOUT=*
000088 //SQLÄOUT  DD SYSOUT=*
000089 //SQLÄIN   DD *
000090       SET CURRENT SQLID = 'ADMR4';
000091
000092       )IF DATABASE DSNDB04 NOT_EXISTS
000093          CREATE DATABASE DSNDB04
000094             STOGROUP   SYSDEFLT
000095             BUFFERPOOL BP0;
000096
000097          COMMIT;
```

Figure 6-173 Create default database for exception tables

Figure 6-174 on page 272 shows the Check Data exceptions table creation. There is a check to verify whether you have already created them. The names are from Figure 6-161 on page 261 (Database, Tablespace, and exception table name).

```
 ADMR4.DATA.JCL(CHECK100) - 01.00               Columns 00001 00072
==>                                                 Scroll ===> HALF
    )ENDIF

    )IF TABLESPACE DSNDB04.EXCEP100 NOT_EXISTS
       CREATE TABLESPACE EXCEP100 IN DSNDB04
          USING STOGROUP   SYSDEFLT
                PRIQTY     120
                SECQTY     120
                ERASE      NO
          BUFFERPOOL BP0
          LOCKSIZE   TABLESPACE
          CLOSE      NO
          SEGSIZE    24;

       COMMIT;
    )ENDIF

    )IF TABLE DB2AUTH."GLWTDPTA_EXCP" NOT_EXISTS
       CREATE TABLE DB2AUTH."GLWTDPTA_EXCP"
         LIKE GLWSAMP."GLWTDPT"
         IN  DSNDB04.EXCEP100;

       COMMIT;
```

Figure 6-174 Check Data exceptions table space and table GLWTDPTA_EXCP creation

Figure 6-175 shows the ALTER for adding a timestamp.

```
ADMR4.DATA.JCL(CHECK100) - 01.00               Columns 00001 00072
=>                                                  Scroll ===> HALF
      ALTER TABLE DB2AUTH."GLWTDPTA_EXCP"
        ADD COLUMN CHECK_TIMESTAMP  TIMESTAMP;

      COMMIT;
   )ENDIF

   )IF TABLE DB2AUTH."GLWTEMPA_EXCP" NOT_EXISTS
      CREATE TABLE DB2AUTH."GLWTEMPA_EXCP"
        LIKE GLWSAMP."GLWTEMP"
        IN  DSNDB04.EXCEP100;

      COMMIT;

      ALTER TABLE DB2AUTH."GLWTEMPA_EXCP"
        ADD COLUMN CHECK_ROWID  CHAR(5);

      COMMIT;

      ALTER TABLE DB2AUTH."GLWTEMPA_EXCP"
        ADD COLUMN CHECK_TIMESTAMP  TIMESTAMP;

      COMMIT;
   )ENDIF
```

Figure 6-175 Adding timestamp to the exception table

In Figure 6-175 on page 272, the job does ALTER TABLE for adding a rowid column. Further down, it also adds a TIMESTAMP column.

Example 6-6 is the job invoking the utility.

Example 6-6 Job invoking Check Data

```
ADMR4.DATA.JCL(CHECK100) - 01.00               Columns 00001 00072
 ===>                                                 Scroll ===> HALF
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//*  Step:     CKD00103                                                 *
//*                                                                     *
//*  Desc:     This step will invoke the IBM Checkdata Utility          *
//*                                                                     *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//CKD00103 EXEC  PGM=DSNUTILB,REGION=0000M,COND=(4,LT),
//          PARM=(DB0A,'RICHECKDATA')
//*
//STEPLIB  DD DSN=HAA.SHAALOAD,DISP=SHR
//         DD DSN=FEC.SFECLOAD,DISP=SHR
//         DD DSN=DB0AT.SDSNEXIT,DISP=SHR
//         DD DSN=DB0AT.SDSNLOAD,DISP=SHR
//SYSOUT   DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD DSN=ADMR4.ADMR4D.CKD00103.CHKDATA.SYSUT1,
//            DISP=(MOD,DELETE,CATLG),
//            UNIT=(SYSALLDA,5),SPACE=(CYL,(5,1),,,ROUND)
//SORTOUT  DD DSN=ADMR4.ADMR4D.CKD00103.CHKDATA.SORTOUT,
//            DISP=(MOD,DELETE,CATLG),
//            UNIT=(SYSALLDA,5),SPACE=(CYL,(5,1),,,ROUND)
//SYSPUNCH DD DSN=ADMR4.ADMR4D.CKD00103.CHKDATA.SYSPUNCH,
//            DISP=(MOD,CATLG,CATLG),
//            UNIT=(SYSALLDA,5),SPACE=(CYL,(5,1),,,ROUND)
//SYSERR   DD DSN=ADMR4.ADMR4D.CKD00103.CHKDATA.SYSERR,
//SYSIN    DD *
  CHECK DATA
    TABLESPACE GLWSAMP.GLWSDPT
    TABLESPACE GLWSAMP.GLWSEMP
    TABLESPACE GLWSAMP.GLWSEPA
    TABLESPACE GLWSAMP.GLWSPJA
    TABLESPACE GLWSAMP.GLWSPRJ
    TABLESPACE GLWSAMP.GLWSSPL
    TABLESPACE GLWSAMP.GLWS001
    TABLESPACE GLWSAMP.GLWS002
    TABLESPACE GLWSAMP.GLWS003
      SHRLEVEL CHANGE
      SCOPE PENDING
      AUXERROR REPORT
      LOBERROR REPORT
      XMLERROR REPORT
      FOR EXCEPTION
       IN "GLWSAMP"."GLWTDPT"
         USE "DB2AUTH"."GLWTDPTA_EXCP"
```

```
 IN "GLWSAMP"."GLWTEMP"
   USE "DB2AUTH"."GLWTEMPA_EXCP"
 IN "GLWSAMP"."GLWTEPA"
   USE "DB2AUTH"."GLWTEPAA_EXCP"
 IN "GLWSAMP"."GLWTPJA"
   USE "DB2AUTH"."GLWTPJAA_EXCP"
 IN "GLWSAMP"."GLWTPRJ"
   USE "DB2AUTH"."GLWTPRJA_EXCP"
 IN "GLWSAMP"."GLWTSPL"
   USE "DB2AUTH"."GLWTSTRA_EXCP"
 IN "GLWSAMP"."GLWTTWN"
   USE "DB2AUTH"."GLWTTWNA_EXCP"
 IN "GLWSAMP"."GLWTVRN"
   USE "DB2AUTH"."GLWTVRNA_EXCP"
 IN "GLWSAMP"."GLWTDNG"
   USE "DB2AUTH"."GLWTDNGA_EXCP"
 IN "GLWSAMP"."GLWTENG"
   USE "DB2AUTH"."GLWTENGA_EXCP"
 IN "GLWSAMP"."GLWTLNG"
   USE "DB2AUTH"."GLWTLNGA_EXCP"
 IN "GLWSAMP"."GLWTPNG"
   USE "DB2AUTH"."GLWTPNGA_EXCP"
 IN "GLWSAMP"."GLWTSQL"
   USE "DB2AUTH"."GLWTSQLA_EXCP"
EXCEPTIONS 0

/*
//*
***************************** Bottom of Data ****************************
```

We then look at the output SDSF; H in Figure 6-176 on page 275.

```
 SDSF OUTPUT DISPLAY ADMR4D   JOB09445  DSID     2 LINE 0       COLUMNS 02- 81
 COMMAND INPUT ===>                                           SCROLL ===> PAGE
******************************** TOP OF DATA *********************************
                  J E S 2  J O B  L O G  --  S Y S T E M  S C 6 3  --  N O D E

12.07.55 JOB09445 ---- TUESDAY,   08 MAY 2012 ----
12.07.55 JOB09445  IRR010I  USERID ADMR4    IS ASSIGNED TO THIS JOB.
12.07.55 JOB09445  ICH70001I ADMR4    LAST ACCESS AT 11:19:28 ON TUESDAY, MAY 8,
12.07.55 JOB09445  ÅHASP373 ADMR4D   STARTED - INIT 3    - CLASS A - SYS SC63
12.07.55 JOB09445  IEF403I ADMR4D - STARTED - TIME=12.07.55 - ASID=0099 - SC63
12.07.55 JOB09445  HAA1204I REGISTRATION WITH SUBSYSTEM DBA  SUCCESSFUL
12.07.55 JOB09445  -                                        --TIMINGS (MINS.)--
12.07.55 JOB09445  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK
12.07.55 JOB09445  -ADMR4D            REG00101    00     45    .00    .00    .00
12.07.56 JOB09445  -ADMR4D            EXC00102    00    157    .00    .00    .00
12.07.56 JOB09445  -ADMR4D            CKD00103    04    756    .00    .00    .00
12.07.56 JOB09445  IEF404I ADMR4D - ENDED - TIME=12.07.56 - ASID=0099 - SC63
12.07.56 JOB09445  -ADMR4D   ENDED.  NAME-DB2 UTILITY          TOTAL CPU TIME=
12.07.56 JOB09445  ÅHASP395 ADMR4D   ENDED
------ JES2 JOB STATISTICS ------
  08 MAY 2012 JOB EXECUTION DATE
          571 CARDS READ
          761 SYSOUT PRINT RECORDS
            0 SYSOUT PUNCH RECORDS
           56 SYSOUT SPOOL KBYTES
         0.01 MINUTES EXECUTION TIME
```

Figure 6-176 Check Data output

We look at some of the pages in the SYSOUT.

Figure 6-177 on page 276 shows the details of the execution of the SQL statements.

```
--*********************************************************************
--*  Sql Instruction Executed  SQLCODE: 0000                          *
--*********************************************************************
--
--    )IF DATABASE DSNDB04 NOT_EXISTS
--*********************************************************************
--* Object exists.  DDL Bypassed.                                     *
--*********************************************************************
--       CREATE DATABASE DSNDB04
--          STOGROUP   SYSDEFLT
--          BUFFERPOOL BP0;                                          w1- xj
--
--       COMMIT;
--    )ENDIF
--
--    )IF TABLESPACE DSNDB04.EXCEP100 NOT_EXISTS
--*********************************************************************
--* Object exists.  DDL Bypassed.                                     *
--*********************************************************************
--       CREATE TABLESPACE EXCEP100 IN DSNDB04
--          USING STOGROUP   SYSDEFLT
--                PRIQTY     120
--                SECQTY     120
--                ERASE      NO
--          BUFFERPOOL BP0
--          LOCKSIZE   TABLESPACE
--          CLOSE      NO
--          SEGSIZE    24;
```

Figure 6-177 SQL statements are bypassed

As we can see, the object already exists, so the Data Definition Language (DDL) is bypassed.

Figure 6-178 on page 277 shows the SDSF output with the details about the Check Data execution.

```
 SDSF OUTPUT DISPLAY ADMR4D   JOB09445  DSID   109 LINE 3       COLUMNS 01- 132
 COMMAND INPUT ===>                                          SCROLL ===> PAGE
ODSNU050I    129 12:07:56.40 DSNUGUTC -  CHECK DATA TABLESPACE GLWSAMP.GLWSDPT TABLESPACE GLWSAMP.GLWSEMP
TABLESPACE
 GLWSAMP.GLWSEPA TABLESPACE GLWSAMP.GLWSPJA TABLESPACE GLWSAMP.GLWSPRJ TABLESPACE GLWSAMP.GLWSSPL TABLESPACE
 GLWSAMP.GLWSO01 TABLESPACE GLWSAMP.GLWSO02 TABLESPACE GLWSAMP.GLWSO03 SHRLEVEL CHANGE SCOPE PENDING AUXERROR
REPORT
  LOBERROR REPORT XMLERROR REPORT FOR EXCEPTION IN "GLWSAMP"."GLWTDPT" USE "DB2AUTH"."GLWTDPTA_EXCP" IN
 "GLWSAMP"."GLWTEMP" USE "DB2AUTH"."GLWTEMPA_EXCP" IN "GLWSAMP"."GLWTEPA" USE "DB2AUTH"."GLWTEPAA_EXCP" IN
 "GLWSAMP"."GLWTPJA" USE "DB2AUTH"."GLWTPJAA_EXCP" IN "GLWSAMP"."GLWTPRJ" USE "DB2AUTH"."GLWTPRJA_EXCP" IN
 "GLWSAMP"."GLWTSPL" USE "DB2AUTH"."GLWTSPLA_EXCP" IN "GLWSAMP"."GLWTACT" USE "DB2AUTH"."GLWTACTA_EXCP" IN
 "GLWSAMP"."GLWTJBS" USE "DB2AUTH"."GLWTJBSA_EXCP" IN "GLWSAMP"."GLWTLCN" USE "DB2AUTH"."GLWTLCNA_EXCP" IN
 "GLWSAMP"."GLWTLNM" USE "DB2AUTH"."GLWTLNMA_EXCP" IN "GLWSAMP"."GLWTPGW" USE "DB2AUTH"."GLWTPGWA_EXCP" IN
 "GLWSAMP"."GLWTSFN" USE "DB2AUTH"."GLWTSFNA_EXCP" IN "GLWSAMP"."GLWTSTR" USE "DB2AUTH"."GLWTSTRA_EXCP" IN
 "GLWSAMP"."GLWTTWN" USE "DB2AUTH"."GLWTTWNA_EXCP" IN "GLWSAMP"."GLWTVRN" USE "DB2AUTH"."GLWTVRNA_EXCP" IN
 "GLWSAMP"."GLWTDNG" USE "DB2AUTH"."GLWTDNGA_EXCP" IN "GLWSAMP"."GLWTENG" USE "DB2AUTH"."GLWTENGA_EXCP" IN
 "GLWSAMP"."GLWTLNG" USE "DB2AUTH"."GLWTLNGA_EXCP" IN "GLWSAMP"."GLWTPNG" USE "DB2AUTH"."GLWTPNGA_EXCP" IN
 "GLWSAMP"."GLWTSQL" USE "DB2AUTH"."GLWTSQLA_EXCP" EXCEPTIONS 0
 DSNU727I  -DBOA 129 12:07:56.41 DSNUKINP - TABLESPACE 'GLWSAMP.GLWSDPT' IS NOT CHECK PENDING
 DSNU727I  -DBOA 129 12:07:56.41 DSNUKINP - TABLESPACE 'GLWSAMP.GLWSEMP' IS NOT CHECK PENDING
 DSNU727I  -DBOA 129 12:07:56.41 DSNUKINP - TABLESPACE 'GLWSAMP.GLWSEPA' IS NOT CHECK PENDING
 DSNU727I  -DBOA 129 12:07:56.41 DSNUKINP - TABLESPACE 'GLWSAMP.GLWSPJA' IS NOT CHECK PENDING
 DSNU727I  -DBOA 129 12:07:56.41 DSNUKINP - TABLESPACE 'GLWSAMP.GLWSPRJ' IS NOT CHECK PENDING
 DSNU727I  -DBOA 129 12:07:56.41 DSNUKINP - TABLESPACE 'GLWSAMP.GLWSSPL' IS NOT CHECK PENDING
 DSNU727I  -DBOA 129 12:07:56.41 DSNUKINP - TABLESPACE 'GLWSAMP.GLWSO01' IS NOT CHECK PENDING
 DSNU727I  -DBOA 129 12:07:56.41 DSNUKINP - TABLESPACE 'GLWSAMP.GLWSO02' IS NOT CHECK PENDING
 DSNU727I  -DBOA 129 12:07:56.41 DSNUKINP - TABLESPACE 'GLWSAMP.GLWSO03' IS NOT CHECK PENDING
```

Figure 6-178 CHECK DATA output

7

IBM DB2 Utilities Enhancement Tool for z/OS

The maintenance window can be managed by the IBM DB2 Utilities Enhancement Tool for z/OS Version 2.2 (DB2 Utilities Enhancement Tool). This tool provides utility governance and transparent thread control, ensuring that the maintenance window is not compromised by unwanted or unnecessary thread delays. This is done through an interface with DSNUTILB, which is called the *DSNUTILB Intercept*. A few scenarios are described to illustrate the added benefit of using DB2 Utilities Enhancement Tool when running your DB2 utilities.

Through the use of the Utility Monitor, DB2 Utilities Enhancement Tool can enforce company-wide standards in utility syntax. This helps the DBA support multiple or diverse environments. The Utility Monitor provides the needs of DBAs so that DBAs can be confident that the utility jobs conform to company syntax standards, therefore, providing a secure execution environment for junior staff. Having syntax standards also reduces errors and provides tighter auditing on utility execution.

DB2 Utilities Enhancement Tool also provides additional functionality to some of the DB2 utilities. For example, DB2 Utilities Enhancement Tool can automatically size and create mapping tables and indexes for use by the REORG utility. The LOAD utility was extended to include the ability to PRESORT your data before beginning the LOAD phase. And, the CHECK DATA utility can discard data rows to a flat file for further processing.

In this chapter, we describe how to govern your utilities by using the DB2 Utilities Enhancement Tool:

- Use the Utility Monitor to enforce the use or disuse of utility parameters to ensure that your utilities are running optimally
- Use the Utility Monitor to allow or disallow users from executing specific utilities against business-critical objects
- Change the severity of a DB2 message to match your shop's requirements
- Use the Utility Monitor to audit who is running what utility and against what object
- Use the Utility Monitor for trend analysis to see how many times an object is being image copied, or reorganized

- Cancel threads automatically to ensure the maintenance within your batch window completes without error or delays

This chapter contains the following sections:

- How the Utility Monitor works
- Enforce the use or disuse of utility syntax
- Allow or disallow users from executing utilities
- Change the severity of a DB2 message
- Suppress repetitive messages in utility SYSPRINT
- Use the Utility Monitor for auditing purposes or trend analysis
- Accelerate LOADs with PRESORT
- Avoid contention or delays during your maintenance window

Using DB2 Utilities Enhancement Tool to monitor all invocations of DSNUTILB not only ensures that utilities will be executed with the allotted maintenance window, but that the DBA staff will not be paged for non-issues. Utilities will be sure to run with the specified parameters, and threads will be canceled for those utilities that must have priority access to an object.

7.1 How the Utility Monitor works

The DB2 Utilities Enhancement Tool has a policy that governs the behavior of the DSNUTILB Intercept. The policy is a set of rules written in a language called XML that defines how DB2 Utilities Enhancement Tool intercepts a DSNUTILB utility and what actions it is to perform.

The rules that are defined within the policy dictate which utilities are monitored, as well as what syntax to enforce. When a DSNUTILB utility job is run, DB2 Utilities Enhancement Tool evaluates the utility statement along with the Utility Monitor rules within the policy. Each rule within the policy relates some utility statement text with a coded option. When the rule is evaluated as true, the option directs DB2 Utilities Enhancement Tool to perform one or a combination of the following actions:

- ADD

  A text string is added to the utility statement if the specified string is not found in the utility statement syntax. The text string is appended to the end of the utility statement only once. For example, if a REORG utility does not have the parameter KEEPDICTIONARY specified, Utilities Enhancement Tool adds the syntax KEEPDICTIONARY, and passes it to the REORG utility for processing.

  - OPTIONIF

    A text string within the ADD element is added to the utility statement based on the presence of the text string defined in OPTIONIF. That is, the string in the element ADD is only added if the string in OPTIONIF is present. In this way, a child parameter can be safely added to utility syntax only if the parent parameter is defined.

- REMOVE

  A text string is removed from the utility statement if the specified string is found in the statement syntax. For example, if the COPY utility contains the parameter SHRLEVEL CHANGE, the parameter can be removed, causing the default value of SHRLEVEL REFERENCE to be used instead.

- SUBSTITUTE

  A text string is replaced in the utility statement if the specified string is found in the statement syntax. The substituted text is placed in the same location within the utility statement as the target string. For example, if a LOAD utility has the LOG YES parameter specified, DB2 Utilities Enhancement Tool substitutes LOG NO in place of LOG YES and passes it to the LOAD utility for processing.

- FAIL

  This action causes the utility job step to fail, and the return code value supplied in this element is issued along with messages containing information about why the utility failed.

- JOURNAL

  This action indicates to DB2 Utilities Enhancement Tool to write this event to the DB2 Utilities Enhancement Tool table for subsequent review or analysis. The information written to the Utilities Enhancement Tool table includes pertinent information about the utility statement and environment, plus the original syntax specified and substituted syntax that was passed to the IBM utility and processed.

A *utility governance policy* is shipped within DB2 Utilities Enhancement Tool. This policy provides rules that are considered best practices by IBM to use when running utilities. Some rules are utility optimization behaviors, and some rules help ensure object recoverability. We review the policy in the next section.

7.1.1 A look inside the utility governance policy

Beginning with DB2 Utilities Enhancement Tool V2.2, a sample utility governance policy is provided within the hlq.SABPSAMP library. This policy determines what utilities to monitor and what actions to take for each utility, as well as what threads to cancel for your business-critical objects.

The sample policy is described in Example 7-1.

Example 7-1 Sample policy of the Utility Monitor

```
<?XML VERSION="1.0" ENCODING="UTF-8"?>
<!DOCTYPE OPTIONS SYSTEM "DD:DTD(ABPDTDPL)">
<!--                                                                  -->
<!-- ************************************************************ -->
<!-- *                                                            * -->
<!-- * ABPPLCY                                                    * -->
<!-- *                                                            * -->
<!-- * IBM DB2 UTILITIES ENHANCEMENT TOOL FOR Z/OS V2.2 (H2AM220)  * -->
<!-- *                                                            * -->
<!-- * DSNUTILB INTERCEPTION POLICY                               * -->
<!-- *                                                            * -->
<!-- ************************************************************ -->
<!-- *                                                            * -->
<!-- * 5655-T58                                                   * -->
<!-- * (C) COPYRIGHT ROCKET SOFTWARE, INC. 2002, 2012 ALL RIGHTS   * -->
<!-- * RESERVED.                                                  * -->
<!-- *                                                            * -->
<!-- ************************************************************ -->
<!-- *                                                            * -->
<!-- * MODIFICATION HISTORY                                       * -->
<!-- *                                                            * -->
<!-- * MM/DD/YYYY   APAR   DESCRIPTION                            * -->
<!-- * ==========  ======= ======================================= * -->
<!-- * 09/21/2012  PM70553 ADD BEST PRACTICES POLICY              * -->
<!-- *                                                            * -->
<!-- ************************************************************ -->
<!--                                                                  -->
<!-- PURPOSE:                                                         -->
<!--                                                                  -->
<!--    THIS MEMBER CONTAINS THE BEST PRACTICES POLICY TO BE USED    -->
<!--    WITH THE DSNUTILB INTERCEPT WITHIN THE                       -->
<!--    DB2 UTILITIES ENHANCEMENT TOOL FOR Z/OS STARTED TASK.        -->
<!--    THE RULES WITHIN THIS POLICY CONTAIN THE USE OR DISUSE OF    -->
<!--    SUGGESTED PARAMETERS WHEN RUNNING VARIOUS DB2 UTILITIES.     -->
<!--                                                                  -->
<!-- ************************************************************ -->
<DSNUTILB_INTERCEPT>
<!-- DEFINE THE UTILITIES FOR DB2 9 ON WHICH TO CHANGE SYNTAX        -->
   <PRACTICE NAME="UTILITY_RULES_DB2V9">
       <UTILITY NAME="RUNSTATS">
          <MONITOR>
             <SYNTAX ADD="SAMPLE 25"/>
          </MONITOR>
       </UTILITY>
       <UTILITY NAME="LOAD">
```

```
         <MONITOR>
            <SYNTAX ADD="PRESORTED YES" OPTIONIF="PRESORT"/>
            <SYNTAX VALUE="LOG %" SUBSTITUTE="LOG NO"/>
            <SYNTAX ADD="KEEPDICTIONARY"/>
            <SYNTAX ADD="REUSE"/>
         </MONITOR>
      </UTILITY>
      <UTILITY NAME="REORG_TABLESPACE">
         <MONITOR>
             <SYNTAX ADD="SAMPLE 25" OPTIONIF="STATISTICS"/>
             <SYNTAX ADD="KEEPDICTIONARY"/>
             <MESSAGE ID="DSNU126I" RETURN_CODE="8"/>
         </MONITOR>
      </UTILITY>
   </PRACTICE>
<!-- DEFINE THE UTILITIES FOR DB2 10 ON WHICH TO CHANGE SYNTAX       -->
   <PRACTICE NAME="UTILITY_RULES_DB2V10">
      <UTILITY NAME="RUNSTATS">
         <MONITOR>
            <SYNTAX REMOVE="SAMPLE %"/>
            <SYNTAX ADD="TABLESAMPLE SYSTEM AUTO"/>
         </MONITOR>
      </UTILITY>
      <UTILITY NAME="LOAD">
         <MONITOR>
            <SYNTAX ADD="PRESORTED YES" OPTIONIF="PRESORT"/>
            <SYNTAX REMOVE="FORMAT SPANNED YES"/>
            <SYNTAX VALUE="LOG %" SUBSTITUTE="LOG NO"/>
            <SYNTAX ADD="KEEPDICTIONARY"/>
            <SYNTAX ADD="REUSE"/>
              <MESSAGE ID="DSNU115OI" RETURN_CODE="4"/>
         </MONITOR>
      </UTILITY>
      <UTILITY NAME="REORG_TABLESPACE">
         <MONITOR>
             <SYNTAX ADD="KEEPDICTIONARY"/>
             <SYNTAX REMOVE="FORCE %"/>
             <SYNTAX ADD="FORCE READERS"/>
             <MESSAGE ID="DSNU126I" RETURN_CODE="8"/>
         </MONITOR>
      </UTILITY>
   </PRACTICE>
<!-- ************************************************************ -->
<!-- BLOCK/CANCEL THREADS AND MONITOR UTILITIES FOR SSID XXXX        -->
<!-- ************************************************************ -->
 <POLICY>
  <DB2SYSTEM SSID="XXXX" ACTION="MONITOR_UTILITY">
   <USE_PRACTICE NAME="UTILITY_RULES_DB2V9"/>
     <INCLUDE>
       <RULE UTILITY_JOBNAME="MYJOB%"/>
     </INCLUDE>
  </DB2SYSTEM>
<!-- ************************************************************ -->
<!-- BLOCK/CANCEL THREADS AND MONITOR UTILITIES FOR SSID YYYY        -->
<!-- ************************************************************ -->
```

```
  <DB2SYSTEM SSID="YYYY" ACTION="MONITOR_UTILITY">
   <USE_PRACTICE NAME="UTILITY_RULES_DB2V10"/>
     <INCLUDE>
       <RULE UTILITY_JOBNAME="MYJOB%"/>
     </INCLUDE>
  </DB2SYSTEM>
 </POLICY>
</DSNUTILB_INTERCEPT>
```

The utility governance policy is broken down for further explanation before we begin the scenarios. Because each version of DB2 has unique functionality that might be available in one version of DB2 but not in another, DB2 9 and DB2 10 are divided up into their own sections. They can be identified by the comment line that indicates the version of DB2 that the section supports.

The best practice rules for DB2 9 are displayed in Example 7-2.

Example 7-2 Utility governance policy rules for DB2 9

```
<!-- DEFINE THE UTILITIES FOR DB2 9 ON WHICH TO CHANGE SYNTAX        -->
   <PRACTICE NAME="UTILITY_RULES_DB2V9">
       <UTILITY NAME="RUNSTATS">
          <MONITOR>
             <SYNTAX ADD="SAMPLE 25"/>
          </MONITOR>
       </UTILITY>
       <UTILITY NAME="LOAD">
          <MONITOR>
             <SYNTAX ADD="PRESORTED YES" OPTIONIF="PRESORT"/>
             <SYNTAX VALUE="LOG %" SUBSTITUTE="LOG NO"/>
             <SYNTAX ADD="KEEPDICTIONARY"/>
             <SYNTAX ADD="REUSE"/>
              <MESSAGE ID="DSNU353I"  SUPPRESS="YES"/>
              <MESSAGE ID="DSNU1150I" RETURN_CODE="4"/>
          </MONITOR>
       </UTILITY>
       <UTILITY NAME="REORG_TABLESPACE">
          <MONITOR>
              <SYNTAX ADD="SAMPLE 50" OPTIONIF="STATISTICS"/>
              <SYNTAX ADD="KEEPDICTIONARY"/>
          </MONITOR>
       </UTILITY>
   </PRACTICE>
```

The policy rules for use on a DB2 9 subsystem are identified by the practice name "UTILITY_RULES_DB2V9". Within that section, the RUNSTATS, LOAD, and REORG TABLESPACE utilities are being monitored for specific utility parameters.

For the RUNSTATS utility, add SAMPLE 25 if sampling is not being used. This helps reduce the overall elapsed time when capturing fairly accurate statistics for your tables.

For the LOAD utility, the PRESORTED YES parameter is added if you had manually added the DB2 Utilities Enhancement Tool parameter PRESORT to the LOAD utility syntax. The DB2 Utilities Enhancement Tool parameter PRESORT sorts the data prior to the LOAD utility starting, reducing the overall elapsed time and CPU consumption of the LOAD utility.

In addition, by adding PRESORTED YES, the sort phase is not started for the clustering indexes. So, for any object that you are loading that only has clustering indexes, additional elapsed time and CPU consumption can be eliminated by not starting the sort phase.

In addition, LOG NO is added to the utility if it is not there, or LOG YES replaces LOG NO, if present. The KEEPDICTIONARY parameter is added if it is not there to avoid rebuilding it. It is ignored by the LOAD utility if the object is not compressed. The parameter REUSE is added, if it is not present, to ensure that the underlying data sets are reused.

And finally, if message DSNU1150I is issued in the SYSPRINT (which denotes the number of data records that were not loaded), the severity is changed from a 0 to a 4 to signal to the DBA that the SYSPRINT should be reviewed. Likewise, if message DSNU353I is repeated in the SYSPRINT more than once, they are suppressed, and logged in the Utility Monitor tables. This eliminates the DBA from sifting through potentially large amounts of SYSPRINT information, when message DSNU1150I indicates how many data records were not loaded.

For the REORG TABLESPACE utility, SAMPLE 25 is added if the parameter STATISTICS is present within the utility statement. The KEEPDICTIONARY parameter is added if it is not there. It is ignored by the REORG utility if the table space is not compressed.

The best practice rules for DB2 10 are displayed in Example 7-3.

Example 7-3 Utility governance policy rules for DB2 10

```
<!-- DEFINE THE UTILITIES FOR DB2 10 ON WHICH TO CHANGE SYNTAX       -->
   <PRACTICE NAME="UTILITY_RULES_DB2V10">
       <UTILITY NAME="RUNSTATS">
          <MONITOR>
             <SYNTAX REMOVE="SAMPLE %"/>
             <SYNTAX ADD="TABLESAMPLE SYSTEM AUTO"/>
          </MONITOR>
       </UTILITY>
       <UTILITY NAME="LOAD">
          <MONITOR>
             <SYNTAX ADD="PRESORTED YES" OPTIONIF="PRESORT"/>
             <SYNTAX REMOVE="FORMAT SPANNED YES"/>
             <SYNTAX VALUE="LOG %" SUBSTITUTE="LOG NO"/>
             <SYNTAX ADD="KEEPDICTIONARY"/>
             <SYNTAX ADD="REUSE"/>
               <MESSAGE ID="DSNU353I"  SUPPRESS="YES"/>
               <MESSAGE ID="DSNU1150I" RETURN_CODE="8"/>
          </MONITOR>
       </UTILITY>
       <UTILITY NAME="REORG_TABLESPACE">
          <MONITOR>
              <SYNTAX ADD="KEEPDICTIONARY"/>
              <SYNTAX REMOVE="FORCE %"/>
              <SYNTAX ADD="FORCE READERS" OPTIONIF="SHRLEVEL CHANGE"/>
              <MESSAGE ID="DSNU126I" RETURN_CODE="8"/>
          </MONITOR>
       </UTILITY>
   </PRACTICE>
```

The policy rules for use on a DB2 10 subsystem are identified by the practice name "UTILITY_RULES_DB2V10". Within that section, the RUNSTATS, LOAD, and REORG TABLESPACE utilities are being monitored for specific utility parameters that are similar to those being monitored for DB2 9. A few additions and changes are described next.

For the RUNSTATS utility, SAMPLE 50 is being removed if it is present, and TABLESAMPLE SYSTEM AUTO is used instead. This leverages the newer technology for clients on DB2 10. For any table space that has multiple tables that are not partitioned or are segmented, TABLESAMPLE is ignored and SAMPLE 25 is used by default instead.

For the LOAD utility, the parameters FORMAT SPANNED YES are removed to enable parallel processing.

For the REORG TABLESPACE utility, the FORCE parameter is being removed and replaced by FORCE READERS if SHRLEVEL CHANGE is defined to enable the drain phase to complete prior to the switch phase. In this way, the REORG utility is sure to run as a true online utility and force off only those threads that prevent the utility from completing successfully. If the DSNU126I message is present in the SYSPRINT (which signals that a REORG on a LOB table space with SHRLELVEL NONE is no longer supported), the severity is changed from a 0 to an 8, indicating to the DBA that the REORG did not take place for one or more objects.

Now that the utility rules are defined, they are enforced on a subsystem-by-subsystem basis in the section that begins with the tag 'POLICY', as shown in Example 7-4.

Example 7-4 Putting the Utility Monitor to use on a subsystem-by-subsystem basis

```
<!-- ************************************************************ -->
<!-- BLOCK/CANCEL THREADS AND MONITOR UTILITIES FOR SSID XXXX      -->
<!-- ************************************************************ -->
 <POLICY>
  <DB2SYSTEM SSID="XXXX" ACTION="MONITOR_UTILITY">
   <USE_PRACTICE NAME="UTILITY_RULES_DB2V9"/>
     <INCLUDE>
       <RULE UTILITY_JOBNAME="MYJOB%"/>
     </INCLUDE>
  </DB2SYSTEM>
<!-- ************************************************************ -->
<!-- BLOCK/CANCEL THREADS AND MONITOR UTILITIES FOR SSID YYYY      -->
<!-- ************************************************************ -->
  <DB2SYSTEM SSID="YYYY" ACTION="MONITOR_UTILITY">
   <USE_PRACTICE NAME="UTILITY_RULES_DB2V10"/>
     <INCLUDE>
       <RULE UTILITY_JOBNAME="MYJOB%"/>
     </INCLUDE>
  </DB2SYSTEM>
 </POLICY>
</DSNUTILB_INTERCEPT>
```

This portion of the utility governance policy must be customized and changed to reflect the names of the DB2 subsystems on which you want to monitor utility syntax. In this example, the SSID values are 'XXXX' and 'YYYY', neither of which is likely to actually match subsystem names on your system. Therefore, to begin using the utility governance rules within this policy, the SSID values need to be changed to match actual subsystem names that are defined on your subsystem.

The element USE_PRACTICE NAME="UTILITY_RULES_DB2V9" invokes the Utility Monitor defined with that name. To invoke the Utility Monitor, a rule needs to be defined that, when matched, calls the utility actions within practice "UTILITY_RULES_DB2V9". In this example, a utility job name of 'MYJOB%' is defined. In this manner, only those batch utility jobs matching the wildcard value of MYJOB% cause the Utility Monitor to start to take action on the utility syntax.

7.1.2 The policy rules are used to invoke the Utility Monitor

When the DB2 Utilities Enhancement Tool is installed and customized, a utility governance policy is created in the hlq.SABPSAMP library containing default rules. The policy is used by the started task any time that DSNUTILB is invoked. The policy rules dictate whether the DB2 Utilities Enhancement Tool started task is to intercept a DSNUTILB utility and what actions to perform.

By default, the name of the policy is created in the hlq.SABPSAMP library as abpidPLCY, where abpid is the 4-character identifier for the DB2 Utilities Enhancement Tool started task that you define during customization. For example, an ABPID can be named ABP1. In this case, the policy name is ABP1PLCY. If any of the rules within the policy evaluate as true, the action described by the policy rule invokes the Utility Monitor.

The purpose of the Utility Monitor component of DB2 Utilities Enhancement Tool is to help DBAs ensure that DB2 utilities are not being run with syntax that violates their company standards, goes against best practices, or causes a potential data recoverability issue. It can be used as an added layer of security each time that a DB2 utility is executed from program DSNUTILB by ensuring that specific utilities aren't executed against business-critical objects, or by preventing certain users from running utilities. You can optionally use the Utility Monitor to simply log all DSNUTILB invocations and see how often specific utilities are being run on specific objects over time.

Running DB2 utilities in conjunction with a best practices methodology ensures that your utilities are running optimally and that the DBA staff will not be paged during off-hours for utilities that contained parameters in conflict with company policies.

We show the following scenarios using DB2 Utilities Enhancement Tool:

- Enforce the use or disuse of utility syntax
- Allow or disallow users from executing utilities
- Change the severity of a DB2 message
- Suppress repetitive messages in utility SYSPRINT
- Use the Utility Monitor for auditing purposes or trend analysis
- Avoid contention or delays during your maintenance window

7.2 Enforce the use or disuse of utility syntax

Have you ever realize, after the fact, that someone ran a utility with some parameters that they should not have? Maybe, someone ran a LOAD utility with REPLACE PART n instead of PART n REPLACE and wiped out data.

Sometimes, running DB2 utilities with certain parameters can cause unforeseen problems later. What if you could prevent these issues from happening in the first place? By using the Utility Monitor, you can enforce the use or disuse of specific utility parameters. By setting up a utility governance policy, you cannot only change the utility parameters at run time, but you can also track each time that the Utility Monitor changes utility syntax so that you can notify the user of the company standards. Now, you will know who the culprit is.

Example 7-5 displays a sample policy where utility syntax is being changed to not only remove parameters that should not be there, but to add parameters that should.

Example 7-5 Add utility parameters that are not there, and remove ones that are

```
<UTILITY NAME="LOAD">
   <MONITOR JOURNAL="YES">
      <SYNTAX ADD="PRESORTED YES" OPTIONIF="PRESORT"/>
      <SYNTAX VALUE="LOG %" SUBSTITUTE="LOG NO"/>
      <SYNTAX ADD="KEEPDICTIONARY"/>
      <SYNTAX ADD="REUSE"/>
        <MESSAGE ID="DSNU353I"  SUPPRESS="YES"/>
        <MESSAGE ID="DSNU1150I" RETURN_CODE="8"/>
   </MONITOR>
</UTILITY>
```

Within the JCL example in Example 7-6, the utility governance policy from Example x will be used to illustrate how you can add parameters to a utility that are not present but should be, and to remove parameters that should not be present.

In this utility JCL in Example 7-6, the policy is set up to affect the LOAD utility in the following manner:

- The DB2 Utilities Enhancement Tool-specific parameter called PRESORT is present in the syntax. This will cause the data within the SYSREC file to be presorted prior to the LOAD utility starting. Because it is present in the LOAD utility, the parameter PRESORTED YES will be added to indicate to the utility that the sort phase does not need to invoke to sort the data for the clustering index, since the data has already been sorted.
- The value LOG NO will be substituted in the syntax for whatever value was specified for LOG to ensure that the utility is running with the optimal parameters.
- The parameter KEEPDICTIONARY will be added to the syntax to retain the existing compression dictionary and avoid the cost of rebuilding it.
- The parameter REUSE will be added to the syntax, but will be ignored in this specific example, since RESUME was defined instead of REPLACE.

Furthermore, the policy defines that message DSNU353I (`RECORD 'n' WILL BE DISCARDED DUE TO CHECK CONSTRAINT constraint-name VIOLATION ON TABLE table-name`) will be suppressed in the SYSPRINT, and message DSNU1150I (`(RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS NOT LOADED=nnnn`) will cause the job to complete with a return code 8.

Example 7-6 Sample LOAD utility JCL that will be modified by the policy

```
//LOAD     EXEC PGM=DSNUTILB,PARM='DB1S,UETLAB6B'
//*
//STEPLIB  DD DISP=SHR,DSN=DB2.V9R1.SDSNLOAD
//*
//SYSPRINT DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
```

```
//*
//SYSREC   DD DSN=DNET000.DB1S.ABPLAB6.ABPTS6.SYSREC,
//            DISP=SHR
//*
//SYSMAP   DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            SPACE=(CYL,(1,1))
//SYSUT1   DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            SPACE=(CYL,(1,1))
//SORTOUT  DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            SPACE=(CYL,(1,1))
//SYSDISC  DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            SPACE=(CYL,(1,1))
//SYSERR   DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            SPACE=(CYL,(1,1))
//*
//SYSIN    DD *
LOAD DATA INDDN SYSREC PRESORT LOG YES  RESUME YES
 EBCDIC  CCSID(00037,00000,00000)
 INTO TABLE "ABPL6"."ABPTB6"
 ( "ID_SHOP"
  POSITION(  00003:00004) SMALLINT
 , "ID_GOOD"
  POSITION(  00005:00006) SMALLINT
 , "QUANTITY"
  POSITION(  00007:00010) INTEGER
 , "DATESALE"
  POSITION(  00011:00020) DATE EXTERNAL
 )
//* ------------------------------------------------
```

When the utility JCL is submitted, the resultant output is defined in Example 7-7. Within the SYSPRINT, message ABPU5330I displays the original LOAD syntax that was submitted to program DSNUTILB. The parameter PRESORT is still present. In addition, PRESORTED YES, KEEPDICTIONARY, and REUSE have already been added to the end of the syntax by the Utility Monitor.

The actual utility control syntax that was executed does not contain the DB2 Utilities Enhancement Tool-specific parameter PRESORT. It was removed before the LOAD utility was executed.

Example 7-7 LOAD utility SYSPRINT showing the changed syntax and suppressed message

```
ABPU5330I 341 12:28:31.85 Original DSNUTILB syntax follows:
ABPU5331I 341 12:28:31.85 LOAD DATA INDDN SYSREC PRESORT LOG NO RESUME YES EBCDIC CCSID(00037,00000,00000)
ABPU5331I 341 12:28:31.85  INTO TABLE "ABPL6"."ABPTB6" ( "ID_SHOP" POSITION( 00003:00004) SMALLINT ,
ABPU5331I 341 12:28:31.85 "ID_GOOD" POSITION( 00005:00006) SMALLINT , "QUANTITY" POSITION( 00007:00010)
ABPU5331I 341 12:28:31.85 INTEGER , "DATESALE" POSITION( 00011:00020) DATE EXTERNAL ) PRESORTED YES
ABPU5331I 341 12:28:31.85 KEEPDICTIONARY REUSE
ABPU5332I 341 12:28:31.85 End of original DSNUTILB syntax listing.
DSNU000I    341 12:28:31.89 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = UETLAB6B
DSNU1044I   341 12:28:31.91 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
DSNU050I    341 12:28:31.92 DSNUGUTC -  LOAD DATA LOG NO RESUME YES EBCDIC CCSID(37, 0, 0)
DSNU650I  DB1S 341 12:28:31.92 DSNURWI -   INTO TABLE "ABPL6"."ABPTB6"
DSNU650I  DB1S 341 12:28:31.92 DSNURWI -    ("ID_SHOP" POSITION(3:4) SMALLINT,
DSNU650I  DB1S 341 12:28:31.92 DSNURWI -     "ID_GOOD" POSITION(5:6) SMALLINT,
DSNU650I  DB1S 341 12:28:31.92 DSNURWI -     "QUANTITY" POSITION(7:10) INTEGER,
DSNU650I  DB1S 341 12:28:31.92 DSNURWI -     "DATESALE" POSITION(11:20) DATE EXTERNAL) PRESORTED YES KEEPDICTIONARY
 REUSE INDDN ABPREC SORTKEYS 0
DSNU232I  DB1S 341 12:28:31.92 DSNURWI - KEEPDICTIONARY REQUESTED BUT COMPRESS ATTRIBUTE NOT DEFINED FOR TABLE SPACE
```

```
ABPLAB6.ABPTS6
DSNU359I  DB1S 341 12:28:31.98 DSNURWI - KEYWORD 'SORTKEYS' SPECIFIED BUT NO INDEX OR FOREIGN KEYS EXIST, KEYWORD IS
IGNORED.
DSNU188I  DB1S 341 12:28:31.98 DSNURWI - OPTION PRESORTED YES IS NOT VALID WHEN USED WITH LOAD ON A TABLE WITH NO
INDEXES.  OPTION IS IGNORED
DSNU304I  DB1S 341 12:28:32.10 DSNURWT - (RE)LOAD PHASE STATISTICS - NUMBER OF RECORDS=5 FOR TABLE ABPL6.ABPTB6
DSNU1147I DB1S 341 12:28:32.10 DSNURWT - (RE)LOAD PHASE STATISTICS - TOTAL NUMBER OF RECORDS LOADED=5 FOR TABLESPACE
ABPLAB6.ABPTS6
DSNU1150I DB1S 341 12:28:32.10 DSNURWT - (RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS NOT LOADED=1
DSNU302I    341 12:28:32.11 DSNURILD - (RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS PROCESSED=6
DSNU300I    341 12:28:32.11 DSNURILD - (RE)LOAD PHASE COMPLETE, ELAPSED TIME=00:00:00
DSNU375I    341 12:28:32.21 DSNURDNP - DISCARD PHASE STATISTICS - 1 INPUT DATA SET RECORDS DISCARDED
DSNU376I    341 12:28:32.21 DSNURDIS - DISCARD PHASE COMPLETE, ELAPSED TIME=00:00:00
DSNU381I  DB1S 341 12:28:32.22 DSNUGSRX - TABLESPACE ABPLAB6.ABPTS6 IS IN COPY PENDING
DSNU563I  DB1S 341 12:28:32.22 DSNUGSRX - TABLESPACE ABPLAB6.ABPTS6 IS IN CHECK PENDING
DSNU010I    341 12:28:32.23 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=4
ABPU5405I 341 12:28:33.86 Utility return code altered by policy practice UTILITY_RULES_DB2V9
ABPU5403I 341 12:28:33.86 Utility statement altered by policy practice UTILITY_RULES_DB2V9
ABPU5009I 341 12:28:33.86 Utility execution completed. SYS=0000, USR=0004
ABPU5010I 341 12:28:33.86 Allow threads started. Step=1
ABPU5011I 341 12:28:34.02 Allow threads completed. RC=00000000
ABPU5003I 341 12:28:34.04 DB2 Utilities Enhancement Tool intercept completed.
******************************** BOTTOM OF DATA ****************************************************
```

Notice within the LOAD utility SYSPRINT that message DSNU1150I defines the number of records that were not loaded. This is your indication that message DSNU353I must have been present, but was removed based on the utility governance policy rules.

The policy also specified to end with a return code 8 if message DSNU1150I is issued in the job output, as our example showed. As a result, the JES job log is displayed in Example 7-8 to illustrate that the LOAD ended with a return code 8.

Example 7-8 LOAD utility job log showing the changed job return code

```
J E S 2  J O B  L O G  --  S Y S T E M  M V S A  --  N O D E

12.28.22 JOB05733 ---- THURSDAY,  06 DEC 2012 ----
12.28.22 JOB05733  IRR010I  USERID DNET000  IS ASSIGNED TO THIS JOB.
12.28.22 JOB05733  ICH70001I DNET000  LAST ACCESS AT 12:24:58 ON THURSDAY, DECEM
12.28.22 JOB05733  $HASP373 UETLAB6  STARTED - INIT 23   - CLASS A - SYS MVSA
12.28.22 JOB05733  IEF403I UETLAB6 - STARTED - TIME=12.28.22
12.28.23 JOB05733  -                                          --TIMINGS (MINS.)--
12.28.23 JOB05733  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK
12.28.23 JOB05733  -UETLAB6           CREATE      00    209    .00    .00    .00
12.28.23 JOB05733  -UETLAB6           INSERT      00    208    .00    .00    .00
12.28.23 JOB05733  -UETLAB6           DELETE      00     53    .00    .00    .00
12.28.27 JOB05733  -UETLAB6           UNLOAD      00  10605    .00    .00    .07
12.28.28 JOB05733  -UETLAB6           ALTER       04    210    .00    .00    .00
12.28.33 JOB05733  -UETLAB6           LOAD        08  16545    .00    .00    .11
12.28.33 JOB05733  -UETLAB6           DROP        00    205    .00    .00    .00
12.28.33 JOB05733  IEF404I UETLAB6 - ENDED - TIME=12.28.35
```

Querying the DB2 Utilities Enhancement Tool tables will display the logged actions taken by the utility governance policy. In the query results shown in Example 7-9 on page 291, the suppression of message DSNU353I is logged, as is the job return code of 8 for message DSNU1150I.

Example 7-9 By default, all of the policy actions are logged in the DB2 Utilities Enhancement Tool tables

```
PAGE    7
                        +-------------------------------------------------------------------------------
                        | JOBNAME  |     ID     | SUPPRESS |  RETURN_CODE   |          INSERTED          |
                        +-------------------------------------------------------------------------------
                      1_| UETLAB6  | DSNU353I   | Y        |             -1 | 2012-12-06-12.28.33.848390 |
                      2_| UETLAB6  | DSNU353I   | Y        |             -1 | 2012-12-06-12.28.33.853934 |
                      3_| UETLAB6  | DSNU1150I  | N        |              8 | 2012-12-06-12.28.33.858240 |
                        +-------------------------------------------------------------------------------
PAGE    8
     ------------------------------------------------------------------------------------------------------------------------
     |                                                        TEXT
     ------------------------------------------------------------------------------------------------------------------------
   1_|  DSNU353I  DB1S 341 12:28:32.06 DSNURWBF - RECORD (1) WILL BE DISCARDED DUE TO CHECK CONSTRAINT ID_GOOD VIOL
   2_|  TABLE ABPL6.ABPTB6
   3_|  DSNU1150I DB1S 341 12:28:32.10 DSNURWT - (RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS NOT LOADED=1
     ------------------------------------------------------------------------------------------------------------------------
SUCCESSFUL RETRIEVAL OF        3 ROW(S)
PAGE    1
```

In this example, executing batch job UETLAB6 resulted in two messages being written to the DB2 Utilities Enhancement Tool tables for historical reference: DSNU353I and DSNU1150I. Within example x, you will see that message DSNU353I occupies two rows in the table, because the original message issued in the JES job log occupied two lines. DB2 Utilities Enhancement Tool logged the message exactly as it appeared in the SYSPRINT.

In the column SUPPRESS, message DSNU353I is logged with a Y indicating that the policy contained a rule to suppress that message. Message DSNU1150I contains an N, which means that there was no policy rule to suppress this message. The column RETURN_CODE contains a negative 1 for message DSNU353I, meaning there was not a rule defined in the policy to change the return code for that message. However, there is an 8 for message DSNU1150I, so this means that the policy contained a rule for that message to end the job with the specified return code.

Within the column TEXT, the associated message text for each of the DB2 messages is written for comprehensiveness. This way, the DBA can see exactly what messages were written to the history tables.

By using the Utility Monitor, you can customize the execution of each DB2 utility that runs in your environment. Enforce the use or disuse of specific utility parameters, change the job return code based on messages in the SYSPRINT, or suppress repeated messages from consuming too much spool, all from one utility governance policy. But have you ever wanted to prevent users from running a specific utility? Have you wanted to prevent data loss or downtime from happening because someone simply ran the wrong utility on the wrong object? Read on to see how the Utility Monitor can help you ensure that your data remains available.

7.3 Allow or disallow users from executing utilities

Has your business' data availability ever had a problem because someone either ran a utility at an inappropriate time, or they ran a utility that they should not have run? In today's world of instant data access, many companies must remain accessible on an almost 24-hour basis. With so much business conducted online, users have come to expect to do business at all hours of the day. To ensure that your business data remains accessible, planned maintenance windows are the only time during which disruptive utilities can be run.

To ensure that access to your data is not interrupted, the Utility Monitor can fail those utilities that cause an object to become inaccessible for some length of time, or might cause an issue with data recoverability. By setting up a utility governance policy, you cannot only prevent an invasive utility from being run to avoid application downtime, but the Utility Monitor can log when it fails a utility so that you can notify the user why their utility was prevented from running.

Example 7-10 displays a sample policy where a utility is failed that would cause a problem with data recoverability.

Example 7-10 Prevent users from running specific utilities

```
EDIT       DB2DUET.V2R2.PARMLIB(TDE1PLCY) - 01.09               Columns 00001 00080
Command ===>                                                      Scroll ===> CSR
000116        </UTILITY>
000117        <UTILITY NAME="MODIFY_RECOVERY">
000118           <MONITOR FAIL="8"/>
000119        </UTILITY>
000120        <UTILITY NAME="REPAIR">
000121           <MONITOR FAIL="8"/>
000122        </UTILITY>
```

In this example, if the MODIFY utility is being executed against an object that could cause an issue with recoverability, the DB2 Utilities Enhancement Tool policy can be defined to fail the utility with a return code 8 if it is executed. Likewise, the policy can be set up to only fail when the utility is being executed on specific objects, or only when the utility is being executed by a specific user or set of users.

When a MODIFY RECOVERY utility is executed using the sample policy in Example 7-10, the informational messages shown in Example 7-11 will appear within the utility SYSPRINT.

Example 7-11 MODIFY RECOVERY utility SYSPRINT info messages when preventing utility execution

```
SDSF OUTPUT DISPLAY UETLAB8  JOB04283  DSID   112 LINE 0       COLUMNS 02- 133
 COMMAND INPUT ===>                                          SCROLL ===> CSR
******************************** TOP OF DATA ***************************************
ABPU5001I 304 15:01:26.12 IBM DB2 Utilities Enhancement Tool Version 0220, FMID=H2AM220, CO
ABPU5012I 304 15:01:26.12 Connected to started task ABPID=TDE1
ABPU5002I 304 15:01:26.14 Initialization is complete.

-------------------------------------------------------------------------------------

ABPU5004I 304 15:01:28.12 Analysis started. Step=1
ABPU5400E 304 15:01:28.12 Utility processing failed by policy practice UTILITY_RULES_DB2V10
ABPU5005I 304 15:01:28.13 Analysis completed. RC=8
ABPU5003I 304 15:01:28.13 DB2 Utilities Enhancement Tool intercept completed.
******************************** BOTTOM OF DATA ************************************
```

In Example 7-11, message ABPU5400E lets the user know that their job was prevented from running by a policy rule. The user can then check with their IT staff as to why their job failed, enabling the IT staff to communicate their company standards and policies to the user.

A *policy rule* can be defined to conditionally fail a utility, so that not every invocation of a specific utility is prevented from running, as in the MODIFY RECOVERY example. Additional criteria can be added to the policy to further qualify when a utility should be prevented from running. Criteria, such as object name or user IDs, can be defined to the policy as an added layer of security, as shown in Example 7-12 on page 293.

Example 7-12 Additional criteria can be used to determine when to stop a utility

```
EDIT       DB2DUET.V2R2.PARMLIB(TDE1PLCY) - 01.15                Columns 00001 00080
Command ===>                                                        Scroll ===> CSR
000039    <RULESET NAME="USERIDS">
000040     <EXCLUDE>
000041       <RULE UTILITY_USERID="%"/>
000042     </EXCLUDE>
000043     <INCLUDE>
000044       <RULE UTILITY_USERID="DNET000"/>
000045     </INCLUDE>
000046    </RULESET>
```

By default, all user IDs are excluded from evaluation, and only those that match the included values will be evaluated. In Example 7-12, a user ID is being included for evaluation to prevent a utility from running. All user IDs are excluded, as indicated by the <EXCLUDE> tag on lines 40 through 42. And, an additional rule defines the user ID to include for evaluation, as indicated by the <INCLUDE> tag on lines 43 through 45. If a MODIFY RECOVERY job is run by a user whose ID matches the value specified in the <INCLUDE> section of the policy, the utility will be prevented from running.

Utilities can also be conditionally failed based on the presence of parameters that should not be in the utility syntax, or by the absence of parameters that should be in the utility syntax but are missing. In Example 7-13, a LOAD utility will be prevented from running if the parameter REPLACE is found within the utility syntax, as defined on line 106. A return code of 8 will be issued and a message in the SYSPRINT will explain which parameter caused the utility from running.

Example 7-13 Disallow a utility to execute based on the presence of a parameter

```
EDIT       DB2DUET.V2R2.PARMLIB(TDE1PLCY) - 01.09                Columns 00001 00080
Command ===>                                                        Scroll ===> CSR
000099        <UTILITY NAME="LOAD">
000100           <MONITOR>
000101              <SYNTAX ADD="PRESORTED YES" OPTIONIF="PRESORT"/>
000102              <SYNTAX REMOVE="FORMAT SPANNED YES"/>
000103              <SYNTAX VALUE="LOG %" SUBSTITUTE="LOG NO"/>
000104              <SYNTAX ADD="KEEPDICTIONARY"/>
000105              <SYNTAX ADD="REUSE"/>
000106              <SYNTAX VALUE="REPLACE" FAIL="8"/>
000107                <MESSAGE ID="DSNU1150I" RETURN_CODE="4"/>
000108           </MONITOR>
```

When the LOAD utility is submitted, the following messages will be issued within the utility SYSPRINT. Message ABPU5400E will explain that the job was prevented from running. Message ABPU5401E will explain which parameters should not be present within the syntax. In Example 7-14, the syntax 'REPLACE' caused the utility to stop processing. This indicates to the user that there was a problem with the utility. The user can then check with their IT staff as to why their job failed, enabling the IT staff to communicate their company standards and policies to the user. The user can then resubmit their utility once the syntax has been corrected.

Example 7-14 LOAD utility SYSPRINT messages when preventing utility execution

```
SDSF OUTPUT DISPLAY UETLAB9  JOB04316  DSID   120 LINE 0       COLUMNS 02- 133
 COMMAND INPUT ===>                                          SCROLL ===> CSR
******************************** TOP OF DATA ***************************************
ABPU5001I 304 16:04:17.92 IBM DB2 Utilities Enhancement Tool Version 0220, FMID=H2AM220, CO
ABPU5012I 304 16:04:17.92 Connected to started task ABPID=TDE1
```

```
ABPU5002I 304 16:04:17.93 Initialization is complete.

--------------------------------------------------------------------------------

ABPU5004I 304 16:04:20.36 Analysis started. Step=1
ABPU5400E 304 16:04:20.36 Utility processing failed by policy practice UTILITY_RULES_DB2V10
ABPU5401E 304 16:04:20.36 Syntax denied: REPLACE
ABPU5005I 304 16:04:20.37 Analysis completed. RC=8
ABPU5003I 304 16:04:20.37 DB2 Utilities Enhancement Tool intercept completed.
******************************** BOTTOM OF DATA ****************************************
```

The ability to stop a utility from running ensures that DBAs are not scrambling to correct a data outage, or recover data that may be lost due to a utility executing against business-critical objects, or a utility executing with an inappropriate combination of utility parameters. This added layer of security will ensure that everyone is aware of what utility activity is permissible. In addition, DB2 Utilities Enhancement Tool will track every action made by the policy, providing a historical log of events for DBAs to easily see what activity is happening on the system. For more information about auditing utility activity, see 7.6, “Use the Utility Monitor for auditing purposes or trend analysis” on page 298.

7.4 Change the severity of a DB2 message

Have you ever been paged at an inconvenient hour because a return code 4 was issued within a utility batch job, but the actual warning was a non-issue? Or, have you come into the office one morning thinking that the utilities maintenance window must have gone smoothly because you were not been paged the night before, only to discover that a LOAD utility had some data rows that were not loaded? By using the Utility Monitor, you can take control of your utility execution from start to finish. By defining which DB2 messages should be elevated or reduced in severity, you now will not be erroneously paged for a non-issue, and you are certain to be notified of an actual issue.

In Example 7-15, the policy is defined to end with a return code 8 after the LOAD utility completes if message DSNU1150I is issued in the output. This message documents how many rows were not loaded into a table.

Example 7-15 Policy rule to change the return code if a message is present in the SYSPRINT

```
VIEW       DB2DUET.V2R2.PARMLIB(TDE1PLCY) - 01.27                Columns 00001 00080
Command ===>                                                        Scroll ===> CSR
000081        <UTILITY NAME="LOAD">
000082           <MONITOR>
000083              <SYNTAX ADD="PRESORTED YES" OPTIONIF="PRESORT"/>
000084              <SYNTAX VALUE="LOG %" SUBSTITUTE="LOG NO"/>
000085              <SYNTAX ADD="KEEPDICTIONARY"/>
000086              <SYNTAX ADD="REUSE"/>
000087                <MESSAGE ID="DSNU1150I" RETURN_CODE="8"/>
000088           </MONITOR>
```

When DB2 Utilities Enhancement Tool intercepts the LOAD utility to change the return code of a message, it does not prevent the utility from completing. The LOAD utility processes through to completion as it normally does, and Utilities Enhancement Tool merely changes the job return code when the LOAD utility is complete. Example 7-16 on page 295 shows the Utilities Enhancement Tool messages in the SYSPRINT that indicate that the job return code was changed.

Example 7-16 LOAD utility SYSPRINT messages with the return code altered

```
ABPU5001I 305 11:15:57.09 IBM DB2 Utilities Enhancement Tool Version 0220, FMID=H2AM220, COMP_ID=5655-T58
 ABPU5012I 305 11:15:57.09 Connected to started task ABPID=TDE1
 ABPU5002I 305 11:15:57.11 Initialization is complete.
 ------------------------------------------------------------------------------------------------------
 ABPU5004I 305 11:15:59.35 Analysis started. Step=1
 ABPU5005I 305 11:15:59.46 Analysis completed. RC=0
 ABPU5006I 305 11:15:59.46 Thread cancel started. Step=1
 ABPU5007I 305 11:16:00.24 Thread cancel completed. RC=00000004
 ABPU5008I 305 11:16:00.36 Utility execution started. Step=1
 ABPU5330I 305 11:16:00.36 Original DSNUTILB syntax follows:
 ABPU5331I 305 11:16:00.36 LOAD DATA INDDN SYSREC PRESORT LOG NO RESUME YES EBCDIC CCSID(00037,00000,00000)
 ABPU5331I 305 11:16:00.36  INTO TABLE "ABPL6"."ABPTB6" ( "ID_SHOP" POSITION( 00003:00004) SMALLINT ,
 ABPU5331I 305 11:16:00.36 "ID_GOOD" POSITION( 00005:00006) SMALLINT , "QUANTITY" POSITION( 00007:00010)
 ABPU5331I 305 11:16:00.36 INTEGER , "DATESALE" POSITION( 00011:00020) DATE EXTERNAL ) PRESORTED YES
 ABPU5331I 305 11:16:00.36 KEEPDICTIONARY REUSE
 ABPU5332I 305 11:16:00.36 End of original DSNUTILB syntax listing.
1DSNU000I    305 11:16:00.41 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = UETLAB6B
 DSNU1044I   305 11:16:00.42 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
0DSNU050I    305 11:16:00.43 DSNUGUTC -  LOAD DATA LOG NO RESUME YES EBCDIC CCSID(37, 0, 0)
 DSNU650I  DB1S 305 11:16:00.43 DSNURWI -   INTO TABLE "ABPL6"."ABPTB6"
 DSNU650I  DB1S 305 11:16:00.43 DSNURWI -    ("ID_SHOP" POSITION(3:4) SMALLINT,
 DSNU650I  DB1S 305 11:16:00.43 DSNURWI -     "ID_GOOD" POSITION(5:6) SMALLINT,
 DSNU650I  DB1S 305 11:16:00.43 DSNURWI -     "QUANTITY" POSITION(7:10) INTEGER,
 DSNU650I  DB1S 305 11:16:00.43 DSNURWI -     "DATESALE" POSITION(11:20) DATE EXTERNAL) PRESORTED YES KEEPDICTIONARY
REUSE INDDN ABPREC SORTKEYS 0
DSNU353I  DB1S 305 11:16:00.55 DSNURWBF - RECORD (1) WILL BE DISCARDED DUE TO CHECK CONSTRAINT ID_GOOD VIOLATION ON
TABLE ABPL6.ABPTB6
 DSNU304I  DB1S 305 11:16:00.58 DSNURWT - (RE)LOAD PHASE STATISTICS - NUMBER OF RECORDS=5 FOR TABLE ABPL6.ABPTB6
 DSNU1147I DB1S 305 11:16:00.58 DSNURWT - (RE)LOAD PHASE STATISTICS - TOTAL NUMBER OF RECORDS LOADED=5 FOR TABLESPACE
ABPLAB6.ABPTS6
 DSNU1150I DB1S 305 11:16:00.58 DSNURWT - (RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS NOT LOADED=1
 DSNU302I    305 11:16:00.58 DSNURILD - (RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS PROCESSED=6
 DSNU300I    305 11:16:00.58 DSNURILD - (RE)LOAD PHASE COMPLETE, ELAPSED TIME=00:00:00
 DSNU375I    305 11:16:00.67 DSNURDNP - DISCARD PHASE STATISTICS - 1 INPUT DATA SET RECORDS DISCARDED
 DSNU376I    305 11:16:00.68 DSNURDIS - DISCARD PHASE COMPLETE, ELAPSED TIME=00:00:00
 DSNU381I  DB1S 305 11:16:00.68 DSNUGSRX - TABLESPACE ABPLAB6.ABPTS6 IS IN COPY PENDING
 DSNU563I  DB1S 305 11:16:00.68 DSNUGSRX - TABLESPACE ABPLAB6.ABPTS6 IS IN CHECK PENDING
 DSNU010I    305 11:16:00.69 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=4
 ABPU5405I 305 11:16:01.86 Utility return code altered by policy practice UTILITY_RULES_DB2V9
 ABPU5403I 305 11:16:01.87 Utility statement altered by policy practice UTILITY_RULES_DB2V9
 ABPU5009I 305 11:16:01.87 Utility execution completed. SYS=0000, USR=0004
 ABPU5010I 305 11:16:01.87 Allow threads started. Step=1
 ABPU5011I 305 11:16:02.03 Allow threads completed. RC=00000000
 ABPU5003I 305 11:16:02.04 DB2 Utilities Enhancement Tool intercept completed.
```

Message ABPU5405I lets the user know that the return code was altered. Within this SYSPRINT, message "DSNU010I UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE = 4" indicates that the normal return code is 4. But when the entire job output is viewed, the return code that DB2 Utilities Enhancement Tool changed is visible, as shown in Example 7-17.

Example 7-17 Return code is altered by DB2 Utilities Enhancement Tool

```
 SDSF OUTPUT DISPLAY UETLAB6  JOB04946  DSID     2 LINE 0       COLUMNS 01- 132
 COMMAND INPUT ===>                                          SCROLL ===> CSR
******************************** TOP OF DATA ***************************************************
1                  J E S 2  J O B  L O G  --  S Y S T E M  M V S A  --  N O D E  D E M O M V S
0
 11.15.51 JOB04946 ---- WEDNESDAY, 31 OCT 2012 ----
 11.15.51 JOB04946  IRR010I  USERID DNET000  IS ASSIGNED TO THIS JOB.
 11.15.51 JOB04946  ICH70001I DNET000  LAST ACCESS AT 11:11:22 ON WEDNESDAY, OCTOBER 31, 2012
 11.15.51 JOB04946  $HASP373 UETLAB6  STARTED - INIT 5    - CLASS A - SYS MVSA
 11.15.51 JOB04946  IEF403I UETLAB6 - STARTED - TIME=11.15.51
```

```
 11.15.52 JOB04946  -                                        --TIMINGS (MINS.)--            ---
 11.15.52 JOB04946  -JOBNAME  STEPNAME PROCSTEP    RC    EXCP    CPU    SRB  CLOCK   SERV  PG
 11.15.52 JOB04946  -UETLAB6           CREATE      00     216    .00    .00    .00   2757   0
 11.15.52 JOB04946  -UETLAB6           INSERT      00     215    .00    .00    .00   1174   0
 11.15.52 JOB04946  -UETLAB6           DELETE      00      54    .00    .00    .00    396   0
 11.15.56 JOB04946  -UETLAB6           UNLOAD      00   10604    .00    .00    .07  16698   0
 11.15.56 JOB04946  -UETLAB6           ALTER       04     217    .00    .00    .00   1120   0
 11.16.03 JOB04946  -UETLAB6           LOAD        08   16533    .00    .00    .10  25180   0
 11.16.03 JOB04946  -UETLAB6           DROP        00     212    .00    .00    .00   2060   0
 11.16.03 JOB04946  IEF404I UETLAB6 - ENDED - TIME=11.16.03
11.16.03 JOB04946  $HASP395 UETLAB6  ENDED
```

In Example 7-17 on page 295, the job step LOAD ended with a return code 8, as the policy rule indicated.

The ability to reduce or elevate the severity of a DB2 message means that DBAs can customize the behavior of DB2 to the needs of any shop. DBAs can now be assured that actual issues can be flagged for their attention, while non-issues do not cause a false alarm. In addition, DB2 Utilities Enhancement Tool will track every action made by the policy, providing a historical log of events for DBAs to easily see what activity is happening on the system. For more information about auditing utility activity, see 7.6, “Use the Utility Monitor for auditing purposes or trend analysis” on page 298.

7.5 Suppress repetitive messages in utility SYSPRINT

The Utility Monitor can also suppress large numbers of repetitive messages from being issued in the job output. By simply defining a rule in the policy to suppress repetitive messages, Utilities Enhancement Tool can suppress the consecutive messages that appear more than once while simultaneously logging this action. This enables DBAs to be kept up-to-date on all actions DB2 Utilities Enhancement Tool takes within each DB2 utility batch job.

In Example 7-18, the policy is defined to suppress message DSNU353I within the LOAD utility if it is repeated more than once. The purpose of this message is to document which data rows were not loaded into a table.

Example 7-18 Policy rule suppressing a message if issued more than once in the SYSPRINT

```
VIEW       DB2DUET.V2R2.PARMLIB(TDE1PLCY) - 01.29                Columns 00001 00080
Command ===>                                                        Scroll ===> CSR
000109        <UTILITY NAME="LOAD">
000110           <MONITOR>
000111              <SYNTAX ADD="PRESORTED YES" OPTIONIF="PRESORT"/>
000112              <SYNTAX REMOVE="FORMAT SPANNED YES"/>
000113              <SYNTAX VALUE="LOG %" SUBSTITUTE="LOG NO"/>
000114                <MESSAGE ID="DSNU353I"  SUPPRESS="YES"/>
000115                <MESSAGE ID="DSNU1150I" RETURN_CODE="8"/>
000116           </MONITOR>
000117        </UTILITY>
```

When there are hundreds or thousands of records not loaded, these messages can overwhelm the content’s utility SYSPRINT. The SYSPRINT content in Example 7-19 on page 297 displays several DSNU353I messages that indicate which SYSREC records were not loaded into the table. In addition, message DSNU1150I displays the final tally of records that were not loaded into the table.

Example 7-19 LOAD utility SYSPRINT containing messages DSNU353I and DSNU1150I

```
1DSNU000I    305 12:17:47.37 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = UETLAB6B
 DSNU1044I   305 12:17:47.39 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
0DSNU050I    305 12:17:47.39 DSNUGUTC -  LOAD DATA INDDN SYSREC LOG NO RESUME YES EBCDIC CCSID(37, 0, 0)
 DSNU650I  - 305 12:17:47.39 DSNURWI -  INTO TABLE "ABPL6"."ABPTB6"
 DSNU650I  - 305 12:17:47.40 DSNURWI -   ("ID_SHOP" POSITION(3:4) SMALLINT,
 DSNU650I  - 305 12:17:47.40 DSNURWI -    "ID_GOOD" POSITION(5:6) SMALLINT,
 DSNU650I  - 305 12:17:47.40 DSNURWI -    "QUANTITY" POSITION(7:10) INTEGER,
 DSNU650I  - 305 12:17:47.40 DSNURWI -    "DATESALE" POSITION(11:20) DATE EXTERNAL)
 DSNU353I  - 305 12:17:47.45 DSNURWBF - RECORD (1) WILL BE DISCARDED DUE TO CHECK CONSTRAINT ID_GOOD VIOLATION ON TABLE
ABPL6.ABPTB6
 DSNU353I  - 305 12:17:47.45 DSNURWBF - RECORD (2) WILL BE DISCARDED DUE TO CHECK CONSTRAINT ID_GOOD VIOLATION ON TABLE
ABPL6.ABPTB6
 DSNU353I  - 305 12:17:47.45 DSNURWBF - RECORD (3) WILL BE DISCARDED DUE TO CHECK CONSTRAINT ID_GOOD VIOLATION ON TABLE
ABPL6.ABPTB6
 DSNU353I  - 305 12:17:47.45 DSNURWBF - RECORD (4) WILL BE DISCARDED DUE TO CHECK CONSTRAINT ID_GOOD VIOLATION ON TABLE
ABPL6.ABPTB6
 DSNU353I  - 305 12:17:47.45 DSNURWBF - RECORD (5) WILL BE DISCARDED DUE TO CHECK CONSTRAINT ID_GOOD VIOLATION ON TABLE
ABPL6.ABPTB6
 DSNU304I  - 305 12:17:47.47 DSNURWT - (RE)LOAD PHASE STATISTICS - NUMBER OF RECORDS=5 FOR TABLE ABPL6.ABPTB6
 DSNU1147I - 305 12:17:47.47 DSNURWT - (RE)LOAD PHASE STATISTICS - TOTAL NUMBER OF RECORDS LOADED=5 FOR TABLESPACE
ABPLAB6.ABPTS6
 DSNU1150I - 305 12:17:47.47 DSNURWT - (RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS NOT LOADED=5
 DSNU302I    305 12:17:47.47 DSNURILD - (RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS PROCESSED=10
 DSNU300I    305 12:17:47.47 DSNURILD - (RE)LOAD PHASE COMPLETE, ELAPSED TIME=00:00:00
 DSNU375I    305 12:17:47.48 DSNURDNP - DISCARD PHASE STATISTICS - 5 INPUT DATA SET RECORDS DISCARDED
 DSNU376I    305 12:17:47.48 DSNURDIS - DISCARD PHASE COMPLETE, ELAPSED TIME=00:00:00
 DSNU381I  - 305 12:17:47.48 DSNUGSRX - TABLESPACE ABPLAB6.ABPTS6 IS IN COPY PENDING
 DSNU563I  - 305 12:17:47.48 DSNUGSRX - TABLESPACE ABPLAB6.ABPTS6 IS IN CHECK PENDING
 DSNU010I    305 12:17:47.48 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=4
```

Using the DB2 Utilities Enhancement Tool Utility Monitor policy rules to determine when to print or suppress a message, the modified LOAD utility SYSPRINT contents will look like the SYSPRINT in Example 7-20.

Example 7-20 LOAD utility SYSPRINT containing only DSNU1150I

```
1DSNU000I    305 12:15:20.38 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = UETLAB6B
 DSNU1044I   305 12:15:20.40 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
0DSNU050I    305 12:15:20.40 DSNUGUTC -  LOAD DATA LOG NO RESUME YES EBCDIC CCSID(37, 0, 0)
 DSNU650I  - 305 12:15:20.40 DSNURWI -  INTO TABLE "ABPL6"."ABPTB6"
 DSNU650I  - 305 12:15:20.40 DSNURWI -   ("ID_SHOP" POSITION(3:4) SMALLINT,
 DSNU650I  - 305 12:15:20.40 DSNURWI -    "ID_GOOD" POSITION(5:6) SMALLINT,
 DSNU650I  - 305 12:15:20.40 DSNURWI -    "QUANTITY" POSITION(7:10) INTEGER,
 DSNU650I  - 305 12:15:20.40 DSNURWI -    "DATESALE" POSITION(11:20) DATE EXTERNAL) PRESORTED YES INDDN ABPREC  SORTKEYS
0
DSNU304I  - 305 12:15:20.48 DSNURWT - (RE)LOAD PHASE STATISTICS - NUMBER OF RECORDS=5 FOR TABLE ABPL6.ABPTB6
 DSNU1147I - 305 12:15:20.48 DSNURWT - (RE)LOAD PHASE STATISTICS - TOTAL NUMBER OF RECORDS LOADED=5 FOR TABLESPACE
ABPLAB6.ABPTS6
 DSNU1150I - 305 12:15:20.48 DSNURWT - (RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS NOT LOADED=5
 DSNU302I    305 12:15:20.48 DSNURILD - (RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS PROCESSED=10
 DSNU300I    305 12:15:20.48 DSNURILD - (RE)LOAD PHASE COMPLETE, ELAPSED TIME=00:00:00
 DSNU375I    305 12:15:20.57 DSNURDNP - DISCARD PHASE STATISTICS - 5 INPUT DATA SET RECORDS DISCARDED
 DSNU376I    305 12:15:20.57 DSNURDIS - DISCARD PHASE COMPLETE, ELAPSED TIME=00:00:00
 DSNU381I  - 305 12:15:20.57 DSNUGSRX - TABLESPACE ABPLAB6.ABPTS6 IS IN COPY PENDING
 DSNU563I  - 305 12:15:20.57 DSNUGSRX - TABLESPACE ABPLAB6.ABPTS6 IS IN CHECK PENDING
 DSNU010I    305 12:15:20.58 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=4
 ABPU5405I 305 12:15:22.09 Utility return code altered by policy practice UTILITY_RULES_DB2V10
 ABPU5403I 305 12:15:22.10 Utility statement altered by policy practice UTILITY_RULES_DB2V10
 ABPU5009I 305 12:15:22.10 Utility execution completed. SYS=0000, USR=0004
 ABPU5010I 305 12:15:22.10 Allow threads started. Step=1
 ABPU5011I 305 12:15:22.23 Allow threads completed. RC=00000000
 ABPU5003I 305 12:15:22.23 DB2 Utilities Enhancement Tool intercept completed.
```

DB2 message DSNU353I is not present within the utility SYSPRINT, and message DSNU1150I caused the job step to end with a return code 8. The Utility Monitor logs the actions defined in the policy within a set of tables within the DB2 Utilities Enhancement Tool database. By defining a rule within the policy to suppress repetitive messages, DBAs are still able to verify which messages were suppressed by using a simple query, as shown in Example 7-21 and Example 7-22.

Example 7-21 Log contents when changing return codes or suppressing messages (Part 1 of 2)

```
   +-------------------------------------------------------------------------------
   | JOBNAME  |     ID     | SUPPRESS |  RETURN_CODE   |          INSERTED          |
   +-------------------------------------------------------------------------------
 1_| UETLAB6  | DSNU353I   | Y        |             -1 | 2012-10-31-12.42.15.972619 |
 2_| UETLAB6  | DSNU353I   | Y        |             -1 | 2012-10-31-12.42.15.974714 |
 3_| UETLAB6  | DSNU353I   | Y        |             -1 | 2012-10-31-12.42.15.976389 |
 4_| UETLAB6  | DSNU353I   | Y        |             -1 | 2012-10-31-12.42.15.977879 |
 5_| UETLAB6  | DSNU353I   | Y        |             -1 | 2012-10-31-12.42.15.979681 |
 6_| UETLAB6  | DSNU353I   | Y        |             -1 | 2012-10-31-12.42.15.981295 |
 7_| UETLAB6  | DSNU353I   | Y        |             -1 | 2012-10-31-12.42.15.982705 |
 8_| UETLAB6  | DSNU353I   | Y        |             -1 | 2012-10-31-12.42.15.983842 |
 9_| UETLAB6  | DSNU353I   | Y        |             -1 | 2012-10-31-12.42.15.985154 |
10_| UETLAB6  | DSNU353I   | Y        |             -1 | 2012-10-31-12.42.15.986329 |
11_| UETLAB6  | DSNU1150I  | N        |              8 | 2012-10-31-12.42.15.987807 |
   +-------------------------------------------------------------------------------
```

Example 7-22 Log contents when changing return codes or suppressing messages (Part 2 of 2)

```
   ----------------------------------------------------------------------------------------------+
   |                                      TEXT                                                    |
   ----------------------------------------------------------------------------------------------+
 1_|  DSNU353I  - 305 12:42:14.52 DSNURWBF - RECORD (1) WILL BE DISCARDED DUE TO CHECK CONSTRAINT  |
 2_|  ID_GOOD VIOLATION ON TABLE ABPL6.ABPTB6                                                      |
 3_|  DSNU353I  - 305 12:42:14.52 DSNURWBF - RECORD (2) WILL BE DISCARDED DUE TO CHECK CONSTRAINT  |
 4_|  ID_GOOD VIOLATION ON TABLE ABPL6.ABPTB6                                                      |
 5_|  DSNU353I  - 305 12:42:14.52 DSNURWBF - RECORD (3) WILL BE DISCARDED DUE TO CHECK CONSTRAINT  |
 6_|  ID_GOOD VIOLATION ON TABLE ABPL6.ABPTB6                                                      |
 7_|  DSNU353I  - 305 12:42:14.52 DSNURWBF - RECORD (4) WILL BE DISCARDED DUE TO CHECK CONSTRAINT  |
 8_|  ID_GOOD VIOLATION ON TABLE ABPL6.ABPTB6                                                      |
 9_|  DSNU353I  - 305 12:42:14.52 DSNURWBF - RECORD (5) WILL BE DISCARDED DUE TO CHECK CONSTRAINT  |
10_|  ID_GOOD VIOLATION ON TABLE ABPL6.ABPTB6                                                      |
11_|  DSNU1150I - 305 12:42:14.55 DSNURWT - (RE)LOAD PHASE STATISTICS - NUMBER RECORDS NOT LOADED=5|
   ----------------------------------------------------------------------------------------------+
OSUCCESSFUL RETRIEVAL OF         11 ROW(S)
```

7.6 Use the Utility Monitor for auditing purposes or trend analysis

Security within DB2 is mature and continues to evolve over time with each new DB2 version that is released to the public. In addition, security applications are available to further restrict access to underlying VSAM files. Users without the appropriate access to perform certain actions are not able to do so. But for those users that do have access to perform various functions, the Utility Monitor can track who is doing what across all of your subsystems. The policy can be set up to simply take no action and only track the utility invocations to see who is running which utilities and on what objects.

Having this type of information available is helpful for trend analysis. DBAs can now view how many times an object is being reorganized or image-copied over time. Are your business-critical objects being image-copied frequently enough? Are they being reorganized too frequently, perhaps before they really need it? Or, have you ever wondered who rebound a package that caused a problem with performance? Now, DBAs will have this information available to them by using a simple policy rule, as shown in Example 7-23.

Example 7-23 Policy rule to simply audit utility executions

```
EDIT       DB2DUET.V2R2.PARMLIB(TDE1PLCY) - 01.31                Columns 00001 00080
Command ===>                                                        Scroll ===> CSR
000100        <UTILITY NAME="LOAD">
000101           <MONITOR JOURNAL="YES">
000102              <SYNTAX ADD="PRESORTED YES" OPTIONIF="PRESORT"/>
000103              <SYNTAX REMOVE="FORMAT SPANNED YES"/>
000104              <SYNTAX VALUE="LOG %" SUBSTITUTE="LOG NO"/>
000105                <MESSAGE ID="DSNU353I"  SUPPRESS="YES"/>
000106                <MESSAGE ID="DSNU1150I" RETURN_CODE="8"/>
000107           </MONITOR>
000108        </UTILITY>
```

Within Example 7-23, the LOAD utility is defined with the JOURNAL parameter to log all activity within the specified utility. By default, logging is turned on, but can be set differently for different utilities. To turn off logging for a particular utility, such as the REPORT utility, the JOURNAL parameter can be defined as shown in Example 7-24.

Example 7-24 Policy rule to turn off logging of the REPORT utility

```
EDIT       DB2DUET.V2R2.PARMLIB(TDE1PLCY) - 01.32                Columns 00001 00080
Command ===>                                                        Scroll ===> CSR
000109        <UTILITY NAME="REPORT">
000110           <MONITOR JOURNAL="NO"/>
000111        </UTILITY>
```

Using the rule defined in the policy in Example 7-24, no invocations of the REPORT utility will be tracked in the DB2 Utilities Enhancement Tool tables. In addition, there are no syntax rules defined for the REPORT utility as there are for the LOAD utility.

By using a simple SQL query, DBAs can view the information within the DB2 journal tables to see who ran which utilities, and on what objects.

7.7 Accelerate LOADs with PRESORT

Many modern businesses are operating globally now and the amount of data that needs to be processed by the company's IT systems increases dramatically year to year. Longer batch windows are required to process new data, but hardware upgrades are delayed as IT budgets are staying flat or shrinking. This makes it extremely hard to meet defined service level agreements (SLAs), and IT is constantly searching for ways to do "more with less". In this scenario, we show how DB2 Utilities Enhancement Tool's PRESORT option can accelerate LOAD processing and save precious time during the batch window.

There are situations when presorting a data set before loading can lead to reduced overall processing times. DB2 Utilities Enhancement Tool extends the LOAD syntax by adding the PRESORT keyword. The PRESORT keyword instructs the DB2 Utilities Enhancement Tool to presort the input data set in clustering index order before calling the LOAD utility with PRESORTED option set to YES. When LOAD is called with PRESORTED YES, it can run RELOAD and BUILD phases in parallel and skip the sorting of the clustering index. The DB2 Utilities Enhancement Tool determines the sort criteria by using the rules described in Table 7-1.

Table 7-1 DB2 Utilities Enhancement Tool presort rules

Table space/table type	Sort rows by
Partitioned or universal	By clustering index key, or if no clustering index key exists, by the oldest defined index if available.
Simple or segmented	By table OBID first and then by clustering index key. If no clustering index key is available, sort rows by the oldest defined index instead.
ORGANIZE BY HASH	Hash key

If DB2 Sort is available, DB2 Utilities Enhancement Tool will use it to presort the input data set.

In this scenario, we will demonstrate how executing a LOAD with PRESORT can reduce the load elapsed time for a hash organized table. Our test system is configured with DB2 Sort 1.3, so it will be used by DB2 Utilities Enhancement Tool to presort the input data set.

The table that is going to be loaded is based on stored procedure workload's GLWTEPA table, but is organized by hash. The full table Data Definition Language (DDL) is shown in Example 7-25.

Example 7-25 Target table DDL

```
CREATE TABLE UET.TEST_TAB_HASH (
  EMP_NOINTEGERNOT NULL,
  PROJ_NOINTEGERNOT NULL,
  ACT_NOINTEGERNOT NULL,
  EMPTIMEDECIMAL(5, 2)WITH DEFAULT NULL,
  EMSTDATEDATENOT NULL,
  EMENDATEDATEWITH DEFAULT NULL,
  CREATED_TSTIMESTAMPNOT NULLWITH DEFAULT,
  CREATED_BYCHARACTER(8) FOR SBCS DATANOT NULLWITH DEFAULT,
  UPDATED_TSTIMESTAMPNOT NULLWITH DEFAULT,
  UPDATED_BYCHARACTER(8) FOR SBCS DATANOT NULLWITH DEFAULT
  )
  PARTITION BY SIZE EVERY 4G
  ORGANIZE BY HASH UNIQUE (EMP_NO, PROJ_NO, ACT_NO) HASH SPACE 65536K
  IN DATABASE TMPDB
  AUDIT NONE
  DATA CAPTURE NONE
  NOT VOLATILE
  LOGGED
  COMPRESS NO
  BUFFERPOOL BP0
  CCSID EBCDIC;
```

The input data set has 5,000,000 rows that have been unloaded in random order. First, we will execute the LOAD without the PRESORT option. The LOAD job JCL is listed in Example 7-26.

Example 7-26 LOAD without PRESORT

```
//LOAD02 JOB (999,POK),'RE',
//         REGION=0M,NOTIFY=&SYSUID,
//         MSGCLASS=X,CLASS=T
/*JOBPARM S=SC63
//*************************************************************
//STEP1    EXEC DSNUPROC,UID='LOAD04',UTPROC='',SYSTEM='DB0A'
//STEPLIB  DD DSN=DB0AT.SDSNLOAD,DISP=SHR
//         DD DSN=DB0AT.SDSNEXIT,DISP=SHR
//SORTOUT  DD DSN=ADMR4.TEMP.D01,UNIT=SYSDA,SPACE=(CYL,(1000,100)),
//         DISP=(NEW,DELETE,DELETE)
//SYSREC   DD DSN=ADMR4.HUGE.TS01.P00001,
//         DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD DSN=ADMR4.TEMP.D02,UNIT=SYSDA,SPACE=(CYL,(1000,500)),
//         DISP=(MOD,DELETE,DELETE)
//SYSIN    DD *
 LOAD DATA REPLACE
  LOG NO  ENFORCE NO
 EBCDIC CCSID(0037)

 INTO TABLE UET.TEST_TAB_HASH
 (
 EMP_NO
      POSITION (     1  )         INTEGER EXTERNAL (    11 )
  ,
 PROJ_NO
      POSITION (    12  )         INTEGER EXTERNAL (    11 )
  ,
 ACT_NO
      POSITION (    23  )         INTEGER EXTERNAL (    11 )
  ,
 EMPTIME
      POSITION (    34  )         DECIMAL EXTERNAL(     7,  2 )
      NULLIF(    41 ) = '?'
  ,
 EMSTDATE
      POSITION (    42  )         DATE EXTERNAL (    10 )
  ,
 EMENDATE
     POSITION (    52  )         DATE EXTERNAL (    10 )
     NULLIF(    62 ) = '?'
 ,
 CREATED_TS
     POSITION (    63  )         TIMESTAMP EXTERNAL (    26 )
 ,
 CREATED_BY
     POSITION (    89  )         CHAR (     8 )
 ,
 UPDATED_TS
     POSITION (    97  )         TIMESTAMP EXTERNAL (    26 )
```

```
 ,
 UPDATED_BY
     POSITION (   123  )         CHAR (     8 )
)
```

As you can see from Example 7-26 on page 301, we are using a fairly simple LOAD job that replaces all data in the target table. The execution SYSPRINT is shown in Example 7-27.

Example 7-27 LOAD without PRESORT SYSPRINT

```
1
 ABPU5001I 066 13:40:55.64 IBM DB2 Utilities Enhancement Tool Version 0220, FMID=H2AM220, COMP_ID=5655-T58
 ABPU5012I 066 13:40:55.64 Connected to started task ABPID=ABP1
 ABPG8008I 066 13:40:55.65 System=SC63    ,Job=LOAD02  ,Job Id=JOB20433,Step=DSNUPROC,Program=DSNUTILB,User=ADMR4
 ABPU5002I 066 13:40:55.66 Initialization is complete.

 ---------------------------------------------------------------------------------------

 ABPU5004I 066 13:40:56.67 Analysis started. Step=1
 ABPU5301I 066 13:40:56.85 Thread cancel prevented by policy.
 ABPU5005I 066 13:40:56.85 Analysis completed. RC=0
 ABPU5008I 066 13:40:56.85 Utility execution started. Step=1
1DSNU000I    066 13:40:56.92 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = LOAD04
 DSNU1044I   066 13:40:56.94 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
0DSNU050I    066 13:40:56.94 DSNUGUTC -  LOAD DATA REPLACE LOG NO ENFORCE NO EBCDIC CCSID(37)
 DSNU650I  -DBOA 066 13:40:56.94 DSNURWI -  INTO TABLE UET.TEST_TAB_HASH
 DSNU650I  -DBOA 066 13:40:56.94 DSNURWI -   (EMP_NO POSITION(1) INTEGER EXTERNAL(11),
 DSNU650I  -DBOA 066 13:40:56.94 DSNURWI -    PROJ_NO POSITION(12) INTEGER EXTERNAL(11),
 DSNU650I  -DBOA 066 13:40:56.94 DSNURWI -    ACT_NO POSITION(23) INTEGER EXTERNAL(11),
 DSNU650I  -DBOA 066 13:40:56.94 DSNURWI -    EMPTIME POSITION(34) DECIMAL EXTERNAL(7, 2) NULLIF(41)='?',
 DSNU650I  -DBOA 066 13:40:56.94 DSNURWI -    EMSTDATE POSITION(42) DATE EXTERNAL(10),
 DSNU650I  -DBOA 066 13:40:56.94 DSNURWI -    EMENDATE POSITION(52) DATE EXTERNAL(10) NULLIF(62)='?',
 DSNU650I  -DBOA 066 13:40:56.94 DSNURWI -    CREATED_TS POSITION(63) TIMESTAMP EXTERNAL(26),
 DSNU650I  -DBOA 066 13:40:56.94 DSNURWI -    CREATED_BY POSITION(89) CHAR(8),
 DSNU650I  -DBOA 066 13:40:56.94 DSNURWI -    UPDATED_TS POSITION(97) TIMESTAMP EXTERNAL(26),
 DSNU650I  -DBOA 066 13:40:56.94 DSNURWI -    UPDATED_BY POSITION(123) CHAR(8)) KEEPDICTIONARY REUSE
 DSNU242I  -DBOA 066 13:40:56.94 DSNURWI - KEEPDICTIONARY OR COPYDICTIONARY REQUESTED BUT COMPRESS ATTRIBUTE NOT DEFINED
 FOR TABLE SPACE TMPDB.TESTRTAB, PARTITION 1
 DSNU2802I -DBOA 066 13:40:57.21 DSNUGHSH - THE HASH SPACE HAS BEEN PREALLOCATED AT 16381 FOR TABLESPACE TMPDB.TESTRTAB
 PART 0
 DSNU350I  -DBOA 066 13:40:58.24 DSNURRST - EXISTING RECORDS DELETED FROM TABLESPACE
 DSNU304I  -DBOA 066 13:44:55.50 DSNURWT - (RE)LOAD PHASE STATISTICS - NUMBER OF RECORDS=5000000 FOR TABLE
 UET.TEST_TAB_HASH
 DSNU1147I -DBOA 066 13:44:55.50 DSNURWT - (RE)LOAD PHASE STATISTICS - TOTAL NUMBER OF RECORDS LOADED=5000000 FOR
 TABLESPACE TMPDB.TESTRTAB
 DSNU302I    066 13:44:55.50 DSNURILD - (RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS PROCESSED=5000000
 DSNU300I    066 13:44:55.50 DSNURILD - (RE)LOAD PHASE COMPLETE, ELAPSED TIME=00:03:58
 DSNU334OI   066 13:44:55.51 DSNUGSOR - UTILITY PERFORMS DYNAMIC ALLOCATION OF SORT DISK SPACE
 DSNU042I    066 13:44:59.75 DSNUGSOR - SORT PHASE STATISTICS -
                       NUMBER OF RECORDS=4066283
                       ELAPSED TIME=00:00:04
 DSNU349I  -DBOA 066 13:45:04.10 DSNURBXA - BUILD PHASE STATISTICS - NUMBER OF KEYS=4066283 FOR INDEX UET.TEST_TAB_#_ZKG
 DSNU258I    066 13:45:04.11 DSNURBXD - BUILD PHASE STATISTICS - NUMBER OF INDEXES=1
 DSNU259I    066 13:45:04.11 DSNURBXD - BUILD PHASE COMPLETE, ELAPSED TIME=00:00:04
 DSNU380I  -DBOA 066 13:45:04.11 DSNUGSRX - TABLESPACE TMPDB.TESTRTAB PARTITION 1 IS IN COPY PENDING
 DSNU010I    066 13:45:04.11 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=4
 ABPU5403I 066 13:45:04.71 Utility statement altered by policy practice UTILITY_RULES_DB2V10
 ABPU5009I 066 13:45:04.71 Utility execution completed. SYS=0000, USR=0004
 ABPU5003I 066 13:45:04.72 DB2 Utilities Enhancement Tool intercept completed.
```

DB2 Utilities Enhancement Tool was active during the job execution. We can see that the input was analyzed by DB2 Utilities Enhancement Tool and KEEPDICTIONARY REUSE keywords were added to the LOAD control card. Our target table is not compressed, so adding this attribute forces LOAD to issue a warning message. The RELOAD phase took almost four minutes, while the build phase lasted only four seconds. This is normal as we do not have any indexes defined on the table. The JES job log is shown in Example 7-28.

Example 7-28 JES Job log for LOAD without PRESORT

```
1                J E S 2  J O B  L O G  --  S Y S T E M  S C 6 3  --  N O D E  W T S C P L X 2
0
 13.40.55 JOB20433 ---- THURSDAY,  07 MAR 2013 ----
 13.40.55 JOB20433  IRR010I  USERID ADMR4    IS ASSIGNED TO THIS JOB.
 13.40.55 JOB20433  ICH70001I ADMR4    LAST ACCESS AT 13:38:14 ON THURSDAY, MARCH 7, 2013
 13.40.55 JOB20433  $HASP373 LOAD02   STARTED - INIT TWS  - CLASS T - SYS SC63
 13.40.55 JOB20433  IEF403I LOAD02 - STARTED - TIME=13.40.55 - ASID=0047 - SC63
 13.40.55 JOB20433  IGD01008I STORCLAS SET TO
 13.40.55 JOB20433  IGD01008I STORCLAS SET TO
 13.40.55 JOB20433  IGD01008I STORCLAS SET TO
 13.40.56 JOB20433  IGD01008I STORCLAS SET TO
 13.40.56 JOB20433  IGD01008I STORCLAS SET TO
 13.40.56 JOB20433  IGD01008I STORCLAS SET TO
 13.45.05 JOB20433  IGD01008I STORCLAS SET TO
 13.45.05 JOB20433  -                                        --TIMINGS (MINS.)--            ----PAGING COUNTS---
 13.45.05 JOB20433  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK   SERV  PG   PAGE   SWAP    VIO SWAPS
STEPNO
 13.45.05 JOB20433  -LOAD02   STEP1    DSNUPROC    04  48453    .60    .00   4.15  4534K   0     10      0      0     0
1
 13.45.05 JOB20433  IEF404I LOAD02 - ENDED - TIME=13.45.05 - ASID=0047 - SC63
 13.45.05 JOB20433  -LOAD02   ENDED.  NAME-RE                  TOTAL CPU TIME=   .60  TOTAL ELAPSED TIME=  4.15
 13.45.05 JOB20433  $HASP395 LOAD02   ENDED
0------ JES2 JOB STATISTICS ------
-  07 MAR 2013 JOB EXECUTION DATE
-           54 CARDS READ
-          336 SYSOUT PRINT RECORDS
-            0 SYSOUT PUNCH RECORDS
-           22 SYSOUT SPOOL KBYTES
-         4.16 MINUTES EXECUTION TIME
```

From Example 7-28, we can see that the job consumed 0.6 minutes of CPU time and ran for 4.16 minutes.

Next, we see how PRESORT can help make this load run faster. Hash-organized tables benefit most from the presort. The new JCL is listed in Example 7-29.

Example 7-29 LOAD PRESORT JCL

```
//LOAD02 JOB (999,POK),'RE',
//         REGION=0M,NOTIFY=&SYSUID,
//         MSGCLASS=X,CLASS=T
/*JOBPARM S=SC63
//*****************************************************************
//STEP1    EXEC DSNUPROC,UID='LOAD04',UTPROC='',SYSTEM='DBOA'
//STEPLIB  DD DSN=DBOAT.SDSNLOAD,DISP=SHR
//         DD DSN=DBOAT.SDSNEXIT,DISP=SHR
//SORTOUT  DD DSN=ADMR4.TEMP.D01,UNIT=SYSDA,SPACE=(CYL,(1000,100)),
//         DISP=(NEW,DELETE,DELETE)
//SYSREC   DD DSN=ADMR4.HUGE.TS01.P00001,
//         DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD DSN=ADMR4.TEMP.D02,UNIT=SYSDA,SPACE=(CYL,(1000,500)),
```

```
//         DISP=(MOD,DELETE,DELETE)
//SYSIN    DD *
 LOAD PRESORT DATA REPLACE
  LOG NO  ENFORCE NO
 EBCDIC CCSID(0037)

 INTO TABLE UET.TEST_TAB_HASH
 (
 EMP_NO
      POSITION (     1  )         INTEGER EXTERNAL (    11 )
  ,
 PROJ_NO
      POSITION (    12  )         INTEGER EXTERNAL (    11 )
  ,
 ACT_NO
      POSITION (    23  )         INTEGER EXTERNAL (    11 )
  ,
 EMPTIME
      POSITION (    34  )         DECIMAL EXTERNAL(     7,  2 )
      NULLIF(    41 ) = '?'
  ,
 EMSTDATE
      POSITION (    42  )         DATE EXTERNAL (    10 )
  ,
 EMENDATE
      POSITION (    52  )         DATE EXTERNAL (    10 )
      NULLIF(    62 ) = '?'
  ,
 CREATED_TS
      POSITION (    63  )         TIMESTAMP EXTERNAL (    26 )
  ,
 CREATED_BY
      POSITION (    89  )         CHAR (     8 )
  ,
 UPDATED_TS
      POSITION (    97  )         TIMESTAMP EXTERNAL (    26 )
  ,
 UPDATED_BY
      POSITION (   123  )         CHAR (     8 )
 )
```

The only difference compared to the original LOAD in Example 7-26 on page 301 is the PRESORT keyword added right after the LOAD command. The job SYSPRINT is shown in Example 7-30.

Example 7-30 LOAD PRESORT SYSPRINT

```
ABPU50O1I O66 13:45:42.18 IBM DB2 Utilities Enhancement Tool Version O22O, FMID=H2AM22O, COMP_ID=5655-T58
 ABPU5O12I O66 13:45:42.18 Connected to started task ABPID=ABP1
 ABPG80O8I O66 13:45:42.2O System=SC63    ,Job=LOADO2  ,Job Id=JOB2O434,Step=DSNUPROC,Program=DSNUTILB,User=ADMR4
 ABPU5OO2I O66 13:45:42.21 Initialization is complete.

 ------------------------------------------------------------------------------------------

 ABPU5OO4I O66 13:45:43.25 Analysis started. Step=1
 ABPU53O1I O66 13:45:43.42 Thread cancel prevented by policy.
 ABPU5OO5I O66 13:45:43.42 Analysis completed. RC=O
```

```
 ABPU5908I 066 13:45:43.42 IBM DB2 SORT found and will be used
 ABPU5008I 066 13:46:11.55 Utility execution started. Step=1
 ABPU5330I 066 13:46:11.55 Original DSNUTILB syntax follows:
 ABPU5331I 066 13:46:11.55 LOAD PRESORT DATA REPLACE LOG NO ENFORCE NO EBCDIC CCSID(0037) INTO TABLE UET.
 ABPU5331I 066 13:46:11.55 TEST_TAB_HASH ( EMP_NO POSITION ( 1 ) INTEGER EXTERNAL ( 11 ) , PROJ_NO
 ABPU5331I 066 13:46:11.55 POSITION ( 12 ) INTEGER EXTERNAL ( 11 ) , ACT_NO POSITION ( 23 ) INTEGER
 ABPU5331I 066 13:46:11.55 EXTERNAL ( 11 ) , EMPTIME POSITION ( 34 ) DECIMAL EXTERNAL( 7, 2 ) NULLIF( 41 )
 ABPU5331I 066 13:46:11.55 = '?' , EMSTDATE POSITION ( 42 ) DATE EXTERNAL ( 10 ) , EMENDATE POSITION ( 52 )
 ABPU5331I 066 13:46:11.55  DATE EXTERNAL ( 10 ) NULLIF( 62 ) = '?' , CREATED_TS POSITION ( 63 ) TIMESTAMP
 ABPU5331I 066 13:46:11.55 EXTERNAL ( 26 ) , CREATED_BY POSITION ( 89 ) CHAR ( 8 ) , UPDATED_TS POSITION (
 ABPU5331I 066 13:46:11.55 97 ) TIMESTAMP EXTERNAL ( 26 ) , UPDATED_BY POSITION ( 123 ) CHAR ( 8 ) )
 ABPU5331I 066 13:46:11.55 PRESORTED YES KEEPDICTIONARY REUSE
 ABPU5332I 066 13:46:11.55 End of original DSNUTILB syntax listing.
1DSNU000I    066 13:46:11.73 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = LOAD04
 DSNU1044I   066 13:46:11.74 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
0DSNU050I    066 13:46:11.75 DSNUGUTC -  LOAD DATA REPLACE LOG NO ENFORCE NO EBCDIC CCSID(37)
 DSNU650I  -DBOA 066 13:46:11.75 DSNURWI -  INTO TABLE UET.TEST_TAB_HASH
 DSNU650I  -DBOA 066 13:46:11.75 DSNURWI -   (EMP_NO POSITION(1) INTEGER EXTERNAL(11),
 DSNU650I  -DBOA 066 13:46:11.75 DSNURWI -    PROJ_NO POSITION(12) INTEGER EXTERNAL(11),
 DSNU650I  -DBOA 066 13:46:11.75 DSNURWI -    ACT_NO POSITION(23) INTEGER EXTERNAL(11),
 DSNU650I  -DBOA 066 13:46:11.75 DSNURWI -    EMPTIME POSITION(34) DECIMAL EXTERNAL(7, 2) NULLIF(41)='?',
 DSNU650I  -DBOA 066 13:46:11.75 DSNURWI -    EMSTDATE POSITION(42) DATE EXTERNAL(10),
 DSNU650I  -DBOA 066 13:46:11.75 DSNURWI -    EMENDATE POSITION(52) DATE EXTERNAL(10) NULLIF(62)='?',
 DSNU650I  -DBOA 066 13:46:11.75 DSNURWI -    CREATED_TS POSITION(63) TIMESTAMP EXTERNAL(26),
 DSNU650I  -DBOA 066 13:46:11.75 DSNURWI -    CREATED_BY POSITION(89) CHAR(8),
 DSNU650I  -DBOA 066 13:46:11.75 DSNURWI -    UPDATED_TS POSITION(97) TIMESTAMP EXTERNAL(26),
 DSNU650I  -DBOA 066 13:46:11.75 DSNURWI -    UPDATED_BY POSITION(123) CHAR(8)) PRESORTED YES KEEPDICTIONARY REUSE
 INDDN ABPREC SORTKEYS 5000000
 DSNU242I  -DBOA 066 13:46:11.75 DSNURWI - KEEPDICTIONARY OR COPYDICTIONARY REQUESTED BUT COMPRESS ATTRIBUTE NOT DEFINED
 FOR TABLE SPACE TMPDB.TESTRTAB, PARTITION 1
 DSNU188I  -DBOA 066 13:46:11.75 DSNURWI - OPTION PRESORTED YES IS NOT VALID WHEN USED WITH LOAD ON A HASH TABLE SPACE.
 OPTION IS IGNORED
 DSNU280I -DBOA 066 13:46:12.12 DSNUGHSH - THE HASH SPACE HAS BEEN PREALLOCATED AT 16381 FOR TABLESPACE TMPDB.TESTRTAB
 PART 0
 DSNU350I  -DBOA 066 13:46:13.08 DSNURRST - EXISTING RECORDS DELETED FROM TABLESPACE
 DSNU334OI   066 13:46:13.12 DSNUGSOR - UTILITY PERFORMS DYNAMIC ALLOCATION OF SORT DISK SPACE
 DSNU304I  -DBOA 066 13:46:49.55 DSNURWT - (RE)LOAD PHASE STATISTICS - NUMBER OF RECORDS=5000000 FOR TABLE
 UET.TEST_TAB_HASH
 DSNU1147I -DBOA 066 13:46:49.55 DSNURWT - (RE)LOAD PHASE STATISTICS - TOTAL NUMBER OF RECORDS LOADED=5000000 FOR
 TABLESPACE TMPDB.TESTRTAB
 DSNU302I    066 13:46:49.55 DSNURILD - (RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS PROCESSED=5000000
 DSNU300I    066 13:46:49.55 DSNURILD - (RE)LOAD PHASE COMPLETE, ELAPSED TIME=00:00:37
 DSNU042I    066 13:46:49.57 DSNUGSOR - SORT PHASE STATISTICS -
                       NUMBER OF RECORDS=4066283
                       ELAPSED TIME=00:00:00
 DSNU349I  -DBOA 066 13:46:53.04 DSNURBXA - BUILD PHASE STATISTICS - NUMBER OF KEYS=4066283 FOR INDEX UET.TEST_TAB_#_ZKG
 DSNU258I    066 13:46:53.04 DSNURBXD - BUILD PHASE STATISTICS - NUMBER OF INDEXES=1
 DSNU259I    066 13:46:53.04 DSNURBXD - BUILD PHASE COMPLETE, ELAPSED TIME=00:00:03
 DSNU380I  -DBOA 066 13:46:53.08 DSNUGSRX - TABLESPACE TMPDB.TESTRTAB PARTITION 1 IS IN COPY PENDING
 DSNU010I    066 13:46:53.09 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=4
 ABPU5403I 066 13:46:53.69 Utility statement altered by policy practice UTILITY_RULES_DB2V10
 ABPU5009I 066 13:46:53.69 Utility execution completed. SYS=0000, USR=0004
 ABPU5003I 066 13:46:53.70 DB2 Utilities Enhancement Tool intercept completed.
```

As you can see from Example 7-30 on page 304, the same load now took only 37 seconds in a RELOAD phase, which is significantly faster than more than four minutes during the original LOAD processing. The JES job log is shown in Example 7-31.

Example 7-31 LOAD PRESORT JES job log

```
1                J E S 2  J O B  L O G  --  S Y S T E M  S C 6 3  --  N O D E  W T S C P L X 2
0
 13.45.41 JOB20434 ---- THURSDAY,  07 MAR 2013 ----
 13.45.41 JOB20434  IRR010I  USERID ADMR4    IS ASSIGNED TO THIS JOB.
 13.45.42 JOB20434  ICH70001I ADMR4    LAST ACCESS AT 13:40:55 ON THURSDAY, MARCH 7, 2013
```

```
 13.45.42 JOB20434  $HASP373 LOAD02   STARTED - INIT TWS  - CLASS T - SYS SC63
 13.45.42 JOB20434  IEF403I LOAD02 - STARTED - TIME=13.45.42 - ASID=0047 - SC63
 13.45.42 JOB20434  IGD01008I STORCLAS SET TO
 13.45.42 JOB20434  IGD01008I STORCLAS SET TO
 13.45.42 JOB20434  IGD01008I STORCLAS SET TO
 13.45.43 JOB20434  IGD01008I STORCLAS SET TO
 13.45.43 JOB20434  IGD01008I STORCLAS SET TO
 13.46.11 JOB20434  IGD01008I STORCLAS SET TO
 13.46.54 JOB20434  IGD01008I STORCLAS SET TO
 13.46.54 JOB20434  -                                        --TIMINGS (MINS.)--            ----PAGING COUNTS---
 13.46.54 JOB20434  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK   SERV  PG   PAGE   SWAP    VIO SWAPS
STEPNO
 13.46.54 JOB20434  -LOAD02   STEP1    DSNUPROC    04  35204    .83    .00   1.20 12980K   0     10      0      0     0
1
 13.46.54 JOB20434  IEF404I LOAD02 - ENDED - TIME=13.46.54 - ASID=0047 - SC63
 13.46.54 JOB20434  -LOAD02   ENDED.  NAME-RE                  TOTAL CPU TIME=   .83  TOTAL ELAPSED TIME=  1.20
 13.46.54 JOB20434  $HASP395 LOAD02   ENDED
0------ JES2 JOB STATISTICS ------
-  07 MAR 2013 JOB EXECUTION DATE
-           54 CARDS READ
-          395 SYSOUT PRINT RECORDS
-            0 SYSOUT PUNCH RECORDS
-           26 SYSOUT SPOOL KBYTES
-         1.20 MINUTES EXECUTION TIME
```

We can see that the job consumed 0.83 seconds of CPU time and ran for 1.2 minutes. From the SYSPRINT, we know that the LOAD utility consumed about 40 seconds in total, so about 32 seconds were used by DB2 Utilities Enhancement Tool to presort the input data set using the DB2 Sort.

Our results clearly demonstrate the value of DB2 Utilities Enhancement Tool in conjunction with DB2 sort. The overall elapsed time was 3.5 times less than the elapsed time of the original LOAD. This can bring significant relief to the LOAD-heavy shops with shrinking maintenance windows.

7.8 Avoid contention or delays during your maintenance window

Have you ever been called for support at an inconvenient hour because a thread had a lock on an object, preventing maintenance from running? No matter the time of day or night, if you are the designated on-call DBA, you may be required to log in to the system right away to diagnose and address the issue. And how frustrating is it to only discover that the cause was something simple, like a long-running thread preventing the utility from completing successfully? Take advantage of the Utility Monitor's automatic thread canceling and blocking feature within the DB2 Utilities Enhancement Tool policy.

Using the DB2 Utilities Enhancement Tool policy rules, threads can be automatically canceled for the object defined within the DB2 utility, and additional threads can be blocked from obtaining locks until the utility completes. This can be done automatically anytime that a utility is invoked through the program DSNUTILB.

The manner in which thread blocking and canceling is turned on is through the DB2 subsystem identifier (SSID) section in the policy, as shown in Example 7-32 on page 307.

Example 7-32 Sample policy rule to invoke thread canceling

```
<DB2SYSTEM SSID="DB0A" ACTION="MONITOR_UTILITY">
   <USE_PRACTICE NAME="UTILITY_RULES_DB2V10"/>
     <INCLUDE>
       <RULE UTILITY_JOBNAME="LOAD%"/>
     </INCLUDE>
  </DB2SYSTEM>
  <DB2SYSTEM SSID="DB0A" ACTION="BLOCK_AND_CANCEL_THREADS">
   <USE_PRACTICE NAME="UTILITY_RULES_DB2V10"/>
     <INCLUDE>
       <RULE UTILITY_JOBNAME="LOAD%"/>
     </INCLUDE>
  </DB2SYSTEM>
```

In Example 7-32, the subsystem DB0A will have two actions take place when certain criteria resolve to be true. DB2SYSTEM with action MONITOR_UTILITY will cause the Utility Monitor to invoke anytime that the utility job name being executed matches the wildcard criteria of 'LOAD%'. In addition, DB2SYSTEM with action BLOCK_AND_CANCEL_THREADS will cause any active threads to be canceled on the objects defined in the executing utility job, and will block any additional threads from taking a lock.

In order to demonstrate the thread cancellation feature of DB2 Utilities Enhancement Tool, we will use the LOAD utility and a query described in Example 7-33 that will act as a blocker. The LOAD will use the same control statements as used in 7.7, “Accelerate LOADs with PRESORT” on page 299. See Example 7-29 on page 303.

Example 7-33 Blocker JCL

```
//RUNSQL JOB (999,POK),'RE',
//         REGION=0M,NOTIFY=&SYSUID,
//         MSGCLASS=X,CLASS=T
/*JOBPARM S=SC63
//RUNTEP2 EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD SYSOUT=*
//STEPLIB  DD DSN=DB0AT.SDSNLOAD,DISP=SHR
//         DD DSN=DB0AT.SDSNEXIT,DISP=SHR
//SYSTSIN  DD *
 DSN SYSTEM(DB0A)
 RUN  PROGRAM(DSNTEP2) PLAN(DSNTEP10) -
      PARMS('/ALIGN(LHS) MIXED -
      TOLWARN(YES)') -
      LIB('DB0AM.RUNLIB.LOAD')
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN    DD *
SELECT COUNT (*)
  FROM (SELECT B.EMP_NO,
               A.PROJ_NO,
               A.ACT_NO,
               ROW_NUMBER ()
               OVER (PARTITION BY A.EMP_NO, A.PROJ_NO, A.ACT_NO
                     ORDER BY A.EMP_NO)
                  AS RN
          FROM UET.TEST_TAB_HASH A, UET.TEST_TAB_HASH B
         WHERE     A.EMP_NO = B.EMP_NO
```

```
                AND A.ACT_NO = B.ACT_NO
                AND A.PROJ_NO = B.PROJ_NO) C
 WHERE C.RN = 1 WITH RS
```

As you can see, we are using the DSNTEP2 program to execute a query that joins the TEST_TAB_HASH table with itself and orders the row in every EMP_NO, PROJ_NO, and ACT_NO group by EMP_NO. The query does not have much meaning, but it is complicated enough to run for several minutes in our environment. We also use the read stability (RS) isolation level in order to acquire locks that will be detected by DB2 Utilities Enhancement Tool. Depending on the isolation level and various bind options, queries can run without acquiring locks on cursor stability (CS) and uncommitted read (UR) isolation levels. Due to current limitations, if we do not specify the RS isolation level, DB2 Utilities Enhancement Tool is not able to detect that our query is accessing the TEST_TAB_HASH table.

We submit the LOAD22 job after submitting the RUNSQL job but while it is still running. DB2 Utilities Enhancement Tool will intercept the load call and analyze the objects that are referenced in the LOAD control cards. In our case, the only object accessed is the UET.TEST_TAB_HASH table. Next, it will look through all of the threads that are active on the system and will determine the objects that they are accessing using lock and claim information. If any of the threads are accessing the objects referenced in utility control cards in an incompatible mode, they will be canceled using the CANCEL THREAD command. Further references to these objects will be blocked by placing the objects into the utility access mode.

For every utility invocation that leads to thread cancellation, DB2 Utilities Enhancement Tool will generate five reports.

The All Active Threads Report (REPT0002) lists all active threads that were running when the utility job was submitted. Example 7-34 shows the content of this report for our LOAD execution.

Example 7-34 All active threads

```
** All Active Threads Report for CANCEL_THREADS Request=1      PAGE=1    **
 ____________________________________________________________________________

 SEQNO CAN TTOKEN ELAPTIME/   PLANNAME/    CONNID/      AUTHID/  JOBNAME/ PROGRAM  INDB2 TIME/     COMMIT/  SQL CALL/
                  STATUS/     CORRID       CONNTYPE     ORIGAUTH ASIDX             INDB2 CPU       ABORT    GETPAGE
                  COLLID
 ----- --- ------ ----------- ------------ ------------ -------- -------- -------- --------------- -------- ---------
     1 ***   9488 00:00:00.00 DSNTEP10     BATCH        ADMR4    RUNSQL   DSNTEP2  00:00:00.000206        1          6
                  T           RUNSQL       TSO          ADMR4    0047              00:00:00.000183        0      10528
                  DSNTEP2
     2       9461 00:00:00.06 ABP22PLN     DB2CALL      STC      ABP1PROC          00:00:00.063493       51         10
                  T           ABP1PROC     CAF          STC      0095              00:00:00.007807        0        744

     3       9471 00:00:00.17 ABP22PLN     DB2CALL      STC      ABP1PROC          00:00:00.174674       76        482
                  T           ABP1PROC     CAF          STC      0095              00:00:00.036486        0        347

     4          2 00:00:03.60 ?RRSAF       RRSAF        STC      DBOAADMT          00:00:03.605009     6131       3067
                  T           DBOAADMT_DMN RRSAF        STC      0094              00:00:02.916455        0      73755

     5          3 00:00:00.02 ?RRSAF       RRSAF        STC      DBOAADMT          00:00:00.028162       33        164
                  T           DBOAADMT_II  RRSAF        STC      0094              00:00:00.008035        0         40
 ** END OF REPORT **
```

We had our blocker thread with JOBNAME=RUNSQL, PROGRAM=DSNTEP2, and several threads for DB2 Utilities Enhancement Tool and Administrative Task Scheduler. Notice three asterisks in the CAN column for our blocker. If the thread is marked with three asterisks in the

CAN column, a cancel was attempted on that thread. This report also contains the information on the number of SQL calls, GETPAGE, and CLASS 1 and CLASS 2 CPU and elapsed times. You can use this information to see the whole picture of what was executing on the system, to validate that canceling the thread was the correct choice, and if not, adjusting your batch schedule to avoid conflicts in the future.

The All Active Threads Objects Referenced Report (REP0004) lists the objects referenced by the active threads. See Example 7-35.

Example 7-35 Referenced objects

```
** All Active Threads Objects Referenced Report for CANCEL_THREADS Request=1     PAGE=1    **
 ________________________________________________________________________________________

 SEQNO CAN TTOKEN ELAPTIME    PLANNAME  AUTHID   MATCH  DBNAME   PSNAME   PART  LOCK_TYPE    STATE  LOCK_COUNT
 ----- --- ------ ----------- --------  -------- -----  -------- -------- ----- ------------ -----  ----------
     1 ***   9488 00:00:00.00  DSNTEP10  ADMR4          DSNDB07   DSN4K00      0  PAGESET      IS           1
                                                                              0  ROW          S            1
                                                     TMPDB     TESTRTAB      1  PART_SPL     IS           1
     2       9461 00:00:00.06  ABP22PLN  STC              **NO OBJECTS REFERENCED**
     3       9471 00:00:00.17  ABP22PLN  STC              **NO OBJECTS REFERENCED**
 ** END OF REPORT **
```

Our blocker thread is holding an IS partition lock on the TESTDB.TESTRTAB table space that contains the UET.TEST_TAB_HASH table. This is obviously in conflict with the LOAD REPLACE that we are performing on this table, so DB2 Utilities Enhancement Tool will attempt to cancel the thread.

The All Active Threads Unit of Recovery Report (REP0003) is shown in Example 7-36.

Example 7-36 Active units of recovery report

```
** All Active Threads Unit of Recovery Report for CANCEL_THREADS Request=1      PAGE=1    **
 ________________________________________________________________________________________

 SEQNO CAN TTOKEN PLANNAME AUTHID   UR_STRT_DT UR_STRT_TIME    UR_STATE UR_ELAP_TIME    UR_LOG_SRBA  UR_LOG_ERBA  LOG_PAGES LOG_RECS
 ----- --- ------ -------- -------- ========== --------------- -------- --------------- ------------ ------------ --------- ---------
     1 ***   9488 DSNTEP10 ADMR4    **NO UR**
     2       9461 ABP22PLN STC      03/08/2013 00:57:38.181364          00:00:00.112854 000029C44BD1 000029C45178         1       305
     3       9471 ABP22PLN STC      03/08/2013 00:57:38.181396          00:00:00.124004 000029C44E1E 000029C45144         1       623
     4          2 ?RRSAF   STC      03/04/2013 18:50:37.038164          03-06:07:012    0000C0769ED6 0000C07722B4         9        32
     5          3 ?RRSAF   STC      03/04/2013 18:50:32.447949          03-06:07:058    0000C075AE10 0000C0761141         7        24
 ** END OF REPORT **
```

Our blocker thread did not run any data modification statements, so DB2 Utilities Enhancement Tool is showing that it has no unit of recovery. In the case of the blockers with units of recovery, this information will help you to make an informed decision about the actions that need to be taken to finish the canceled processing after the batch window.

The Threads Canceled Report (REP0000) lists the threads that were canceled by DB2 Utilities Enhancement Tool. Its contents are an excerpt from the All Active Threads Report. See Example 7-37.

Example 7-37 Canceled threads report

```
** Threads Canceled Report for CANCEL_THREADS Request=1     PAGE=1    **
 ________________________________________________________________________

 SEQNO TTOKEN ELAPTIME/   PLANNAME/   CONNID/     AUTHID/  JOBNAME/ PROGRAM  INDB2 TIME/    COMMIT/  SQL CALL/
            STATUS/     CORRID      CONNTYPE    ORIGAUTH ASIDX           INDB2 CPU      ABORT    GETPAGE
             COLLID
 ----- ------ ----------- ----------- ----------- -------- -------- -------- --------------- -------- ----------
     1   9488 00:00:00.00 DSNTEP10     BATCH       ADMR4    RUNSQL   DSNTEP2  00:00:00.000206       1         6
            T          RUNSQL      TSO         ADMR4    0047            00:00:00.000183       0     105283
```

```
            DSNTEP2
 ** END OF REPORT **
```

The Threads Canceled Unit of Recovery Report (REP0001) is an excerpt from the All Active Threads Unit of Recovery Report. See Example 7-38.

Example 7-38 Canceled units of recovery

```
** Threads Canceled Unit of Recovery Report for CANCEL_THREADS Request=1      PAGE=1    **
 ________________________________________________________________________________

 SEQNO TTOKEN PLANNAME AUTHID   UR_STRT_DT UR_STRT_TIME    UR_STATE UR_ELAP_TIME    UR_LOG_SRBA  UR_LOG_ERBA  LOG_PAGES LOG_RECS
 ----- ------ -------- -------- ========== --------------- -------- --------------- ------------ ------------ --------- ---------
     1   9488 DSNTEP10 ADMR4    **NO UR**
 ** END OF REPORT **
```

SPRT0001 DD contains the general information about the thread cancellation invocation. See Example 7-39.

Example 7-39 Thread cancellation processing report

```
ABPB6120I Processing CANCEL_THREADS request=1   , Started at 19:57:13.1; Parameters are as follows:
 ABPB6998I   TABLE "UET"."TEST_TAB_HASH"
 ABPB6410I Thread list build was successful: Active thread count=5
 ABPB6411I Thread filtering applied: Threads of cancel interest count=1
 ABPB6543W Failed to determine the objects referenced by thread 2     for report REPT0004, RC=00000004
 ABPB6543W Failed to determine the objects referenced by thread 3     for report REPT0004, RC=00000004
 ABPB6450I Thread cancel issued:  PLAN=DSNTEP10  CONN=TSO       CORR=RUNSQL         AUTH=ADMR4     OAUTH=ADMR4
TOKN=9488
 ABPB6451I                     JOBN=RUNSQL    ASID=0047     PROGNAME=DSNTEP2   COLLID=DSNTEP2
 ABPS0220I TCB: 007CA8E0 Session: 25145440 - CANCEL THREAD requested for thread token 9488
 ABPS0211I  DSNV426I  -DBOA DSNVCT THREAD '009488' HAS BEEN CANCELED

 ABPB6470I Thread status checking for canceled threads started. Retry count=20   Retry interval=3
 ABPB6461I Thread has terminated: PLAN=DSNTEP10  CONN=TSO       CORR=RUNSQL         AUTH=ADMR4     OAUTH=ADMR4
TOKN=9488
 ABPB6471I Thread status checking ended; all canceled threads have terminated
 ABPB6121I Processing of CANCEL_THREADS request=1   has ended at 19:57:13.4: Threads canceled=1   , RC=00000004
 ABPB6700I DB2 Utilities Enhancement Tool cancel execution ended. Highest RC=00000004
```

Example 7-39 lists the total number of active threads (ABP6410I) and the number of threads of cancel interest (ABP6411I). ABP6543W indicates that there were two threads for which DB2 Utilities Enhancement Tool was not able to determine the list of referenced objects. If we match the thread sequence numbers (2 and 3) to the All Active Thread Report from Example 7-34 on page 308, we can see that these are DB2 Utilities Enhancement Tool's own threads so both of these warnings can be safely ignored.

The ABPS0220I message shows that -CANCEL THREAD was called for the thread with token 9488, which is our blocker thread. ABPS0211I confirms that the thread has been canceled successfully. DB2 Utilities Enhancement Tool checks to see whether all canceled threads have terminated. If not, DB2 Utilities Enhancement Tool will retry the cancel for the specified number of times (CHECK_THDTERM_RETRY_COUNT parameter), delaying each retry by the specified number of seconds (CHECK_THDTERM_RETRY_INTERVAL parameter). ABPB6471I will be issued when all of the canceled threads have terminated. In some circumstances, the CANCEL THREAD command might not actually terminate a thread. To accommodate this situation, DB2 Utilities Enhancement Tool provides an alternative escalated cancellation method. When you perform an escalated cancellation, DB2 Utilities Enhancement Tool issues the z/OS Cancel command to terminate the batch job, TSO user, or started task that is associated with the thread.

After the utility processing is finished, the affected objects are returned to the normal read/write access mode.

DB2 Utilities Enhancement Tool thread cancellation functionality is not limited to automatic invocations during utility executions. The DB2 Utilities Enhancement Tool batch interface enables you to create a batch job step that cancels threads on the DB2 objects that an application or utility needs to access during batch processing. Optionally, this job step can also block new threads from forming on the objects that a utility needs until after the utility completes.

By incorporating a thread-cancellation job step into your batch jobs, you can ensure that the applications and utilities that run during the batch window can access the DB2 resources that they need.

Part 3

IBM DB2 Fast Copy Solution Pack

In this part, we discuss the tools included in the Fast Copy Solution Pack:

8

IBM DB2 Recovery Expert for z/OS

IBM DB2 Recovery Expert for z/OS is a self-managing backup and recovery solution to help protect the mission-critical data that runs your business. DB2 Recovery Expert for z/OS recovers database objects safely, precisely, and quickly without having to resort to full database recovery. It can help you avoid accidental data loss or corruption by providing a faster, less costly method of recovery when time is of the essence.

In this chapter, we first provide a general description of DB2 Recovery Expert for z/OS V3.1 (DB2 Recovery Expert) and then we demonstrate four DB2 Recovery Expert usage scenarios:

- The first scenario is DB2 subsystem level backup. We discuss the system level backup (SLB) requirements and how Recovery Expert can help you verify your configuration to ensure that it is recoverable. Then, we show how to alter your subsystem with Recovery Expert to make it compliant with these requirements. The scenario also demonstrates how to create DB2 and Recovery Expert SLBs and some additional features provided by Recovery Expert.
- The second scenario is a subsystem restore. We show you how the system level restores can be executed in Recovery Expert and what additional functionality is present to make the DBA's life easier.
- The third scenario is object level restore. We demonstrate how to create an Object Profile and use it to generate several recovery plans.
- The last scenario is a dropped object restore. We show how to use Recovery Expert web interface to drive log-based dropped object recovery.

The chapter contains the following sections:

- DB2 Recovery Expert
- System level backup scenario
- System level restore
- Object recovery
- Log-based dropped object recovery using web interface

8.1 DB2 Recovery Expert

DB2 Recovery Expert helps you avoid accidental data loss or corruption by providing the fastest, least costly method of backup and recovery.

DB2 Recovery Expert provides a fast and easy-to-use implementation of a DB2 system backup and recovery methodology. It reduces backup windows by leveraging storage-based, fast replication so that backups of multi-terabyte databases can be performed in seconds. It simplifies backup and recovery methodologies by allowing full-system, application, and disaster recoveries to be performed from a common system-level backup.

Consistent backups can be created using “full” or “data-only” SLB options. It provides DB2 system backup and recovery support for complex applications, where all of the application’s data must be backed up, restored, and recovered as a unit. System backups can be taken while the DB2 system remains active. In addition, when creating SLBs, DB2 Recovery Expert invokes storage-based fast-replication facilities through appropriate storage processor APIs, reducing host CPU and I/O resource utilization and enabling existing data copy methods to be used while the DB2 system is down.

DB2 Recovery Expert has integrated intelligent recovery and disaster recovery managers that analyze recovery assets and establish optimal recovery procedures to minimize recovery time and recovery point objectives. Recovery jobs are tailored specifically to available backup and hardware resources.

The Intelligent Recovery Manager supplies the ability to perform local recoveries efficiently by using all available recovery resources. Restore operations that invoke fast-replication facilities through appropriate storage processor APIs and parallel recovery can significantly reduce recovery time and complexity.

The Intelligent Disaster Recovery Manager uses local site procedures to prepare for offsite disaster recovery or disaster restart in advance. The information that is acquired allows Intelligent Disaster Recovery Manager to intelligently perform remote site restoration operations and appropriate recovery or restart procedures.

DB2 Recovery Expert offers several unique and significant features that you can use to significantly improve your DB2 backup and recovery methodology.

The Intelligent Recovery Manager takes into account all the recovery resources and all the recovery events (quiesce points, log no utilities, and so on). It produces a comprehensive set of recovery plans, detailing all the possible methods to recover a set of objects. It also automatically includes related referential integrity (RI), XML, and LOB spaces so that related DB2 objects remain in sync after the recovery. It can also generate the recovery jobs to run in parallel, which can significantly reduce recover time.

In general, it produces a comprehensive set of recovery plans and orders them from fastest to slowest, allowing a DBA to easily choose the most effective recovery path.

DB2 Recovery Expert offers these features:

- ISPF interface:

  DB2 Recovery Expert provides an easy-to-use ISPF interface to manage all of its main functions. Through the ISPF interface, users can easily create system backup, object, and disaster recovery profiles that contain all the necessary information to run backup, restore, and disaster recovery jobs.

- DB2 system backup and recovery:

  DB2 Recovery Expert provides the ability to back up an entire DB2 system (full image or data only) or a partial DB2 system at the volume level through the use of system backup profiles. These profiles designate the DB2 system, user options, and resources that will be used to perform the backup. When executed, DB2 Recovery Expert will validate that all DB2 data is included in the backup by performing dynamic discovery of all the data sets and their associated volumes. This ensures that the entire DB2 system is backed up.

  The system backup can be taken while the DB2 system is active by using fast-replication storage devices. DB2 Recovery Expert also provides the ability to “offload” or copy the system backup to tape. A system backup can be used at the local site to restore an entire DB2 system, or at the remote site to restore the DB2 system for disaster recovery purposes. In addition, DB2 Recovery Expert can restore individual DB2 objects from a system backup.

- DB2 system backup configuration and management

  DB2 Recovery Expert includes a System Setup feature that can be used to discover a DB2 system and recommend layout and configuration changes so that the DB2 system can be set up appropriately to accommodate a system backup and recovery methodology.

- Backup validation

  DB2 Recovery Expert provides extensive backup validation to ensure that the system backup contains all DB2 files and catalog structures required for a successful recovery. It also checks the state of all table space and index spaces to ensure that they are in an appropriate state to back up. For example, if a table space is in recover pending, it cannot be restored from the SLB.

- Tape offload support

  DB2 Recovery Expert provides tape offload support to automate copying a system backup or partial system backup from disk to tape. Backups created on disk can be copied to tape using DFSMSdss or Fast Dump and Restore (FDR) so that the backup disk volume pool can be reused. DB2 Recovery Expert allows you to encrypt the data when offloading to tape or disk. Data encryption can be specified for either DFSMSdss or FDR offloads. DB2 Recovery Expert provides a report of backups and offloaded tapes for offsite support. A subsequent DB2 system restore operation restores the backup from disk or tape depending on system backup availability and recovery scope. DB2 Recovery Expert uses the most appropriate backup for application-level recovery and restores the databases from disk, tape, or a previous image copy, depending on which backup provides the most expedient recovery process.

- Object level backup

  DB2 Recovery Expert provides the ability to create data set level fast replication backups. This feature can be used to drive EMC Snap Dataset or IBM Dataset FlashCopy to create a backup for an individual table space and index or groups of table spaces and indexes. Users can create two types of object backups by using fast replication. One type is created using traditional image copies that are registered in SYSIBM.SYSCOPY and usable by any recovery tool or other process that uses image copies. Another type is created by using VSAM type copies that are registered in the DB2 Recovery Expert internal repository. These backups are usable for recovery purposes when DB2 Recovery Expert generates recovery JCL.

- Object level recovery

  DB2 Recovery Expert object level recovery enables users to recover individual DB2 objects or groups of related objects from a system backup or from image copies. Users create Object Profiles that contain the information that will be used to recover a DB2 subsystem’s objects to a desired point in time. When recovery is necessary, DB2

Recovery Expert analyzes all the available backup resources to generate the most appropriate recovery JCL to recover all the objects in the profile. DB2 Recovery Expert can also invoke additional recovery utilities after restoring the databases to bring them to a more current point-in-time. In addition, object-level recovery leverages storage-based data set fast replication facilities. The use of storage-based data set fast replication allows object recovery to be performed in parallel to the database restore process, thus significantly reducing the overall recovery time. Object or database recoveries that traditionally have taken many hours can be performed in minutes or seconds by using DB2 Recovery Expert.

- Tape-based disaster restart

  DB2 Recovery Expert provides disaster recovery support by transforming traditional DB2 disaster recovery procedures into a tape-based disaster restart methodology. System backups can be tagged for offsite transport to a disaster recovery site during the offload process. The tapes can be delivered to the disaster recovery site via a remotely replicated virtual tape system, a remote tape drive, or via the old-fashioned truck method. The tape-based disaster restart methodology loads the system backup tapes and restarts DB2 at the disaster recovery site. The DB2 restart process transforms the system backup into a transactionally consistent DB2 system that is ready to accept work. Using DB2 Recovery Expert to implement a DB2 tape-based disaster restart methodology simplifies disaster recovery procedures and reduces recovery time objectives.

- Automation and management of disaster recovery

  Using DB2 Recovery Expert to automate and manage traditional disaster recovery processes simplifies disaster recovery procedures, reduces recovery time, and makes the recovery process less error-prone. Users create disaster recovery profiles, which contain the recovery assets that are sent to the recovery site. These assets can include system backups, archive logs, change accumulation files, and image copies. The DB2 Recovery Expert Intelligent Disaster Recovery Manager runs at some set interval at the local site, performing the following functions.

- System restore interface

  DB2 Recovery Expert provides an ISPF interface to display all the system backups that have been performed. DB2 systems can be restored by selecting a system backup and specifying restore and recovery options. DB2 Recovery Expert then builds JCL that can be executed to restore and recover the entire DB2 system from the system backup and other recovery resources created since the system backup.

- Copy blades

  DB2 Recovery Expert copy blades provide storage processor integration and extensibility to support heterogeneous storage platforms and fast-replication features. DB2 Recovery Expert supports IBM, EMC, and HDS storage systems and fast-replication facilities using integrated copy blades.

- Metadata repository DB2

  DB2 Recovery Expert provides a comprehensive metadata repository to record backup information, such as backup time, backup type, log byte addresses, and volumes used for the backup. Reports can be generated to monitor information, such as backup methods and operations, storage volume usage, system backup volume usage, and archived backups.

- Multi-purpose SLB

  DB2 Recovery Expert-generated SLBs can be used for multiple purposes, saving storage and processing resources. An SLB can be used for DB2 system recovery, application recovery, object recovery, and for disaster restart or recovery. With this ability, significant CPU, I/O, and storage resources that would otherwise be required to make multiple backups for different purposes are saved.

- DB2 recovery performed efficiently

  DB2 Recovery Expert reduces recovery time by running restore and recovery operations in parallel. Storage-based fast-replication facilities are used to restore backups quickly while invoking DB2 recovery processes in parallel to reduce overall recovery time and minimize DB2 and application downtime. DB2 systems are restored using volume-based fast replication, and DB2 applications and objects are restored using data set-based fast-replication facilities.

- DB2 version support

  DB2 Recovery Expert supports DB2 versions 8, 9, and 10 in either data sharing or non-data sharing modes of operation.

There are two relatively recent IBM Redbooks publications about IBM DB2 Recovery Expert for z/OS:

- *Optimizing Restore and Recovery Solutions with DB2 Recovery Expert for z/OS,* SG24-7606
- *IBM DB2 Recovery Expert for z/OS User Scenarios,* SG24-7226

DB2 Recovery Expert can perform a RESTORE from an SLB that has been taken with the BACKUP SYSTEM or one of its own SLBs. It automates many processes around the restore. It can restore the log volumes. It automatically clears coupling facility structures when restoring a data sharing system. It also automatically detects when objects need further attention after the log has been applied. This can happen if the log range being applied contains an online REORG for a table space. This table space will be in recovery pending after the system restore. DB2 Recovery Expert can detect this situation and automatically generate a RECOVER utility to recover these table spaces from an image copy.

In the next sections, we cover the new functions and enhancements of DB2 Recovery Expert V3.1.

8.1.1 What is new in DB2 Recovery Expert V3.1

DB2 Recovery Expert V3.1 includes many new features that enhance an already powerful recovery solution. These new features focus on making it easier to protect mission-critical data and continue to provide the fastest, least costly method of recovery when time is of the essence:

- Integration with IBM Tools Customizer for z/OS, providing one interface to customize and generate installation JCL for all DB2 Tools.
- Enhanced ISPF interface that includes the recovery and log analysis functionality previously found only in the V2.2 GUI interface.
- Completely redesigned web browser interface replaces the V2.2 GUI interface. The look and feel are more in sync with the ISPF interface. It eliminates the need to separately install a local version for each user.
- Support for IPv6 in DB2 Recovery Expert server, agent, and client. The agent and server support and accept connections from both IPv4 and IPv6 clients.

- Ability to generate Data Definition Language (DDL) for object authorizations, enabling the restoration of object authorizations when recovering dropped objects.
- Ability to generate BIND statements for related plans and packages, enabling the binding of plans and packages for objects being recovered. This feature is available for the following situations:
  - Existing object recovery or dropped object recovery from the SLR[1]
  - Dropped object recovery using Log-Based Dropped Object Recovery
  - When generating binds for the new DDL generation option
- Expanded the object types that can be recovered to include indexes, views, synonyms, aliases, data types, triggers, functions, stored procedures, sequences, and roles. It added the ability to perform dropped object recovery or DDL generation for all of the added object types.

8.1.2 Base architecture

IBM DB2 Recovery Expert works like a client/server architecture, where you can have one server connect to several agents, and each agent connects to one DB2 for z/OS. Figure 8-1 shows the architecture for the product.

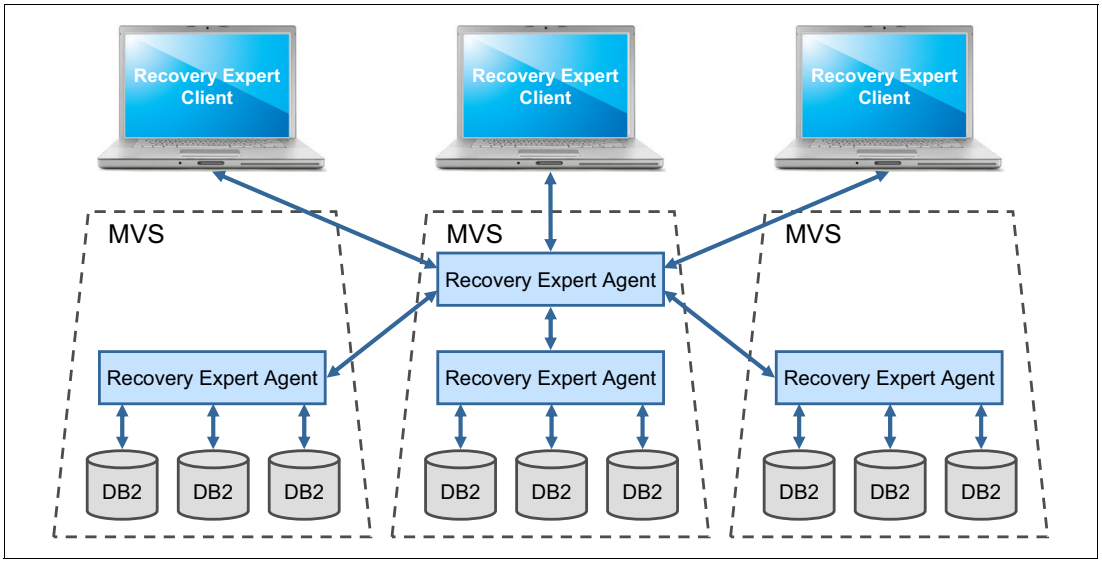

Figure 8-1 DB2 Recovery Expert 3.1 for z/OS base architecture

IBM DB2 Recovery for z/OS consists of these main components:

- DB2 Recovery Expert Server

  The DB2 Recovery Expert for z/OS server centrally manages and controls all DB2 Recovery Expert for z/OS functions that are performed on behalf of user requests. You must run at least one instance of the server to manage all of your DB2 subsystems and data sharing groups and to support all of your DB2 Recovery Expert for z/OS user clients. Using TCP/IP connections, the DB2 Recovery Expert for z/OS server, clients, and agents communicate with each other to perform the recovery functions.
- DB2 Recovery Expert for z/OS agent

[1] DB2 Recovery Expert offers the option of capturing DB2 system catalog information and storing it in a set of DB2 tables referred to as the schema level repository (SLR).

The DB2 Recovery Expert for z/OS agent provides access to database and system services in support of the DB2 Recovery Expert for z/OS server and remote clients. You must run one instance of the agent on every system or logical partition (LPAR) that hosts

DB2 subsystems or data sharing groups that you want to access with DB2 Recovery Expert for z/OS. Each agent communicates with the DB2 Recovery Expert for z/OS server to provide services.

- DB2 Recovery Expert for z/OS WEB client

  This component is a web-based simple, easy-to-use GUI with menus that make recoveries to a point in time current, quick, and precise.

- DB2 Recovery Expert for z/OS ISPF Interface

  An easy-to-use ISPF interface is provided to assist you with setting the options used when creating and using SLBs managed by DB2 Recovery Expert for z/OS.

Product important data sets:

Control file	The name of the VSAM data set that contains the control information for DB2 Recovery Expert. This KSDS VSAM file contains product customization information, including DB2 specific information such as plan names. After installation, the control file can be further modified by using option 0 from the primary selection menu. This can be the same control file used in other DB2 tools products.
BREPORT	The name of the VSAM repository data set that contains information about system backup reports.
OBJECTS	The name of the VSAM repository data set that contains objects in an Object Profile.
OFFOPTS	The name of the VSAM repository data set that contains information about system backup offload options.
PROFILE.CATS	The name of the VSAM repository data set that contains information about a DB2 system's user catalogs.
PROFILE.MAPS	The name of the VSAM repository data set that contains information about profile volume mappings.
PROFILES	The name of the VSAM repository data set that contains information about profiles.
RBADATA	The name of the VSAM repository data set that contains information stored by the relative byte address (RBA) Capture Utility.
SYSBACK	The name of the VSAM repository data set that contains information about system backups.
SYSBACK.OBJS	The name of the VSAM repository data set that contains information about objects that were backed up.
SYSBACK.SSIDS	The name of the VSAM repository data set that contains information about system backup system IDs.
SYSBACK.VOLS	The name of the VSAM repository data set that contains information about system backup volumes.
SYSCONFG	The name of the VSAM repository data set that contains information about subsystem configuration.

ISPF and web interface

Although most features of DB2 Recovery Expert can be used through both interfaces, they have some differences in the way that you work with them. The ISPF interface works based on backup and restore profiles when the web GUI is based on recovery advisors.

You can use the DB2 Recovery Expert ISPF interface to perform many of the backup and restore functions that are offered by DB2 Recovery Expert. The backup and restore functions include these tasks:

- Creating backup profiles
- Building and submitting backup jobs
- Creating Object Profiles
- Building and submitting object recovery jobs
- Restoring subsystems
- Creating image copies from SLBs
- Setting up disaster recovery jobs
- Setting up subsystems to work with backup and restore utilities
- Setting up the RBA Capture Utility
- Analyzing the log to find quiet times for selected objects

From the main window of the DB2 Recovery Expert web interface, you can recover an object; perform log analysis; perform a system restore; recover data and dropped objects using DB2 logs; and view previously saved recovery, system restore, and log analysis actions.

Table 8-1 Comparison between web interface and ISPF interface capabilities

Function	Option	Web	ISPF
Object recovery	– To current – To LRSN/RBA or Quiesce point – To Copy – To prior version with DDL	Y Y Y Y	Y Y Y Y
Database/table space object recovery	– To current – To log record sequence number (LRSN)/RBA or quiesce point – To copy – To prior version with DDL – From DB2 Recovery Expert created system-level backup	Y Y Y Y Y	Y Y Y Y Y
Dropped object recovery	– To current – To LRSN/RBA or quiesce point – To copy – To prior version with DDL – From DB2 Recovery Expert created system-level backup	Y Y Y Y Y	Y Y Y Y Y
Recovery of groups of objects	– Based on referential integrity groups – Based on IBM DB2 Automation Tool for z/OS Object Profiles – Based on DB2 Recovery Expert object recovery profiles	Y Y N	Y Y Y

Function	Option	Web	ISPF
Recovery of entire Subsystem	– From DB2 Recovery Expert created system-level backup - From DB2 created system-level backup - Automatically recover/rebuild objects in recovery-pending (RECP) or rebuild-pending (RDBP) status after restore	Y Y N	Y N Y
DB2 log analysis	Identify quiet points in the DB2 log that can be used as recovery points	Y	Y
Subsystem setup utility	Assist with preparing a DB2 subsystem for using the DB2 BACKUP SYSTEM utility	N	Y
Manage DB2 disaster recovery	Create JCL for jobs necessary to recover a DB2 subsystem at a remote location	N	Y
Create system-level backups	Generate JCL needed to create DB2 Recovery Expert managed system-level backups	N	Y
System backup health check	Assess the relative health of a system backup based on non-logged events occurring after the system backup	N	Y
Fast replication image copies	Create object level image copies using fast replication	N	Y
Create image copies from a system-level backup	Create JCL to create image copies for an object or a set of objects and register them in SYSCOPY	N	Y

Intelligent Recovery Manager

DB2 Recovery Expert has integrated, intelligent recovery and disaster recovery managers that analyze recovery assets and establish optimal recovery procedures to minimize recovery time and recovery point objectives. Recovery jobs are tailored specifically to available backup and hardware resources:

- The Intelligent Recovery Manager supplies the ability to perform local recoveries efficiently by using all available recovery resources. Restore operations that invoke fast-replication facilities through appropriate storage processor APIs and parallel recovery can significantly reduce recovery time and complexity.
- The Intelligent Disaster Recovery Manager uses local site procedures to prepare for offsite disaster recovery or disaster restart in advance. The information that is acquired allows Intelligent Disaster Recovery Manager to intelligently perform remote site restoration operations and appropriate recovery or restart procedures.

RBA Capture utility

The DB2 Recovery Expert RBA Capture utility records the current RBA of a DB2 subsystem at regular intervals based on the store clock time. This utility is optional. It uses a started task to capture the RBAs and clock times, and stores the data in its own repository that is separate from the backup and restore system repository.

8.1.3 DB2 system, object, and application recovery

DB2 Recovery Expert guides you through the process of restoring objects or complete systems from an SLB.

DB2 Recovery Expert automates DB2 system, application, or object recovery from a system backup. System recovery is performed by using the System Restore and Offload ISPF panels. Application or object recovery is performed by using the Object Profiles panel interface. Both menu options are available from the main ISPF panel.

There are two options to execute a system restore. Both of these options require that the SLB is performed through Recovery Expert:

Full It is similar to recover to copy. DB2 Recovery Expert restores both data and log volumes. No DB2 log apply recovery is performed on the restored volumes. A full system restore basically restores the DB2 system to the point in time that the system backup was taken. Any units of work that were in-flight at the time of the SLB are backed out.

Data-only It recovers only data from the system backup. The data-only option allows recovery to a selected point in time after the SLB was created. This is done by applying log records to the entire system after the data volumes have been restored.

When restoring a system by using data-only option, DB2 Recovery Expert uses the Intelligent Recovery Manager to determine which steps need to be taken to bring the entire system to the selected point in time.

The Intelligent Recovery Manager offers several options on the recovery plan to help you choose the best fit on a particular recovery procedure. The recovery processes include fast data set restores and running one or more of the following utilities:

- Recovery Utility
- Rebuild Index Utility
- SQL Undo/Redo
- DSN1COPY
- Unload and Load
- Check data
- DFSMSdss
- Check Index
- DDL and Data Manipulation Language (DML)

To recover a business object or a single object, DB2 Recovery Expert use the Object Profile option. A business object is a set of objects (table spaces and indexes) related to an application or a business. For instance, all objects related in a referential integrity rule make one business object. This information can be stored in the metadata repository, creating an Object Profile, and then used anytime that you need to recover this set of objects. Within the Object Profile, you can specify a recovery point and then DB2 Recovery Expert will present several options for how you want to recover the object:

- Restore from SLB and RECOVER LOGONLY
- RECOVER
- RECOVER with backout
- Restore of VSAM data set and RECOVER LOGONLY
- DSN1COPY and RECOVER LOGONLY
- RECOVER to image copy (IC) and redo SQL
- DSN1COPY of IC and redo SQL

IBM FlashCopy considerations

When using the FlashCopy blade, the DB2 data must reside on FlashCopy capable storage subsystems and the DB2 system backup profile must define identically sized source and target volumes. In addition, the source and target volumes must both be located in the same storage subsystem. Users can specify target volume ranges or target storage management subsystem (SMS) groups so there is no need to update a backup profile when the DB2 subsystem expands to new volumes.

FlashCopy commands issued through the FlashCopy blade interface back up all the DB2 system volumes. The FlashCopy blade uses volume-based copy services to create the DB2 SLB. It uses volume or data set copy services to restore DB2 systems, an application, or objects. Application or object recovery will be performed through DFSMSdss. DFSMSdss will use fast replication if possible and will use host-based I/O (slow copy) if the FlashCopy background copy process is not available.

There are options to either use FlashCopy consistency or issue a DB2 log suspend. It is better to use FlashCopy consistency rather than log suspend. The backup window will be shorter and updates to DB2 will be suspended for a shorter period of time.

The IBM FlashCopy blade provides DB2 Recovery Expert interface support for IBM FlashCopy. The FlashCopy blade uses the native IBM ANTRQST macro interface to invoke FlashCopy.

When using the FlashCopy blade, the backup target volumes can be kept online or offline, depending on how you specify your target volumes:

- If target UNITs are specified, Recovery Expert performs all the necessary commands to bring the volumes online when they are needed to copy the volumes to tape or to perform application recovery through DFSMSdss or FDR. When DB2 Recovery Expert needs to bring a backup volume online temporarily, it will relabel the backup volume, vary it online, and then read the data from the backup volume. When the backup volume is no longer needed, DB2 Recovery Expert will vary the volume offline and then relabel it to the original volume serial.
- If target SMS storage groups are specified, DB2 Recovery Expert will keep the target volumes online during the entire process.

  Performance: The second option performs better than the first option.

- System level backup

  We discuss the SLB requirements and how DB2 Recovery Expert can help you verify your configuration to ensure that it is recoverable. Then, we show how to alter your subsystem with DB2 Recovery Expert to make it compliant with these requirements. The scenario also demonstrates how to create DB2 and DB2 Recovery Expert SLBs and some additional features provided by DB2 Recovery Expert.

- Subsystem restore

  We show how the system level restores can be executed in DB2 Recovery Expert and additional functionality to make the DBA's life easier.

- Dropped object restore

  We show how we can automate a recovery of a dropped database schema that includes a database with multiple tables, a set of stored procedures, views, and triggers.

8.1.4 System level backup

To ensure the most complete and accurate subsystem restoration, the DB2 subsystem should follow these recommendations.

The user catalogs for the DB2 log and boot strap data sets should be separate from the user catalogs for DB2 object data (table spaces and indexes) and reside on the same volumes as any other DB2 object data or DB2 object data catalogs.

The DB2 boot strap, active log, and archive log data sets should reside on separate volumes from the DB2 object data.

Start state

Subsystem DB0C is used as a base for this scenario. It has been set up so that data, logs, archivists, and bootstrap data sets (BSDSs) are under one high-level qualifier (HLQ) and in one integrated catalog facility (ICF) catalog - UCAT.DN0CD. All of the data sets are placed in the DB0CDATA storage group, which is part of the DSN$DB0C$DB copypool. The ICF catalog is placed into the DB0CMISC storage group, which is not part of either the data copypool or the log copypool.

We start by analyzing DB0C to see if it is possible to use SLBs. For that analysis, we choose option 5. DB2 Subsystem Analysis and Configuration on the IBM DB2 Recovery Expert for z/OS main menu. See Figure 8-2.

```
RCVYXPRT V3R1 ------- IBM DB2 Recovery Expert for z/OS                  -------
Option  ===> 5

                                                        2013/02/27  14:38:35
                                                        User: ADMR4   - ARY

--------------------------------------------------------------------------------

     0. User Settings

     1. System Backup Profiles

     2. System Restore and Offload

     3. Object Profiles

     4. Disaster Recovery Profiles

     5. DB2 Subsystem Analysis and Configuration

     6. Coordinated Application Profiles

     X. Exit
```

Figure 8-2 Recovery Expert main menu

The next step is to choose the subsystem that you want to back up and the backup method. DB2 Recovery Expert supports both DB2 backup system as well as low-level fast replication technologies from IBM and other vendors. We select D for DB2 system level backup and DB0C as the subsystem. See Figure 8-3 on page 327.

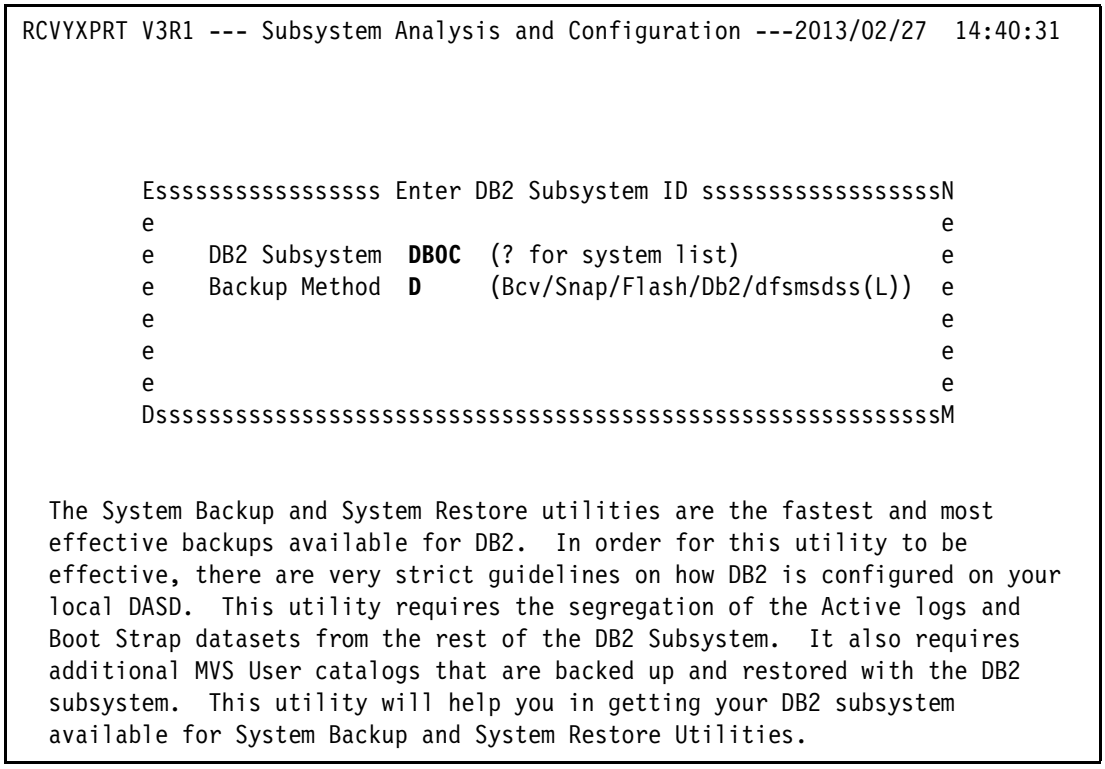

Figure 8-3 Backup method and subsystem selection

DB2 Recovery Expert will analyze the subsystem for SLB with the specified method and present the results. DB2 Recovery Expert pulls information from various sources (DB2 catalog, DSNZPARMs, ICF catalogs, DFSMS, and so on) and checks whether the current system layout can be backed up by using the specified method. The analysis results are shown on Figure 8-4 on page 328 and Figure 8-5 on page 329.

On Figure 8-4 on page 328, we can see that our subsystem is using one ICF catalog UCAT.DB0CD for logs, BSDSs, and data. This approach is in violation of DB2 SLB requirements, as BACKUP system will back up both logs and data and the only possible restore point in time is the backup time. So, the overall analysis result is "`Subsystem configuration prevents system level backup`".

The Analysis panel also contains useful information about the system layout. We can see the copypool information and have an option to list the volumes in every pool. Backup pools are also shown and again there is an option to look at volumes that belong to that pool.

DB2 Recovery Expert also presents a list of all volumes used by the subsystem. Each volume is color-coded. The colors have the following meaning:

- Dark blue - Volume is optimal.
- Light blue - The volume contains data other than DB2 data.
- Pink - Both log and object data reside on the volume.
- Red - The volume cannot be backed up by DB2 Recovery Expert.

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/27   14:42:21
Option ===> _____________________________________________________ Scroll ===> CSR

Commands: ANALYZE  REANALYZE

--------------------------------------------------------------------------------
 Subsystem: DB0C     Active: Yes     Datasharing: No
 Date of Last Analysis: 02/27/2013   Analysis Recommended: N
 Message: Subsystem configuration prevents system level backup.
-------------------------------------------------           Row 1 of 56     +
  New MVS User Catalogs to be used by this subsystem
_    Log/BSDS Catl  ____________________________________________ Volume  ______
_    DB2 Data Catl  ____________________________________________ Volume  ______
     Line Cmds: (C-Create, A-Add Alias, D-Dataset Disp, U-Update, V-View Alias)

  Existing MVS User Catalogs used by this subsystem
_    Data Log Other UCAT.DB0CD                                   Volume  SBOXJS
     Line Cmds: (D-Dataset Display, V-View Aliases)

  Storage Copy Pools
_    Data Copy Poo  DSN$DB0C$DB
_    Log Copy Pool  DSN$DB0C$LG
_    Backup Pool    DB0CCP8
     Line Cmds: (V-Volume List)

_ Boot Strap Datasets
_    DB0C - BSDS 1  DB0CD.BSDS01                                 Volume  X66526
_    DB0C - BSDS 2  DB0CD.BSDS02                                 Volume  X66226
     Line Cmds: (R-Rename BSDS, M-Move BSDS)

_ Active Log Datasets
_    DB0C - Log 1   DB0CD.LOGCOPY1.DS01                          Volume  SBOXJN
_    DB0C - Log 1   DB0CD.LOGCOPY1.DS02                          Volume  X66125
_    DB0C - Log 1   DB0CD.LOGCOPY1.DS03                          Volume  X66329
_    DB0C - Log 2   DB0CD.LOGCOPY2.DS01                          Volume  SBOXJL
_    DB0C - Log 2   DB0CD.LOGCOPY2.DS02                          Volume  SBOXJM
_    DB0C - Log 2   DB0CD.LOGCOPY2.DS03                          Volume  X6652A
     Line Cmds: (R-Rename Log, M-Move Log)

  Alias used with associated MVS User Catalogs
_    DB0CD          UCAT.DB0CD                                   Data Log Other
```

Figure 8-4 Analysis results (Page 1 of 2)

The SBOXJS volume is colored red because it contains DB2 subsystem-related data, but it is not included in the copypool definition.

We can browse the data sets on a volume by typing D next to the volume. The contents of SBOXJS are shown on Figure 8-6 on page 331. We can see that the ICF catalog is colored blue as it is used by the subsystem. The rest of the data sets are colored light blue, which means that they are not part of the subsystem, but will be backed up as a parameter of the SLB.

The DB2 ICF catalog is outside of the copypools, which also prevents DB2 SLB. Remember, one of the SLB requirements is to have ICF catalogs co-located with the logs and data. If ICF catalogs are not backed up and restored along with the DB2 data and logs, there will be inconsistencies between DB2 and the catalog information. For example, if you reorganize a table with SHRLEVEL REFERENCE or CHANGE, the resulting data set can be on a different volume. And if DB2 is restored to a point in time prior to reorganization, it will not be able to find the object as the catalog record will point to a wrong volume. This is only one example.

Many other unpleasant surprises can happen if ICF catalogs are not co-located with the logs and data.

Recovery Expert can also optimize the storage utilization by your subsystem. As you can see on Figure 8-5, there is a line command to move all data sets from the volume. This can be very useful in a situation where there are many volumes with low utilization. You can consolidate the data on these volumes to fewer volumes and save on storage. Now, saving a couple of disks might not seem significant, but when you consider that you also save the same number of disks for every version of the backup, it starts to make much more sense. A usage example of this feature is not included in this book, but it is similar to log or bootstrap data set (BSDS) moves that are shown later in this chapter. Recovery Expert generates all of the required IDCAMS commands to move the data sets and builds a JCL job that you can edit and submit.

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/27   14:43:41
Option ===> ____________________________________________________ Scroll ===> PAGE

Commands: ANALYZE  REANALYZE

--------------------------------------------------------------------------------
 Subsystem: DB0C     Active: Yes     Datasharing: No
 Date of Last Analysis: 02/27/2013   Analysis Recommended: N
 Message: Subsystem configuration prevents system level backup.
--------------------------------------------------------          Row 32 of 56    -
_    DB0CX          UCAT.DB0CD                                      Data
     Line Cmds: (D-Dataset Display, M-Merge catalog entries, R-Rename Alias)

  Volumes used by this subsystem
     Volume         Data   DataCat   ActLog   ActCat   ArcLog   ArcCat   Other   Pool
_    -NONE-         Yes    No        No       No       Yes      No       No      N/A
_    SBOXJK         Yes    No        No       No       No       No       Yes     Yes
_    SBOXJL         Yes    No        Yes      No       No       No       Yes     Yes
_    SBOXJM         Yes    No        Yes      No       No       No       Yes     Yes
_    SBOXJN         Yes    No        Yes      No       No       No       Yes     Yes
D    SBOXJS         No     Yes       No       Yes      No       No       Yes     No
_    X66125         Yes    No        Yes      No       No       No       Yes     Yes
_    X66226         Yes    No        Yes      No       No       No       No      Yes
_    X66326         Yes    No        No       No       No       No       No      Yes
_    X66327         Yes    No        No       No       No       No       Yes     Yes
_    X66328         Yes    No        No       No       No       No       No      Yes
_    X66329         Yes    No        Yes      No       No       No       Yes     Yes
_    X66526         Yes    No        Yes      No       No       No       No      Yes
_    X6652A         Yes    No        Yes      No       No       No       No      Yes
_    X66628         Yes    No        No       No       No       No       No      Yes
_    X66727         Yes    No        No       No       No       No       Yes     Yes
_    XBD01D         Yes    No        No       No       No       No       No      Yes
_    XBD11C         Yes    No        No       No       No       No       No      Yes

     Line Cmds: (D-Dataset Display, M-Move all Datasets on Volume)

******************************** Bottom of Data ********************************
```

Figure 8-5 Analysis results (Page 2 of 2)

Note: As you can see on Figure 8-5 on page 329, there is a volume called -NONE-. There are three reasons why a data set would show under NONE:

- The objects are listed in the DB2 catalog but the underlying VSAM data set was deleted.
- The table space was created as DEFINE NO and the underlying VSAM data sets have not been created yet.
- The BSDS contains archive log data sets that no longer exist.

So according to DB2 Recovery Expert, two serious issues are preventing us from taking a consistent and recoverable SLB. Now, we try to run a DB2 SLB using the BACKUP SYSTEM utility. The JCL job listing is in Figure 8-7 on page 331. The job output is in Example 8-1 on page 332.

The DB2 BACKUP SYSTEM utility's ability to check various conditions that prevent recoverable SLB execution is limited. The utility issues DFSMShsm commands to back up the copypools. The backup was executed successfully and returned 0.

```
RCVYXPRT V3R1   ------ Volume Dataset List Display ------ 2013/02/27   14:45:39
Option ===> _____________________________________________________ Scroll ===> PAGE

Command: Newvol

-------------------------------------------------------------------------------
 Subsystem: DB0C        Volume Serial: SBOXJS
 Free Trks: 146,646     DB2 Data Trks:    300      DB2 Log Trks:    300
-------------------------------------------------           Row 1 of 30
Cmd Dataset Name                                    Tracks Volume  DB2 Logs Data
    UCAT.DB0CD                                         300         Yes Yes  Yes
    UCAT.DB0CD.CATINDEX                                  2         Yes Yes  Yes
    DB0CM.COPYDDN.DSN8D10P.DSN8S10C                      3 ______  No  No   No
    DB0CM.DBRMLIB.DATA                                   9 ______  No  No   No
    DB0CM.DB0CWLMJ.JAVAENV                               1 ______  No  No   No
    DB0CM.DSN8.LISTDEF                                   1 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.PART3                        2 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.P00001                      15 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.P00003                      15 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.P00004                      15 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.REORGCPY                    15 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.REORGDSC                    15 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.REORGPUN                    15 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.SYSPUNCH                    15 ______  No  No   No
    DB0CM.NEW.SDSNCLST                                  76 ______  No  No   No
    DB0CM.NEW.SDSNSAMP                                 401 ______  No  No   No
    DB0CM.NEW.SDSNSAMP.BACKUP                          402 ______  No  No   No
    DB0CM.NEW.SDSNTEMP                                 101 ______  No  No   No
    DB0CM.PROCLIB                                       15 ______  No  No   No
    DB0CM.RUNLIB.LOAD                                   35 ______  No  No   No
    DB0CM.SDSNMACS.CLONE                               151 ______  No  No   No
    DB0CM.SRCLIB.DATA                                   14 ______  No  No   No
    DB0CM.SYSCOPY.DSN8D10A.DSN8S10D                     15 ______  No  No   No
    DB0CM.SYSCOPY.DSN8D10A.DSN8S10E                     15 ______  No  No   No
    DB0CM.SYSCOPY.DSN8D10A.DSN8S10P                     15 ______  No  No   No
    DB0CM.SYSCOPY.DSN8D10P.DSN8S10C                     15 ______  No  No   No
    SYS1.VTOCIX.SBOXJS                                  30 ______  No  No   No
    SYS1.VVDS.VSBOXJS                                   10 ______  No  No   No
    UCAT.DB0CMISC                                     1500 ______  No  No   No
    UCAT.DBOCMISC.CATINDEX                             300 ______  No  No   No

****************************** Bottom of Data ********************************
```

Figure 8-6 List of data sets on volume SBOXJS

```
//BACKUPO JOB ADMR4,CLASS=A,NOTIFY=&SYSUID
/*JOBPARM S=SC63
//STEP1    EXEC DSNUPROC,TIME=1440,
//         UTPROC='',
//         SYSTEM='DB0C'
//STEPLIB  DD DSN=DB0CT.SDSNEXIT,DISP=SHR
//         DD DSN=DB0CT.SDSNLOAD,DISP=SHR
//SYSIN    DD *
  BACKUP  SYSTEM
/*
```

Figure 8-7 BACKUP SYSTEM job

Example 8-1 BACKUP SYSTEM job output

```
ABPU5001I 058 14:51:27.74 IBM DB2 Utilities Enhancement Tool Version 0220, FMID=H2AM220,
COMP_ID=5655-T58
 ABPU5012I 058 14:51:27.74 Connected to started task ABPID=ABP1
 ABPG8008I 058 14:51:27.76 System=SC63    ,Job=BACKUP0 ,Job
Id=JOB18214,Step=DSNUPROC,Program=DSNUTILB,User=ADMR4
 ABPU5002I 058 14:51:27.77 Initialization is complete.


-------------------------------------------------------------------------------------------

 ABPU5004I 058 14:51:29.28 Analysis started. Step=1
 ABPU5005I 058 14:51:29.28 Analysis completed. RC=0
 ABPU5008I 058 14:51:29.28 Utility execution started. Step=1
1DSNU000I    058 14:51:29.35 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = ADMR4.BACKUP0
 DSNU1044I   058 14:51:29.37 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
0DSNU050I    058 14:51:29.38 DSNUGUTC -  BACKUP SYSTEM
 DSNU1600I   058 14:51:29.38 DSNUVBBD - BACKUP SYSTEM UTILITY FOR DATA STARTING,
                         COPYPOOL = DSN$DB0C$DB
                         TOKEN = X'C4C2F0C3CAFCA63DFB43D32B001916D0F6D5'.
 DSNU1614I   058 14:51:52.22 DSNUVBBD - BACKUP SYSTEM UTILITY FOR DATA COMPLETED SUCCESSFULLY,
                         COPYPOOL = DSN$DB0C$DB
                         TOKEN = X'C4C2F0C3CAFCA63DFB43D32B001916D0F6D5'
                         DATA COMPLETE LRSN = X'001916D17C46'
                         ELAPSED TIME = 00:00:22.
 DSNU1600I   058 14:51:52.22 DSNUVBBD - BACKUP SYSTEM UTILITY FOR LOGS STARTING,
                         COPYPOOL = DSN$DB0C$LG
                         TOKEN = X'C4C2F0C3CAFCA63DFB43D32B001916D0F6D5'.
 DSNU1614I   058 14:51:53.04 DSNUVBBD - BACKUP SYSTEM UTILITY FOR LOGS COMPLETED SUCCESSFULLY,
                         COPYPOOL = DSN$DB0C$LG
                         TOKEN = X'C4C2F0C3CAFCA63DFB43D32B001916D0F6D5'
                         DATA COMPLETE LRSN = X'001916D17C46'
                         ELAPSED TIME = 00:00:00.
 DSNU1602I   058 14:51:53.35 DSNUVBBD - BACKUP SYSTEM UTILITY COMPLETED, ELAPSED TIME =
00:00:23.
 DSNU010I    058 14:51:53.36 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
 ABPU5009I 058 14:51:53.98 Utility execution completed. SYS=0000, USR=0000
 ABPU5003I 058 14:51:54.14 DB2 Utilities Enhancement Tool intercept completed.
```

Now, we try to execute the same DB2 SLB, but we let Recovery Expert drive it. SLBs in Recovery Expert can only be executed from the ISPF interface. We start by selecting option 1. System Backup Profiles from the main menu on Figure 8-8 on page 333.

```
RCVYXPRT V3R1 ------- IBM DB2 Recovery Expert for z/OS                  -------
Option  ===> 1

                                                        2013/02/27  14:54:48
                                                        User: ADMR4   - ARY

--------------------------------------------------------------------------------

     0. User Settings

     1. System Backup Profiles

     2. System Restore and Offload

     3. Object Profiles

     4. Disaster Recovery Profiles

     5. DB2 Subsystem Analysis and Configuration

     6. Coordinated Application Profiles

     X. Exit
```

Figure 8-8 Recovery Expert main menu

There are no backup profiles on our test system, so Recovery Expert suggests that we create a backup profile. On Figure 8-9, C is correct, so we press Enter.

```
 RCVYXPRT V3R1    -------- Backup Profile Display -------- 2013/02/27  14:55:44
 Option  ===>                                                Scroll ===> PAGE

 Line Commands: B - Build  U - Update  C - Create  V - View  D - Delete
                R - Rename


--------------------------------------------------------------------------------
  Profile Like  *                                    SSID Like  DBOC
  Creator Like  *                                             Row 1 of 1    >

--------------------------------------------------------------------------------

 Cmd  Name                           Creator   SSID  Updt
  C   Press Enter to Create Profile
 ****************************** Bottom of Data ********************************



EssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssN
e ARYRO31I - No Profiles were found that match your selection criteria. Press e
e enter to create a new profile or change the selection criteria.            e
DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 8-9 Create a new SLB profile

Backup profile settings are shown on Figure 8-10. Recovery Expert can drive multiple types of SLBs. It uses low-level storage APIs to drive fast replication technologies from different vendors. It can also drive DB2 SLBs by utilizing BACKUP SYSTEM, which is what we want to use, so we choose D for Backup Method. The value M for Source/Target Mapping is the only valid option for DB2 SLBs, so we leave it. We explain the definitions of the other values for this parameter later. We also choose U for the Update Option field to let other users change our profile, if needed.

```
RCVYXPRT V3R1    -------- Backup Profile Display -------- 2013/02/27  14:59:29
Option  ===>                                                Scroll ===> PAGE

Line Commands: B - Build  U - Update  C - Create  V - View  D - Delete
               R - Rename

------------------------------------------------------------------------------
 Profile Like  *                                    SSID Like  DBOC
 Creator Like  *                                             Row 1 of 1    >
------- Esssssssssssss Enter New Backup Profile Options sssssssssssssN -------
        e                                                              e
Cmd  Na e    Creator          ADMR4                                    e
 C   Pr e                                                              e
******* e    Profile Name     DB2 SLB                                  e *******
        e                                                              e
        e    Description      BACKUP SYSTEM                            e
        e                                                              e
        e    DB2 SSID         DBOC  (? for system list)                 e
        e                                                              e
        e    Backup Method    D  (Bcv/Snap/Flash/Db2/dfsmsdss(L))      e
        e                                                              e
        e    Source/          M  (Auto discover/pool mapping,          e
        e    Target Mapping       Stogroup discover/pool mapping,     e
        e                         Manual)                              e
        e                                                              e
        e    Update Option    U  (Update, View only, No access)        e
        e                                                              e
        e                                                              e
        e                                                              e
        e                                                              e
        DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 8-10 Create backup profile

We also enter a profile name and a meaningful description and press Enter to close the panel and create the profile. The Update Backup Profile panel is shown on Figure 8-11 on page 335.

As this is the first time we run the profile, Recovery Expert flags this profile as “Setup Needed”. Profile setup is a validation process performed by DB2 Recovery Expert before a backup of a subsystem can be taken. This process authenticates the volumes for the subsystem, checks the locations of the user data, logs, and user catalogs, and performs other validations to ensure that the backup can proceed and that the resulting backup will be usable. For the full list of validations performed by Recovery Expert, see Chapter 7 of *IBM DB2 Recovery Expert for z/OS Version 3 Release 1 User’s Guide*, SC19-3687.

A backup profile must successfully complete the profile setup before an SLB can be generated. Once a backup profile has been set up, it does not need to be set up again unless changes are made to the source or target volume configuration or unless DB2 Recovery Expert detects certain errors while building a backup job.

The “Issue Log Suspend” field is also set to N and is not editable because we are using the BACKUP SYSTEM utility, which can provide consistency without issuing SET LOG SUSPEND or using storage level consistency.

```
RCVYXPRT  V3R1 ---------- Update Backup Profile --------- 2013/02/27  15:00:17
Option  ===>                                                Scroll ===> PAGE

     Commands: ? - Show all commands
Line Commands: I - Insert  D - Delete  X - Exclude  U - Undo from exclude

------------------------------------------------------------------------------
 Creator: ADMR4      Name: DB2 SLB                          SSID: DBOC
 Share Option: U  (Upd,View,No)    Description: BACKUP SYSTEM
------------------------------- Backup Options -------------------------------
Backup Method     ==> D  (B/S/F/D/L)          Current Generation==> 00
Backup Scope      ==> F  (Full/Data)          Setup Needed      ==> Y
Backup Generations==> 04 (00 - 99)            Issue Log Suspend ==> N  (Yes/No)
Offload Options   ==> N  (Yes/No/Update)      Validate DB2 Vols ==> Y  (Yes/No)
                                              Enable Obj Restore==> Y  (Yes/No)
------------------ Volume Mappings -----------------         Row 1 of 23
     Source   Dev     Src   Target
Cmd  Volumes  Type    Unit  Volumes   Message Area
     SBOXJI   3390-7  611B  X66127
     SBOXJJ   3390-7  621F  X5642D
     SBOXJK   3390-9  D83B  SBOXKM
     SBOXJL   3390-9  D90F  SBOXKK
     SBOXJM   3390-9  DA0F  SBOXKN
     SBOXJN   3390-9  DB36  SBOXKL
     SBOXJQ   3390-9  D83C  XBD070
     SBOXJR   3390-9  DA3A  X5DA0C
     SBOXJU   3390-7  661D  X5662C
     SBOXJV   3390-7  671D  X66128
     X66125   3390-7  6125  SBOXKI
     X66226   3390-7  6226  SBOXKJ
     X66326   3390-7  6326  X5632C
     X66327   3390-7  6327  X5632D
     X66328   3390-7  6328  X5652C
     X66329   3390-7  6329  X5652D
     X66526   3390-7  6526  X5672D
     X6652A   3390-7  652A  X5672C
     X66628   3390-7  6628  X5642C
     X66727   3390-7  6727  X66126
     X76029   3390-7  6029  X5662D
     XBD01D   3390-9  D01D  SBOXK0
     XBD11C   3390-9  D11C  SBOXKP
***************************** Bottom of Data ********************************
```

Figure 8-11 Update Backup Profile panel

8.2 System level backup scenario

We describe the requirements for SLBs in DB2 and walk through several cases of SLB execution using various methods provided by Recovery Expert. We also look at additional functionality in Recovery Expert that allows us to check whether the subsystem can be backed up and, if not, to make configuration changes to comply with the requirements.

System level backup requirements

To ensure the most complete and accurate subsystem restore, the DB2 subsystem should follow these recommendations.

The user catalogs for the DB2 log and boot strap data sets should be separate from the user catalogs for the DB2 object data (table spaces and indexes) and reside on the same volumes as any other DB2 object data or DB2 object data catalogs.

The DB2 bootstrap, active log, and archive log data sets should reside on separate volumes from the DB2 object data.

Subsystem analysis

The DB0C subsystem will be used as a base for this scenario. It has been set up so that data, logs, archive logs, and BSDSs are under one HLQ and in one ICF catalog - UCAT.DN0CD. All of the data sets are placed in the DB0CDATA storage group, which is part of the DSN$DB0C$DB copypool. The ICF catalog is placed into the DB0CMISC storage group, which is not part of either the data copypool or log copypool.

We start by analyzing DB0C to see whether it is possible to use SLBs. For that analysis, we choose option 5. DB2 Subsystem Analysis and Configuration in the Recovery Expert main menu. See Figure 8-12.

```
RCVYXPRT V3R1 ------- IBM DB2 Recovery Expert for z/OS                  -------
Option  ===> 5

                                                        2013/02/27  14:38:35
                                                        User: ADMR4   - ARY

--------------------------------------------------------------------------------

     0. User Settings

     1. System Backup Profiles

     2. System Restore and Offload

     3. Object Profiles

     4. Disaster Recovery Profiles

     5. DB2 Subsystem Analysis and Configuration

     6. Coordinated Application Profiles

     X. Exit
```

Figure 8-12 Recovery Expert main menu

The next step is to choose the subsystem that you want to back up and the backup method. Recovery Expert supports both the DB2 backup system as well as low-level fast replication technologies from IBM and other vendors. We select D for DB2 system level backup and DB0C as the subsystem. See Figure 8-13.

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/27  14:40:31




        Essssssssssssssss Enter DB2 Subsystem ID sssssssssssssssssN
        e                                                         e
        e    DB2 Subsystem  DB0C  (? for system list)             e
        e    Backup Method  D     (Bcv/Snap/Flash/Db2/dfsmsdss(L)) e
        e                                                         e
        e                                                         e
        e                                                         e
        DssssssssssssssssssssssssssssssssssssssssssssssssssssssssM


  The System Backup and System Restore utilities are the fastest and most
  effective backups available for DB2.  In order for this utility to be
  effective, there are very strict guidelines on how DB2 is configured on your
  local DASD.  This utility requires the segregation of the Active logs and
  Boot Strap datasets from the rest of the DB2 Subsystem.  It also requires
  additional MVS User catalogs that are backed up and restored with the DB2
  subsystem.  This utility will help you in getting your DB2 subsystem
  available for System Backup and System Restore Utilities.
```

Figure 8-13 Backup method and subsystem selection

Recovery Expert will analyze the subsystem for the possibility of SLB using the specified backup method and present the results. Recovery Expert pulls information from various sources (DB2 catalog, DSNZPARMs, ICF catalogs, DFSMS, and so on) and checks whether the current system layout can be backed up by using the specified method. The analysis results are shown on Figure 8-14 on page 338 and Figure 8-15 on page 339.

On Figure 8-14 on page 338, we can see that our subsystem is using one ICF catalog UCAT.DB0CD for logs, BSDSs, and data. This is in violation of DB2 SLB requirements, as BACKUP SYSTEM will back up both logs and data and the only possible restore point in time will be the backup time. So, the overall analysis result is "`Subsystem configuration prevents SLB`".

The Subsystem Analysis and Configuration panel also contains useful information about the system layout. We can see the copypool information and can optionally list the volumes in every pool. Backup pools are also shown and again there is an option to look at member volumes.

Recovery Expert also presents a list of all volumes used by the subsystem. Each volume is color-coded. The colors have the following meaning:

- Dark blue - Volume is optimal.
- Light blue - The volume contains data other than DB2 data.
- Pink - Both log and object data reside on the volume.
- Red - The volume cannot be backed up by DB2 Recovery Expert.

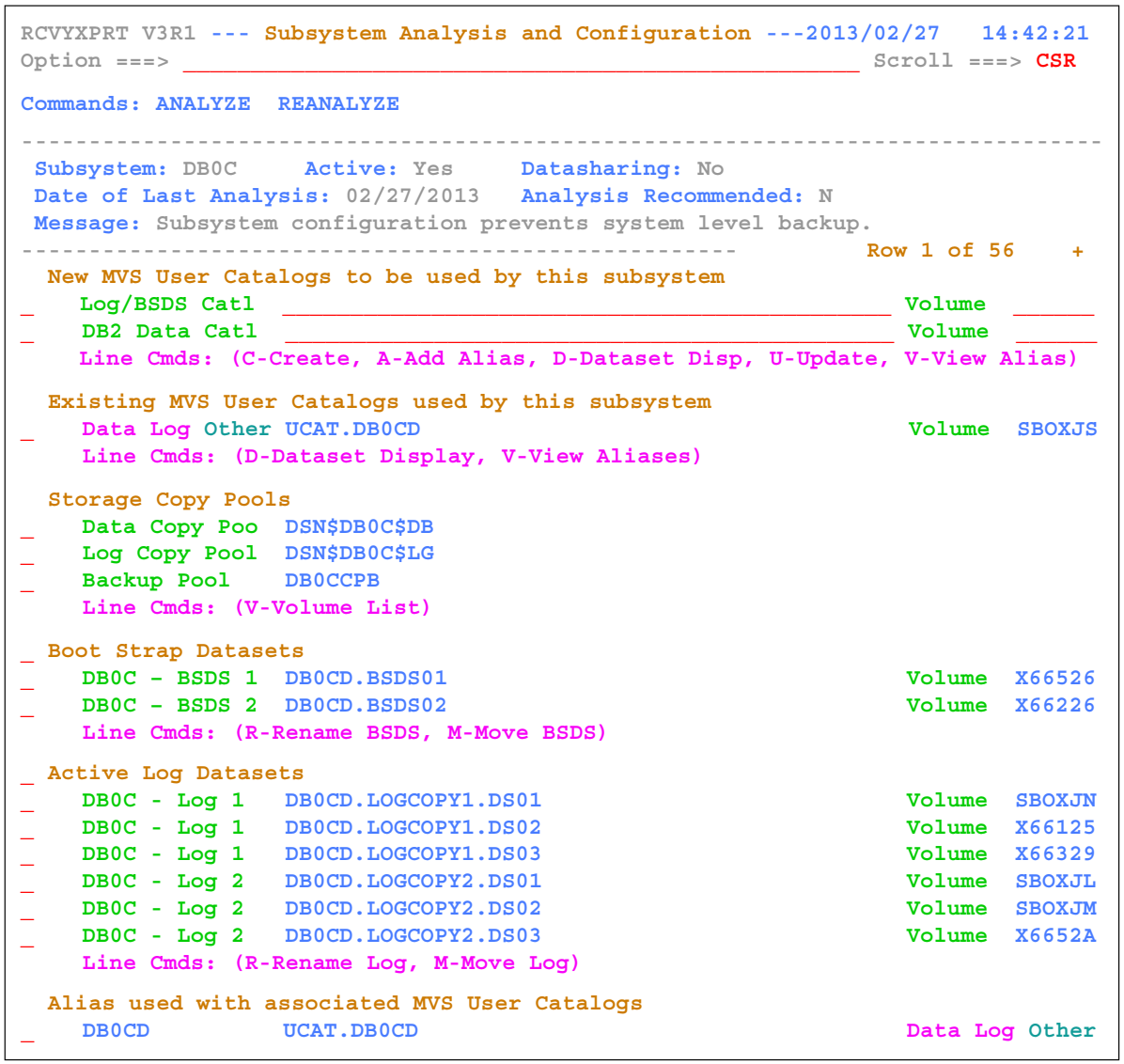

Figure 8-14 Analysis results (Page 1 of 2)

The SBOXJS volume is colored red because it contains the ICF catalog used by DB0C, but it is not included into the copypool definitions.

We can browse the data sets on a volume by typing D next to the volume name. The contents of SBOXJS are shown on Figure 8-16 on page 341. We can see that the ICF catalog is colored blue as it is used by the subsystem. The rest of the data sets are colored light blue, which means that they are not part of the subsystem, but they will be backed up as part of the SLB.

The DB2 ICF catalog is outside of the copypools, which also prevents DB2 SLB. Remember, one of the SLB requirements is to have ICF catalogs co-located with the logs and data. If ICF catalogs are not backed up and restored along with the DB2 data and logs, there will be inconsistencies between DB2 and the catalog information. For example, if you reorganize a table with SHRLEVEL REFERENCE or CHANGE, the resulting data set can be on a different volume. And, if DB2 is restored to a point in time prior to reorganization, it will not be able to find the object as the catalog record will point to a wrong volume.

This is only one example, many other unpleasant surprises can happen if ICF catalogs are not co-located with the logs and data. The ICF catalog is also required on the backed-up copypools in order to enable object level recovery.

Recovery Expert can also optimize the storage utilization by your subsystem. As you can see from Figure 8-15, there is a line command to move all data sets from the volume. This can be very useful in a situation when there are many volumes with low utilization. You can consolidate the data on fewer volumes and save on storage. Now, saving a couple of disks might not seem significant, but when you consider that you also save the same number of disks for every version of the backup that is configured for the copypool, it starts to make much more sense. A usage example of this feature is not included in this book, but it is similar to log or bootstrap data set moves that are shown later. Recovery Expert generates all of the required IDCAMS commands to move the data sets and builds a JCL job that you can edit and submit.

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/27   14:43:41
Option ===> ____________________________________________________ Scroll ===> PAGE

Commands: ANALYZE  REANALYZE

--------------------------------------------------------------------------------
 Subsystem: DB0C     Active: Yes     Datasharing: No
 Date of Last Analysis: 02/27/2013   Analysis Recommended: N
 Message: Subsystem configuration prevents system level backup.
-------------------------------------------------          Row 32 of 56    -
_    DB0CX          UCAT.DB0CD                                 Data
     Line Cmds: (D-Dataset Display, M-Merge catalog entries, R-Rename Alias)

  Volumes used by this subsystem
     Volume         Data   DataCat   ActLog   ActCat   ArcLog   ArcCat   Other   Pool
_    -NONE-         Yes    No        No       No       Yes      No       No      N/A
_    SBOXJK         Yes    No        No       No       No       No       Yes     Yes
_    SBOXJL         Yes    No        Yes      No       No       No       Yes     Yes
_    SBOXJM         Yes    No        Yes      No       No       No       Yes     Yes
_    SBOXJN         Yes    No        Yes      No       No       No       Yes     Yes
D    SBOXJS         No     Yes       No       Yes      No       No       Yes     No
_    X66125         Yes    No        Yes      No       No       No       Yes     Yes
_    X66226         Yes    No        Yes      No       No       No       No      Yes
_    X66326         Yes    No        No       No       No       No       No      Yes
_    X66327         Yes    No        No       No       No       No       Yes     Yes
_    X66328         Yes    No        No       No       No       No       No      Yes
_    X66329         Yes    No        Yes      No       No       No       Yes     Yes
_    X66526         Yes    No        Yes      No       No       No       No      Yes
_    X6652A         Yes    No        Yes      No       No       No       No      Yes
_    X66628         Yes    No        No       No       No       No       No      Yes
_    X66727         Yes    No        No       No       No       No       Yes     Yes
_    XBD01D         Yes    No        No       No       No       No       No      Yes
_    XBD11C         Yes    No        No       No       No       No       No      Yes

     Line Cmds: (D-Dataset Display, M-Move all Datasets on Volume)

******************************** Bottom of Data ********************************
```

Figure 8-15 Analysis results (Page 2 of 2)

Note: As you can see on Figure 8-15, there is a volume called -NONE-. There are three reasons why a data set shows under NONE:

- The objects are listed in the DB2 catalog but the underlying VSAM data set was deleted.
- The table space was created as DEFINE NO and the underlying VSAM data sets have not been created yet.
- The bootstrap data set contains archive log data sets that no longer exist.

System level backup using BACKUP SYSTEM

So according to Recovery Expert, we have two serious issues that are preventing us from taking a consistent and recoverable SLB. Now, we try to run a DB2 SLB using BACKUP SYSTEM utility. The JCL job listing is shown on Figure 8-17 on page 341. The job output is shown in Example 8-2 on page 342.

```
RCVYXPRT V3R1   ------ Volume Dataset List Display ------ 2013/02/27   14:45:39
Option ===> _____________________________________________________ Scroll ===> PAGE

Command: Newvol

-------------------------------------------------------------------------------
 Subsystem: DB0C        Volume Serial: SBOXJS
 Free Trks: 146,646     DB2 Data Trks:    300      DB2 Log Trks:    300
-------------------------------------------------           Row 1 of 30
Cmd Dataset Name                                    Tracks Volume  DB2 Logs Data
    UCAT.DB0CD                                         300         Yes Yes  Yes
    UCAT.DB0CD.CATINDEX                                  2         Yes Yes  Yes
    DB0CM.COPYDDN.DSN8D10P.DSN8S10C                      3 ______  No  No   No
    DB0CM.DBRMLIB.DATA                                   9 ______  No  No   No
    DB0CM.DB0CWLMJ.JAVAENV                               1 ______  No  No   No
    DB0CM.DSN8.LISTDEF                                   1 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.PART3                        2 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.P00001                      15 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.P00003                      15 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.P00004                      15 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.REORGCPY                    15 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.REORGDSC                    15 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.REORGPUN                    15 ______  No  No   No
    DB0CM.DSN8D10A.DSN8S10E.SYSPUNCH                    15 ______  No  No   No
    DB0CM.NEW.SDSNCLST                                  76 ______  No  No   No
    DB0CM.NEW.SDSNSAMP                                 401 ______  No  No   No
    DB0CM.NEW.SDSNSAMP.BACKUP                          402 ______  No  No   No
    DB0CM.NEW.SDSNTEMP                                 101 ______  No  No   No
    DB0CM.PROCLIB                                       15 ______  No  No   No
    DB0CM.RUNLIB.LOAD                                   35 ______  No  No   No
    DB0CM.SDSNMACS.CLONE                               151 ______  No  No   No
    DB0CM.SRCLIB.DATA                                   14 ______  No  No   No
    DB0CM.SYSCOPY.DSN8D10A.DSN8S10D                     15 ______  No  No   No
    DB0CM.SYSCOPY.DSN8D10A.DSN8S10E                     15 ______  No  No   No
    DB0CM.SYSCOPY.DSN8D10A.DSN8S10P                     15 ______  No  No   No
    DB0CM.SYSCOPY.DSN8D10P.DSN8S10C                     15 ______  No  No   No
    SYS1.VTOCIX.SBOXJS                                  30 ______  No  No   No
    SYS1.VVDS.VSBOXJS                                   10 ______  No  No   No
    UCAT.DB0CMISC                                     1500 ______  No  No   No
    UCAT.DBOCMISC.CATINDEX                             300 ______  No  No   No

****************************** Bottom of Data ********************************
```

Figure 8-16 List of data sets on volume SBOXJS

```
//BACKUPO JOB ADMR4,CLASS=A,NOTIFY=&SYSUID
/*JOBPARM S=SC63
//STEP1    EXEC DSNUPROC,TIME=1440,
//         UTPROC='',
//         SYSTEM='DB0C'
//STEPLIB  DD DSN=DB0CT.SDSNEXIT,DISP=SHR
//         DD DSN=DB0CT.SDSNLOAD,DISP=SHR
//SYSIN    DD *
  BACKUP  SYSTEM
/*
```

Figure 8-17 BACKUP SYSTEM job

The DB2 BACKUP SYSTEM utility's ability to check various conditions that prevent recoverable SLB execution is limited. The utility issued DFSMShsm commands to back up the copypools. The backup executed successfully and the utility returned 0.

Example 8-2 BACKUP SYSTEM job output

```
ABPU5001I 058 14:51:27.74 IBM DB2 Utilities Enhancement Tool Version 0220, FMID=H2AM220,
COMP_ID=5655-T58
 ABPU5012I 058 14:51:27.74 Connected to started task ABPID=ABP1
 ABPG8008I 058 14:51:27.76 System=SC63    ,Job=BACKUP0 ,Job
Id=JOB18214,Step=DSNUPROC,Program=DSNUTILB,User=ADMR4
 ABPU5002I 058 14:51:27.77 Initialization is complete.


--------------------------------------------------------------------------------------------

 ABPU5004I 058 14:51:29.28 Analysis started. Step=1
 ABPU5005I 058 14:51:29.28 Analysis completed. RC=0
 ABPU5008I 058 14:51:29.28 Utility execution started. Step=1
1DSNU000I    058 14:51:29.35 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = ADMR4.BACKUP0
 DSNU1044I   058 14:51:29.37 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
0DSNU050I    058 14:51:29.38 DSNUGUTC -  BACKUP SYSTEM
 DSNU1600I   058 14:51:29.38 DSNUVBBD - BACKUP SYSTEM UTILITY FOR DATA STARTING,
                         COPYPOOL = DSN$DB0C$DB
                         TOKEN = X'C4C2F0C3CAFCA63DFB43D32B001916D0F6D5'.
 DSNU1614I   058 14:51:52.22 DSNUVBBD - BACKUP SYSTEM UTILITY FOR DATA COMPLETED SUCCESSFULLY,
                         COPYPOOL = DSN$DB0C$DB
                         TOKEN = X'C4C2F0C3CAFCA63DFB43D32B001916D0F6D5'
                         DATA COMPLETE LRSN = X'001916D17C46'
                         ELAPSED TIME = 00:00:22.
 DSNU1600I   058 14:51:52.22 DSNUVBBD - BACKUP SYSTEM UTILITY FOR LOGS STARTING,
                         COPYPOOL = DSN$DB0C$LG
                         TOKEN = X'C4C2F0C3CAFCA63DFB43D32B001916D0F6D5'.
 DSNU1614I   058 14:51:53.04 DSNUVBBD - BACKUP SYSTEM UTILITY FOR LOGS COMPLETED SUCCESSFULLY,
                         COPYPOOL = DSN$DB0C$LG
                         TOKEN = X'C4C2F0C3CAFCA63DFB43D32B001916D0F6D5'
                         DATA COMPLETE LRSN = X'001916D17C46'
                         ELAPSED TIME = 00:00:00.
 DSNU1602I   058 14:51:53.35 DSNUVBBD - BACKUP SYSTEM UTILITY COMPLETED, ELAPSED TIME =
00:00:23.
 DSNU010I    058 14:51:53.36 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
 ABPU5009I 058 14:51:53.98 Utility execution completed. SYS=0000, USR=0000
 ABPU5003I 058 14:51:54.14 DB2 Utilities Enhancement Tool intercept completed.
```

Now, we try to execute the same DB2 SLB, but we let Recovery Expert drive it. SLBs in Recovery Expert can only be executed from the ISPF interface. We start by selecting option 1. System Backup Profiles from the IBM DB2 Recovery Expert for z/OS main menu. See Figure 8-18 on page 343.

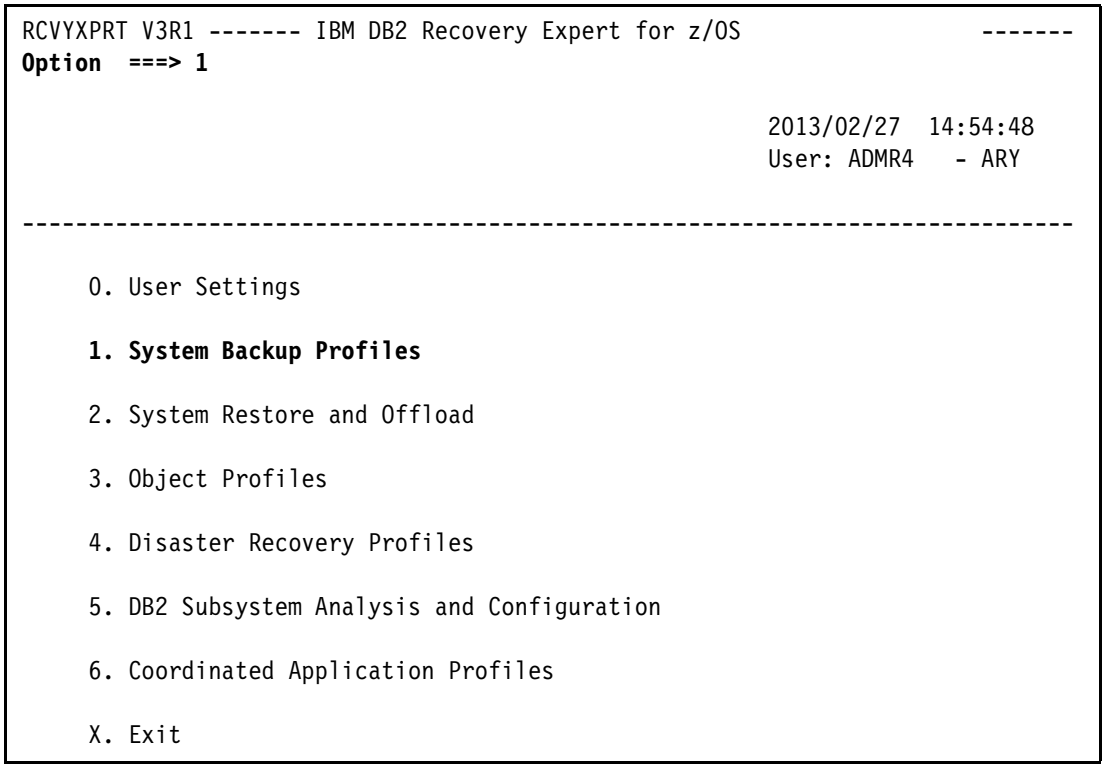

Figure 8-18 Recovery Expert main menu

There are no backup profiles defined for DB0C, so Recovery Expert prompts to create a backup profile. It has already placed the C line command in the correct place, so we continue by pressing Enter. See Figure 8-19 on page 344.

```
 RCVYXPRT V3R1    -------- Backup Profile Display -------- 2013/02/27  14:55:44
 Option  ===>                                                Scroll ===> PAGE

 Line Commands: B - Build  U - Update  C - Create  V - View  D - Delete
                R - Rename


------------------------------------------------------------------------------
  Profile Like  *                                    SSID Like  DBOC
  Creator Like  *                                             Row 1 of 1    >

------------------------------------------------------------------------------

 Cmd  Name                            Creator   SSID  Updt
  C   Press Enter to Create Profile
 ***************************** Bottom of Data ********************************



EsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssN
e ARYR031I - No Profiles were found that match your selection criteria. Press e
e enter to create a new profile or change the selection criteria.            e
DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 8-19 Create new SLB profile

The backup profile settings are shown on Figure 8-20 on page 345. Recovery Expert can drive multiple types of SLBs. It uses low-level storage APIs to drive fast replication technologies from different vendors. It can also drive DB2 SLBs by utilizing BACKUP SYSTEM, which is what we want to use. We choose D for the backup method. The value M for Source/Target Mapping is the only valid option for DB2 SLBs, so we leave it. The meaning of the other values for this parameter will be explained later. We also choose U for the Update Option field to let other users change our profile, if needed.

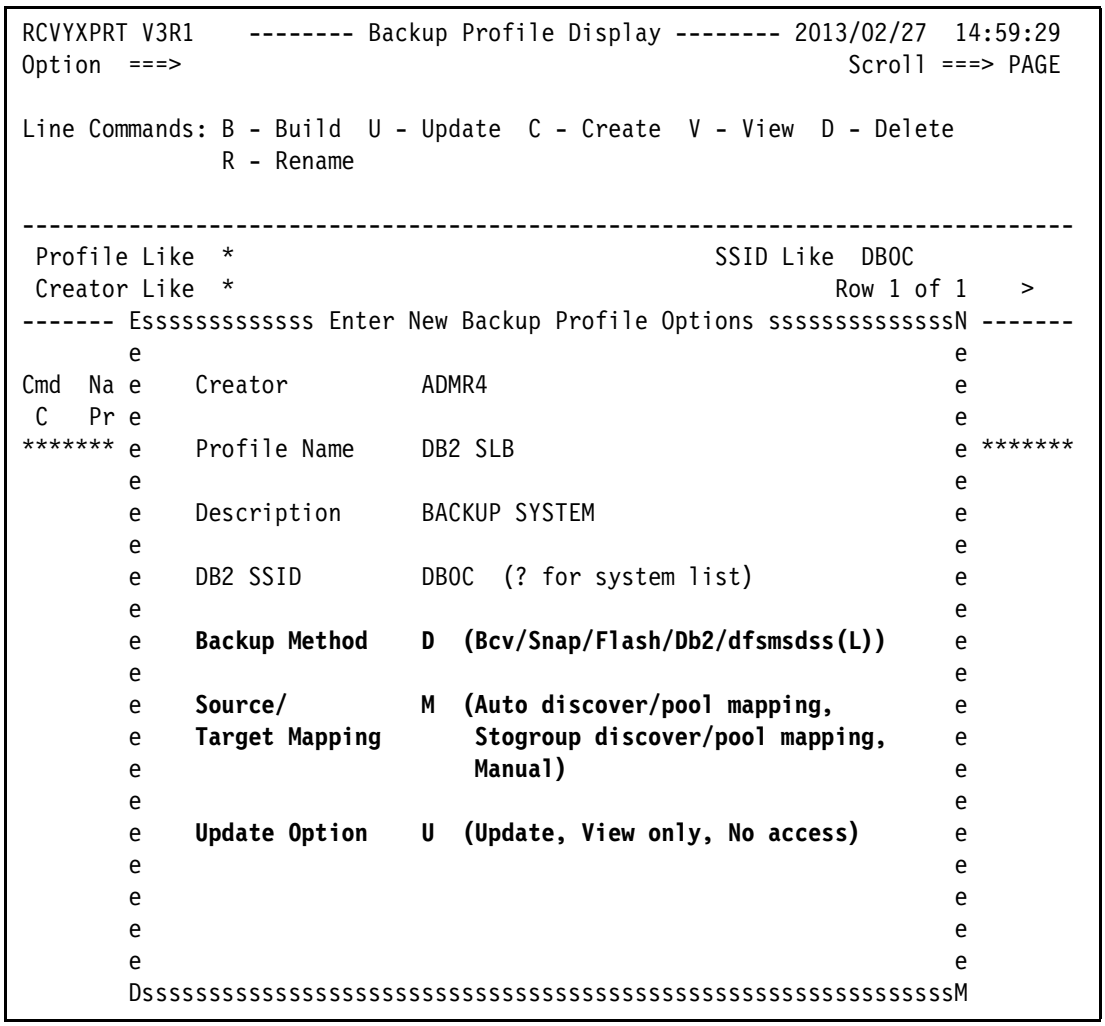

Figure 8-20 Create a backup profile

We also give a name to the new profile and provide a meaningful description. Next, we press Enter to move to the Update Backup Profile panel (Figure 8-21 on page 346).

Because this is the first time that we run the profile, Recovery Expert flags this profile as “Setup Needed”. The profile setup is a validation process performed by DB2 Recovery Expert before a backup of a subsystem can be taken. This process authenticates the volumes for the subsystem, checks the locations of the user data, logs, and user catalogs, and performs other validations to ensure that the backup can proceed and that the resulting backup will be usable. For the full list of validations performed by Recovery Expert, see Chapter 7 of *IBM DB2 Recovery Expert for z/OS Version 3 Release 1 User’s Guide*, SC19-3687.

A backup profile must successfully complete the profile setup before an SLB can be generated. Once a backup profile has been set up, it does not need to be set up again unless changes are made to the source or target volume configuration or unless DB2 Recovery Expert detects certain errors while building a backup job.

The “Issue Log Suspend” field is set to N and is not editable because we are using the BACKUP SYSTEM utility. The BACKUP SYSTEM utility can help provide consistency without issuing SET LOG SUSPEND or using storage level consistency (this is done by blocking some operations, that is, data set extends, creations, deletions, pseudo close, and

checkpoints). The number of backup generations is extracted from the copypool definitions in DFSMShsm and cannot be changed. The Current Generation field shows the generation used for the latest backup.

Tip: Backup generations in DSFSMShsm are used in a circular fashion. After all of the configured generations are used, the next backup will roll off the earliest backup.

```
RCVYXPRT  V3R1 ---------- Update Backup Profile --------- 2013/02/27  15:00:17
Option  ===>                                                Scroll ===> PAGE

     Commands: ? - Show all commands
Line Commands: I - Insert  D - Delete  X - Exclude  U - Undo from exclude

--------------------------------------------------------------------------------
 Creator: ADMR4      Name: DB2 SLB                          SSID: DBOC
 Share Option: U  (Upd,View,No)    Description: BACKUP SYSTEM
------------------------------ Backup Options ------------------------------
Backup Method     ==> D  (B/S/F/D/L)          Current Generation==> 00
Backup Scope      ==> F  (Full/Data)          Setup Needed      ==> Y
Backup Generations==> 04 (00 - 99)            Issue Log Suspend ==> N  (Yes/No)
Offload Options   ==> N  (Yes/No/Update)      Validate DB2 Vols ==> Y  (Yes/No)
                                              Enable Obj Restore==> Y  (Yes/No)
------------------ Volume Mappings -----------------         Row 1 of 23
     Source   Dev     Src   Target
Cmd  Volumes  Type    Unit  Volumes   Message Area
     SBOXJI   3390-7  611B  X66127
     SBOXJJ   3390-7  621F  X5642D
     SBOXJK   3390-9  D83B  SBOXKM
     SBOXJL   3390-9  D90F  SBOXKK
     SBOXJM   3390-9  DA0F  SBOXKN
     ......... (similar lines removed for brevity)
     X66628   3390-7  6628  X5642C
     X66727   3390-7  6727  X66126
     X76029   3390-7  6029  X5662D
     XBD01D   3390-9  D01D  SBOXK0
     XBD11C   3390-9  D11C  SBOXKP
****************************** Bottom of Data ***********************************
```

Figure 8-21 Update Backup Profile panel

We type Y for “Enable Obj Restore” to be able to use this SLB for object level recoveries and press PF3 to exit this panel. The next step is to type B next to the newly created backup profile to build the JCL for the backup. See Figure 8-22 on page 347.

Note: The mappings displayed in Volumes Mapping section are extracted from DFSMShsm volume mappings defined during the FRBACKUP PREPARE execution. Recovery Expert executes the FRBACKUP PREPARE when needed, for example, if the number of configured backup generations changes.

```
RCVYXPRT V3R1    -------- Backup Profile Display -------- 2013/02/27  15:01:21
Option  ===>                                                Scroll ===> PAGE

Line Commands: B - Build  U - Update  C - Create  V - View  D - Delete
               R - Rename

------------------------------------------------------------------------------
 Profile Like  *                                    SSID Like  DBOC
 Creator Like  *                                             Row 1 of 1    >
------------------------------------------------------------------------------

Cmd  Name                           Creator   SSID  Updt
 B   DB2 SLB                        ADMR4     DBOC   U
***************************** Bottom of Data ********************************
```

Figure 8-22 Build JCL for a backup profile

Next, we enter the JOBCARD for the generated JOB. You can also see that the Backup Repository field is not editable and set to N. See Figure 8-23.

```
RCVYXPRT V3R1    -------- Backup Profile Display -------- 2013/02/27  15:01:21
Option  ===>                                                Scroll ===> PAGE

Line Commands: B - Build  U - Update  C - Create  V - View  D - Delete
               R - Rename
  Esssssssssssssssssssssssss Build Backup Job sssssssssssssssssssssssssN
- e                                                                     e -
  e    Edit Generated Job     Y    (Yes/No)                            e
  e    Setup Job              Y    (Yes/No)                            e
- e    Perform Offload        N    (Yes/No)                            e -
  e    Backup Repository      N    (Yes/No)                            e
C e                                                                     e
  e    Build job in Dataset   ADMR4.ARY.CNTL                           e
* e                Member    BACKUPC                                   e *
  e                                                                     e
  e    Job Cards:                                                       e
  e ==> //BACKSLB JOB ADMR4,CLASS=A,NOTIFY=&SYSUID                     e
  e ==> /*JOBPARM S=SC63                                               e
  e ==> //*                                                            e
  e ==> //*                                                            e
  DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 8-23 Backup job build settings

The Backup Repository field controls whether the Recovery Expert repository will be backed up along with the subsystem. This option is only available after the setup has been executed. We press Enter to continue and Recovery Expert generates and presents the backup system job. The listing is in Example 8-3.

Example 8-3 Recovery Expert DB2 SLB JCL

```
//BACKSLB JOB ADMR4,CLASS=A,NOTIFY=&SYSUID
/*JOBPARM S=SC63
//*
```

```
//*
//*
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//*  Profile:  ADMR4.DB2 SLB                                            *
//*  Job:      01 of 01                                                 *
//*  Desc:     SLB USING BACKUP SYSTEM                                  *
//*  User:     ADMR4                                                    *
//*  Date:     Wednesday February 27, 2013                              *
//*  Time:     15:03:01.83                                              *
//*                                                                     *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//*  Step:     ARYBACK                                                  *
//*                                                                     *
//*  Desc:     This step will invoke the System Backup job.             *
//*                                                                     *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//ARYBACK  EXEC  PGM=ARY@MAIN,REGION=0O6M,COND=(4,LT)
//*
//STEPLIB  DD DISP=SHR,DSN=DBTLSP.SARYLOAD
//         DD DISP=SHR,DSN=DBOCT.SDSNEXIT
//         DD DISP=SHR,DSN=DBOCT.SDSNLOAD
//DB2PARMS DD DISP=SHR,DSN=DB2TOOLS.ARY.CONTROL
//ARYBPROF DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILES
//ARYBOFFL DD DISP=SHR,DSN=DB2TOOLS.ARY.OFFOPTS
//ARYBPMAP DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILE.MAPS
//ARYBPCAT DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILE.CATS
//ARYSBACK DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK
//ARYSBOBJ DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.OBJS
//ARYSBVOL DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.VOLS
//ARYSBSSD DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.SSIDS
//ARYBREPT DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.REPORT
//ARYPOBJS DD DISP=SHR,DSN=DB2TOOLS.ARY.OBJECTS
//SYSOUT   DD SYSOUT=*
//ARYOUT   DD SYSOUT=*
//ARY#REPT DD SYSOUT=*
//ARYSNAPO DD SYSOUT=*
//ARY#PARM DD DISP=SHR,
//         DSN=DBTLSP.SARYSAMP(ARY#PARM)
//ARYIN    DD *
   BACKUP "ADMR4"."DB2 SLB"
          SETUP
/*
//*
```

The job output is shown in Example 8-4 on page 349. The job ended with an RC=8 as Recovery Expert detected that the ICF catalog used for everything in DB0C was not included in the backed-up copypools. This example shows the value that Recovery Expert offers beyond standard backup system functionality. A situation where the client's only backup is unrecoverable can cause serious outages with financial and reputation impacts. Recovery Expert's extended validation functionality allows you to mitigate this risk.

Example 8-4 Recovery Expert DB2 backup system job output

```
 15:04:40 ARYS001I - IBM DB2 Recovery Expert for z/OS Starting. Version 03.01.001
 15:04:40 ARYS003I - Control Cards:
 15:04:40 ARYS004I -   BACKUP "ADMR4"."DB2 SLB"
 15:04:40 ARYS004I -         SETUP
 15:04:40 ARYS004I -
 15:04:40 ARYS013I - Profile ADMR4.DB2 SLB was read from the repository.
 15:04:40 ARYS038I - Performing profile volume map validation...
 15:04:40 ARYS075I - Performing subsystem source volume validation...
 15:05:03 ARYS102W - Source volume SBOXJL contains Data, Active log.
 15:05:03 ARYS102W - Source volume SBOXJM contains Data, Active log.
 15:05:03 ARYS102W - Source volume SBOXJN contains Data, Active log.
15:05:03 ARYS102W - Source volume SBOXJS contains Data Usercat, Active log Usercat.
 15:05:03 ARYS102W - Source volume X66125 contains Data, Active log.
 15:05:03 ARYS102W - Source volume X66226 contains Data, Active log.
 15:05:03 ARYS102W - Source volume X66329 contains Data, Active log.
 15:05:03 ARYS102W - Source volume X66526 contains Data, Active log.
 15:05:03 ARYS102W - Source volume X6652A contains Data, Active log.
 15:05:03 ARYS103W - All recoveries of this backup must include log recovery.
 15:05:03 ARYS021E - The following volumes are being used by Subsystem DBOC but are not being
backed up:
 15:05:03 ARYS166E -   Volume SBOXJS is not in this profile.  These subsystem datasets reside on
this volume:
 15:05:03 ARYS167E -    Dataset: UCAT.DBOCD
 15:05:03 ARYS002I - IBM DB2 Recovery Expert for z/OS complete. RC=008.
```

System level backup using Recovery Expert's FlashCopy method

Now, we look at what Recovery Expert's FlashCopy backup method can provide in the current situation. To ensure that the SLB will be usable, we start by executing a subsystem analysis. The process is similar to the process described previously for the DB2 SLB method, but this time we need to specify F for the backup method. See Figure 8-24.

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/27  15:06:53




        Essssssssssssssss Enter DB2 Subsystem ID sssssssssssssssssN
        e                                                         e
        e    DB2 Subsystem  DBOC  (? for system list)             e
        e    Backup Method  F     (Bcv/Snap/Flash/Db2/dfsmsdss(L))  e
        e                                                         e
        e                                                         e
        e                                                         e
        DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 8-24 Analyze subsystem for FlashCopy backup method

The results are shown on Figure 8-25 on page 350 and Figure 8-26 on page 351.

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/27   15:08:32
Option ===> ____________________________________________________ Scroll ===> PAGE

Commands: ANALYZE  REANALYZE

--------------------------------------------------------------------------------
 Subsystem: DB0C     Active: Yes     Datasharing: No
 Date of Last Analysis: 02/27/2013   Analysis Recommended: N
 Message: Subsystem configuration allows for application recovery or
          full system restore only.
--------------------------------------------------------          Row 1 of 50    +
  New MVS User Catalogs to be used by this subsystem
_    Log/BSDS Catl  ____________________________________________ Volume  ______
_    DB2 Data Catl  ____________________________________________ Volume  ______
     Line Cmds: (C-Create, A-Add Alias, D-Dataset Disp, U-Update, V-View Alias)

  Existing MVS User Catalogs used by this subsystem
_    Data Log Other UCAT.DB0CD                                  Volume  SBOXJS
     Line Cmds: (D-Dataset Display, V-View Aliases)

_ Boot Strap Datasets
_    DB0C - BSDS 1  DB0CD.BSDS01                                Volume  X66526
_    DB0C - BSDS 2  DB0CD.BSDS02                                Volume  X66226
     Line Cmds: (R-Rename BSDS, M-Move BSDS)

_ Active Log Datasets
_    DB0C - Log 1   DB0CD.LOGCOPY1.DS01                         Volume  SBOXJN
_    DB0C - Log 1   DB0CD.LOGCOPY1.DS02                         Volume  X66125
_    DB0C - Log 1   DB0CD.LOGCOPY1.DS03                         Volume  X66329
_    DB0C - Log 2   DB0CD.LOGCOPY2.DS01                         Volume  SBOXJL
_    DB0C - Log 2   DB0CD.LOGCOPY2.DS02                         Volume  SBOXJM
_    DB0C - Log 2   DB0CD.LOGCOPY2.DS03                         Volume  X6652A
     Line Cmds: (R-Rename Log, M-Move Log)

  Alias used with associated MVS User Catalogs
_    DB0CD          UCAT.DB0CD                                  Data Log Other
_    DB0CX          UCAT.DB0CD                                  Data
     Line Cmds: (D-Dataset Display, M-Merge catalog entries, R-Rename Alias)

  Volumes used by this subsystem
     Volume         Data  DataCat   ActLog  ActCat  ArcLog  ArcCat  Other   Flash
_    -NONE-         Yes   No        No      No      Yes     No      No      N/A
```

Figure 8-25 FlashCopy method backup analysis results (Page 1 of 2)

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/27   15:10:20
Option ===> ____________________________________________________ Scroll ===> PAGE

Commands: ANALYZE  REANALYZE

--------------------------------------------------------------------------------
 Subsystem: DB0C     Active: Yes     Datasharing: No
 Date of Last Analysis: 02/27/2013   Analysis Recommended: N
 Message: Subsystem configuration allows for application recovery or
          full system restore only.
-------------------------------------------------------          Row 32 of 50    -
_    SBOXJK         Yes    No       No       No       No       No       Yes      Yes
_    SBOXJL         Yes    No       Yes      No       No       No       Yes      Yes
_    SBOXJM         Yes    No       Yes      No       No       No       Yes      Yes
_    SBOXJN         Yes    No       Yes      No       No       No       Yes      Yes
_    SBOXJS         No     Yes      No       Yes      No       No       Yes      Yes
_    X66125         Yes    No       Yes      No       No       No       Yes      Yes
_    X66226         Yes    No       Yes      No       No       No       No       Yes
_    X66326         Yes    No       No       No       No       No       No       Yes
_    X66327         Yes    No       No       No       No       No       Yes      Yes
_    X66328         Yes    No       No       No       No       No       No       Yes
_    X66329         Yes    No       Yes      No       No       No       Yes      Yes
_    X66526         Yes    No       Yes      No       No       No       No       Yes
_    X6652A         Yes    No       Yes      No       No       No       No       Yes
_    X66628         Yes    No       No       No       No       No       No       Yes
_    X66727         Yes    No       No       No       No       No       Yes      Yes
_    XBD01D         Yes    No       No       No       No       No       No       Yes
_    XBD11C         Yes    No       No       No       No       No       No       Yes

     Line Cmds: (D-Dataset Display, M-Move all Datasets on Volume)

****************************** Bottom of Data ********************************
```

Figure 8-26 FlashCopy method backup analysis results (Page 2 of 2)

The analysis by Recovery Expert states that "`Subsystem configuration allows for application recovery or full system restore only`". This is true because when using the FlashCopy backup method, Recovery Expert can find all of the volumes used by the subsystem and back them up using low-level FlashCopy APIs. So when using this method, Recovery Exert is not using DFSMShsm and its copypool constructs. As you can see on Figure 8-26, the volume with the ICF catalog SBOXJS, which caused one of the issues for DB2 SLB, is included in the backup. Recovery Expert can also recover both logs and data, even when using the DB2 backup method. Another good feature of Recovery Expert is its ability to use FlashCopy method SLBs for object level restores.

Note: Object level restore from a FlashCopy method SLB is currently not supported for dropped object recovery, but there are plans to implement this functionality in the future. Alternatives for dropped object restore will be covered later in the chapter.

In order to execute the backup, we need to build another backup profile and generate a JCL job for it. We start by creating a new profile. For that, we need to return to the main menu and select option 1. System Backup Profiles. Then, we type C on any line to create a new profile. The new profile options are displayed on Figure 8-27 on page 352.

The backup method should be set to F, which stands for FlashCopy.

Note: Recovery Expert supports storage systems from IBM, EMC, and Hitachi Data Systems. It can run various types of fast replication technologies from these vendors. The DB2 BACKUP SYSTEM utility supports only IBM FlashCopy. Along with their own types of fast replication technologies, EMC and Hitachi have implemented IBM FlashCopy in their storage controllers (on the microcode level), but its usage will require additional licensing.

The Source/Target Mapping field controls how Recovery Expert will identify source and target volumes. There are three options:

- Auto discover/pool mapping - Recovery Expert will identify the source volumes used by DB2. The user has to provide the target storage groups.
- Stogroup discover/pool mapping - The user has to provide the source and target storage groups.
- Manual - The user has to provide the source volumes and the target volumes.

```
RCVYXPRT V3R1    -------- Backup Profile Display -------- 2013/02/27  15:14:58
Option  ===>                                                Scroll ===> PAGE

Line Commands: B - Build  U - Update  C - Create  V - View  D - Delete
               R - Rename

------------------------------------------------------------------------------
 Profile Like  *                                  SSID Like  DBOC
 Creator Like  *                                           Row 1 of 1    >
------- Essssssssssss Enter New Backup Profile Options sssssssssssssN -------
        e                                                            e
Cmd  Na e    Creator          ADMR4                                  e
 C   DB e                                                            e
******* e    Profile Name     RE SYSTEM BACKUP                       e *******
        e                                                            e
        e    Description      FLASHCOPY BACKUP                       e
        e                                                            e
        e    DB2 SSID         DBOC  (? for system list)               e
        e                                                            e
        e    Backup Method    F  (Bcv/Snap/Flash/Db2/dfsmsdss(L))    e
        e                                                            e
        e    Source/          A  (Auto discover/pool mapping,        e
        e    Target Mapping       Stogroup discover/pool mapping,    e
        e                         Manual)                            e
        e                                                            e
        e    Update Option    U  (Update, View only, No access)      e
        e                                                            e
        e                                                            e
        e                                                            e
        e                                                            e
        DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 8-27 FlashCopy method backup profile

We use A for Auto discover/pool mapping for this scenario. We press Enter to continue and Recovery Expert presents the Update Backup Profile panel. See Figure 8-28 on page 353. We have a choice to either use the SET LOG SUSPEND command or to use storage level consistency. This is controlled by the “Issue Log Suspend” parameter. We select Y for our scenario. We also enable object level restore from this SLB by setting the “Enable Obj Restore” field to Y. The only remaining task is to specify the target storage group, so we type Y next to Target Pool and press Enter.

Recovery Expert presents a panel to enter the target storage groups as shown on Figure 8-29. We enter the name of the backup copypool storage group and press PF3 to return to the Backup Profile Update panel. Then, we press PF3 again to return to the profile list.

```
RCVYXPRT  V3R1 ---------- Update Backup Profile --------- 2013/02/27  15:17:55
Option  ===>                                                Scroll ===> PAGE

     Commands: ? - Show all commands
Line Commands: I - Insert  D - Delete  X - Exclude  U - Undo from exclude

--------------------------------------------------------------------------------
 Creator: ADMR4      Name: RE SYSTEM BACKUP                SSID: DBOC
 Share Option: U  (Upd,View,No)    Description: FLASHCOPY BACKUP
------------------------------- Backup Options -------------------------------
Backup Method     ==> F  (B/S/F/D/L)          Current Generation==> 00
Backup Scope      ==> F  (Full/Data)          Setup Needed      ==> Y
Backup Generations==> 01 (01 - 99)            Issue Log Suspend ==> N  (Yes/No)
Offload Options   ==> N  (Yes/No/Update)      Validate DB2 Vols ==> Y  (Yes/No)
Target Pool       ==> Y  (Yes/No/Update)      Enable Obj Restore==> Y  (Yes/No)
----------- Volume Inclusions/Exclusions -----------          Row 1 of 1
     Source   Dev     Src   Target
Cmd  Volumes  Type    Unit  Units     Message Area
 I   PRESS ENTER FOR NEW LINE
***************************** Bottom of Data *********************************
```

Figure 8-28 FlashCopy backup method profile settings

Notice the Backup Generations and Current Generation fields on Figure 8-28. The first value sets the number of backups maintained by Recovery Expert, and the second value shows the generation number that used by the last backup.

Next, we build the JCL for the new profile. See Figure 8-29.

The steps are similar to the steps described in “System level backup using BACKUP SYSTEM” on page 340.

```
RCVYXPRT  V3R1 ---------- Target Pool Selection --------- 2013/02/27  16:37:08
Option  ===>                                                Scroll ===> PAGE

Line Commands: I - Enter  D - Delete

--------------------------------------------------------------------------------
 Creator: ADMR4      Name: RE SYSTEM BACKUP                SSID: DBOC
 Share Option: U  (Upd,View,No)    Description: FLASHCOPY BACKUP
---------------------------- Target Range Options ----------------------------

Enter by Unit or Stogroup   ==> S  (Unit/Stogroup)

--------------- Enter Storage Groups ---------------          Row 1 of 1
Cmd  Stogroup
     DBOCCPB
***************************** Bottom of Data *********************************
```

Figure 8-29 Target Pool Selection

The generated JCL is listed in Example 8-5.

Example 8-5 Recovery Expert FlashCopy method backup

```
//BACKFCP JOB ADMR4,CLASS=A,NOTIFY=&SYSUID
/*JOBPARM S=SC63
//*
//*
//*
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//*  Profile:  ADMR4.RE SYSTEM BACKUP                                   *
//*  Job:      01 of 01                                                 *
//*  Desc:     FLASHCOPY BACKUP                                         *
//*  User:     ADMR4                                                    *
//*  Date:     Wednesday February 27, 2013                              *
//*  Time:     16:40:00.24                                              *
//*                                                                     *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//*  Step:     ARYBACK                                                  *
//*                                                                     *
//*  Desc:     This step will invoke the System Backup job.             *
//*                                                                     *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//ARYBACK  EXEC  PGM=ARY@MAIN,REGION=006M,COND=(4,LT)
//*
//STEPLIB  DD DISP=SHR,DSN=DBTLSP.SARYLOAD
//         DD DISP=SHR,DSN=DBOCT.SDSNEXIT
//         DD DISP=SHR,DSN=DBOCT.SDSNLOAD
//DB2PARMS DD DISP=SHR,DSN=DB2TOOLS.ARY.CONTROL
//ARYBPROF DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILES
//ARYBOFFL DD DISP=SHR,DSN=DB2TOOLS.ARY.OFFOPTS
//ARYBPMAP DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILE.MAPS
//ARYBPCAT DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILE.CATS
//ARYSBACK DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK
//ARYSBOBJ DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.OBJS
//ARYSBVOL DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.VOLS
//ARYSBSSD DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.SSIDS
//ARYBREPT DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.REPORT
//ARYPOBJS DD DISP=SHR,DSN=DB2TOOLS.ARY.OBJECTS
//SYSOUT   DD SYSOUT=*
//ARYOUT   DD SYSOUT=*
//ARY#REPT DD SYSOUT=*
//ARYSNAPO DD SYSOUT=*
//ARY#PARM DD DISP=SHR,
//         DSN=DBTLSP.SARYSAMP(ARY#PARM)
//ARYIN    DD *
   BACKUP "ADMR4"."RE SYSTEM BACKUP"
          SETUP
          INCLUDE-ARCHIVE-VOLS
/*
```

```
//*
```

The job output is shown in Example 8-6. The job lists all of the volumes found in the target storage group and maps the source volumes to the target volumes. Because we are executing the job for the first time, Recovery Expert placed a SETUP clause into the backup control statements, so this execution was only a setup. We need to remove the setup clause and rerun the job to get the actual backup.

Example 8-6 Recovery Expert FlashCopy method backup output with the SETUP clause

```
 16:40:14 ARYS001I - IBM DB2 Recovery Expert for z/OS Starting. Version 03.01.001
 16:40:14 ARYS003I - Control Cards:
 16:40:14 ARYS004I -   BACKUP "ADMR4"."RE SYSTEM BACKUP"
 16:40:14 ARYS004I -          SETUP
 16:40:14 ARYS004I -         INCLUDE-ARCHIVE-VOLS
 16:40:14 ARYS004I -
 16:40:14 ARYS013I - Profile ADMR4.RE SYSTEM BACKUP was read from the repository.
 16:40:14 ARYS038I - Performing profile volume map validation...
 16:40:14 ARYS004I - Discovering DB2 source volumes...
 16:40:36 ARYS370I - Performing target volume validation...
 16:40:36 ARYS275I - Fetching volumes in target storage group: DBOCCPB
 16:40:36 ARYS450I -  Volume SBOXKI was found in storage group.
 16:40:36 ARYS450I -  Volume SBOXKJ was found in storage group.
......... (similar lines removed for brevity)
 16:40:36 ARYS450I -  Volume X8D500 was found in storage group.
 16:40:36 ARYS450I -  Volume X8D80D was found in storage group.
 16:40:36 ARYS404I - Source Volser: SBOXJK has been mapped to Target Volser: X8D002
 16:40:36 ARYS404I - Source Volser: SBOXJL has been mapped to Target Volser: XBD070
.......... (similar lines removed for brevity)
 16:40:36 ARYS404I - Source Volser: X66727 has been mapped to Target Volser: SBOXKI
 16:40:36 ARYS404I - Source Volser: XBD01D has been mapped to Target Volser: XBD14E
 16:40:36 ARYS404I - Source Volser: XBD11C has been mapped to Target Volser: SBOXKM
 16:40:36 ARYS371I - Target volume validation complete.
 16:40:36 ARYS039I - Volume map validation complete.
 16:40:36 ARYS079I - Profile setup is complete.  Remove the "SETUP" control card to run the backup.
 16:40:36 ARYS002I - IBM DB2 Recovery Expert for z/OS complete. RC=000.
```

Example 8-7 shows the output for a job without the SETUP clause.

Example 8-7 Recovery Expert FlashCopy method backup output

```
 16:41:13 ARYS001I - IBM DB2 Recovery Expert for z/OS Starting. Version 03.01.001
 16:41:13 ARYS003I - Control Cards:
 16:41:13 ARYS004I -   BACKUP "ADMR4"."RE SYSTEM BACKUP"
 16:41:13 ARYS004I -         INCLUDE-ARCHIVE-VOLS
 16:41:13 ARYS004I -
 16:41:13 ARYS013I - Profile ADMR4.RE SYSTEM BACKUP was read from the repository.
 16:41:13 ARYS004I - Discovering DB2 source volumes...
 16:41:34 ARYS370I - Performing target volume validation...
 16:41:34 ARYS275I - Fetching volumes in target storage group: DBOCCPB
 16:41:34 ARYS450I -  Volume SBOXKI was found in storage group.
 16:41:34 ARYS450I -  Volume SBOXKJ was found in storage group.
 16:41:34 ARYS450I -  Volume SBOXKK was found in storage group.
......... (similar lines removed for brevity)
 16:41:34 ARYS450I -  Volume X8D002 was found in storage group.
 16:41:34 ARYS450I -  Volume X8D101 was found in storage group.
 16:41:34 ARYS450I -  Volume X8D301 was found in storage group.
 16:41:34 ARYS450I -  Volume X8D400 was found in storage group.
 16:41:34 ARYS450I -  Volume X8D500 was found in storage group.
 16:41:34 ARYS450I -  Volume X8D80D was found in storage group.
```

```
 16:41:34 ARYS404I - Source Volser: SBOXJK has been mapped to Target Volser: X8D002
 16:41:34 ARYS404I - Source Volser: SBOXJL has been mapped to Target Volser: XBD070
......... (similar lines removed for brevity)
16:41:34 ARYS404I - Source Volser: X66628 has been mapped to Target Volser: X6622A
 16:41:34 ARYS404I - Source Volser: X66727 has been mapped to Target Volser: SBOXKI
 16:41:34 ARYS404I - Source Volser: XBD01D has been mapped to Target Volser: XBD14E
 16:41:34 ARYS404I - Source Volser: XBD11C has been mapped to Target Volser: SBOXKM
 16:41:34 ARYS371I - Target volume validation complete.
 16:41:34 ARYS039I - Volume map validation complete.
 16:41:35 ARYS275I - DB2 checkpoint taken at RBA/LRSN 00191709F906
 16:41:35 ARYS240I - Performing fast replication to create backup...
 16:41:35 ARYS241I - Backup via flash volume from source volser SBOXJN to unit D12C has completed.
 16:41:35 ARYS241I - Backup via flash volume from source volser SBOXJL to unit D070 has completed.
 16:41:35 ARYS241I - Backup via flash volume from source volser SBOXJK to unit D002 has completed.
 16:41:35 ARYS241I - Backup via flash volume from source volser SBOXJM to unit D101 has completed.
 16:41:35 ARYS241I - Backup via flash volume from source volser SBOXJS to unit D12D has completed.
......... (similar lines removed for brevity)
 16:41:35 ARYS241I - Backup via flash volume from source volser X66327 to unit 6129 has completed.
 16:41:35 ARYS241I - Backup via flash volume from source volser XBD11C to unit D226 has completed.
 16:41:35 ARYS241I - Backup via flash volume from source volser X66727 to unit 631B has completed.
 16:41:35 ARYS241I - Backup via flash volume from source volser X66628 to unit 622A has completed.
 16:41:36 ARYS084I - Backup of profile ADMR4.RE SYSTEM BACKUP has been created.
 16:41:36 ARYS004I - Performing volume label cleanup...
 16:41:36 ARYS004I - Volume cleanup complete.
 16:41:36 ARYS004I - Collecting dataset information for object level recovery...
 16:42:04 ARYS004I - Dataset information collection complete.
16:42:04 ARYS080I - Backup with timestamp 2013/02/27-16:41:35, generation 01 was saved in the repository.
 16:42:05 ARYS002I - IBM DB2 Recovery Expert for z/OS complete. RC=000.
```

As you can see in Example 8-7 on page 355, Recovery Expert is dumping the volumes used by DB2 using FlashCopy and later collecting the data set information for object level recovery. We did not specify the source storage groups or volumes. Recovery Expert is able to detect all of the volumes used by the subsystem and include them in the backup. It does not rely on DFSMShsm to determine the source storage groups. Imagine a scenario when a DBA allocated additional objects on a new storage group and forgot to include the new storage group into the copypool definition. These new objects will be left out when using BACKUP SYSTEM utility without Recovery Expert.

Note: FlashCopy method backups taken by Recovery Expert cannot be used as an input for the RESTORE SYSTEM utility and information about such backups is not stored in the BSDSs. It is stored in the Recovery Expert repository. The only way to restore from a FlashCopy backup is by using Recovery Expert.

Configuring the subsystem for system level backup

The only way that we can back up DB0C in its current condition is by using the FlashCopy backup method, and it only allows for object level recovery and full (data and logs) recovery. This imposes serious limitations on recovery points in time, so we need to change the subsystem layout to comply with the SLB requirements listed in “System level backup requirements” on page 336. First, we need to move the BSDSs and active logs into a separate storage group and ICF catalog. Then, we need to ensure that the ICF catalogs for the data and logs are included in the SLB. Currently, the UCAT.DB0CD catalog is used for logs, BSDSs, and subsystem data and is located outside of the defined copypools.

Renaming or moving bootstrap data sets and active logs

We start by moving the BSDSs into a separate storage group and ICF catalog. We have asked the storage administrator to prepare two storage groups: DB0CLOG1 for the first copy of the logs and BSDS and DB0CLOG2 to store the second copies of the logs and BSDS. We also asked to allocate a new catalog for logs and BSDSs with the following aliases: DB0CL for active logs, DB0CA for archive logs, and DB0CB for BSDSs.

All of the configuration changes to the subsystems are executed in the Subsystem Analysis and Configuration panel, which can be accessed by selecting option 5 on the main menu. We navigate there and typing D for backup method and typing Y when Recovery Expert asks whether it should reanalyze the subsystem.

Note: DB2 subsystem should be shut down before any renames or moves can happen.

The actual renaming of the BSDS is done by specifying the new catalog name in the Log/BSDS Catl field and typing an R near the data set name. See Figure 8-30. Even though the function is called rename, Recovery Expert allows you to simultaneously move the data set to a different location.

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/27  17:17:40
Option  ===>                                                  Scroll ===> PAGE

Commands: ANALYZE  REANALYZE

------------------------------------------------------------------------------
 Subsystem: DB0C    Active: No      Datasharing: No
 Date of Last Analysis: 02/27/2013  Analysis Recommended: N
 Message: Subsystem configuration allows for application recovery or
          full system restore only.
--------------------------------------------------          Row 1 of 50    +
  New MVS User Catalogs to be used by this subsystem
    Log/BSDS Catl UCAT.DB0CL                                    Volume
    DB2 Data Catl                                               Volume
    Line Cmds: (C-Create, A-Add Alias, D-Dataset Disp, U-Update, V-View Alias)

  Existing MVS User Catalogs used by this subsystem
    Data Log Other UCAT.DB0CD                                   Volume  SB0XJS
    Line Cmds: (D-Dataset Display, V-View Aliases)

  Boot Strap Datasets
R   DB0C - BSDS 1  DB0CD.BSDS01                                 Volume  X66526
    DB0C - BSDS 2  DB0CD.BSDS02                                 Volume  X66226
    Line Cmds: (R-Rename BSDS, M-Move BSDS)
```

Figure 8-30 Renaming the bootstrap data set

Recovery Expert shows a panel to enter the rename or move options. See Figure 8-31 on page 358. We specify the new alias and SMS storage class. Our SMS automatic class selection (ACS) routines are set up so that the DB0CLOG1 storage class will map the data set to the correct storage group.

```
RCVYXPRT  V3R1 ------- Boot Strap Dataset Rename  ------- 2013/02/27  18:27:15
Option  ===>
--------------------------------------------------------------------------------

  Subsystem: DBOC

  You have selected the Rename Boot Strap Dataset Function.  You have
  entered this command on a detail line which updates the alias for
  DBOCD.BSDS01.
  The New Boot Strap Alias will be substituted for the first node of the
  the existing dataset name.  You may use a symbolic of &SSID for
  substituting the DB2 subsystem name.

     New Boot Strap Alias: DBOCB                        Ex: DB2&SSID

  You may optionally specify the following:

     New Volume:
     New SMS Storage Class: DBOCLOG1
```

Figure 8-31 Boot Strap Dataset Rename panel

The new alias must belong to the catalog specified on Figure 8-30 on page 357. After pressing Enter, the Recovery Expert presents a panel with IDCAMS control statements to rename/move the data set. Now, we have the ability to edit and submit the generated job. See Figure 8-32.

```
RCVYXPRT  V3R1 --------- IDCAMS Interface Module -------- 2013/02/27  18:28:23
Option  ===>                                              Scroll ===> PAGE

Action   ==>         (E - EDIT, B - BATCH SUBMISSION)

---------------- IDCAMS Input Cards ----------------         Row 1 of 16    >

  /*-------------------------------------------------------*/

  DEFINE CLUSTER -
        (NAME('DBOCB.BSDS01') -
         STORAGECLASS(DBOCLOG1) -
         MODEL('DBOCD.BSDS01'))

  IF MAXCC > 0 THEN CANCEL

  REPRO  INDATASET('DBOCD.BSDS01') -
         OUTDATASET('DBOCB.BSDS01') -
         REPLACE

  IF MAXCC > 0 THEN CANCEL

  DELETE 'DBOCD.BSDS01'
***************************** Bottom of Data ********************************
```

Figure 8-32 Rename/move boot strap data set IDCAMS control statements

The control statements define the new VSAM cluster with a new name and storage group using the original cluster as a model, execute the REPRO command to copy the contents of the original cluster to the new cluster, and delete the original cluster.

After the jobs are successfully executed for both data sets, we return to the analysis panel and see that it still shows the old BSDS names. We need to execute the REANALYZE command as shown on Figure 8-33 to see the updates.

```
 RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/27  18:31:59
 Option  ===> REANALYZE                                          Scroll ===> PAGE

 Commands: ANALYZE  REANALYZE


--------------------------------------------------------------------------------
  Subsystem: DBOC    Active: No      Datasharing: No
  Date of Last Analysis: 02/27/2013  Analysis Recommended: Y
  Message: Subsystem configuration allows for application recovery or
           full system restore only.
```

Figure 8-33 Executing the REANALYZE command

Note: The DB2 subsystem should be started before running the analysis. For it to start, we need to change the BSDS DD in the MSTR address space startup procedure.

The subsystem state after the reanalysis is shown on Figure 8-34 on page 360. Both BSDSs have been renamed and are now in a different ICF catalog.

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/27  18:35:47
Option  ===>                                                  Scroll ===> PAGE

Commands: ANALYZE  REANALYZE

--------------------------------------------------------------------------------
 Subsystem: DBOC    Active: Yes     Datasharing: No
 Date of Last Analysis: 02/27/2013  Analysis Recommended: N
 Message: Subsystem configuration prevents system level backup.

-------------------------------------------------          Row 1 of 60    +
  New MVS User Catalogs to be used by this subsystem
    Log/BSDS Catl  UCAT.DBOCL                                 Volume  SBOXJQ
    DB2 Data Catl                                             Volume
    Line Cmds: (C-Create, A-Add Alias, D-Dataset Disp, U-Update, V-View Alias)

  Existing MVS User Catalogs used by this subsystem
    Data Log Other UCAT.DBOCD                                 Volume  SBOXJS
    Log            UCAT.DBOCL                                 Volume  SBOXJQ
    Line Cmds: (D-Dataset Display, V-View Aliases)

  Storage Copy Pools
    Data Copy Pool DSN$DBOC$DB
    Log Copy Pool  DSN$DBOC$LG
    Backup Pool    DBOCCPB
    Line Cmds: (V-Volume List)

  Boot Strap Datasets
    DBOC - BSDS 1  DBOCB.BSDS01                               Volume  SBOXJQ
    DBOC - BSDS 2  DBOCB.BSDS02                               Volume  SBOXJR
    Line Cmds: (R-Rename BSDS, M-Move BSDS)

  Active Log Datasets
    DBOC - Log 1   DBOCD.LOGCOPY1.DS01                        Volume  SBOXJN
    DBOC - Log 1   DBOCD.LOGCOPY1.DS02                        Volume  X66125
    DBOC - Log 1   DBOCD.LOGCOPY1.DS03                        Volume  X66329
    DBOC - Log 2   DBOCD.LOGCOPY2.DS01                        Volume  SBOXJL
    DBOC - Log 2   DBOCD.LOGCOPY2.DS02                        Volume  SBOXJM
    DBOC - Log 2   DBOCD.LOGCOPY2.DS03                        Volume  X6652A
    Line Cmds: (R-Rename Log, M-Move Log)
```

Figure 8-34 Subsystem state after reanalysis

The next step is to rename and move the active logs. The process is similar to renaming and moving BSDSs. Our storage administrator defined the DB0CL alias for the active logs, so we move the first and second active log data set there. We intentionally leave the third copy untouched. The system state after the move is shown on Figure 8-35 on page 361.

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/27  19:06:09
Option  ===>                                                  Scroll ===> PAGE

Commands: ANALYZE  REANALYZE

--------------------------------------------------------------------------------
 Subsystem: DBOC    Active: Yes     Datasharing: No
 Date of Last Analysis: 02/27/2013  Analysis Recommended: N
 Message: Subsystem configuration prevents system level backup.

-------------------------------------------------          Row 1 of 61    +
  New MVS User Catalogs to be used by this subsystem
    Log/BSDS Catl  UCAT.DBOCL                                 Volume  SBOXJQ
    DB2 Data Catl                                             Volume
    Line Cmds: (C-Create, A-Add Alias, D-Dataset Disp, U-Update, V-View Alias)

  Existing MVS User Catalogs used by this subsystem
    Data Log Other UCAT.DBOCD                                 Volume  SBOXJS
    Log            UCAT.DBOCL                                 Volume  SBOXJQ
    Line Cmds: (D-Dataset Display, V-View Aliases)

  Storage Copy Pools
    Data Copy Pool DSN$DBOC$DB
    Log Copy Pool  DSN$DBOC$LG
    Backup Pool    DBOCCPB
    Line Cmds: (V-Volume List)

  Boot Strap Datasets
    DBOC - BSDS 1  DBOCB.BSDS01                               Volume  SBOXJQ
    DBOC - BSDS 2  DBOCB.BSDS02                               Volume  SBOXJR
    Line Cmds: (R-Rename BSDS, M-Move BSDS)

  Active Log Datasets
    DBOC - Log 1   DBOCD.LOGCOPY1.DS03                        Volume  X66326
    DBOC - Log 2   DBOCD.LOGCOPY2.DS03                        Volume  X6652A
    DBOC - Log 1   DBOCL.LOGCOPY1.DS01                        Volume  SBOXJQ
    DBOC - Log 1   DBOCL.LOGCOPY1.DS02                        Volume  SBOXJQ
    DBOC - Log 2   DBOCL.LOGCOPY2.DS01                        Volume  SBOXJR
    DBOC - Log 2   DBOCL.LOGCOPY2.DS02                        Volume  SBOXJR
    Line Cmds: (R-Rename Log, M-Move Log)
```

Figure 8-35 Subsystem state after first and second active logs have been moved

Both copies of the third active log will still be backed up and restored with DB2 data. Therefore, during the RESTORE SYSTEM, the third active log will be overwritten, which can lead to errors while applying logs during crash recovery or roll forward operations. We see whether this condition will be flagged by BACKUP SYSTEM. We use the same job. The JCL is listed in Figure 8-7 on page 331. The job output is shown in Example 8-8.

Example 8-8 Backup system job output

```
ABPU5001I 059 12:09:59.71 IBM DB2 Utilities Enhancement Tool Version 0220, FMID=H2AM220, COMP_ID=5655-T58
 ABPU5012I 059 12:09:59.71 Connected to started task ABPID=ABP1
 ABPG8008I 059 12:09:59.78 System=SC63    ,Job=BACKUP0 ,Job
Id=JOB18337,Step=DSNUPROC,Program=DSNUTILB,User=ADMR4
```

```
 ABPU5002I 059 12:09:59.80 Initialization is complete.

-------------------------------------------------------------------------------------------------

 ABPU5004I 059 12:10:00.94 Analysis started. Step=1
 ABPU5005I 059 12:10:00.94 Analysis completed. RC=0
 ABPU5008I 059 12:10:00.94 Utility execution started. Step=1
1DSNU000I    059 12:10:01.00 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = ADMR4.BACKUP0
 DSNU1044I   059 12:10:01.03 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
0DSNU050I    059 12:10:01.04 DSNUGUTC -  BACKUP SYSTEM
 DSNU1600I   059 12:10:01.04 DSNUVBBD - BACKUP SYSTEM UTILITY FOR DATA STARTING,
                         COPYPOOL = DSN$DB0C$DB
                         TOKEN = X'C4C2F0C3CAFDC403EADD1C2900191A5AB37E'.
 DSNU1614I   059 12:10:15.13 DSNUVBBD - BACKUP SYSTEM UTILITY FOR DATA COMPLETED SUCCESSFULLY,
                         COPYPOOL = DSN$DB0C$DB
                         TOKEN = X'C4C2F0C3CAFDC403EADD1C2900191A5AB37E'
                         DATA COMPLETE LRSN = X'00191A5B3893'
                         ELAPSED TIME = 00:00:14.
 DSNU1600I   059 12:10:15.13 DSNUVBBD - BACKUP SYSTEM UTILITY FOR LOGS STARTING,
                         COPYPOOL = DSN$DB0C$LG
                         TOKEN = X'C4C2F0C3CAFDC403EADD1C2900191A5AB37E'.
 DSNU1614I   059 12:10:17.50 DSNUVBBD - BACKUP SYSTEM UTILITY FOR LOGS COMPLETED SUCCESSFULLY,
                         COPYPOOL = DSN$DB0C$LG
                         TOKEN = X'C4C2F0C3CAFDC403EADD1C2900191A5AB37E'
                         DATA COMPLETE LRSN = X'00191A5B3893'
                         ELAPSED TIME = 00:00:02.
 DSNU1602I   059 12:10:17.57 DSNUVBBD - BACKUP SYSTEM UTILITY COMPLETED, ELAPSED TIME = 00:00:16.
 DSNU010I    059 12:10:17.59 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
 ABPU5009I 059 12:10:18.19 Utility execution completed. SYS=0000, USR=0000
 ABPU5003I 059 12:10:18.19 DB2 Utilities Enhancement Tool intercept completed.
```

BACKUP SYSTEM is backing up the configured copypools without checking for possible issues with the subsystem layout. In the same situation, Recovery Expert signals that an issue exists; see Example 8-9.

Example 8-9 Recovery Expert DB2 SLB job output

```
 03:10:35 ARYS001I - IBM DB2 Recovery Expert for z/OS Starting. Version 03.01.001
 03:10:35 ARYS003I - Control Cards:
 03:10:35 ARYS004I -    BACKUP "ADMR4"."DB2 SLB"
 03:10:35 ARYS004I -
 03:10:35 ARYS013I - Profile ADMR4.DB2 SLB was read from the repository.
 03:10:35 ARYS075I - Performing subsystem source volume validation...
 03:10:49 ARYS102W - Source volume SBOXJL contains Data, Data Usercat, Active log Usercat.
 03:10:49 ARYS102W - Source volume SBOXJN contains Data, Active log.
 03:10:49 ARYS102W - Source volume X66329 contains Data, Active log.
 03:10:49 ARYS103W - All recoveries of this backup must include log recovery.
 03:10:49 ARYS076I - Subsystem source volume validation complete. All source volumes not explicitly
 excluded are being copied.
 03:10:49 ARYS039I - Volume map validation complete.
 03:10:50 ARYS275I - DB2 checkpoint taken at RBA/LRSN 00192F70CF2B
 03:10:50 ARYS004I - Invoking IBM System Level Backup utility...
 03:11:08 ARYS020I - IBM System Backup Utility Complete. Token = C4C2F0C3CB0D9655D036BD2A00192F70D55C
 03:11:09 ARYS004I - Collecting dataset information for object level recovery...
 03:11:43 ARYS004I - Dataset information collection complete.
 03:11:43 ARYS080I - Backup with timestamp 2013/03/13-03:11:07, generation 02 was saved in the repository.
 03:11:44 ARYS002I - IBM DB2 Recovery Expert for z/OS complete. RC=004.
```

As shown in Example 8-9 on page 362, Recovery Expert warns that all recoveries will have to be performed for both data and logs because some volumes contain both logs and data.

Note: It is not possible to restore logs with RECOVER SYSTEM. You must use DFSMShsm FRRECOV to restore logs. Recovery Expert can restore both data and logs.

The next step is to fix the active log layout by moving both copies of the third log data set to the correct storage group under the DB0CL high-level qualifier. The process is the same. First, we stop DB2, use the rename option on both copies of the third active log data set, and start DB2. After the subsystem is started, we navigate back to the Subsystem Analysis and Configuration panel to execute the REANALYZE command. The final system state is shown on Figure 8-36 on page 364 and Figure 8-38 on page 366.

The active logs and BSDSs are now placed into two separate storage groups that are part of the log's copypool. A separate ICF catalog holds the aliases related to logs, archive logs, and BSDSs. This catalog is also inside the copypool.

The only remaining problem is the data ICF catalog. It is still on the SBOXJS volume that is not included in the data copypool. See Figure 8-38 on page 366 where the SBOXJS volume is colored red. We have to either move the catalog to a new location or create a new catalog and merge the aliases used for data there. The first option has to be executed outside of Recovery Expert and is usually performed by storage administrators.

Tip: Detailed step-by-step instructions for catalog movement are shown in informational APAR II13354: MOVING ICF CATALOGS - STEP-BY-STEP SCENARIOS, INCLUDING SHARED CATALOGS.

The second approach can be implemented in Recovery Expert and we use it in our scenario. First, we need to create the new catalog. In this scenario, the catalog name is UCAT.DB0CDN (Figure 8-37 on page 365). Our ACS routines have been set up so that this data set will be placed in the DB0CDATA storage group that is part of the data copypool.

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/28   12:20:28
Option ===> ____________________________________________________ Scroll ===> PAGE

Commands: ANALYZE  REANALYZE

--------------------------------------------------------------------------------
 Subsystem: DB0C     Active: Yes     Datasharing: No
 Date of Last Analysis: 02/28/2013   Analysis Recommended: N
 Message: Subsystem configuration prevents system level backup.
-------------------------------------------------          Row 1 of 66     +
  New MVS User Catalogs to be used by this subsystem
_    Log/BSDS Catl  UCAT.DB0CL                                Volume  SBOXJQ
_    DB2 Data Catl  ________________________________________  Volume  ______
     Line Cmds: (C-Create, A-Add Alias, D-Dataset Disp, U-Update, V-View Alias)

  Existing MVS User Catalogs used by this subsystem
_    Data Other     UCAT.DB0CD                                Volume  SBOXJS
_    Log            UCAT.DB0CL                                Volume  SBOXJQ
     Line Cmds: (D-Dataset Display, V-View Aliases)

  Storage Copy Pools
_    Data Copy Pool DSN$DB0C$DB
_    Log Copy Pool  DSN$DB0C$LG
_    Backup Pool    DB0CCPB
     Line Cmds: (V-Volume List)

_ Boot Strap Datasets
_    DB0C - BSDS 1  DB0CB.BSDS01                             Volume  SBOXJQ
_    DB0C - BSDS 2  DB0CB.BSDS02                             Volume  SBOXJR
     Line Cmds: (R-Rename BSDS, M-Move BSDS)

_ Active Log Datasets
_    DB0C - Log 1   DB0CL.LOGCOPY1.DS01                      Volume  SBOXJQ
_    DB0C - Log 1   DB0CL.LOGCOPY1.DS02                      Volume  SBOXJQ
_    DB0C - Log 1   DB0CL.LOGCOPY1.DS03                      Volume  SBOXJQ
_    DB0C - Log 2   DB0CL.LOGCOPY2.DS01                      Volume  SBOXJR
_    DB0C - Log 2   DB0CL.LOGCOPY2.DS02                      Volume  SBOXJR
_    DB0C - Log 2   DB0CL.LOGCOPY2.DS03                      Volume  SBOXJR
     Line Cmds: (R-Rename Log, M-Move Log)

  Alias used with associated MVS User Catalogs
```

Figure 8-36 DB0C final state after log and BSDS move (Page 1 of 2)

We start by typing the new catalog name in the DB2 Data Catl field and typing C on that line as shown in Figure 8-37 on page 365.

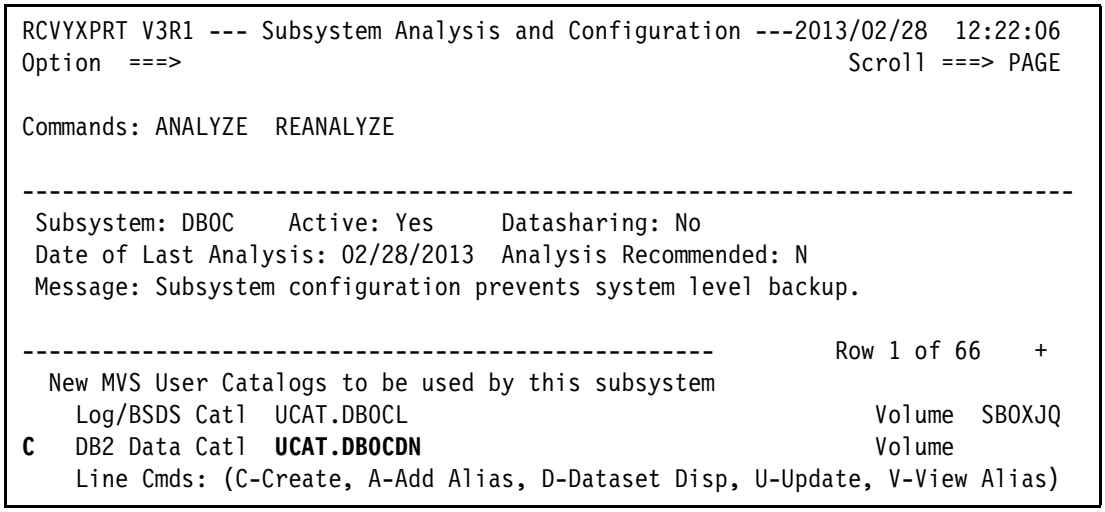

Figure 8-37 Create a new catalog for DB2 data

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/28   12:21:13
Option ===> ____________________________________________________ Scroll ===> PAGE

Commands: ANALYZE  REANALYZE

------------------------------------------------------------------------------
 Subsystem: DB0C     Active: Yes     Datasharing: No
 Date of Last Analysis: 02/28/2013   Analysis Recommended: N
 Message: Subsystem configuration prevents system level backup.
------------------------------------------------------          Row 32 of 66   -+
_    DB0CA          UCAT.DB0CL                                      Log
_    DB0CB          UCAT.DB0CL                                      Log
_    DB0CD          UCAT.DB0CD                                      Data Other
_    DB0CL          UCAT.DB0CL                                      Log
_    DB0CX          UCAT.DB0CD                                      Data
     Line Cmds: (D-Dataset Display, M-Merge catalog entries, R-Rename Alias)

  Volumes used by this subsystem
     Volume         Data   DataCat   ActLog   ActCat   ArcLog   ArcCat   Other   Pool
_    -NONE-         Yes    No        No       No       Yes      No       No      N/A
_    SBOXJI         No     No        No       No       No       No       No      Yes
_    SBOXJJ         No     No        No       No       No       No       No      Yes
_    SBOXJK         Yes    No        No       No       No       No       Yes     Yes
_    SBOXJL         Yes    No        No       Yes      No       No       Yes     Yes
_    SBOXJM         Yes    No        No       No       No       No       Yes     Yes
_    SBOXJN         Yes    No        No       No       No       No       Yes     Yes
_    SBOXJQ         No     No        Yes      Yes      No       Yes      No      Yes
_    SBOXJR         No     No        No       No       No       No       No      Yes
_    SBOXJS         No     Yes       No       No       No       No       Yes     No
_    SBOXJV         No     No        Yes      No       No       No       No      Yes
_    X66125         Yes    No        Yes      No       No       No       Yes     Yes
_    X66226         Yes    No        Yes      No       No       No       No      Yes
_    X66326         Yes    No        No       No       No       No       No      Yes
_    X66327         Yes    No        No       No       No       No       Yes     Yes
_    X66328         Yes    No        No       No       No       No       No      Yes
_    X66329         Yes    No        No       No       No       No       Yes     Yes
_    X66526         Yes    No        No       No       No       No       No      Yes
_    X6652A         Yes    No        No       No       No       No       No      Yes
_    X66628         Yes    No        No       No       No       No       No      Yes
_    X66727         Yes    No        No       No       No       No       Yes     Yes
_    X76029         No     No        No       No       No       No       No      Yes
```

Figure 8-38 DB0C final state after log and BSDS move (Page 2 of 2)

Recovery Expert will present a panel with catalog allocation options. See Figure 8-39 on page 367. We specify the name of the new ICF catalog and the name of the SMS storage class and leave the defaults for the rest. Pressing Enter displays the panel with generated IDCAMS control statements as shown on Figure 8-40 on page 367. We continue by specifying B on the command line, and Recovery Expert generates the job JCL. We submit the JCL and verify that the new catalog has been allocated to the DB0CDATA storage group.

```
RCVYXPRT  V3R1 ----- Update User Catalog Information ---- 2013/02/28  12:55:21
Option  ===>

  Subsystem: DBOC

  User Catalog Name      ==>  UCAT.DBOCDN
  User Catalog Volume    ==>
  SMS Storage Class      ==>  DBOCDATA

  Data Parameters
    Tracks or Cylinders  ==>  C    (Tracks/Cylinders)
    Primary Quantity     ==>  20
    Secondary Quantity   ==>  5
    Data Buffers         ==>  4

  Index Parameters
    Tracks or Cylinders  ==>  C    (Tracks/Cylinders)
    Primary Quantity     ==>  1
    Secondary Quantity   ==>  1
    Index Buffers        ==>  4

  User Catalog Aliases   ==>
```

Figure 8-39 New catalog options

The SMS ACS routines have been set up to place the UCAT.DB0CDN data set into the correct storage group.

```
RCVYXPRT  V3R1 --------- IDCAMS Interface Module -------- 2013/02/28  12:56:21
Option  ===>                                                Scroll ===> PAGE

Action   ==> B       (E - EDIT, B - BATCH SUBMISSION)

---------------- IDCAMS Input Cards ----------------         Row 1 of 15    >

    SET MAXCC = 0
    DEFINE  -
      USERCATALOG(NAME('UCAT.DBOCDN') -
        STORAGECLASS(DBOCDATA) -
        CYLINDERS(20 5) -
        ICFCATALOG) -
      DATA -
        (BUFND(4) -
        CYLINDERS(20 5) -
        CONTROLINTERVALSIZE(4096)) -
      INDEX -
        (BUFNI(4) -
        CYLINDERS(1 1) -
        CONTROLINTERVALSIZE(2048)) -
        CAT(MCAT.SANDBOX.Z1C.SBOX00)
****************************** Bottom of Data ********************************
```

Figure 8-40 Catalog allocation IDCAMS control statements

The next step is to merge all the aliases that belong to the old catalog UCAT.DB0CD into the new catalog UCAT.DB0CDN. The list of aliases for each catalog is on Figure 8-38 on page 366, where we can see that DB0CD and DB0CX belong to UCAT.DB0CD. Merging the alias can be done by using the M command to merge the catalog entries. We type M next to the alias in the Recovery Expert Subsystem Analysis Results panel. The name of the target catalog is specified in the “New MVS™ User Catalogs to be used by this subsystem” section of the Analysis Results panel as shown on Figure 8-41. The DB2 Data Catl field should already be populated with the correct name because we used Recovery Expert to create the catalog.

```
------------------------------------------------          Row 1 of 66    +
  New MVS User Catalogs to be used by this subsystem
    Log/BSDS Catl  UCAT.DB0CL                                Volume  SB0XJQ
    DB2 Data Catl  UCAT.DB0CDN                               Volume  SB0XJL
    Line Cmds: (C-Create, A-Add Alias, D-Dataset Disp, U-Update, V-View Alias)
```

Figure 8-41 Specifying the target catalog

We start by merging the DB0CX alias. We type M next to the alias and press Enter. See Figure 8-42.

```
 Alias used with associated MVS User Catalogs
    DB0CA          UCAT.DB0CL                                Log
    DB0CB          UCAT.DB0CL                                Log
    DB0CD          UCAT.DB0CD                                Data Other
    DB0CL          UCAT.DB0CL                                Log
M   DB0CX          UCAT.DB0CD                                Data
    Line Cmds: (D-Dataset Display, M-Merge catalog entries, R-Rename Alias)
```

Figure 8-42 Merging the alias

Recovery Expert generates the job as shown in Example 8-10.

Example 8-10 Job to alter catalog

```
//ADILET5 JOB ADMR4,CLASS=A,NOTIFY=&SYSUID
/*JOBPARM S=SC63
//*
//*
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//*  STEP:     DB2STOP                                                  *
//*                                                                     *
//*  DESC:     THIS STEP WILL STOP DB2 SUBSYSTEM DB0C                   *
//*                                                                     *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//DB2STOP  EXEC PGM=ARY#SDB2,COND=(4,LT),PARM=(DB0C,STOP)
//STEPLIB  DD DISP=SHR,DSN=DBTLSP.SARYLOAD
//DB2PARMS DD DISP=SHR,DSN=DB2TOOLS.ARY.CONTROL
//SYSPRINT DD SYSOUT=*
//*
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//*  STEP:     IDCAMS                                                   *
```

```
//*                                                                     *
//*  DESC:     THIS STEP WILL EXECUTE THE IDCAMS JOB.                   *
//*                                                                     *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*
//*****************************************************************
//*  IDCAMS JOB STEP                                             *
//*****************************************************************
//IDCAMS   EXEC PGM=IDCAMS,REGION=OO6M,COND=(4,LT)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  ALTER UCAT.DBOCD LOCK
  IF MAXCC > 0 THEN CANCEL
  ALTER UCAT.DBOCDN LOCK
  IF MAXCC > 0 THEN CANCEL
  REPRO -
    INDATASET(UCAT.DBOCD) -
    OUTDATASET(UCAT.DBOCDN) -
    LEVEL(DBOCX) -
    MERGECAT
  IF MAXCC > 0 THEN CANCEL
  DELETE  DBOCX ALIAS
  DEFINE  ALIAS (NAME(DBOCX) REL(UCAT.DBOCDN))
  ALTER UCAT.DBOCD UNLOCK
  ALTER UCAT.DBOCDN UNLOCK

//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//*  STEP:     DB2START                                                 *
//*                                                                     *
//*  DESC:     THIS STEP WILL START DB2 SUBSYSTEM DBOC                  *
//*                                                                     *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//DB2START  EXEC PGM=ARY#SDB2,COND=(4,LT),PARM=(DBOC,START)
//STEPLIB  DD DISP=SHR,DSN=DBTLSP.SARYLOAD
//DB2PARMS DD DISP=SHR,DSN=DB2TOOLS.ARY.CONTROL
//SYSPRINT DD SYSOUT=*
//*
/*
//*
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//*  STEP:     UPDSTAT                                                  *
//*                                                                     *
//*  DESC:     THIS STEP WILL UPDATE THE ANALYSIS STATUS INDICATOR.     *
//*                                                                     *
//** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//*                                                                     *
//*****************************************************************
//*  UPDATE ANALYSIS STATUS INDICATOR                              *
//*****************************************************************
//UPDSTAT  EXEC PGM=ARY#SUPD,COND=(4,LT),PARM=(DBOC)
//STEPLIB  DD DISP=SHR,DSN=DBTLSP.SARYLOAD
//ARYMOVER DD DISP=SHR,DSN=DB2TOOLS.ARY.CONFIG
```

```
//SYSPRINT DD SYSOUT=*
//*
```

Recovery Expert calls a special program that will stop the subsystem. Then, it feeds the control statements to IDCAMS. The actual movement is the IDCAMS REPRO with MERGECAT. The job also ensures that both catalogs are locked before the merge.

Tip: Generally, the MERGECAT job can be rerun, but you need to ensure that there are no programs that are currently accessing the data sets that belong to the alias that is being moved. Otherwise, IDCAMS is unable to uncatalog the data set from the old catalog.

As the final step, Recovery Expert starts the subsystem. The job output for the IDCAMS step is shown in Example 8-11.

Example 8-11 Catalog IDCAMS

```
1IDCAMS  SYSTEM SERVICES                                    TIME: 13:13:31       02/28/13     PAGE
1
0

   ALTER UCAT.DBOCD LOCK
0IDC0531I ENTRY UCAT.DBOCD ALTERED
0IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
0

   IF MAXCC > 0 THEN CANCEL

   ALTER UCAT.DBOCDN LOCK
0IDC0531I ENTRY UCAT.DBOCDN ALTERED
0IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
0

   IF MAXCC > 0 THEN CANCEL

   REPRO -
     INDATASET(UCAT.DBOCD) -
     OUTDATASET(UCAT.DBOCDN) -
     LEVEL(DBOCX) -
     MERGECAT
0IDC0639I SPHERE CONVERSION STARTED FOR
 IDC0639I DBOCX.DSNDBC.CNKDBNAM.CNKIDX11.I0001.A001
0IDC01402I SPHERE CONVERSION COMPLETED FOR
 IDC01402I DBOCX.DSNDBC.CNKDBNAM.CNKIDX11.I0001.A001
0IDC0639I SPHERE CONVERSION STARTED FOR
 IDC0639I DBOCX.DSNDBC.CNKDBNAM.CNKIDX12.I0001.A001
.......
0IDC0639I SPHERE CONVERSION STARTED FOR
 IDC0639I DBOCX.DSNDBC.CNKDBNAM.CNKTBSP1.I0001.A020
0IDC01402I SPHERE CONVERSION COMPLETED FOR
 IDC01402I DBOCX.DSNDBC.CNKDBNAM.CNKTBSP1.I0001.A020
0IDC01460I THE NUMBER OF ENTRIES MERGED WAS 25
0IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
0

   IF MAXCC > 0 THEN CANCEL

   DELETE  DBOCX ALIAS
0IDC0550I ENTRY (X) DBOCX DELETED
```

```
0IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
0
   DEFINE  ALIAS (NAME(DBOCX) REL(UCAT.DBOCDN))
0IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
0

   ALTER UCAT.DBOCD UNLOCK
0IDC0531I ENTRY UCAT.DBOCD ALTERED
0IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
1IDCAMS  SYSTEM SERVICES                                    TIME: 13:13:31       02/28/13     PAGE
5
0
   ALTER UCAT.DBOCDN UNLOCK
0IDC0531I ENTRY UCAT.DBOCDN ALTERED
0IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
0

0IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

Next, we execute the same procedure for the DB0CD alias and execute the REANALYZE command. The state of the system is shown on Figure 8-43 on page 372 and Figure 8-44 on page 373.

Note the final status provided by Recovery Expert. Now, it has changed to "`Other non-DB2 data will be backed up and restored`". There are no conditions that inhibit SLB and restore. All of the logs, BSDSs, and archive logs are now in a separate catalog under separate aliases. The ICF catalog is located on a volume that is part of the configured copypools. DB2 volumes contain data sets that are not part of the subsystem. They will be backed up and restored along with DB2 data, which can be either good or bad, depending on the nature of these data sets. If these data sets are application data that has to be consistent with DB2, it is probably a positive effect, but if the data sets are not related to DB2 in any way, restoring them to a prior point in time can cause issues. The DBA has to exercise caution when backing up and restoring non-DB2 data during DB2 SLBs.

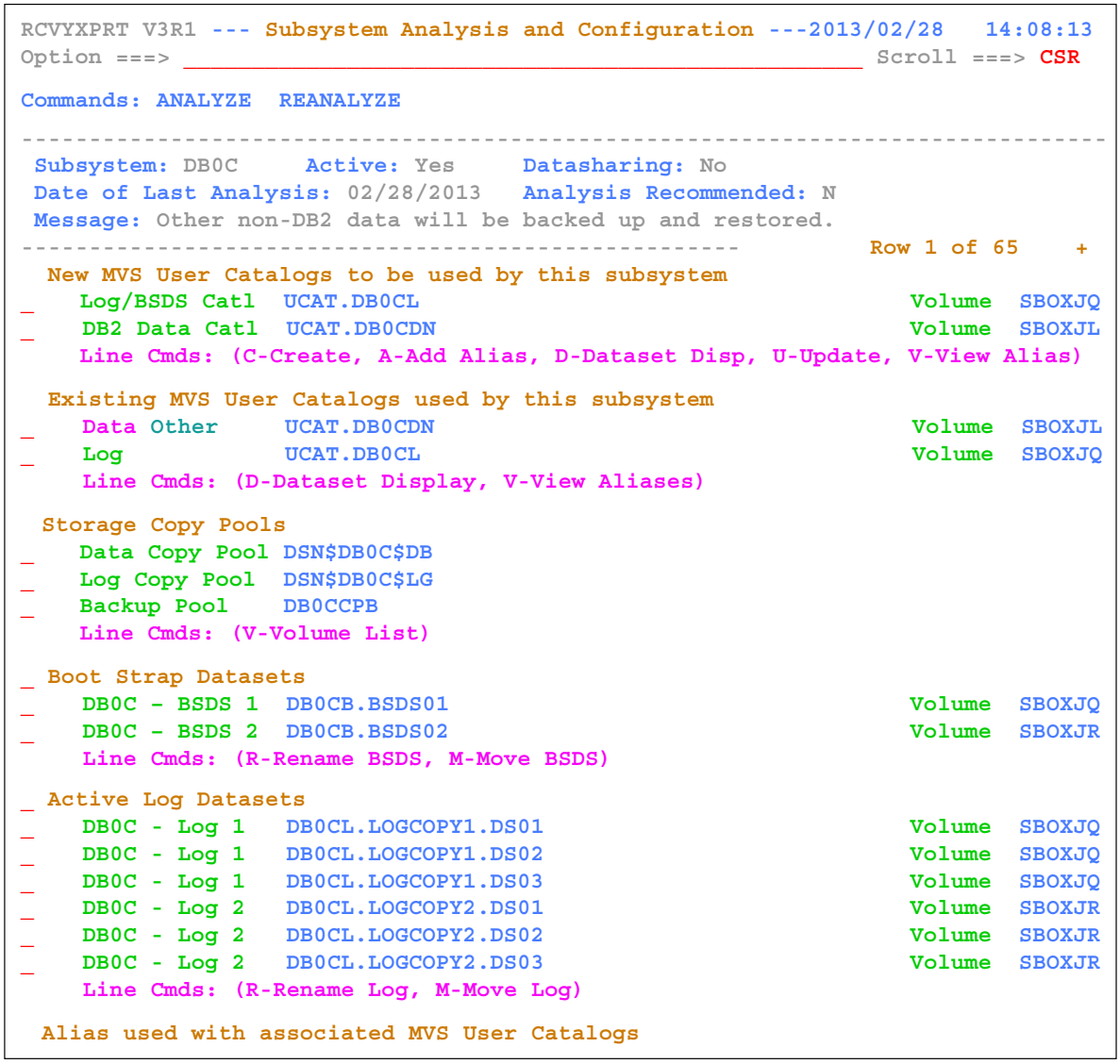

Figure 8-43 Subsystem state after the catalog move (Page 1 of 2)

On Figure 8-44 on page 373, we can see that there are no red volumes any more. The volumes that are colored in light blue contain non-DB2 data. We can look at the data sets on each volume by typing D next to the volume name.

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2013/02/28   14:10:25
Option ===> ____________________________________________________ Scroll ===> PAGE

Commands: ANALYZE  REANALYZE

--------------------------------------------------------------------------------
 Subsystem: DB0C     Active: Yes     Datasharing: No
 Date of Last Analysis: 02/28/2013   Analysis Recommended: N
 Message: Other non-DB2 data will be backed up and restored.
--------------------------------------------------------          Row 32 of 65   -+
_    DB0CA          UCAT.DB0CL                                        Log
_    DB0CB          UCAT.DB0CL                                        Log
_    DB0CD          UCAT.DB0CDN                                       Data Other
_    DB0CL          UCAT.DB0CL                                        Log
_    DB0CX          UCAT.DB0CDN                                       Data
     Line Cmds: (D-Dataset Display, M-Merge catalog entries, R-Rename Alias)

  Volumes used by this subsystem
     Volume         Data   DataCat   ActLog   ActCat   ArcLog   ArcCat   Other   Pool
_    -NONE-         Yes    No        No       No       Yes      No       No      N/A
_    SBOXJI         No     No        No       No       No       No       No      Yes
_    SBOXJJ         No     No        No       No       No       No       No      Yes
_    SBOXJK         Yes    No        No       No       No       No       Yes     Yes
_    SBOXJL         Yes    Yes       No       No       No       No       Yes     Yes
_    SBOXJM         Yes    No        No       No       No       No       Yes     Yes
_    SBOXJN         Yes    No        No       No       No       No       Yes     Yes
_    SBOXJQ         No     No        Yes      Yes      No       Yes      No      Yes
_    SBOXJR         No     No        Yes      No       No       No       No      Yes
_    SBOXJV         No     No        No       No       Yes      No       No      Yes
_    X66125         Yes    No        No       No       No       No       Yes     Yes
_    X66226         Yes    No        No       No       No       No       No      Yes
_    X66326         Yes    No        No       No       No       No       No      Yes
_    X66327         Yes    No        No       No       No       No       Yes     Yes
_    X66328         Yes    No        No       No       No       No       No      Yes
_    X66329         Yes    No        No       No       No       No       Yes     Yes
_    X66526         Yes    No        No       No       No       No       No      Yes
_    X6652A         Yes    No        No       No       No       No       No      Yes
_    X66628         Yes    No        No       No       No       No       No      Yes
_    X66727         Yes    No        No       No       No       No       Yes     Yes
_    X76029         No     No        No       No       Yes      No       No      Yes
_    XBD01D         Yes    No        No       No       No       No       No      Yes
```

Figure 8-44 Subsystem state after the catalog move (Page 2 of 2)

Execute DB2 system level backup through Recovery Expert

Now that we have configured our subsystem to comply with all BACKUP SYSTEM requirements, we can actually execute the SLB. We start by going back to the main menu and selecting option 1. System Backup Profiles. Recovery Expert displays the configured backup profiles as shown on Figure 8-45 on page 374. We type B next to the DB2 SLB to build the backup job JCL. The job consists of two steps: the first step is to back up the database and the second step is to back up the Recovery Expert repository. The backup step JCL is listed in Figure 8-45 on page 374.

```
RCVYXPRT V3R1    -------- Backup Profile Display -------- 2013/02/28  14:12:07
Option  ===>                                                Scroll ===> PAGE

Line Commands: B - Build  U - Update  C - Create  V - View  D - Delete
               R - Rename

------------------------------------------------------------------------------
 Profile Like  *                                      SSID Like  DBOC
 Creator Like  *                                               Row 1 of 2    >
------------------------------------------------------------------------------

Cmd  Name                            Creator   SSID  Updt
 B   DB2 SLB                         ADMR4     DBOC   U
     RE SYSTEM BACKUP                ADMR4     DBOC   U
***************************** Bottom of Data ********************************
```

Figure 8-45 SLB profile list

The required backup profile was previously configured, so we reuse it. We have manually added the WAIT-FOR-BG-COPY parameter, which makes the job return only after the background copy process completes. See Figure 8-46.

```
//ARYBACK  EXEC  PGM=ARY@MAIN,REGION=0O6M,COND=(4,LT)
//*
//STEPLIB  DD DISP=SHR,DSN=DBTLSP.SARYLOAD
//         DD DISP=SHR,DSN=DBOCT.SDSNEXIT
//         DD DISP=SHR,DSN=DBOCT.SDSNLOAD
//DB2PARMS DD DISP=SHR,DSN=DB2TOOLS.ARY.CONTROL
//ARYBPROF DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILES
//ARYBOFFL DD DISP=SHR,DSN=DB2TOOLS.ARY.OFFOPTS
//ARYBPMAP DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILE.MAPS
//ARYBPCAT DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILE.CATS
//ARYSBACK DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK
//ARYSBOBJ DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.OBJS
//ARYSBVOL DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.VOLS
//ARYSBSSD DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.SSIDS
//ARYBREPT DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.REPORT
//ARYPOBJS DD DISP=SHR,DSN=DB2TOOLS.ARY.OBJECTS
//SYSOUT   DD SYSOUT=*
//ARYOUT   DD SYSOUT=*
//ARY#REPT DD SYSOUT=*
//ARYSNAPO DD SYSOUT=*
//ARY#PARM DD DISP=SHR,
//         DSN=DBTLSP.SARYSAMP(ARY#PARM)
//ARYIN    DD *
   BACKUP "ADMR4"."DB2 SLB"
   WAIT-FOR-BG-COPY
/*
```

Figure 8-46 Generated backup step JCL

The backup step output is shown on Figure 8-47 on page 375. Recovery Expert will track the background copy progress and report it in the output.

```
17:43:39 ARYS001I - IBM DB2 Recovery Expert for z/OS Starting. Version 03.01.001
17:43:39 ARYS003I - Control Cards:
17:43:39 ARYS004I -   BACKUP "ADMR4"."DB2 SLB"
17:43:39 ARYS004I -   WAIT-FOR-BG-COPY
17:43:39 ARYS004I -
17:43:39 ARYS013I - Profile ADMR4.DB2 SLB was read from the repository.
17:43:40 ARYS075I - Performing subsystem source volume validation...
 17:43:54 ARYS076I - Subsystem source volume validation complete. All source volumes not
explicitly excluded are being copied.
17:43:54 ARYS039I - Volume map validation complete.
17:43:55 ARYS275I - DB2 checkpoint taken at RBA/LRSN 001930B5BBA6
17:43:55 ARYS004I - Invoking IBM System Level Backup utility...
 17:43:58 ARYS020I - IBM System Backup Utility Complete. Token =
C4C2F0C3CAFF5083E16F89AB001930B5C041
17:43:59 ARYS004I - Collecting dataset information for object level recovery...
17:44:29 ARYS004I - Dataset information collection complete.
 17:44:29 ARYS080I - Backup with timestamp 2013/03/01-17:43:58, generation 04 was saved
in the repository.
17:44:30 ARYS350I - Remaining tracks to be copied:     679,784
17:44:45 ARYS350I - Remaining tracks to be copied:     649,085
17:45:00 ARYS350I - Remaining tracks to be copied:     582,497
...........
18:13:33 ARYS350I - Remaining tracks to be copied:       4,049
18:13:48 ARYS350I - Remaining tracks to be copied:       1,229
18:14:03 ARYS004I - The background copy process is complete.
18:14:03 ARYS002I - IBM DB2 Recovery Expert for z/OS complete. RC=000.
```

Figure 8-47 Backup job step output

Another important Recovery Expert feature is the backup report that lists the objects in recovery-related restrictive states. The report generated by the previously executed backup is listed in Example 8-12.

Example 8-12 Recovery Expert backup report

```
IBM DB2 Recovery Expert for z/OS
                  Backup Summary Report

     Utility Executed:......... Backup
     Profile Name:.............. ADMR4.DB2 SLB
      DB2 Subsystem:............ DBOC
      DB2 Version:.............. 1010
    Backup Type:............... DB2 System Level Backup
    Backup Contains:.......... Object Data and Log Data
     Partial Backup:........... No
     Nbr of Volumes:........... 0023
   HSM Backup Token:.......... C4C2F0C3CAFF5083E16F89AB001930B5C041
    Backup RBA:................ 001930B5FD91
    Last Checkpoint RBA:...... 001930B5BBA6
    HPGRBLP RBA:.............. 001930B5C041
    Backup Date:.............. 03/01/2013
   Backup Time:.............. 2013-03-01-17.43.55.663032
    Consistency Method:....... DB2 System Level Backup
     Supports Object Restore:.. Yes
    I/O Suspend Time:......... 2013-03-01-17.43.55.573991
    I/O Resume Time:.......... 2013-03-01-17.43.58.428109
```

```
    Backup Elapsed:..... 02.85 Seconds

            IBM DB2 Recovery Expert for z/OS
              Backup Volume Detail Report

 <-Source Volumes-> <-Targets-> <--------Data Types-------->
 Volser Ucb# Devtyp Volser Ucb# Obj OCat ALog ACat RLog RCat
 ------ ---- ------ ------ ---- --- ---- ---- ---- ---- ----
 SBOXJI 611B 3390-7 X66726 6726 No  No  No  No  Yes  No
 SBOXJJ 621F 3390-7 X66227 6227 No  No  No  No  Yes  No
 SBOXJK D83B 3390-9 X5DAOD DAOD Yes No  No  No  No   No
 SBOXJL D90F 3390-9 SBOXKQ D31F Yes Yes No  No  No   No
.............
 XBD11C D11C 3390-9 XBD14E D14E Yes No  No  No  No   No

            IBM DB2 Recovery Expert for z/OS
               Restricted Objects Report

 <-Database-><-Space Name-><-Type-><-Partition-><-Status->
 ----------  ------------  -------  ------------  ----------
  ADMR4     GLWSDPT     TS    0001       RW,RECP
.............
  GLWSAMP   GLWXEPA1    IX    0003       RW,PSRBD
  GLWSAMP   GLWXEPA1    IX    0004       RW,PSRBD
  GLWSAMP   GLWXEPA1    IX    0005       RW,PSRBD
```

The first section, Backup Summary Report, contains the general information regarding the executed backup, for example, the name of the profile that was used for the backup, backup type, taken timestamp, and RBA.

The Backup Volume Detail Report section contains the volume mappings along with the information about the contents of each volume.

The Restricted Objects Report section contains a list of objects that were in restrictive states at the time of the backup. During the restore, Recovery Expert can generate a job that takes these objects out of the restrictive state.

8.3 System level restore

In this scenario, we use the SLB that was taken in “Execute DB2 system level backup through Recovery Expert” on page 373 to restore the subsystem. We demonstrate how Recovery Expert can drive RESTORE SYSTEM and, in addition, detect objects in restrictive states and generate a job that will recover the restricted table spaces and rebuild related indexes.

System level restore is initiated through option 2. System Restore and Offload in the Recovery Expert main menu. So, we start by typing 2 and pressing Enter. Recovery Expert prompts for the subsystem name. We enter DB0C and press Enter again. See Figure 8-48 on page 377.

```
RCVYXPRT V3R1 ------- IBM DB2 Recovery Expert for z/OS                  -------
Option  ===> 2

                                                        2013/03/01  18:20:48
                                                        User: ADMR4   - ARY

--------------------------------------------------------------------------------

     0. User Settings

     1. System Backup Profiles

     2. System Restore and Offload

     3. Object Profiles

     4. Disaster Recovery Profiles

     5. DB2 Subsystem Analysis and Configuration

     6. Coordinated Application Profiles

     X. Exit
```

Figure 8-48 System Restore and Offload

Next, Recovery Expert presents a panel with a list of SLBs taken for the selected database as shown on Figure 8-49 on page 378. For every SLB, Recovery Expert shows the type of the backup (DB2 or FlashCopy method), backup RBA, and backup date and time. In addition, it shows whether this backup is still on disk or has been offloaded to tape along with the possibility of object level restore.

Important: Recovery Expert currently relies on its own repository to list the existing backups. So, if you used BACKUP SYSTEM outside of Recovery Expert, it will not know about the resulting backups.

In addition, there are some implications to mixing SLBs taken in Recovery Expert and through the direct invocation of the BACKUP SYSTEM utility. For example, the FlashCopy SLB uses repository information to map source volumes to targets. It does not check DFSMShsm to find out whether the target volumes are used for any backups taken outside of Recovery Expert. This can lead to some target volumes that hold valid backup data being overwritten by the new backup.

When deleting a DB2 backup in Recovery Expert, it is deleted only from the Recovery Expert repository. Copypool backup generations are not deleted.

```
RCVYXPRT  V3R1   -------- Restore System Display -------- 2013/03/01  18:28:46
Option  ===>                                                Scroll ===> PAGE

     Commands: ? - Show all commands
Line Commands: S-Select for Restore  D-Delete        V-View Reports  O-Offload
               J-Object Report       H-Health Check  I-Image Copies

DB2 Subsystem ID  DBOC
--------------------------------------------------------------------------------
  Select a row from the list of recovery points displayed below.  To enter
  an RBA/LRSN and have a recovery point automatically selected for you,
  enter the RESTORE primary command followed by a DB2 subsystem ID.
--------------------------------------------------------------------------------
                                                                  Row 1 of 6    >
                           Data  Mixed  On    On       Obj   Partial
Cmd  Date        Time      Only  Data   Disk  Offload  Rcvr  Backup   Type
 S   03/01/2013  17:43:58  No    No     Yes   No       Yes   No       DB2
     SSID: DBOC  RBA/LRSN: 001930B5FD91
     03/01/2013  11:01:26  No    No     Yes   No       Yes   No       DB2
     SSID: DBOC  RBA/LRSN: 0019265F9C50
     02/27/2013  19:29:30  No    No     Yes   No       Yes   No       Flash
     SSID: DBOC  RBA/LRSN: 001919F2564B
***************************** Bottom of Data ********************************
```

Figure 8-49 List of SLBs for a subsystem

Recovery Expert presents a panel to select the recovery options as shown on Figure 8-50 on page 379. Recovery Expert can restore both data and logs. We restore data only. We also type Y for “Resolve Recover/Rebuild Pending Objects”, enabling the Recovery Expert to generate a job to recover/rebuild restricted objects after the system level restore.

We can also select a recovery point here, but for simplicity, we stay with the backup RBA and type N for “Select Recovery Point”. If we need to select a recovery point, we can type Y and Recovery Expert displays a panel with several options to identify the required RBA. The sources of the RBA are RBA to timestamp mappings gathered by Recovery Expert (if configured), archive log, checkpoint events, and timestamp to log record sequence number (LRSN) utility for data sharing configurations.

Tip: LRSN can be converted to timestamp using the following simple SQL statement:

```
SELECT TIMESTAMP (CONCAT (LRSN_HEX_STRING, X'0000')) FROM SYSIBM.SYSDUMMY1
```

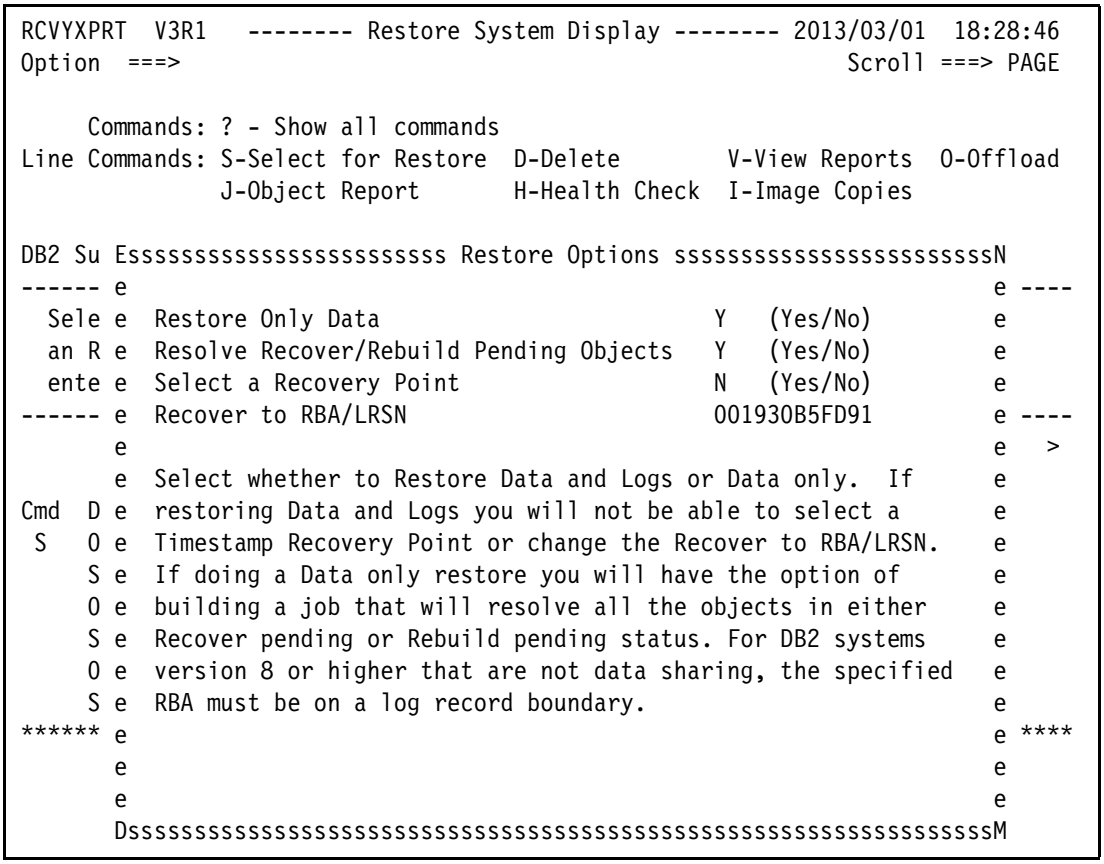

Figure 8-50 Restore options for an SLB

Generally, system level restore requires the following steps:

1. Create a conditional restart point.
2. Restore the volumes that are part of the subsystem.
3. Apply logs to roll the subsystem forward to the required RBA.
4. Recovery Expert will also add an optional step to recover/rebuild restricted objects after the restore.

Recovery Expert shows a panel to set the names for the partitioned data set (PDS) members that will hold the JCL for every step. See Figure 8-51 on page 380. We specify the PDS library to hold all members and enter a name to hold each step member.

```
RCVYXPRT  V3R1 ------------ Recovery Options ------------ 2013/03/01  18:31:41
Option   ===>
--------------------------------------------------------------------------------
Enter the Recovery options for the Recover/Rebuild pending job.

 O Esssssssssssssssssssssssss Build Restore Job sssssssssssssssssssssssssN
 V e                                                                      e
 R e   Edit Generated Job      Y           (Yes/No)                       e
 U e   Edit Recovery Options   N           (Yes/No)                       e
 P e                                                                      e
 N e   Build job in Dataset    ADMR4.ARY.CNTL                             e
 E e                Member     CRM         Conditional Restart Member     e
 O e                Member     RSM         Restore System Member          e
   e                Member     LRM         Log Restore Member             e
   e                Member     RPM         Recover/Rebuild Pending Member e
   e                                                                      e
   e    Job Cards:                                                        e
   e ==> //RJOBCRD JOB ADMR4,CLASS=A,NOTIFY=&SYSUID                       e
   e ==> /*JOBPARM S=SC63                                                 e
   e ==> //*                                                              e
   e ==> //*                                                              e
   e                                                                      e
   e                                                                      e
   e                                                                      e
   DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 8-51 PDS settings for the restore steps

The “create a conditional restart point” job first stops DB2 and then calls the change log inventory utility to create the conditional restart control record. Sample JCL excerpts are listed in Example 8-13.

Example 8-13 Create conditional restart control record

```
........
//*                                                                    *
//DB2STOP  EXEC PGM=ARY#SDB2,COND=(4,LT),PARM=(DBOC,STOP)
//STEPLIB  DD DISP=SHR,DSN=DBTLSP.SARYLOAD
//         DD DISP=SHR,DSN=DBOCT.SDSNEXIT
//         DD DISP=SHR,DSN=DBOCT.SDSNLOAD
//DB2PARMS DD DISP=SHR,DSN=DB2TOOLS.ARY.CONTROL
//SYSPRINT DD SYSOUT=*
//*
........
//ARYCRCR EXEC PGM=DSNJU003,REGION=006M,COND=(4,LT)
//*
//STEPLIB  DD DISP=SHR,DSN=DBOCT.SDSNEXIT
//         DD DISP=SHR,DSN=DBOCT.SDSNLOAD
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD DISP=SHR,DSN=DBOCB.BSDS01
//SYSUT2   DD DISP=SHR,DSN=DBOCB.BSDS02
//SYSIN    DD *
  CRESTART CREATE,SYSPITR=00192B309903,FORWARD=YES,BACKOUT=YES
/*
```

The DSNJU003 execution output is shown in Example 8-14.

Example 8-14 Create conditional restart record output

```
1DSNJCNVB CONVERSION PROGRAM HAS RUN   DDNAME=SYSUT1
 DSNJCNVB CONVERSION PROGRAM HAS RUN   DDNAME=SYSUT2
0  CRESTART CREATE,SYSPITR=00192B309903,FORWARD=YES,BACKOUT=YES
 DSNJ408I  DSNRJFCK CHECKPOINT RBA FOUND, RBA = 00192B300B8E, TIME = 06:47:55 MARCH 04, 2013
 DSNJ411I  DSNRJRCR CRESTART CREATE FOR CRCRID = 0001, DDNAME = SYSUT1
 DSNJ408I  DSNRJFCK CHECKPOINT RBA FOUND, RBA = 00192B300B8E, TIME = 06:47:55 MARCH 04, 2013
 DSNJ411I  DSNRJRCR CRESTART CREATE FOR CRCRID = 0001, DDNAME = SYSUT2
 DSNJ225I  CRESTART OPERATION COMPLETED SUCCESSFULLY
-DSNJ200I  DSNJU003 CHANGE LOG INVENTORY UTILITY PROCESSING COMPLETED SUCCESSFULLY
```

The next step is to execute the actual restore of the DB2 data, which is done by executing the JCL from the Restore System Member. The job has two steps. The first step restores the volumes by using fast replication and the second step starts DB2. Excerpts from the job are listed in Example 8-15.

Example 8-15 Restore job partial listing

```
.........
//ARYREST  EXEC  PGM=ARY@MAIN,REGION=006M,COND=(4,LT)
//*
//STEPLIB  DD DISP=SHR,DSN=DBTLSP.SARYLOAD
//DB2PARMS DD DISP=SHR,DSN=DB2TOOLS.ARY.CONTROL
//ARYBPROF DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILES
//ARYBOFFL DD DISP=SHR,DSN=DB2TOOLS.ARY.OFFOPTS
//ARYBPMAP DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILE.MAPS
//ARYBPCAT DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILE.CATS
//ARYSBACK DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK
//ARYSBOBJ DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.OBJS
//ARYSBVOL DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.VOLS
//ARYSBSSD DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.SSIDS
//ARYBREPT DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.REPORT
//ARYPOBJS DD DISP=SHR,DSN=DB2TOOLS.ARY.OBJECTS
//ARY#REPT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//ARYOUT   DD SYSOUT=*
//ARYSNAPO DD SYSOUT=*
//ARY#PARM DD DISP=SHR,
//         DSN=DBTLSP.SARYSAMP(ARY#PARM)
//ARYIN    DD *
   RESTORE "ADMR4"."DB2 SLB"
           GENERATION 01
           DATE 03/02/2013
           TIME 01:30:56
/*
.........
//*                                                                 *
//DB2START  EXEC PGM=ARY#SDB2,COND=(4,LT),PARM=(DB0C,START)
//STEPLIB  DD DISP=SHR,DSN=DBTLSP.SARYLOAD
//         DD DISP=SHR,DSN=DB0CT.SDSNEXIT
//         DD DISP=SHR,DSN=DB0CT.SDSNLOAD
//DB2PARMS DD DISP=SHR,DSN=DB2TOOLS.ARY.CONTROL
//SYSPRINT DD SYSOUT=*
```

```
//*
```

The ARYREST step output is shown in Example 8-16. The Recovery Expert disconnects the user catalogs and calls DFSMShsm to execute the fast replication.

Example 8-16 Restore system output

```
02:01:15 ARYS001I - IBM DB2 Recovery Expert for z/OS Starting. Version 03.01.001
02:01:15 ARYS003I - Control Cards:
02:01:15 ARYS004I -   RESTORE "ADMR4"."DB2 SLB"
02:01:15 ARYS004I -          GENERATION 01
02:01:15 ARYS004I -          DATE 03/02/2013
02:01:15 ARYS004I -          TIME 01:30:56
02:01:15 ARYS004I -
02:01:15 ARYS123I - Backup ADMR4.DB2 SLB generation 01 was read from the repository.
02:01:16 ARYS013I - Profile ADMR4.DB2 SLB was read from the repository.
02:01:16 ARYS283I - HSM backup with token C4C2F0C3CAFFB8DBEFE1072B00192A58C294 is on DASD and recoverable.
02:01:16 ARYS038I - Performing profile volume map validation...
02:01:16 ARYS146I - Removing volser SBOXJR from this restore.  It contains only log data.
02:01:16 ARYS146I - Removing volser SBOXJQ from this restore.  It contains only log data.
02:01:16 ARYS146I - Removing volser SBOXJJ from this restore.  It contains only log data.
02:01:16 ARYS146I - Removing volser SBOXJU from this restore.  It contains only log data.
02:01:16 ARYS146I - Removing volser X76029 from this restore.  It contains only log data.
02:01:16 ARYS146I - Removing volser SBOXJI from this restore.  It contains only log data.
02:01:16 ARYS146I - Removing volser SBOXJV from this restore.  It contains only log data.
02:01:16 ARYS039I - Volume map validation complete.
02:01:16 ARYS136I - Disconnecting user catalogs.
02:01:16 ARYS217I - User catalog UCAT.DBOCDN disconnected.
02:01:18 ARYS217I - User catalog UCAT.DBOCL disconnected.
02:01:26 ARYS275I - Restoring HSM copypool DSN$DBOC$DB
02:01:26 ARYS004I - FRRECOV CP(DSN$DBOC$DB) VERIFY(N) TOKEN(X'C4C2F0C3CAFFB8DBEFE1072B00192A58C294')
FROMDASD
02:01:35 ARYS002I - IBM DB2 Recovery Expert for z/OS complete. RC=000.
```

Recovery Expert also produces the restore report that is shown in Example 8-17. The report header contains the general information about the restore. The Restore Volume Detail Report section lists the volumes that have been used as sources and targets in FlashCopy operations.

Example 8-17 Restore report

```
1              IBM DB2 Recovery Expert for z/OS
                  Restore Summary Report

     Utility Executed:..... Restore
     Profile Name:..... ADMR4.DB2 SLB
     DB2 Subsystem:..... DBOC
     DB2 Version:..... 1010
    Restore Type:..... DB2 System Level Backup
    Restored:..... Object Data and Log Data
     Nbr of Volumes:..... 0016
   HSM Backup Token:..... C4C2F0C3CAFFB8DBEFE1072B00192A58C294
     Backup RBA:..... 00192A590000
1              IBM DB2 Recovery Expert for z/OS
                Restore Volume Detail Report

              <-Source Volumes-> <-Restore->
              Volser Ucb# Devtyp  Volser Ucb#
```

```
------ ---- ------ ------ ----
SBOXJK D83B 3390-9 X5DA0D DA0D
SBOXJL D90F 3390-9 SBOXKQ D31F
SBOXJM DA0F 3390-9 X8D002 D002
SBOXJN DB36 3390-9 SBOXKR D326
X66125 6125 3390-7 X66129 6129
X66226 6226 3390-7 X6622A 622A
X66326 6326 3390-7 X6632A 632A
X66327 6327 3390-7 X66325 6325
X66328 6328 3390-7 X66525 6525
X66329 6329 3390-7 X66528 6528
X66526 6526 3390-7 X6672A 672A
X6652A 652A 3390-7 X66529 6529
X66628 6628 3390-7 X66225 6225
X66727 6727 3390-7 X66725 6725
XBD01D D01D 3390-9 X8D400 D400
XBD11C D11C 3390-9 XBD14E D14E
```

The last step in the restore job starts the subsystem. The system prompts us to confirm the conditional restart by issuing "`WTOR: DBOA CONDITIONAL RESTART RECORD INDICATES TRUNCATION AT RBA 00192B309903. REPLY Y OR N`". We reply Y and continue with the next step, which is applying logs to roll forward the system to the specified RBA. The JCL job listing is shown in Example 8-18.

Example 8-18 Log apply and object recover/rebuild job partial listing

```
.........
//ARYLOG   EXEC PGM=DSNUTILB,REGION=006M,PARM=(DBOC,)
//*
//STEPLIB  DD DISP=SHR,DSN=DBOCT.SDSNEXIT
//         DD DISP=SHR,DSN=DBOCT.SDSNLOAD
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//SYSIN    DD *
   RESTORE SYSTEM LOGONLY
/*
..........
//RECPEND  EXEC PGM=IKJEFT1A,REGION=006M
//*
//STEPLIB  DD DISP=SHR,DSN=DBTLSP.SARYLOAD
//         DD DISP=SHR,DSN=DBOCT.SDSNEXIT
//         DD DISP=SHR,DSN=DBOCT.SDSNLOAD
//DB2PARMS DD DISP=SHR,DSN=DB2TOOLS.ARY.CONTROL
//ARYBPROF DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILES
//ARYBOFFL DD DISP=SHR,DSN=DB2TOOLS.ARY.OFFOPTS
//ARYBPMAP DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILE.MAPS
//ARYBPCAT DD DISP=SHR,DSN=DB2TOOLS.ARY.PROFILE.CATS
//ARYSBACK DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK
//ARYSBOBJ DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.OBJS
//ARYSBVOL DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.VOLS
//ARYSBSSD DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.SSIDS
//ARYBREPT DD DISP=SHR,DSN=DB2TOOLS.ARY.SYSBACK.REPORT
//ARYPOBJS DD DISP=SHR,DSN=DB2TOOLS.ARY.OBJECTS
//ARYPOBJS DD DISP=SHR,DSN=DB2TOOLS.ARY.OBJECTS
//ARY#PARM DD DISP=SHR,
```

```
//         DSN=DBTLSP.SARYSAMP(ARY#PARM)
//ISPLLIB  DD DISP=SHR,DSN=DBTLSP.SARYLOAD
//         DD DISP=SHR,DSN=ISP.SISPLOAD
//ISPPLIB  DD DISP=SHR,DSN=DBTLSP.SARYPENU
//ISPTLIB  DD DSN=&&TEMP,DISP=(OLD,DELETE,DELETE)
//         DD DISP=SHR,DSN=ISP.SISPTENU
//ISPMLIB  DD DISP=SHR,DSN=DBTLSP.SARYMENU
//         DD DISP=SHR,DSN=ISP.SISPMENU
//ISPSLIB  DD DISP=SHR,DSN=DBTLSP.SARYSLIB
//         DD DISP=SHR,DSN=ADMR4.ARY.CNTL
//ISPPROF  DD DSN=&&PROF,DISP=(NEW,DELETE),
//            UNIT=SYSALLDA,SPACE=(TRK,(2,1,2)),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=800)
//ISPLOG   DD SYSOUT=*,DCB=(RECFM=VA,LRECL=125,BLKSIZE=129)
//ISPWRK1  DD UNIT=SYSALLDA,SPACE=(CYL,(30,30)),
//            DCB=(RECFM=FB,LRECL=133,BLKSIZE=1330)
//ARYOUT   DD SYSOUT=*
//ARY#REPT DD SYSOUT=*
//ARYERROR DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSTSIN  DD *
  PROFILE  NOPREFIX
  ISPSTART PGM(ARY@PEND)
//*
//ARY#DATA DD *
  RESOLVE_PENDINGS     (                                                -
    OBJECTS_PER_STEP   00                                               -
    DB2_SUBSYSTEM      DB0C                                             -
    START_RBA          00192A590000                                     -
    END_RBA            00192B309903                                     -
    GEN_TO_DATASET     ADMR4.ARY.CNTL                                   -
    ICCHECK            N                                                -
    GEN_TO_MEMBER      RPM                                              -
    JOB_CARD_1_1      '//RJOBCRD JOB ADMR4,CLASS=A,NOTIFY=&SYSU'        -
    JOB_CARD_1_2      'ID'                                              -
    JOB_CARD_2_1      '/*JOBPARM S=SC63'                                -
    JOB_CARD_3_1      '//*'                                             -
    JOB_CARD_4_1      '//*'                                             -
    PARML_DSN          DBTLSP.SARYSAMP                                  -
    PARML_MEMBER       ARY#PARM                                         -
    DB2CNTL_DSN        DB2TOOLS.ARY.CONTROL                             -
    REUSE              Y                                                -
    PARALLEL           04                                               -
    TAPEUNITS          02                                               -
    IXREUSE            Y                                                -
    STATS              Y                                                -
    UPDATE_CAT         S                                                -
    UPDATE_HIST        S                                                -
    COLUMN_VALUES      Y                                                -
    REPORT             Y                                                -
    ONLINE_REBUILD     Y                                                -
    SHRLEVEL           C                                                -
    DRAIN_WAIT         0                                                -
    RETRY_DELAY        300                                              -
```

```
    MAXROWS           300                                          -
    LONG_LOG          C                                            -
    DELAY             1200                                         -
                      )
//*
```

The first step executes the RESTORE SYSTEM utility with the LOGONLY option. This utility invocation rolls forward the database to the RBA specified in the conditional restart control record. The second step generates the recover/rebuild index control statements to handle the objects in restrictive states. The resulting JCL is placed in the member that is specified in the Recover/Rebuild Pending Member field on Figure 8-51 on page 380. The RESTORE SYSTEM utility output is shown in Example 8-19.

Example 8-19 RESTORE SYSTEM output

```
1DSNU000I    063 02:02:53.13 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = ADMR4.RJOBCRD
 DSNU1044I   063 02:02:53.15 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
0DSNU050I    063 02:02:53.15 DSNUGUTC -  RESTORE SYSTEM LOGONLY
 DSNU1604I -DBOC 063 02:02:53.22 DSNUVARL - RESTORE SYSTEM PHASE LOG APPLY STARTED AT LOG POINT =
X'00192A58C294'.
 DSNU1629I -DBOC 063 02:03:04.20 DSNUVARL - DB2 PUT ONE OR MORE OBJECTS INTO THE RECOVER-PENDING STATE, THE
 REBUILD-PENDING STATE, OR THE LOGICAL PAGE LIST DURING THE LOG APPLY PHASE.
 DSNU1635I -DBOC 063 02:03:04.20 DSNUVARL - THE RBA RANGE FOR THE LAST CHECKPOINT ISSUED DURING THE
LOGAPPLY PHASE OF THE RESTORE SYSTEM UTILITY IS START_RBA = X'00192B30C776' END_RBA = X'00192B45D11C'
 DSNU1628I   063 02:03:04.20 DSNUVBRD - RESTORE SYSTEM PHASE LOG APPLY COMPLETED, ELAPSED TIME = 00:00:11.
 DSNU010I    063 02:03:04.20 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=4
```

The recover/rebuild pending generation step output is shown in Example 8-20.

Example 8-20 Recover/rebuild pending report

```
1                          IBM DB2 Recovery Expert for z/OS
                    System Recover Pending/Rebuild Pending Report
                                  Subsystem DBOC

 DBNAME/         TSNAME/         Part Type Event Type     IC Dataset/
 IX CRTR          IX NAME                                 Messages
 ________________ ________________ ____ ____ _____________ ____________________
 ADMR4          GLWSDPT        0000 TS  RW,RECP,UTRO   ADMR4.IC.ADMR4.GLWSDPT.D2013063
 ADMR4           GLWSEMP        0001 TS  RW,RECP       ADMR4.IC.ADMR4.GLWSEMP.D2013063
 ADMR4           GLWSEMP        0002 TS  RW,RECP       ADMR4.IC.ADMR4.GLWSEMP.D2013063
 ADMR4           GLWSEMP        0003 TS  RW,RECP       ADMR4.IC.ADMR4.GLWSEMP.D2013063
 ADMR4           GLWSEMP        0004 TS  RW,RECP       ADMR4.IC.ADMR4.GLWSEMP.D2013063
 ADMR4           GLWSEPA        0001 TS  RW,RECP       ADMR4.IC.ADMR4.GLWSEPA.D2013063
 ADMR4           GLWSPJA        0000 TS  RW,RECP       ADMR4.IC.ADMR4.GLWSPJA.D2013063
 ADMR4           GLWSPRJ        0000 TS  RW,RECP       ADMR4.IC.ADMR4.GLWSPRJ.D2013063
 ADMR4           GLWSSPL        0001 TS  RW,RECP       ADMR4.IC.ADMR4.GLWSSPL.D2013063
 ADMR4           GLWSSPL        0002 TS  RW,RECP       ADMR4.IC.ADMR4.GLWSSPL.D2013063
 ADMR4           GLWSSPL        0003 TS  RW,RECP       ADMR4.IC.ADMR4.GLWSSPL.D2013063
 ADMR4           GLWSSPL        0004 TS  RW,RECP       ADMR4.IC.ADMR4.GLWSSPL.D2013063
 ADMR4           GLWS001        0000 TS  RW,RECP       ADMR4.IC.ADMR4.GLWS001.D2013063
 ADMR4           GLWS002        0000 TS  RW,RECP       ADMR4.IC.ADMR4.GLWS002.D2013063
 ADMR4           GLWS003        0000 TS  RW,RECP       ADMR4.IC.ADMR4.GLWS003.D2013063
 ADMR4           XGLW0000       0001 TS  RW,RECP       ADMR4.IC.ADMR4.XGLW0000.D2013063
 ADMR4           XGLW0000       0002 TS  RW,RECP       ADMR4.IC.ADMR4.XGLW0000.D2013063
 ADMR4           XGLW0000       0003 TS  RW,RECP       ADMR4.IC.ADMR4.XGLW0000.D2013063
 ADMR4           XGLW0000       0004 TS  RW,RECP       ADMR4.IC.ADMR4.XGLW0000.D2013063
 ADMR4            GLWXACT1       0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4            GLWXDNG1       0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4            GLWXDPT1       0000 IX  RW,RBDP,UTRO   Index will be rebuilt.
 ADMR4            GLWXDPT2       0000 IX  RW,RBDP,UTRO   Index will be rebuilt.
 ADMR4            GLWXDPT3       0000 IX  RW,RBDP,UTRO   Index will be rebuilt.
```

```
 ADMR4         GLWXDPT4       0000 IX  RW,RBDP,UTRO   Index will be rebuilt.
 ADMR4          GLWXEMP1        0001 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXEMP1        0002 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXEMP1        0003 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXEMP1        0004 IX  RW,RBDP       Index will be rebuilt.
1                        IBM DB2 Recovery Expert for z/OS
                   System Recover Pending/Rebuild Pending Report
                                Subsystem DBOC

   AME/         TSNAME/         Part Type Event Type     IC Dataset/
 IX CRTR          IX NAME                              Messages
 _____________ _____________________ ____ ____ _____________ ________________________
 ADMR4          GLWXEMP2        0000 IX  RW,RBDP,PSRBD   Index will be rebuilt.
 ADMR4          GLWXEMP3        0000 IX  RW,RBDP,PSRBD   Index will be rebuilt.
 ADMR4          GLWXEMP4        0000 IX  RW,RBDP,PSRBD   Index will be rebuilt.
 ADMR4          GLWXEMP5        0000 IX  RW,RBDP,PSRBD   Index will be rebuilt.
 ADMR4          GLWXEMP6        0000 IX  RW,RBDP,PSRBD   Index will be rebuilt.
 ADMR4          GLWXENG1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXEPA1        0000 IX  RW,RBDP,PSRBD   Index will be rebuilt.
 ADMR4          GLWXEPA2        0000 IX  RW,RBDP,PSRBD   Index will be rebuilt.
 ADMR4          GLWXEPA3        0000 IX  RW,RBDP,PSRBD   Index will be rebuilt.
 ADMR4          GLWXJBS1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXLCN1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXLNG1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXLNM1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXPGW1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXPJA1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXPNG1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXPRJ1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXPRJ2        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXPRJ3        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXSFN1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXSQL1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXSTR1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXTWN1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4          GLWXVRN1        0000 IX  RW,RBDP       Index will be rebuilt.
 ADMR4         I_DOCIDGLWTEMP   0000 IX  RW,RBDP,PSRBD   Index will be rebuilt.
ADMR4         I_NODEIDXGLWTEMP  0000 IX  RW,RBDP,PSRBD   Index will be rebuilt.
 GLWSAMP             GLWXEPA1           0000 IX   RW,PSRBD         Index will be rebuilt.
```

Recovery Expert was able to generate the recover and rebuild control statements for all the objects in the restrictive states. The JCL is placed in ADMR4.ARY.CNTL(RPM). A partial listing of this member is shown in Example 8-21.

Example 8-21 Recover/rebuild job partial listing

```
...........
//RCVRFRIC EXEC  PGM=DSNUTILB,REGION=006M,COND=(4,LT),
//           PARM=(DBOC)
//*
//STEPLIB  DD DISP=SHR,DSN=DBOCT.SDSNEXIT
//         DD DISP=SHR,DSN=DBOCT.SDSNLOAD
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//*
//SYSIN    DD *
   RECOVER
       TABLESPACE ADMR4.GLWSDPT
       TABLESPACE ADMR4.GLWSEMP     DSNUM(0001)
       TABLESPACE ADMR4.GLWSEMP     DSNUM(0002)
       TABLESPACE ADMR4.GLWSEMP     DSNUM(0003)
```

```
       TABLESPACE ADMR4.GLWSEMP     DSNUM(0004)
       TABLESPACE ADMR4.GLWSEPA     DSNUM(0001)
       TABLESPACE ADMR4.GLWSPJA
       TABLESPACE ADMR4.GLWSPRJ
       TABLESPACE ADMR4.GLWSSPL     DSNUM(0001)
       TABLESPACE ADMR4.GLWSSPL     DSNUM(0002)
       TABLESPACE ADMR4.GLWSSPL     DSNUM(0003)
       TABLESPACE ADMR4.GLWSSPL     DSNUM(0004)
       TABLESPACE ADMR4.GLWSO01
       TABLESPACE ADMR4.GLWSO02
       TABLESPACE ADMR4.GLWSO03
       TABLESPACE ADMR4.XGLWOOOO    DSNUM(0001)
       TABLESPACE ADMR4.XGLWOOOO    DSNUM(0002)
       TABLESPACE ADMR4.XGLWOOOO    DSNUM(0003)
       TABLESPACE ADMR4.XGLWOOOO    DSNUM(0004)
           PARALLEL(04)
           TAPEUNITS(02)
           REUSE
           LOCALSITE
.........
//REBUILD  EXEC  PGM=DSNUTILB,REGION=006M,COND=(4,LT),
//          PARM=(DB0C)
//*
//STEPLIB  DD DISP=SHR,DSN=DB0CT.SDSNEXIT
//         DD DISP=SHR,DSN=DB0CT.SDSNLOAD
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//*
//SYSIN    DD *
   REBUILD
       INDEX (
              "ADMR4"."GLWXACT1"
             )
           REUSE
           SORTDEVT     SYSALLDA
           SORTNUM      6
           SORTKEYS
           SHRLEVEL     CHANGE
           DRAIN_WAIT   0
           RETRY_DELAY  300
           MAXRO        300
           LONGLOG      CONTINUE
           DELAY        1200
           STATISTICS
             REPORT     YES
             KEYCARD
             UPDATE     SPACE
             HISTORY    SPACE
.........
```

After executing the job, the DB0C subsystem has no objects in restrictive states. Recovery Expert fully automated the complex recovery scenario and brought the subsystem to a fully functional state.

8.4 Object recovery

Our object recovery scenario is based on the stored procedure workload database used throughout the book. Detailed information is in Appendix A.2, “Stored procedures workload” on page 537. We start by building an Object Profile and selecting objects to restore. Next, we look at parallelism settings and how to control the recovery execution through the Recovery Expert ISPF panels.

In this scenario, we restore the GLWEMP (employee) table to a point in time. First, we need to define an Object Profile in Recovery Expert and add this table to the profile. We select option 3. Object Profiles in the Recovery Expert main menu. See Figure 8-52.

```
RCVYXPRT V3R1 ------- IBM DB2 Recovery Expert for z/OS                  -------
Option  ===> 3

                                                        2013/03/04  16:50:21
                                                        User: ADMR4   - ARY

--------------------------------------------------------------------------------

     0. User Settings

     1. System Backup Profiles

     2. System Restore and Offload

     3. Object Profiles

     4. Disaster Recovery Profiles

     5. DB2 Subsystem Analysis and Configuration

     6. Coordinated Application Profiles

     X. Exit
```

Figure 8-52 Recovery Expert main menu

Recovery Expert lists the existing Object Profiles. In our case, this will be the first profile for the DB0A subsystem, so Recovery Expert prompts to create a profile as shown on Figure 8-53 on page 389.

```
 RCVYXPRT V3R1 ---------- Object Profile Display --------- 2013/04/02  03:15:09
 Command ===>                                                Scroll ===> PAGE

 Line Commands: C - Create  U - Update  Q - Quiet Time      V - View
                D - Delete  R - Rename  G - DDL Generation  X - Copy
                M - Import  E - Export  P - Recovery Plans  I - Image Copy

--------------------------------------------------------------------------------
 Creator Like ADMR4    Name Like *                            SSID Like DBOA
 ---------------------------------------------------         Row 1 of 1    >

   Name                             Creator  SSID Updt Description
 C Press Enter to Create a Profile

EssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssN
e ARYR803I - No Profiles were found that match your selection criteria. Press e
e enter to create a new Profile or change the selection criteria.             e
DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 8-53 Create a new Object Profile

Recovery Expert presents a panel with basic settings for a new profile. See Figure 8-54.

```
RCVYXPRT V3R1 ---------- Object Profile Display --------- 2013/03/04  16:51:41
C Essssssssssssssssss Create Object Profile sssssssssssssssssN croll ===> PAGE
  e  Creator           ADMR4                                  e
L e                                                           e View
  e  Profile Name      GLWNEW EMP                             e Copy
  e                                                           e Image Copy
- e  Description       RESTORE EMP TABLE FROM GLW             e ----------------
C e                                                           e  SSID Like DBOA
- e  DB2 SSID          DBOA  (? for subsystem list)           e Row 1 of 1    >
  e                                                           e
  e  LogBased Drop Rec N     (Yes/No)                         e n
C e                                                           e
  e  Share option      U     (Update, View only, No access)   e
  e                                                           e
  e                                                           e
  e                                                           e
  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 8-54 Basic profile settings

We enter the profile name, give it a meaningful description, and set the Share option to U for Update by other users. Next, Recovery Expert presents a panel to select the objects to be included in the Object Profile. See Figure 8-55 on page 390. We can select from a wide range of object types, including databases, table spaces, tables, and indexes. For this scenario, we select a table space that holds the required table by typing S next to Tablespaces.

```
RCVYXPRT V3R1 ---------- Object Types Selection --------- 2013/03/04  16:53:29
Command ===>                                                Scroll ===> PAGE

Line Commands: S - Select an object type
------------------------------------------------------------------------------
Creator: ADMR4    Name: GLWNEW EMP                                  SSID: DBOA
---------------------------------------------------        Row 1 of 17
  Object Types
  Stogroups
  Databases
S Tablespaces
  Tables
  Indexes
  Views
  Synonyms
  Aliases
  Data Types
  Triggers
  Functions
  Procedures
  Sequences
  Roles
  Plans
  Packages
  Auto Tool Profiles
```

Figure 8-55 Add objects to the profile

When we press Enter, Recovery Expert presents a panel with filtering options where we can select a database and enter a wildcard for the table space name. We know that the table that we want to restore belongs to the GLWNEW database, so we enter this value as the database name. We specify % to list all of the table spaces that belong to this database (Figure 8-56 on page 391).

Tip: Object Profiles can hold wildcard specifications as their members. This capability allows Recovery Expert to build the list of objects during recovery job generation. You can, for example, specify that all table spaces that belong to APP1 start with TSAPP1% and include this mask in the profile. This profile definition does not have to be altered after the addition of a new table space.

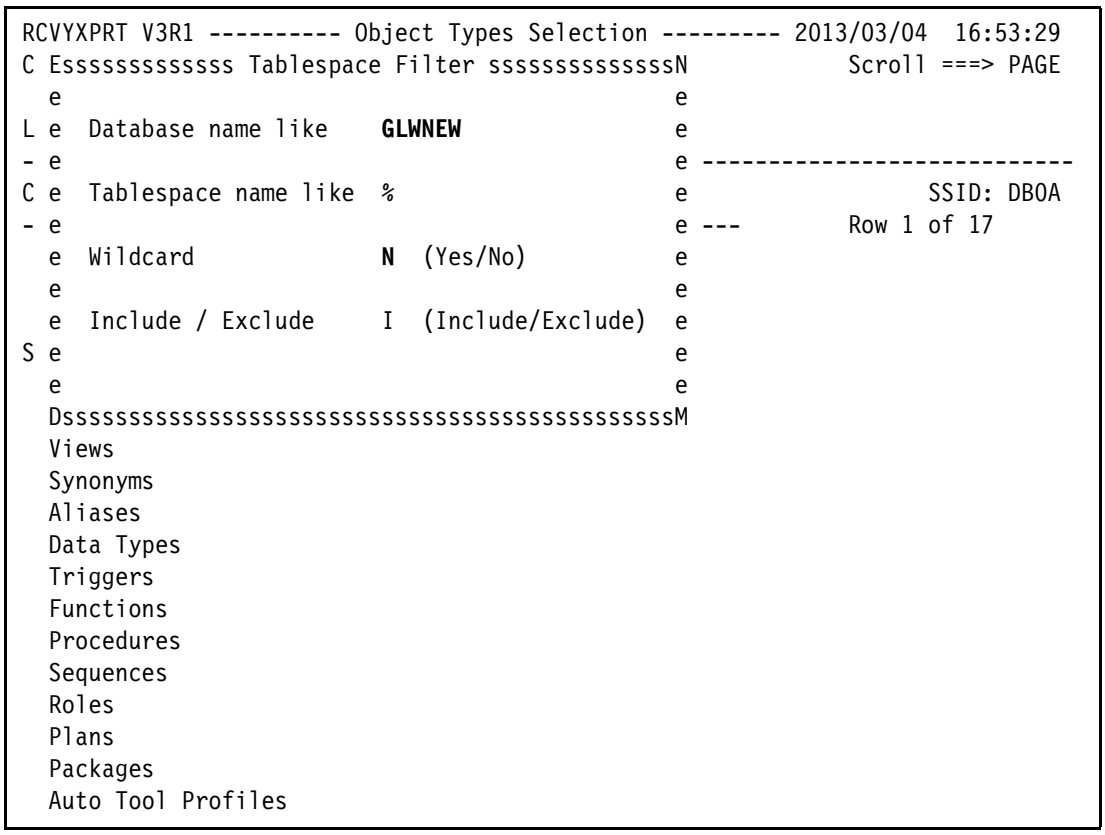

Figure 8-56 Add a table space to an Object Profile

Recovery Expert lists the table spaces that belong to the GLWNEW database as shown on Figure 8-57 on page 392. We type S next to the GLWSEMP table space and press Enter.

```
RCVYXPRT V3R1 ----------- Tablespace Selection ---------- 2013/03/04  16:55:36
Command ===>                                                Scroll ===> PAGE

Line Commands: S - Select  V - Versions  P - Properties
------------------------------------------------------------------------------
Creator: ADMR4    Name: GLWNEW EMP                                  SSID: DBOA
Database name like GLWNEW   Tablespace name like %
-----------------------------------------------------        Row 1 of 23
  Name     DBname   Part Status  Altered Timestamp   Created Timestamp
  GLWS001  GLWNEW      0 Both    2013-03-04-12.34.30 2013-03-04-12.34.30
  GLWS002  GLWNEW      0 Both    2013-03-04-12.34.31 2013-03-04-12.34.31
  GLWS003  GLWNEW      0 Both    2013-03-04-12.34.32 2013-03-04-12.34.32
  GLWSDPT  GLWNEW      0 Both    2013-03-04-12.34.24 2013-03-04-12.34.24
S GLWSEMP  GLWNEW    All Both    2013-03-04-12.34.26 2013-03-04-12.34.26
  GLWSEMP  GLWNEW      1 Both
  GLWSEMP  GLWNEW      2 Both
  GLWSEMP  GLWNEW      3 Both
  GLWSEMP  GLWNEW      4 Both
  GLWSEPA  GLWNEW    All Both    2013-03-04-12.34.28 2013-03-04-12.34.28
  GLWSEPA  GLWNEW      1 Both
  GLWSPJA  GLWNEW      0 Both    2013-03-04-12.34.29 2013-03-04-12.34.29
  GLWSPRJ  GLWNEW      0 Both    2013-03-04-12.34.29 2013-03-04-12.34.29
  GLWSSPL  GLWNEW    All Both    2013-03-04-12.34.29 2013-03-04-12.34.29
  GLWSSPL  GLWNEW      1 Both
  GLWSSPL  GLWNEW      2 Both
  GLWSSPL  GLWNEW      3 Both
  GLWSSPL  GLWNEW      4 Both
  XGLW0000 GLWNEW    All Both    2013-03-04-12.34.26 2013-03-04-12.34.26
  XGLW0000 GLWNEW      1 Both
  XGLW0000 GLWNEW      2 Both
  XGLW0000 GLWNEW      3 Both
  XGLW0000 GLWNEW      4 Both
```

Figure 8-57 List of table spaces that belong to a database

The Object Profile is shown on Figure 8-58 on page 393. The GLWSEMP table space has been added to the Object Profile.

```
RCVYXPRT V3R1 ---------- Update Object Profile ---------- 2013/03/04  16:56:15
Command ===>                                                Scroll ===> PAGE

Commands: EXPLODE
Line Commands: A - Add  R - Remove  V - Versions  P - Properties  U - Update
               I - Include  E - Exclude  X - Explode
--------------------------------------------------------------------------------
Creator: ADMR4    Name: GLWNEW EMP                                  SSID: DBOA
Share option U  (Upd, View, No)  Description RESTORE EMP TABLE FROM GLW
Update IC Options N  (Yes/No)    Update Recovery Options N  (Yes/No)
LBDR N  (Yes/No/Update) Start: N/A                 End: N/A
-------------------------------------------------          Row 1 of 1    >
  Obj  Name                         Schema/Creator     Part Wild Inc Status  A
  Type                              DBName                  Card Exc
  TS   GLWSEMP                      GLWNEW              All No   Inc Both    2
```

Figure 8-58 Object profile with the required table space

DB2 Recovery Expert can generate various recovery plans, including object level restore from SLBs, regular recovery, DSN1COPY with redo SQL, and others. The full list of recovery options is available in Chapter 16. “Recovering a DB2 object using the schema level repository” of *IBM DB2 Recovery Expert for z/OS Version 3 Release 1 User’s Guide*, SC19-3687. Every recovery plan is ranked according to its cost. After the profile has been defined, we type P next to the profile to generate the recovery plans. See Figure 8-59.

```
RCVYXPRT V3R1 ---------- Object Profile Display --------- 2013/03/04  16:56:53
Command ===>                                                Scroll ===> PAGE

Line Commands: C - Create  U - Update  Q - Quiet Time      V - View
               D - Delete  R - Rename  G - DDL Generation  X - Copy
               M - Import  E - Export  P - Recovery Plans  I - Image Copy
--------------------------------------------------------------------------------
Creator Like ADMR4    Name Like *                            SSID Like DBOA
-----------------------------------------------------         Row 1 of 1    >

  Name                            Creator  SSID Updt Description
P GLWNEW EMP                      ADMR4    DBOA U    RESTORE EMP TABLE FROM GL
```

Figure 8-59 Generate recovery plans

DB2 Recovery Expert presents a panel to set the recovery options and recovery point as shown on Figure 8-60 on page 394. The recovery point selection options have been described earlier in the chapter. For this scenario, we type Y next to Update Recovery Options and select 1 (Current), which is the default, for the Recovery Point. We press Enter.

```
RCVYXPRT V3R1 --------- Generate Recovery Plans --------- 2013/03/04  16:57:37
Command ===>
-------------------------------------------------------------------------------
Creator: ADMR4    Name: GLWNEW EMP                                  SSID: DBOA
-------------------------------------------------------------------------------

Update Recovery Options ==> Y  (Yes/No)

Choose the desired recovery point.  You can enter a timestamp or RBA/LRSN
here or enter "Y" for 'Select Recovery point' to display a selection list
of possible recovery points.

Recovery Point          ==> 1  (1-Current, 2-RBA/LRSN, 3-Timestamp)
  Recovery RBA/LRSN     ==>
  Recovery Timestamp    ==>      -    -    -    :    :    :
  Select Recovery Point ==> N  (Yes/No)
```

Figure 8-60 Generate Recovery Plans

Two important settings shown on the Recovery Options panel (Figure 8-61 on page 395) are the number of parallel jobs and the number of concurrent jobs. Recovery Expert can split work into the number of pieces specified by the “Number of parallel jobs” parameter. When doing so, it does not account for the size of objects to be recovered. The splitting is done based on the number of objects. Since object size distribution is usually non-uniform, there is a second parameter that allows us to set the number of concurrent job executions, which is “Number of concurrent jobs”. For a large set of objects, the recommendation is to set the number of parallel jobs to several times higher than the number of concurrent executions. This way, it is more likely that the recovery will be executed in parallel despite non-uniform object size distribution. Since we are restoring only a few small objects, we will set parallelism to 3 and concurrent jobs to 2.

```
RCVYXPRT V3R1 ------------ Recovery Options ------------ 2013/03/04  17:03:27
Command ===>
-------------------------------------------------------------------------------
Creator: ADMR4    Name: GLWNEW EMP                                  SSID: DBOA
-------------------------------------------------------------------------------
                                                                    More:     +
General Options:
  Stop if restricted objects  ==> Y  (Yes/No)

Recover Options:
  Number of Tape Drives       ==> 4  (1-99)
  Site                        ==> D  (Default/Local site/Recovery site)

Copy Options:
  Image Copies after recover  ==> N  (No/Tablespaces/Indexes and tablespaces)
    Local Site Primary        ==> N  (Yes/No)
    Local Site Backup         ==> N  (Yes/No)
    Remote Site Primary       ==> N  (Yes/No)
    Remote Site Backup        ==> N  (Yes/No)
  Number of Tape Drives       ==> 4  (1-99)
  Check Pages                 ==> N  (Yes/No)
  Use DFSMSdss Concurrent Copy==> N  (Yes/No)

Parallel Job Options:
  Number of parallel jobs     ==> 3  (0-99)
  Number of concurrent jobs   ==> 2 (1-99)
```

Figure 8-61 Recovery Options

After we press Enter, Recovery Expert will generate and present the possible recovery plans as shown on Figure 8-62.

```
RCVYXPRT V3R1 -------------- Recovery Plans ------------- 2013/03/04  17:01:31
Command ===>                                              Scroll ===> PAGE

Line Commands: V - Validate  P - Properties  D - Details  B - Build JCL
-------------------------------------------------------------------------------
Creator: ADMR4    Name: GLWNEW EMP                                   SSID: DBOA
-----------------------------------------------------          Row 1 of 6
  Plan Name                                                  Cost
  Using Restore from System Level Backup and RECOVER LOGONLY  89.48
  Using RECOVER                                              238.19
  Using DSN1COPY and RECOVER LOGONLY                         240.49
  Using RECOVER to IC and redo SQL                           1735.00
  Using DSN1COPY of IC and redo SQL                          1736.54
  Recovered Objects                                          n/a
```

Figure 8-62 Recovery plans

Important: When building the recovery plans, Recovery Expert also includes all objects that have a foreign key relationship with the tables in the Object Profile. There is no way to exclude them in the ISPF interface, but you can exclude them in the web-based GUI.

The least expensive plan according to Recovery Expert is to restore the objects from the SLB that was taken during the previous scenario. This makes sense because there have been almost no changes to the objects after the backup was taken. In this plan, Recovery Expert will use FlashCopy to restore the data set and then roll them forward with RECOVER LOGONLY. We choose this option. Detailed information about the plan steps can be accessed by typing D next the chosen recovery plan. See Figure 8-63.

```
RCVYXPRT V3R1 -------------- Recovery Plans ------------- 2013/03/04  17:08:35
Command ===>                                                Scroll ===> CSR

Line Commands: V - Validate  P - Properties  D - Details  B - Build JCL
------------------------------------------------------------------------------
Creator: ADMR4    Name: GLWNEW EMP                                   SSID: DBOA
-----------------------------------------------------          Row 1 of 6
  Plan Name                                                    Cost
D Using Restore from System Level Backup and RECOVER LOGONLY   89.48
  Using RECOVER                                                238.19
  Using DSN1COPY and RECOVER LOGONLY                           240.49
  Using RECOVER to IC and redo SQL                             1735.00
  Using DSN1COPY of IC and redo SQL                            1736.54
  Recovered Objects                                            n/a
```

Figure 8-63 Display recovery plan details

The recovery plan details list all the parallel job groups along with the detailed description of actions that constitute every group. See Figure 8-64 on page 397, which shows the full details only for parallel job group 1. We can see that four table space partitions are restored from the SLB. Then, the RECOVER utility is executed against these table spaces with the LOGONLY option to roll them forward to the required RBA. All of the indexes that belong to the recovered objects have been split into three groups, and the first four of them are rebuilt as part of parallel group 1.

```
RCVYXPRT V3R1 ---------- Recovery Plan Details ---------- 2013/03/04  17:07:33
Command ===>                                                  Scroll ===> CSR

Line Commands: P - Properties

Plan Name: Using Restore from System Level Backup and RECOVER LOGONLY
------------------------------------------------------------------------------
Creator: ADMR4    Name: GLWNEW EMP                                  SSID: DBOA
-------------------------------------------------     Row 104 of 282  -+

   Parallel Job Group
    Parallel Job 1
     Restore from system level backup
      Tablespace partition GLWNEW.GLWSDPT.0
       System Level Backup
      Tablespace partition GLWNEW.GLWSEMP.3
       System Level Backup
      Tablespace partition GLWNEW.GLWSPJA.0
       System Level Backup
      Tablespace partition GLWNEW.XGLWOOOO.2
       System Level Backup
     RECOVER
      Tablespace partition GLWNEW.GLWSDPT.0
      Tablespace partition GLWNEW.GLWSEMP.3
      Tablespace partition GLWNEW.GLWSPJA.0
      Tablespace partition GLWNEW.XGLWOOOO.2
     REBUILD INDEX
      Index GLWNEW.GLWXDPT1
      Index GLWNEW.GLWXDPT2
      Index GLWNEW.GLWXDPT3
      Index GLWNEW.GLWXDPT4
      Index GLWNEW.GLWXPJA1
    Parallel Job 2
     Restore from system level backup
      Tablespace partition GLWNEW.GLWSEMP.1
       System Level Backup
      Tablespace partition GLWNEW.GLWSEMP.4
       System Level Backup
      Tablespace partition GLWNEW.GLWSPRJ.0
       System Level Backup
      Tablespace partition GLWNEW.XGLWOOOO.3
       System Level Backup
```

Figure 8-64 Recovery plan details

Looking at the recovery plan details, we can better understand what Recovery Expert is planning to execute in every step. Users also have an option to view the details of every object that is being restored by typing D next to the object name.

After we review the details of the selected recovery plan, we can generate the JCL for every parallel job group. We exit the plan details and type B next to the selected recovery plan. See Figure 8-65 on page 398.

```
RCVYXPRT V3R1 -------------- Recovery Plans ------------- 2013/03/04  17:08:35
Command ===>                                                Scroll ===> CSR

Line Commands: V - Validate  P - Properties  D - Details  B - Build JCL
------------------------------------------------------------------------------
Creator: ADMR4    Name: GLWNEW EMP                                  SSID: DBOA
-----------------------------------------------------          Row 1 of 6
  Plan Name                                                    Cost
B Using Restore from System Level Backup and RECOVER LOGONLY   89.48
  Using RECOVER                                                238.19
  Using DSN1COPY and RECOVER LOGONLY                           240.49
  Using RECOVER to IC and redo SQL                             1735.00
  Using DSN1COPY of IC and redo SQL                            1736.54
  Recovered Objects                                            n/a
```

Figure 8-65 Build JCL

Recovery Expert will prompt for the name of the library to hold the JCL members and display an editable JOBCARD that will be used for every job. After the required details are provided, Recovery Expert presents the Recovery Plan Jobs panel that allows you to manipulate the generated jobs. See Figure 8-66.

```
RCVYXPRT V3R1 ------------ Recovery Plan Jobs ----------- 2013/03/04  18:11:18
Command ===>                                                Scroll ===> PAGE

Line Commands: B - Browse  E - Edit  V - View  S - Submit

Plan Name: Using Restore from System Level Backup and RECOVER LOGONLY
------------------------------------------------------------------------------
Creator: ADMR4    Name: GLWNEW EMP                                  SSID: DBOA
-----------------------------------------------------          Row 1 of 7

S Recover Job Group
   Serial Job 1 - ADMR4.ARY.CNTL(RECA01)
   Parallel Job Group
    Parallel Job 1 - ADMR4.ARY.CNTL(RECB01)
    Parallel Job 2 - ADMR4.ARY.CNTL(RECB02)
    Parallel Job 3 - ADMR4.ARY.CNTL(RECB03)
   Serial Job 2 - ADMR4.ARY.CNTL(RECC01)
```

Figure 8-66 Recovery Plan Jobs

We have an option to edit each job and submit them either one at a time or submit the whole group. If we opt for the whole group, Recovery Expert will drive the process and make sure that no more than the specified number of concurrent jobs will run in parallel. We will submit on the recovery job group level, so we type S at the top of the tree (next to Recover Job Group), as shown on Figure 8-66. Recovery Expert will start the execution and notify us about the result of each job. See Example 8-22 on page 399.

Example 8-22 Recover job results notification

```
18.43.42 JOB19774 $HASP165 ARYJOB   ENDED AT WTSCPLX2  MAXCC=0000 CN(INTERNAL)
18.43.51 JOB19778 $HASP165 ARYJOC   ENDED AT WTSCPLX2  MAXCC=0000 CN(INTERNAL)
18.43.51 JOB19775 $HASP165 ARYJOB   ENDED AT WTSCPLX2  MAXCC=0000 CN(INTERNAL)
18.43.54 JOB19779 $HASP165 ARYJOD   ENDED AT WTSCPLX2  MAXCC=0000 CN(INTERNAL)
18.44.09 JOB19780 $HASP165 ARYJOB   ENDED AT WTSCPLX2  MAXCC=0000 CN(INTERNAL)
***
```

All of the jobs executed without errors. The selected table and its foreign key-related tables have been successfully restored to the set point in time.

8.5 Log-based dropped object recovery using web interface

We describe the log-based recovery, which is the new feature of Recovery Expert 3.1. Before version 3.1, dropped object recovery relied on a schema level repository (SLR) that had to be updated periodically to capture schema details. The user had to schedule a periodic job to capture the object details and save them to the SLR. If, for some reason, the SLR update job was not executed or the dropped object was created after the last SLR update job execution, there was no way to recover the object.

The new method is based on finding the information about the dropped object in the transaction logs, creating the dropped table, reloading the table from image copy, and generating redo SQL to bring the object to the required point in time. The only requirement for this recovery method is the existence of an image copy. Theoretically, it is possible to restore the table by re-creating the table and applying all of the SQL against the table from the creation moment up to the drop. But, practically, this is nearly impossible for the majority of cases because it means scanning a huge range of logs, some of which might have been deleted. This is why Recovery Expert needs an image copy as a starting point for SQL redo.

In this scenario, we will drop the GLWACT table from the GLWNEW database and recover it by using the log-based recovery. This scenario can be done through the ISPF interface, but this time, we will demonstrate the capabilities of the Recovery Expert web interface.

We begin by logging on to the IBM DB2 Recovery Expert web interface and clicking Log Based Recovery, as shown on Figure 8-67 on page 400.

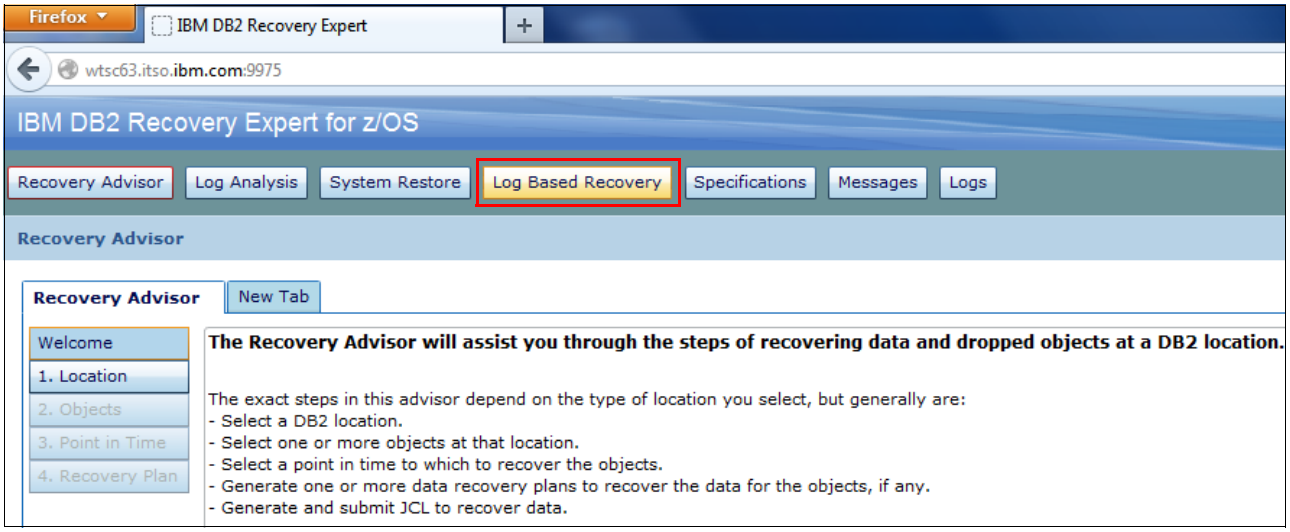

Figure 8-67 Log based recovery

The next step is to select the location (subsystem) that holds the dropped object. We click 1. Location on the left part of the panel. Recovery Expert will list all subsystems on the LPAR as shown on Figure 8-68. We select DB0A, which is the location of the GLWNEW database.

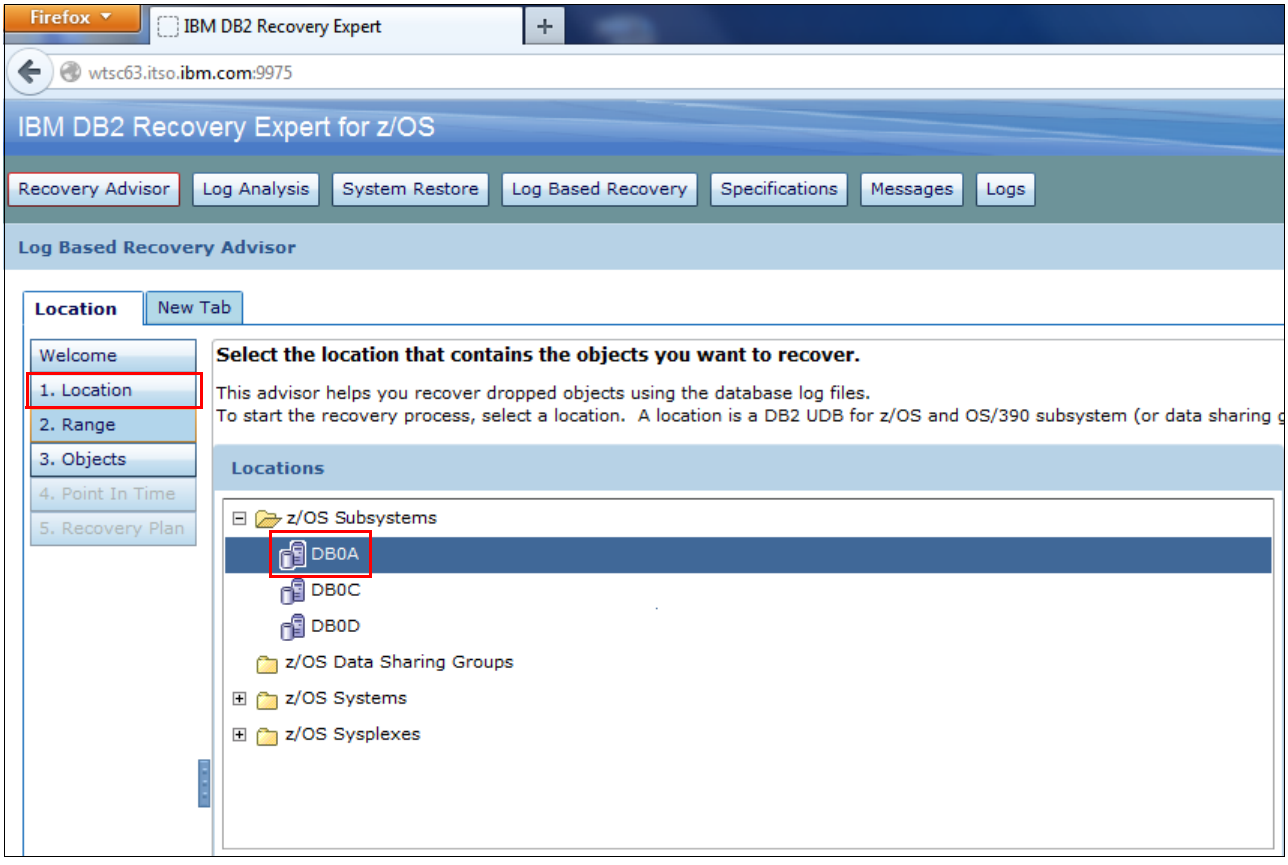

Figure 8-68 Select DB2 subsystem

Next, we need to select the range that will be scanned by Recovery Expert in search of the dropped objects. We can either use a previously scanned range or execute a new scan by specifying the time range. The time range can be specified either in the number of preceding

minutes/hours or by specifying start and end dates and times. Because we have just dropped the table that is going to be recovered, we select to create a new scan based on the preceding 10 minutes. See Figure 8-69.

Tip: If you choose not to use SYSLGNRX, try to keep the scanned range as narrow as possible. Depending on the number of changes that are generated in your subsystem, scanning a long range for dropped tables can be time and resource consuming.

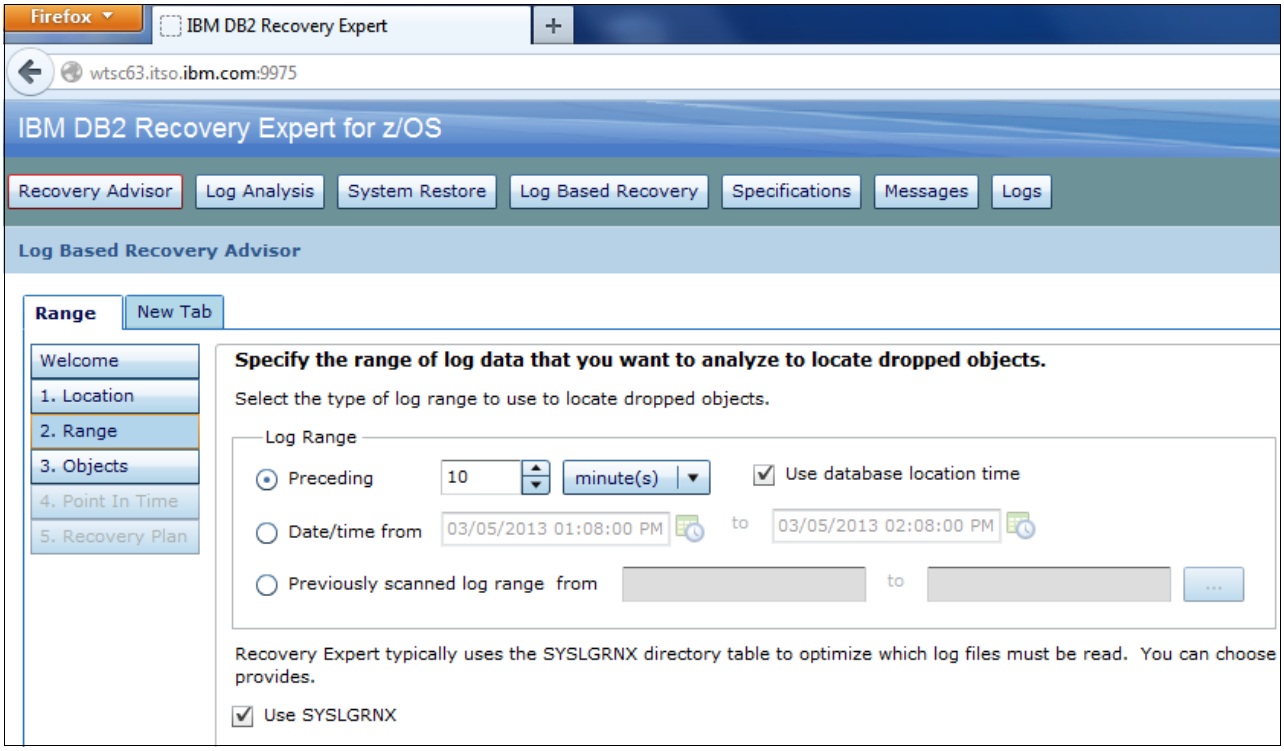

Figure 8-69 Create a new scan range

Recovery Expert will start the scan and will present a panel with the progress. See Figure 8-70.

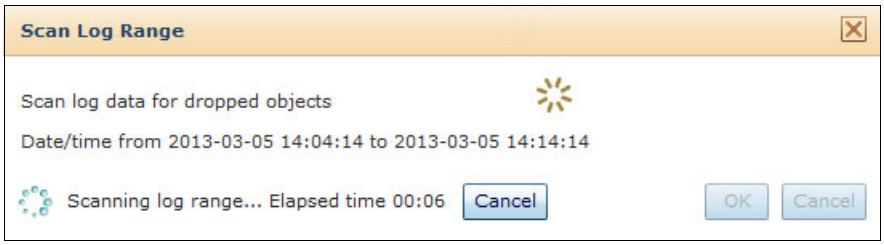

Figure 8-70 Scan progress

As soon as the scan is finished, Recovery Expert will present a hierarchical list of objects where you can select the dropped object. We select the GLWACT table and add it to the list of objects to be recovered. See Figure 8-71 on page 402.

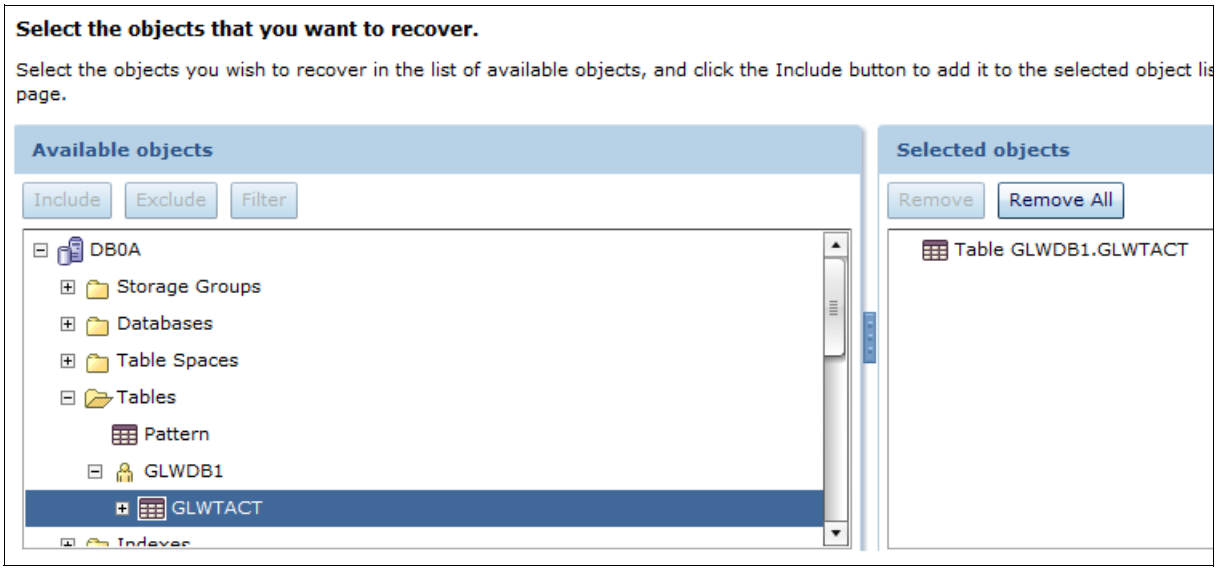

Figure 8-71 Select objects to restore

The next step is to select the recovery timestamp. Recovery Expert will implicitly set the recovery timestamp to the drop timestamp, but we can choose a different point in time by using the GUI dialog. We go with the implicit timestamp, as shown on Figure 8-72.

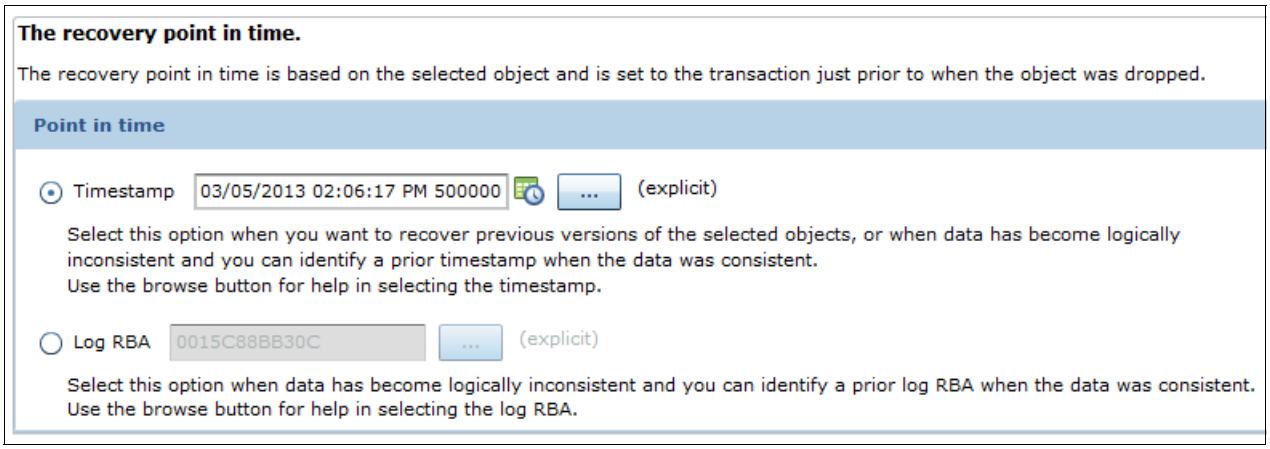

Figure 8-72 Recovery point in time selection

After the point in time has been defined, Recovery Expert will generate a recovery plan for the selected table. There can be only one plan, which is to reload from image copy and apply redo SQL. We continue by clicking Generate to build recovery JCL that will be executed by Recovery Expert. See Figure 8-73 on page 403.

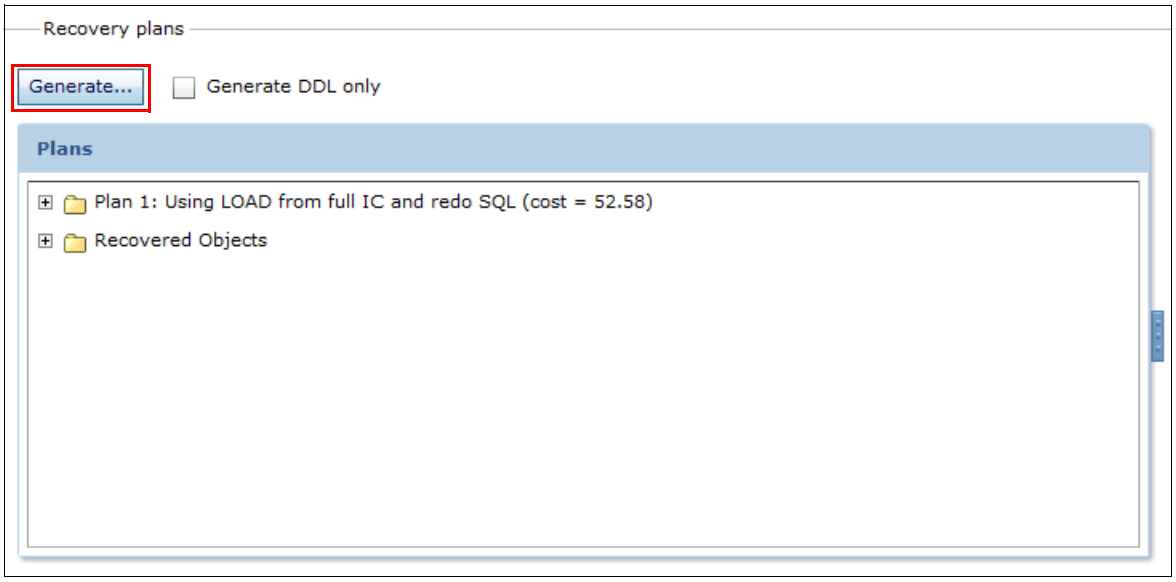

Figure 8-73 Recovery plan

Recovery Expert presents a panel with the generated plan details. See Figure 8-74.

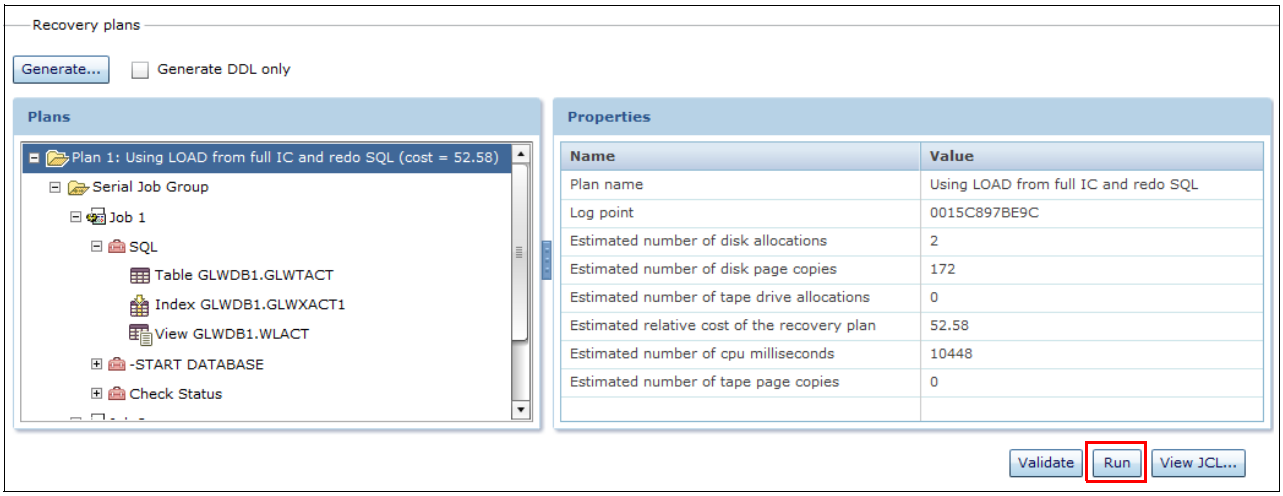

Figure 8-74 Recovery plan details

Recovery Expert re-creates the GLWACT table. It also re-creates the dependent objects, which, in this case, are an index and a view. We can submit the generated JCL from the same panel by clicking Run. Recovery Expert submits the jobs and presents the results panel after the execution. See Figure 8-75 on page 404.

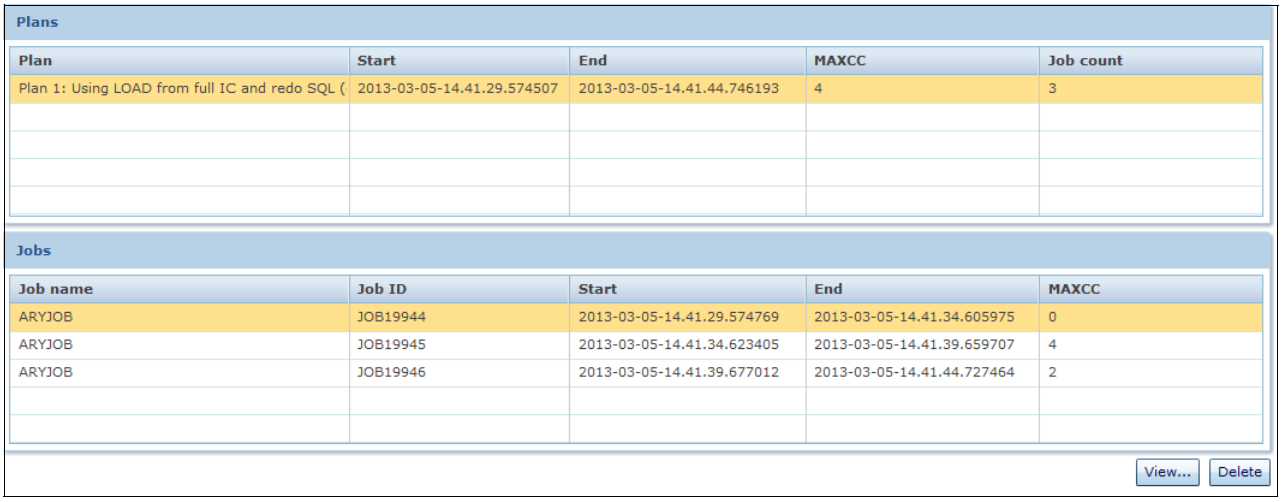

Figure 8-75 Recovery job execution results

All of the recovery jobs executed with no errors and we have successfully recovered the dropped table along with its dependent objects. If there were any errors during the job executions, we could analyze those errors by looking through the output members by selecting View.

9

IBM DB2 Cloning Tool for z/OS

IBM DB2 Cloning Tool for z/OS V3.1 (DB2 Cloning Tool) automates the cloning process to provide usable DB2 clones within minutes, boosting efficiency and freeing up DBA time.

The DB2 Cloning Tool performs these functions:

- Clones DB2 subsystems, DB2 table spaces, or index spaces quickly to create up-to-date test environments
- Automates the cloning process to provide usable DB2 clones within minutes
- Clones a DB2 subsystem by renaming and cataloging the data sets, fixing the volume definitions, and updating the DB2 control information
- Uses fast copy technology to quickly copy DB2 data sets within a subsystem or to a different subsystem

In this chapter, we provide a general overview of the functions that the tool supports at the subsystem and object levels and then describe the corresponding scenarios. We explain these topics:

- DB2 Cloning Tool cloning types
- General considerations for cloning
- Support jobs, commands, and tools
- DB2 subsystem level cloning
- Subsystem cloning with system level backup
- Subsystem level cloning using the stored procedure
- Cloning at the table space level
- Dealing with special objects
- Using DB2 Cloning Tool with the DB2 Administration Tool
- Runtime repository

9.1 DB2 Cloning Tool cloning types

DB2 Cloning Tool is a program product that can clone either an entire subsystem or a set of table spaces. The technique used differs in each case. An important DB2 Cloning Tool concept is that DB2 Cloning Tool does not actually copy anything. DB2 Cloning Tool turns the result of a copy operation into a DB2 clone.

DB2 Cloning Tool can invoke IBM FlashCopy, FlashCopy SE, StorageTek SnapShot, EMC TimeFinder Clone, and IBM DFSMSdss copy utilities. It is compatible with all fast replication and copy utilities, including local and remote mirroring utilities.

In subsystem cloning, the lowest element that can be cloned is an entire DASD volume.

Cloning less than an entire subsystem is referred to as a "*table space refresh*" and in a table space refresh, the lowest element that can be cloned is a data set.

Figure 9-1 shows a typical DB2 environment.

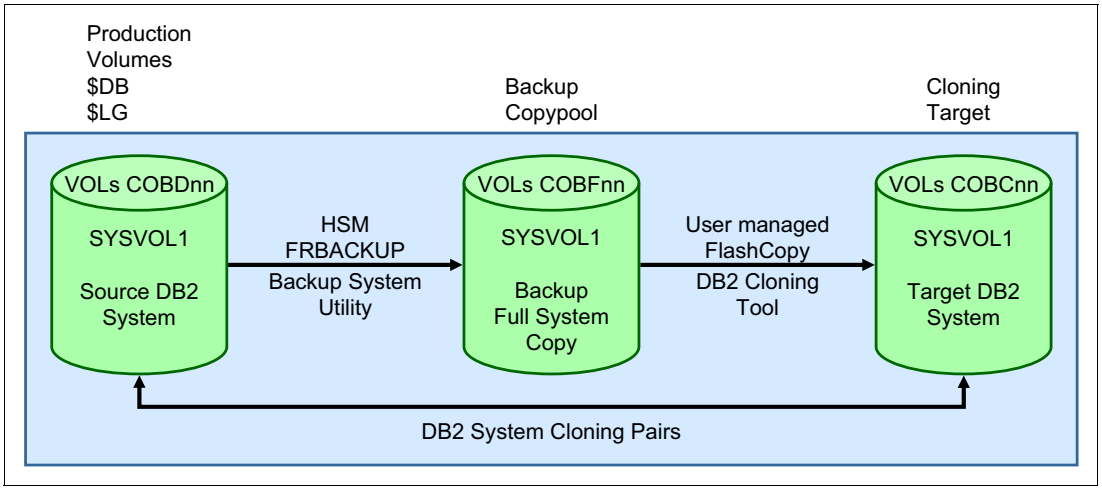

Figure 9-1 DB2 cloning environment

9.1.1 Why cloning

There are many reasons to clone DB2 environments:

- Development
- Disaster recovery testing
- Quality assurance
- Audit
- Creation of a new SAP environment

Cloning procedures are not complex, but using the correct procedure can make it a one-step process that handles terabytes of data in a fraction of time. The cloning process using the DB2 Cloning Tool can create an exact copy of a full subsystem/data sharing group or a subset of objects. DB2 Cloning Tool can consistently duplicate objects without compromising availability. Implementing fast replication copies and fast data set rename steps can make even a fresh copy of large enterprise resource planning (ERP)/customer relationship management (CRM) systems, such as SAP, available in minutes.

It is also worth considering the use of cloning to support a DB2 upgrade or major release testing. A system clone from a key source environment to a test environment then can be used for applying a DB2 upgrade and subsequent testing.

9.1.2 Our cloning test environment

In our test environment, we use three DB2 10 subsystems on one logical partition (LPAR):

- DB0A - Subsystem is used for other projects in this book but used as the source for cloning to DB0C
- DB0C - Cloned from DB0A and then used as the source for further scenarios
- DB0D - A target environment

The database primarily used consists of the GLW tables that are described in Table A-6 on page 538.

The database includes the following definitions:

- Universal table space partitioned by growth
- Universal table space partitioned by range
- Sequences and identity columns
- XML
- Large objects (LOBs)

9.2 General considerations for cloning

We describe several areas to assist you in preparing your environment for either subsystem or object level cloning. In some cases, there is much more detail in the *DB2 Cloning Tool for z/OS, V3.1, User's Guide*, SC19-3493-01, and its updates:

`http://www.ibm.com/support/docview.wss?uid=swg27023876`

The information we provide is not intended to replace the *DB2 Cloning Tool for z/OS, V3.1, User's Guide*, SC19-3493-01, but to complement it.

DB2 software level

Important: Ensure that the load modules and RUNLIBs for the target DB2 system are the same DB2 release and the same or similar maintenance level as the source DB2 system. A different release or maintenance level of DB2 might have dependencies on the DB2 catalog, directory, or bootstrap data set (BSDS) that are not included in the cloning.

After the cloning is complete, the target DB2 can then be migrated to a higher release or maintenance level of DB2.

Online or offline cloning

Consider the point of consistency for the clone:

- An offline DB2 subsystem clone is created by stopping the source DB2 subsystem to achieve your point-in-time copy. Stopping the source DB2 subsystem ensures that all buffers have been flushed, all data has been committed to disk, and that no transactions are in-flight.

- An online DB2 subsystem clone is created by suspending (DB2 SET LOG SUSPEND) the source DB2 subsystem to achieve your point-in-time copy. By suspending the source DB2 subsystem, any pending database writes are forced to disk, update activity is suspended, and the log buffers are flushed to disk. An alternative to suspending the source DB2 subsystem is to use consistent FlashCopy, SnapShot, or TimeFinder/Clone. If you are using mirroring technology to copy the source, you can use the mirror (consistent split or break the mirror) to achieve a point-in-time copy.

In our scenario, we focus on using a system level backup (SLB), which is considered online but does not need to suspend DB2.

For object level cloning, the option exists for a FUZZY clone where the source logs are used to create a consistent point on the target.

Building cloning jobs

We can use the DB2 Cloning Tool ISPF interface to build a profile for each cloning scenario.

The menu-driven interface allows us to easily create and modify cloning jobs with specific command parameters, and then save that information in profiles that can be used again. In addition, subsystem information can be configured once and then is available to all users of the interface.

This is the method that we follow in this book whenever possible.

The DSNZSPEC DSNZPARM

As part of the subsystem cloning, updates need to be made to the DB2 catalog. Some of these changes are not normally possible and therefore we need to set up a special DSNZPARM, DSNZSPEC, for use on the target DB2 subsystem.

This is a very important step that needs to be done before running the clone jobs. We summarize the instructions on DSNZSPEC from the *DB2 Cloning Tool for z/OS, V3.1, User's Guide*, SC19-3493-01.

DSNZSPEC allows the target's DB2 catalog to be updated and defers the backing out of in-flight transactions on the target subsystem. This DSNZPARM should only be used for the time needed to update the target's VCATNAMEs and, optionally, the target DB2 storage group names.

Follow these steps:

1. Allocate a special macro library for DSNZSPEC. It is a small partitioned data set (PDS) with only one member, DSN6SPRC.
2. Copy member DSN6SPRC from the distributed SDSNMACS library to the special macro library.
3. Change the special macro library member, DSN6SPRC, in the following manner:

   Change this line:

   ```
   &SPRMCTU SETC 'O' YES=CATALOG CAN BE UPDATED
   ```

   To this line:

   ```
   &SPRMCTU SETC '1' YES=CATALOG CAN BE UPDATED
   ```

4. Save the modified special macro library member. For example, the modified special DSN6SPRC macro might look similar to this example (in part):

   ```
   ...
   &SPRMNAP SETC 'O' BIT ON - SKIP ADJ. PREFETCH @KYF1570
   ```

```
&SPRMSHP SETC '0' BIT ON - SIMULATE 2G HIPERSPACE
&SPRMCTU SETC '1' YES=CATALOG CAN BE UPDATED
&SPRMXPL SETC '0' YES=GEN ALL EXPLAIN TABLES
&SPRMNHJ SETC '0' YES=TURN OFF HYBRID JOIN
```

5. Create DSNZSPEC.

   Copy it from the normal DSNZPARM, DSNZPAR*x*, that was created for the target DB2 subsystem:

   – Change DSNZSPEC macro DSN6SPRM from RESTART, ALL to DEFER, ALL.

   – Change DSNZSPEC macro DSN6SPRM keyword SYSADM or SYSADM2 to specify the user ID that will execute the SQL statements on the target DB2.

   – Change the JCL for DSNZSPEC so that the special macro library is the first library in the assembly step //SYSLIB DD concatenation.

   – Change all occurrences of DSNZPAR*x* to DSNZSPEC except on the link-edit card INCLUDE ADSNLOAD(DSNZPARM). Assemble and link-edit DSNZSPEC to the target DB2 SDSNEXIT LOAD library.

     For example, the DSNZSPEC might look similar to this example (in part):

```
...
DSN6ENV MVS=XA
DSN6SPRM DEFER, X
ALL, X
...
SYSADM=CKZUSER, X
SYSADM2=DB2ADM, X
...
//SYSLMOD DD DISP=SHR,
// DSN=target.SDSNEXIT
...
ENTRY DSNZMSTR
NAME DSNZSPEC(R)
```

The clone (or replica) is started with the special DSNZPARM to allow the DB2 catalog to be correctly conditioned. Then, the replica is stopped and started with the normal DSNZPARMs as an independent clone.

Simulation

In all the steps where data changes occur, there is a parameter option *SIMULATE* that allows the step to be run without performing the updates. See Example 9-1. As we go through each step, it is useful to run with this option for the first run. As an alternative in the COPY step, you can set the DATA-MOVER program to NONE, for example, `DATA-MOVER(PGM(NONE))`.

Example 9-1 Use of SIMULATE

```
//CKZIN    DD  *
      DB2SQL          -
         SIMULATE -
        DB2-SSID(DBOD) -
        DB2-NAME(SOO1) -
        LISTSQL(Y) -
        WLM-ENVIRONMENT-MASKS( -
          DBOC* DBOD* -
                            ) -
        WLM-ENV-NOT-UPDATED(RC(4)) -
        DATACLAS-NOT-UPDATED(RC(4)) -
```

```
        MGMTCLAS-NOT-UPDATED(RC(4)) -
        STORCLAS-NOT-UPDATED(RC(4)) -
        JOURNAL-DDN(JOURNAL)
```

9.3 Support jobs, commands, and tools

Whether you use the ISPF interface or build your JCL deck yourself, there is still a need for running checks, validating your scenario, or only monitoring. We describe a few support jobs, tools, and commands that can be useful.

9.3.1 DB2 Cloning Tool jobs

The following jobs might not be generated or required to run the clone but we have found them useful in managing a cloning scenario. These commands are shipped with the DB2 Cloning Tool.

COPYCHECK

This DB2 Cloning Tool command serves two purposes. It provides a mechanism to either WAIT for copies to complete, or to WITHDRAW or STOPSNAP (terminate) previously established volume relationships.

- WAIT is used to check for completion of the copies. It lists all the active copies and their percentage completed. An example of WAIT is in Example 9-2.
- WITHDRAW is used to stop the background copy process when FlashCopy is used and STOPSNAP is for stopping the EMC SNAP background copy process relationship. This allows for rerunning the process or canceling it altogether. If you need to rerun the clone and the copy has not yet completed, you can run with this option and the remaining background copy processes are withdrawn.

You need to provide the journal file as used in the cloning process to define the scope of copies to monitor or cancel.

Example 9-2 COPYCHECK WAIT sample output

```
CKZ05630I VOLUME PAIRS STATUS
          SBOXKP/X5DE61 BOTH VOLUME SERIALS ARE AVAILABLE
          SBOXKQ/X5DE62 BOTH VOLUME SERIALS ARE AVAILABLE
          X5DA0D/X5D842 BOTH VOLUME SERIALS ARE AVAILABLE
          X66228/XV6136 BOTH VOLUME SERIALS ARE AVAILABLE
          X66229/XV633F COPY STILL IN PROGRESS,  62% COMPLETED
          X6642A/XV643D BOTH VOLUME SERIALS ARE AVAILABLE
          X66425/XV643F BOTH VOLUME SERIALS ARE AVAILABLE
          X66426/XV6435 BOTH VOLUME SERIALS ARE AVAILABLE
          X66427/XV6439 BOTH VOLUME SERIALS ARE AVAILABLE
          X66525/XV673A COPY STILL IN PROGRESS,  80% COMPLETED
          X66528/X3652E COPY STILL IN PROGRESS,  51% COMPLETED
          X66529/X4612F COPY STILL IN PROGRESS,  75% COMPLETED
          X6672A/X4622F COPY STILL IN PROGRESS,  94% COMPLETED
          X66725/X4632F COPY STILL IN PROGRESS,  58% COMPLETED
          X66726/X4642F COPY STILL IN PROGRESS,  80% COMPLETED
          X66728/X4652F COPY STILL IN PROGRESS,  80% COMPLETED
          X66729/X76624 COPY STILL IN PROGRESS,  75% COMPLETED
          X76724/X9602F BOTH VOLUME SERIALS ARE AVAILABLE
```

Example 9-3 shows a copy that is canceled by using the WITHDRAW parameter.

Example 9-3 COPYCHECK WITHDRAW sample output

```
COPYCHECK           -
      WITHDRAW -
     JOURNAL-DDN(JOURNAL)

CKZO5501I 10.44.41 VOLUME CHECK STARTED - PROGRAM REV=31

CKZO5601I 10.44.41 VOLUME STATUS STARTED - PROGRAM REV=34
CKZO5643I ANTRQST LEVEL=12; ESSRVCS LEVEL=106
CKZO5652I VOLSER PAIR: X66528/X3652E FCWITHDRAW ISSUED
```

If you want to use system commands, you can view the status of the FlashCopy by issuing **fcquery** from the system log. See Example 9-4.

Example 9-4 Fcquery

```
SDSF ULOG  CONSOLE ADMR1                          LINE 47      COLUMNS 42- 1
COMMAND INPUT ===>                                            SCROLL ===> CS
ANTF0421I FCQUERY Relationship 1
  DEVN SSID LSS CCA  CU     SERIAL       ACT    MAX XC PC CC RV SE  SEQ+
  NUM
  6229 8922  02  29 2105 000000021968      1   8159  N  N  N  N NN 0000+
  0000
  RELATIONSHIP DETAIL STARTING TRACK: 00000000
  DEVICE LONG BUSY FOR CG: NO  WRITE INHIBITED: NO
  ------------------------------------------------
    PARTNER    SOURCE   TARGET   S F C C P C T S F P
  LSS CCA SSID START    START    O V O A R R W E S M
  --- --- ---- -------- -------- - - - - - - - - - -
   03  3F 8923 00000001 00000001 Y N Y N N N Y N N N
       NO. OF TRACKS: 00056020  TRACKS TO COPY: 00011321
       ESTABL: 2012/05/09 22:48:44  LAST INCR:  2012/05/09 22:48:44
  ANTF0001I FCQUERY COMMAND COMPLETED FOR DEVICE 6229, COMPLETION CODE:+
   00

  ANTF0421I FCQUERY Relationship 1
```

The **fcwithdraw** command can cancel the FlashCopy relationship. Using the native command is best done by your storage experts. See Example 9-5.

Example 9-5 fcwithdraw example

```
fcwithdr sdevn(6229) tdevn(633f)a.  FCWITHDR COMMAND COMPLETED FOR DEVICE 6229, COMPLETION
CODE: 00
```

FINDUCATS

FINDUCATS identifies which ICF User catalogs point at data sets on the source volumes to be copied.

The COPY step requires pairs of source and target user ICF catalogs to be specified. FINDUCATS does not negate this need. It is intended to be run prior to the initial setup and possibly on an occasional basis to ensure that the user catalogs that should be specified for the COPY step have not changed.

FINDUCATS invokes DCOLLECT to identify ALIAS names of the source volume data sets in order to identify the correct source ICF user catalogs.

Run FINDUCATS, at least initially, to determine the involved ICF user catalogs, and then whenever you want to verify that the ICF user catalogs involved with source volume data sets are as specified in the COPY command. See Figure 9-2.

```
//ADMR1C JOB (999,POK),'FIND UCAT DBOD',
//         REGION=0M,NOTIFY=&SYSUID,MSGCLASS=X,CLASS=T
//PROCLIB JCLLIB ORDER=DBOAM.PROCLIB
/*JOBPARM SYSAFF=SC63
//S01 EXEC PGM=CKZ00010,REGION=0M
//STEPLIB DD DSN=CKZ.SCKZLOAD,DISP=SHR
//CKZINI  DD DSN=CKZ.SCKZPARM(CKZINI),DISP=SHR
//CKZPRINT DD SYSOUT=*
//CKZIN    DD *
 FINDUCATS -
 FROM-STORAGEGROUP(DBODDATA,DBODLOG1,DBODLOG2)
/*
//*
```

Figure 9-2 Sample JCL for FINDUCAT

The output from FINDUCAT is shown in Example 9-6.

Example 9-6 Sample output from FINDUCAT

```
DB2 CLONING TOOL - SUBSYSTEM CLONING
 CKZINI INFORMATION                    MODULE INFORMATION
     REL: V3R1M0                           REL: V3R1M0
    DATE: 14 OCT 2011

CKZ01020I PROGRAM: CKZ01CIO 20110309 10.44 VERS=1.0 REV=35

 FINDUCATS -
 FROM-STORAGEGROUP(DBODDATA,DBODLOG1,DBODLOG2)

CKZ41001I 08.30.03 FINDUCATS STARTED - PROGRAM REV=9
CKZ41086I STORAGE GROUPS/MASKS FOR KEYWORD: FROM-STORAGEGROUP
          DBODDATA DBODLOG1 DBODLOG2

CKZ41087I     28 VOLSERS DERIVED  FOR KEYWORD: FROM-STORAGEGROUP
          XV6136 XV633F XV643D XV643F XV6435 XV6439 XV673A X3652E X4612F X4622F
          X5D844 X5D90C X76624 X9602F X96123 X96223 X9631F Y6612A Y66428 Y66527

CKZ41087I     28 VOLSERS RESOLVED FOR KEYWORD: FROM-STORAGEGROUP
CKZ41101I 08.30.03 VOLUME COLLECTION STARTED - PROGRAM REV=14
CKZ41103I DDNAME=SYS00001 ALLOCATED FOR DSN=**TEMPORARY DCOLLECT DSN
CKZ41135I 08.30.06 VOLSER XV6136 DCOLLECT STARTED
CKZ41135I 08.30.06 VOLSER XV6136 DCOLLECT COMPLETED

CKZ41135I 08.30.06 VOLSER XV6136 DATA PROCESSING STARTED
CKZ41135I 08.30.06 VOLSER XV6136 DATA PROCESSING COMPLETED
.............................................................
CKZ41136I VOLSER    USERCATALOG NAME                          ALIAS NAME OR H
          XV6136    UCAT.DBODD                                DBODD

          XV633F    UCAT.DBODD                                DBODD

          XV643D    UCAT.DBODL                                DBODA

          XV643F    UCAT.DBODL                                DBODA
```

```
          XV6435    UCAT.DBODL                                    DBODA

          XV6439    UCAT.DBODL                                    DBODA

          XV673A    UCAT.DBODD                                    DBODD
.....................................................................
CKZ41137I USERCATALOG NAME
          GARBAGE.DBOADATA
          GARBAGE.DBOALOGS
          MCAT.SANDBOX.Z1C.SBOXOO                    ** MASTER CATALOG **
          UCAT.DBOCD
          UCAT.DBOCL
          UCAT.DBODD
          UCAT.DBODI
          UCAT.DBODL

CKZ41101I 08.30.08 VOLUME COLLECTION COMPLETED; RETURN CODE=0
```

9.3.2 DB2 tasks

In running our scenarios, we identified some useful DB2 functions and commands that assisted us. Some of these functions and commands are relatively new or might not be well known.

ADMIN_SMS_INFO stored procedure

The ADMIN_SMS_INFO stored procedure returns space information about copypools and their storage groups and volumes. See Example 9-7.

To use this function, verify that the following PTFs are applied.

Table 9-1 PTFs for Admin_SMS_INFO

DB2 version	PTF number
DB2 9	UK63214
DB2 10	UK63213

Calling this stored procedure requires the input to be in a global temporary table. To do this, we wrote the REXX procedure ADMSMS as described in Appendix C, “Additional material” on page 551.

Using our REXX code, by specifying the volume, we can identify the storage group. Similarly, knowing the storage group, we can identify the volumes in the groups and verify the space capacity. This information is useful for checking the copypool for the SLB.

Example 9-7 Sample output using the ADMIN_SMS_INFO SP

```
READY
 ADMSMS DBOC V,SBOXJK
################################################################################
# ADMSYSRX V110     ADMIN_SMS_INFO SP to list SMS details  12/06/19 08:03:59 #
################################################################################

--------------------------------------------------------------------------------
VOLUME COPY_POOL    STORGRP  TOTAL_CP   FREE_SPACE   LARGEST_FREE PERCENT_FREE
SBOXJK                          8120M         882M           850M        10.86
READY
```

```
 ADMSMS DBOC S,DBOCDATA
################################################################################
# ADMSYSRX V110     ADMIN_SMS_INFO SP to list SMS details  12/06/19 08:03:59 #
################################################################################

--------------------------------------------------------------------------------
VOLUME COPY_POOL    STORGRP  TOTAL_CP   FREE_SPACE   LARGEST_FREE PERCENT_FREE
SBOXJK              DBOCDATA    8120M         882M          850M        10.86
SBOXJL              DBOCDATA    8120M        2613M         2572M        32.18
READY
 ADMSMS DBOC S,DBOCLOG1
################################################################################
# ADMSYSRX V110     ADMIN_SMS_INFO SP to list SMS details  12/06/19 08:03:59 #
################################################################################

--------------------------------------------------------------------------------
VOLUME COPY_POOL    STORGRP  TOTAL_CP   FREE_SPACE   LARGEST_FREE PERCENT_FREE
SBOXJQ              DBOCLOG1    8120M        7660M         7659M        94.33
READY
 ADMSMS DBOC S,DBOCLOG2
################################################################################
# ADMSYSRX V110     ADMIN_SMS_INFO SP to list SMS details  12/06/19 08:03:59 #
################################################################################

--------------------------------------------------------------------------------
VOLUME COPY_POOL    STORGRP  TOTAL_CP   FREE_SPACE   LARGEST_FREE PERCENT_FREE
SBOXJR              DBOCLOG2    8120M        7660M         7659M        94.33
 ADMSMS DBOC C,DSN$DBOC$DB
################################################################################
# ADMSYSRX V110     ADMIN_SMS_INFO SP to list SMS details  12/06/19 08:03:59 #
################################################################################
--------------------------------------------------------------------------------
VOLUME COPY_POOL    STORGRP  TOTAL_CP   FREE_SPACE   LARGEST_FREE PERCENT_FREE
SBOXJK DSN$DBOC$DB  DBOCDATA    8120M         882M          850M        10.86
SBOXJL DSN$DBOC$DB  DBOCDATA    8120M        2613M         2572M        32.18
SBOXJM DSN$DBOC$DB  DBOCDATA    8120M         978M          785M        12.04
SBOXJN DSN$DBOC$DB  DBOCDATA    8120M         799M          743M        9.840
X66125 DSN$DBOC$DB  DBOCDATA   24359M       11064M        11064M        45.42
X66226 DSN$DBOC$DB  DBOCDATA   24359M       14295M        14295M        58.68
X66326 DSN$DBOC$DB  DBOCDATA   24359M       10729M        10655M        44.05
X66327 DSN$DBOC$DB  DBOCDATA   24359M        5103M         5103M        20.95
X66328 DSN$DBOC$DB  DBOCDATA   24359M        8782M         8617M        36.05
X66329 DSN$DBOC$DB  DBOCDATA   24359M       13887M        13885M        57.01
```

9.3.3 IBM DB2 Recovery Expert Tool for z/OS

Using Recovery Expert, we can identify the status of the DB2 environment and the ability to support a DB2 system backup. Remember, we can also use DB2 Recovery Expert backups as sources of clones.

Figure 9-3 on page 415, Figure 9-4 on page 415, and Figure 9-5 on page 416 show an example of using the analysis feature of Recovery Expert. In this example, the analysis feature shows us that the ICF catalogs reside on volumes outside of the DB2 storage groups. Therefore, a FULL SLB should not be taken until this situation is resolved.

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2012/05/01  15:34:35
Option  ===> _                                                Scroll ===> PAGE

Commands: ANALYZE  REANALYZE

--------------------------------------------------------------------------------
 Subsystem: DB0C    Active: Yes     Datasharing: No
 Date of Last Analysis: 05/01/2012  Analysis Recommended: N
 Message: Subsystem configuration prevents system level backup.

------------------------------------------------          Row 1 of 64    +
  New MVS User Catalogs to be used by this subsystem
_   Log/BSDS Catl  ____________________________________________ Volume  ______
_   DB2 Data Catl  ____________________________________________ Volume  ______
    Line Cmds: (C-Create, A-Add Alias, D-Dataset Disp, U-Update, V-View Alias)

  Existing MVS User Catalogs used by this subsystem
_   Data Other     UCAT.DB0CDATA                                 Volume  SBOX1E
_   Log            UCAT.DB0CLOGS                                 Volume  SBOX2W
    Line Cmds: (D-Dataset Display, V-View Aliases)

  Storage Copy Pools
_   Data Copy Pool DSN$DB0C$DB
```

Figure 9-3 Subsystem Analysis and Configuration from DB2 Recovery Expert (Part 1 of 4)

```
_   Log Copy Pool  DSN$DB0C$LG
_   Backup Pool    DB0CCPB
    Line Cmds: (V-Volume List)

_ Boot Strap Datasets
_   DB0C - BSDS 1  DB0CB.BSDS01                                  Volume  SBOXJR
_   DB0C - BSDS 2  DB0CB.BSDS02                                  Volume  SBOXJQ
    Line Cmds: (R-Rename BSDS, M-Move BSDS)

_ Active Log Datasets
_   DB0C - Log 1   DB0CL.LOGCOPY1.DS01                           Volume  SBOXJR
_   DB0C - Log 1   DB0CL.LOGCOPY1.DS02                           Volume  SBOXJR
```

Figure 9-4 Subsystem Analysis and Configuration from DB2 Recovery Expert (Part 2 of 4)

```
------------------------------------------------------         Row 25 of 64   -+
_   DB0C - Log 1   DB0CL.LOGCOPY1.DS03                          Volume  SBOXJ
_   DB0C - Log 2   DB0CL.LOGCOPY2.DS01                          Volume  SBOXJ
_   DB0C - Log 2   DB0CL.LOGCOPY2.DS02                          Volume  SBOXJ
_   DB0C - Log 2   DB0CL.LOGCOPY2.DS03                          Volume  SBOXJ
    Line Cmds: (R-Rename Log, M-Move Log)

  Alias used with associated MVS User Catalogs
_   DB0CA          UCAT.DB0CLOGS
_   DB0CB          UCAT.DB0CLOGS                                Log
_   DB0CD          UCAT.DB0CDATA                                Data Other
_   DB0CL          UCAT.DB0CLOGS                                Log
    Line Cmds: (D-Dataset Display, M-Merge catalog entries, R-Rename Alias)
```

Figure 9-5 Subsystem Analysis and Configuration from DB2 Recovery Expert (Part 3 of 4)

```
RCVYXPRT V3R1 --- Subsystem Analysis and Configuration ---2012/05/01  13:20:55
Option  ===> ____________________________________________ Scroll ===> PAGE

Commands: ANALYZE  REANALYZE

------------------------------------------------------------------------------
 Subsystem: DB0C    Active: Yes     Datasharing: No
 Date of Last Analysis: 05/01/2012  Analysis Recommended: N
 Message: Subsystem configuration prevents system level backup.

------------------------------------------------------         Row 37 of 64   -+

  Volumes used by this subsystem
    Volume         Data  DataCat  ActLog ActCat  ArcLog ArcCat  Other  Pool
_   -NONE-         Yes   No       No     No      No     No      No     N/A
_   SBOX1E         No    Yes      No     No      No     No      Yes    No
_   SBOX2W         No    No       No     Yes     No     No      Yes    No
_   SBOXJK         Yes   No       No     No      No     No      Yes    Yes
_   SBOXJL         Yes   No       No     No      No     No      Yes    Yes
_   SBOXJM         Yes   No       No     No      No     No      Yes    Yes
_   SBOXJN         Yes   No       No     No      No     No      Yes    Yes
_   SBOXJQ         No    No       Yes    No      No     No      Yes    Yes
_   SBOXJR         No    No       Yes    No      No     No      Yes    Yes
```

Figure 9-6 Subsystem Analysis and Configuration from DB2 Recovery Expert (Part 4 of 4)

The highlighted volumes might have issues that need to be addressed before a valid SLB can be made.

The figures use the following color codes:

- Dark blue: Volume is optimal.
- Light blue: The volume contains data other than DB2 data.
- Pink: Both log and object data reside on the volume.
- Red: The volume cannot be backed up by ARY.

If we look at the highlighted message in Figure 9-6 on page 416, it is clear that an SLB cannot be taken. You get this warning if you are using DB2 Recovery Expert to manage the backup but not if you are using DB2 DSNUTILB with BACKUP SYSTEM FULL. If you ran the backup (using either backup method), you will not be able to perform a system restore from this backup.

9.3.4 System commands

There are many system commands that can be used to assist in preparing, analyzing, and monitoring your clones. We used the following system commands:

- Display the volumes in a storage management subsystem (SMS) storage group by using SMS commands. See Example 9-8.

Example 9-8 Display volumes in an SMS storage group

```
D SMS,SG(DBODDATA),LISTVOL
 IGD002I 20:04:10 DISPLAY SMS 062

 STORGRP  TYPE    SYSTEM= 1 2 3 4
 DBODDATA POOL            + + + +

 VOLUME   UNIT    SYSTEM= 1 2 3 4                          STORGRP NAME
 XV6136   6136            + + + +                            DBODDATA
 XV633F   633F            + + + +                            DBODDATA
 XV643D   643D            + + + +                            DBODDATA
 XV643F   643F            + + + +                            DBODDATA
```

- List a copypool to a data set:

```
HSEND LIST CP(DSN$DBOC$LG) ALLVOLS ODS(ADMR1.CLONE.HSM2)
```

- Display the physical volume by issuing the command in Example 9-9.

Example 9-9 Display volume

```
D U,,,6525,1
RESPONSE=SC63
 IEE457I 13.08.45 UNIT STATUS 354
 UNIT TYPE STATUS        VOLSER     VOLSTATE
 6525 3390 O             X66525     PRIV/RSDNT
```

- Delete aged data sets that have not reached their expiration date by using a mask:

```
TSO DELETE DBOCA.ARCHLOG%.%00002* MASK PURGE
```

9.4 DB2 subsystem level cloning

In some environments, the knowledge of which tables semantically belong together is embedded within the application programs. This then determines the scope of your backups and therefore the need for all objects to be recovered at the same time.

Recovering only a subset of the tables breaks the transactional integrity of the system. An SAP environment is a good example of this type of environment.

The same applies if you need to build copies of the application environment and, therefore, you need to clone at the subsystem level.

Offline or online

An *offline* DB2 subsystem clone is created by stopping the source DB2 subsystem to achieve your point-in-time copy. Stopping the source DB2 subsystem ensures that all buffers have been flushed, all data has been committed to disk, and that no transactions are in-flight.

An *online* DB2 subsystem clone is created by suspending the source DB2 subsystem to achieve your point-in-time copy. By suspending the source DB2 subsystem, any pending database writes are forced to disk, update activity is suspended, and the log buffers are flushed to disk. An alternative to suspending the source DB2 subsystem is to use consistent FlashCopy, SnapShot, or TimeFinder/Clone, or consistent split or break mirror to achieve your point-in-time copy.

The key difference is that an offline clone ensures consistency at a point in time at the expense of an outage to the source subsystem. The online clone conversely maintains availability for the source system at the expense of the possibility of incomplete units of work.

The DB2 Cloning Tool manages the potential for incomplete transactions by applying the logs at the target system. A check (DB2FIX) is also made to ensure that no objects are in logical page list (LPL) or group buffer pool recovery pending (GRECP) status, and if so, a log recovery is started.

The DB2 SLB that is used below is an example of online cloning.

What needs to be changed in DB2

Make these changes to run a DB2 SLB:

- DB2 directory

  The VCATNAMEs:

  - Optionally, the DB2 storage group names
- DB2 catalog updates:
  - The DB2 VCATNAMEs
  - Optionally, the DB2 storage group names
  - Optionally, Workload Manager (WLM) references
- BSDSs' updates:
  - The DB2 catalog name
  - The active log data set names
  - Optionally, distributed data facility (DDF) parameters (including locations)
  - Optionally, archive log data set names and volser

9.4.1 Considerations for system level cloning

Before cloning, review Chapter 7. "Planning for copying and renaming volumes" of the *DB2 Cloning Tool for z/OS, V3.1, User's Guide*, SC19-3493-01.

We list several key points next.

Pre-DB2 system cloning steps

Follow these steps:

1. Set up the target DB2 system definitions.

a. System libraries must be defined, including tools.

b. System automation for subsystem must be set up but suspended during the cloning operation.

2. Verify the availability of the target volumes and storage groups.
3. Set up the special DB2 system maintenance mode DSNZPARM before performing the DB2 cloning automation:
   a. This allows the DB2 catalog to be updated.
   b. This starts the target DB2 in DEFER mode to prevent back-out processing.
4. Identify the source and target ICF catalogs.

DASD volume content

Be aware that system level cloning is performed at the volume level and therefore any data set that is on the source volume is copied to the target. This might mean that you have data sets on the target that are not cataloged.

Verify the data set content of your source volumes and validate their requirement to be on these volumes.

There is the option to rename (with the RENAME command) the copied data sets on the target to a valid target data set name that is capable of being cataloged.

DB2 Cloning Tool requires that a RENAME-MASK or an EXCLUDE-MASK is supplied for every data set on the source volumes. You have the option of leaving excluded data sets on the target volumes or allowing DB2 Cloning Tool to delete them. Deleting those types of data sets might significantly increase the elapsed time of the RENAME job.

The parameter NOTRENAMED of the RENAME command is used to specify the following conditions:

- The disposition of any data sets that are not matched to a rename mask
- The return code, if at least one data set is not renamed

This parameter then puts you in control of determining when to ignore or delete orphaned data sets. Using `NOTRENAMED(KEEP,RC(4))` allows for a soft abend when there are data sets that are not renamed and left on the target volumes.

Using `NOTRENAMED(DELETE,RC(4))` allows for a soft abend when there are data sets that are not renamed and are deleted from the target volumes. The default for this parameter is `RC(8)`, which causes the step to fail.

ICF catalog

Do not have your image copies and archives in the same ICF catalog within your data and logs. These do not need to be cloned unless you plan on recovering to an older point in time than what is on the active logs. Remember that subsystem clones are volume-to-volume copies and take everything on the volume. Therefore, if there is garbage on the volume, you copy the garbage, too, or “garbage in - garbage out” (GIGO).

If you do not rename all the target data sets to use a valid catalog, you will have uncataloged data sets that you may need to clean up. This includes any ICF catalogs that were on the source volumes. An ICF catalog can be renamed, but it will not be usable as an ICF catalog. Using NOTRENAMED with the DELETE option will cause any “not renamed” data sets to be deleted from the target volumes.

The ICF catalog needs to be with your logs and data volumes within their respective storage pools when the System Backup is used as the source. *For this type of clone, the target ICF catalog cannot be on one of the target volumes; it must be on a volume completely outside of the cloning process.*

For full DB2 subsystem clones, the source ICF catalog is copied at the time that the source volumes are copied.

DB2 Cloning Tool catalogs target volume data sets to either a populated or an empty ICF catalog. If a target catalog entry already exists, the RECATALOG option of the RENAME command is required. The default setting is N.

Target ICF catalogs used to catalog the renamed data sets cannot reside on a target volume during the time frame from the volume copy through the completion of the RENAME step. If desired, you can move the target ICF catalogs from the target volume prior to the volume copy, and move the target ICF catalogs back to the target volume after the RENAME has completed.

DB2 catalog

When cloning a subsystem, the DB2 catalog needs to be updated to reflect the location and the VCAT of the target subsystem.

In order to update the DB2 catalog, a special DSNZPARM is required, as mentioned in Appendix B, “DSNZPARMs and general settings” on page 541.

DB2 logs and archives

When you are cloning a full environment (system level), the volumes containing the active logs should be among the source volumes copied to target volumes. The renaming and update of the BSDS is managed by the DB2 Cloning Tool.

Active log data sets in the BSDSs are modified based on the rename masks supplied in the RENAME command.

The source archive logs are not necessarily required at the target. If they are included, the optional ARCHIVE keyword should be used with the DB2UPDATE command to ensure that they are renamed in the BSDS. The archives also need to have a mask applied in the RENAME and cataloged. If the archive logs are required but not copied, the clone can still use them as long as they are cataloged in an ICF catalog shared with the clone’s LPAR. If the source is using them, the clone will wait, and if the clone is using them, the source DB2 will wait. (Of course, cross system enqueues are required.)

If the archives are not required, you can delete the DB2 archive log names in the BSDS that are not on the source volumes by running the DB2ALTERBSDS command. The keyword REMOVE-ARCHIVE-LOGS(NOTRENAMED) can remove the DB2 archive log names in the BSDS that are not on the source volumes, instead of leaving them in there not renamed.

9.5 Subsystem cloning with system level backup

A system level backup (SLB) is created by using the DB2 BACKUP SYSTEM utility or DB2 Recovery Expert. The setup and configuration of the backup are covered in 9.4, “DB2 subsystem level cloning” on page 417.

This would be considered an online clone because there is no requirement to stop the source subsystem.

BACKUP SYSTEM gives you the option to either copy both copypools or to copy the data copypool only. If you want to restore the DB2 system or an object to an arbitrary point-in-time, the option of copying the data copypool only should be sufficient. Alternatively, if you plan to clone the DB2 at system level as we do or if you want to restore DB2 to the point when the backup was created, copying both copypools is required.

When cloning from an SLB, such as DB2 BACKUP SYSTEM, there are two additional benefits:

- No need to stop or suspend the source DB2.
- The source of the clone can be from a past point-in-time.

The same scenarios can be used for system backups that are generated by using DB2 Recovery Expert Tool. See 9.3.3, “IBM DB2 Recovery Expert Tool for z/OS” on page 414. For a general look at system backups, see 8.1, “DB2 Recovery Expert” on page 316.

We provide an A B C example of offline cloning. In summary, the following repetitive procedure occurs for an SLB clone:

1. The source volumes (referred to as set **A**) have been copied to the backup volumes (COPYPOOL set **B**) by a DB2 BACKUP SYSTEM command.
2. The backup volumes (set **B**) are copied to the target volumes (set **C**).
3. The data sets on the target volumes (set **C**) are renamed.

9.5.1 Validating system level backup for cloning

In preparing scenarios for illustrating the use of SLB, we encountered a couple of issues.

It is essential that the location of your source ICF catalogs for your DB2 data and logs are on volumes within your DB2 data and storage pools. The ICF catalogs must be on the backup volumes that will be used by the cloning process as source volumes.

It is unlikely to directly create a clone from a source DB2 via a FlashCopy, SnapShot, or TimeFinder/Clone, where the source ICF catalogs may be outside the volumes that are being copied.

There are a few commands that you can run and tasks that you can perform to verify your environment preparedness to take an SLB.

We discovered that by using the Subsystem Analysis and Configuration function of DB2 Recovery Expert, you can easily analyze your DB2 subsystem and cover these areas:

- DASD
- Catalogs
- BSDS and logs

See 9.3.3, “IBM DB2 Recovery Expert Tool for z/OS” on page 414.

9.5.2 Using the ISPF dialog to prepare a clone from a system level backup

We have used the SLB as the source for this cloning example. We show the process that we followed using the ISPF dialog to build the jobs and execute the clone.

When using the ISPF interface to generate the cloning jobs, you get the following benefits:

- You are freed from having to correctly create and configure the JCL and control parameters.

- The jobs are created in the output JCL library in the order that they need be submitted.

Starting from the Primary Option Menu panel, set the default options for each step by selecting option 0 User settings, as shown in Figure 9-7.

```
DB2 Cloning Tool for z/OS    Primary Option Menu
Option ===> 0

0  User settings                                        User ID . . : ADMR1
1  Clone                                                System ID . : SC63
2  Administrator functions                              Appl ID . . : CKZ
X  Exit                                                 Version . . : 3.1
```

Figure 9-7 ISPF dialog Primary Options Menu selecting option 0 User settings

Then, select Option 1 User DB2 subsystem clone settings. See Figure 9-8.

```
DB2 Cloning Tool for z/OS    User Settings
Option ===> 1

0  User Options
1  User DB2 subsystem clone settings
2  User DB2 tablespace clone settings
```

Figure 9-8 Default settings

The User DB2 subsystem clone settings panel (Figure 9-9) is displayed.

```
CKZ1DSCS              User DB2 subsystem clone settings
Option ===>

Subsystem Clone Profile Default Values:
  Prefix for work data sets  . . ADMR1.CSB
  Work data sets unit device . . SYSDA     (SYSDA, DISK, or etc.)

Valid command selection values are
  1  COPY command
  2  COPYCHECK command
  3  RENAME command
  4  DB2FIX command
  5  DB2SQL command
  6  DB2START command
  7  DB2STOP command
  8  DB2UPDATE command
```

Figure 9-9 Selection menu for setting defaults for each command

In most cases, the default settings can be used. However, overrides are available when defining your Job Profile. Here, we define the default prefix for work data sets, which you can change for each clone operation.

Ensure that the source and target subsystems have been added and properly configured under the Administrator functions options. See Figure 9-10 on page 423.

```
CKZ1PRIM ng Tool for z/OS    Primary Option Menu
Option ===> 2

0  User settings                                        User ID . . : ADMR1
1  Clone                                                System ID . : SC63
2  Administrator functions                              Appl ID . . : CKZ
X  Exit                                                 Version . . : 3.1
--------------------------------------------------------------------------------
CKZ1ADFN ng Tool for z/OS    Administrator functions
 Option ===> 1

 1  DB2 subsystems
```

Figure 9-10 Setting the subsystems via Administrator functions

Select option 1 DB2 subsystems. The DB2 subsystems panel is displayed (Figure 9-11).

```
CKZ1DBSS                        DB2 subsystems
Command ===>                                                Scroll ===> PAGE

Commands:      C - Create
Line Commands: D - Delete  E - Edit  V - View  C - Copy
                                                                  Row 1 of 3
Cmd SSID MBRNAME  GRPNAME  Description
    DB0A                   Source system
    DB0C                   Source and target System
    DB0D                   Target system
```

Figure 9-11 Defining subsystems

On this panel, you define the subsystem environment, including whether the subsystem is to be used as a source, target, or both.

Selecting Create or Edit provides the View DB2 Subsystem panel, which is the environment definitions panel (Figure 9-12 on page 424).

```
CKZ1EDDS                      View DB2 Subsystem
Option ===> 1

SSID . . . . . : DBOC
Description  . : Source and target System

DB2 ZPARMs member  . . . . : DSNZPARM
DB2 BSDS01 data set name . : DBOCB.BSDS01
DB2 BSDS02 data set name . : DBOCB.BSDS02
DB2 Loadlib1 . . . . . : DBOCT.SDSNEXIT
DB2 Loadlib2 . . . . . : DBOCT.SDSNLOAD
DB2 Loadlib3 . . . . . :
DB2 Loadlib4 . . . . . :
DB2 Loadlib5 . . . . . :

Valid command selection values are
  1  Subsystem  cloning information
  2  Tablespace cloning information
```

Figure 9-12 Subsystem environment details (Page 1 of 2)

Choose Option 1 for Subsystem cloning information. See Figure 9-13.

```
CKZ1SBCI               Subsystem cloning information
Command ===>

SSID . . . . . : DBOC
Description  . : Source and target System
                                                                More:     +
Use as Subsystem Cloning Source or Target only . :         (SOURCE, TARGET,
                                                            or blank)
System ID where this DB2 normally runs . : SC63
Group name . . . . . :           (if data sharing)
Member name  . . . . . :         (if data sharing)
Special ZPARMs member  . . . . . : DSNZSPEC
System VCAT  . . . . . : DBOCD

DDF:
  LOCATION . . . . . : DBOC
  GENERIC  . . . . . :
  LUNAME . . . . . : SCPDBOC
  PASSWORD . . . . . :
  PORT . . . . . : 38380  (1-65535, or blank)
  RESPORT  . . . . . : 38381  (1-65535, or blank)
  SECPORT  . . . . . : 38382  (1-65535, or blank)
  IPNAME . . . . . :
```

Figure 9-13 Subsystem environment details (Page 2 of 2)

Note the requirement to provide a name for a special DSNZPARMs member. This special DSNZPARMs member is required to allow updates to the DB2 catalog during the cloning process. See “The DSNZSPEC DSNZPARM” on page 408. However, this special DSNZPARMs member is only necessary when configuring the clone and it is not necessary for the source DB2.

Now, we return to the Primary Option Menu and select option 1 Clone. See Figure 9-14 on page 425.

```
CKZ1PRIM ng Tool for z/OS    Primary Option Menu
Option ===> 1

0  User settings                                        User ID . . : ADMR1
1  Clone                                                System ID . : SC63
2  Administrator functions                              Appl ID . . : CKZ
X  Exit                                                 Version . . : 3.1
```

Figure 9-14 Select option 1 Clone

Selecting option 1 Clone displays Figure 9-15.

```
CKZ1CLON ng Tool for z/OS    Clone
Option ===> 1

1  DB2 subsystem clone
2  DB2 tablespace clone
```

Figure 9-15 Select option 1 for DB2 subsystem clone

We choose option 1 DB2 subsystem clone. We create a profile by using the line command C. See Figure 9-16.

```
CKZ1DSCL             DB2 Subsystem Clone Profile Display
C EsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssN => PAGE
  e CKZ1DSCN                                                              e
C e Enter New DB2 Subsystem Clone Profile Options:                        e
L e   Creator . . . . . : ADMR1                                           e  - Copy
  e   Profile Name  . . . SOURCE DBOC BACKUP SYSTEM                       e
P e   Description . . . . DBOC - DBOD                                     e
C e   Share Option  . . . UPDATE     (Update, View only, No access)      e
  e                                                                       e  1 of 3
C e  Press ENTER or F3 to process                                         e
  e  Press F12 to cancel                                                  e
c e                                                                       e
  e                                                                       e
  DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 9-16 Creating a new clone profile

The Share Option field controls what everyone, except the creator, can do with the profile. The creator has full access to the profile.

Press PF3 to save the profile, and then, we can edit the profile as shown in Figure 9-17 on page 426.

```
CKZ1DSCL            DB2 Subsystem Clone Profile Display
Command ===>                                                Scroll ===> PAGE

Commands:      C - Create
Line Commands: B - Build  D - Delete  E - Edit  R - Rename  V - View  C - Copy

Profile Like . . . *
Creator Like . . . *
                                                                  Row 1 of 3
Cmd Name                          Creator  Share Option Description
    SOURCE DBOA                   ADMR1    UPDATE       DBOA - DBOC
e   SOURCE DBOC BACKUP SYSTEM     ADMR1    UPDATE       DBOC - DBOD
    SSID CLONE                    ADMR1    UPDATE
```

Figure 9-17 Editing the clone profile

We select e to edit the BACKUP SYSTEM procedure. Figure 9-18 displays. On Figure 9-18, we select option 1 Select Source and Target DB2 subsystems.

```
CKZ1ESCP                 Edit DB2 Subsystem Clone Profile
Option ===> 1

Creator  . . . : ADMR1        Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE       Description . . DBOC - DBOD

1  Select Source and Target DB2 subsystems
2  Select Source and Target Volume Pairing
3  Select Source and Target ICF catalogs
4  Select Rename masks
5  Select other parameters
```

Figure 9-18 Choose option 1 Select Source and Target DB2 Subsystems

On Figure 9-19, select command A to add a subsystem pairing.

```
CKZ1SSTS           Select Source and Target DB2 Subsystems
Command ===> a                                              Scroll ===> PAGE

Commands:      A - Add
Line Commands: D - Delete  E - Edit  V - View  T - Cloning Type

Creator  . . . : ADMR1        Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE       Description . : DBOC - DBOD


Cmd SSID MBRNAME  GRPNAME  TGTS TGTMBR   TGTGRP   TYPE    TGTDSA
```

Figure 9-19 Defining source and target subsystems

On Figure 9-20 on page 427, select a subsystem that has been previously defined in the administration panels.

```
CKZ1SSDS                Select Source DB2 Subsystem
Command ===>                                              Scroll ===> PAGE

Line Commands: S - Select a source DB2 subsystem

Creator  . . . : ADMR1        Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE       Description . : DBOC - DBOD

                                                                  Row 1 of 2
Cmd SSID MBRNAME  GRPNAME  Description
    DBOA                   Source system
s   DBOC                   Source and target System
```

Figure 9-20 Select a source DB2 subsystem

We define the type of cloning. See Figure 9-21. To use an SLB, we select ONLINE for the Type of cloning. See “Offline or online” on page 418.

Our scenario is not a data sharing environment.

```
CKZ1SCLT                    Select cloning type
Command ===>

Creator  . . . : ADMR1        Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE       Description . : DBOC - DBOD

Type of cloning  . . . . . . . . . . . . . ONLINE  (ONLINE, OFFLINE)

If the source DB2 is data sharing group:
   Data sharing attributes of target . . : NONDS    (SAME, FEWER, or NONDS)
```

Figure 9-21 Select cloning type

We select the target subsystem.

Note: You can only select a subsystem that has been defined in the administration panels as a target.

See Figure 9-22.

```
CKZ1STDS                Select Target DB2 Subsystem
Command ===>                                              Scroll ===> PAGE

Line Commands: S - Select a target subsystem for source DB2 subsystem DBOC

Creator  . . . : ADMR1        Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE       Description . : DBOC - DBOD

                                                                  Row 1 of 2
Cmd SSID MBRNAME  GRPNAME  Description
    DBOC                   Source and target System
s   DBOD                   Target system
```

Figure 9-22 Select Target DB2 Subsystem

We need to define several settings for the target environment. We need to step through each option, but not all options need definitions. See Figure 9-23. A full description of the options is provided in the *DB2 Cloning Tool for z/OS, V3.1, User's Guide*, SC19-3493-01.

Many of the subsystem values have already been set when defining the subsystems using the Administration option.

```
CKZ1EDCV                    Edit DB2 cloning values
Option ===> 1

Creator  . . . : ADMR1         Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE        Description . : DBOC - DBOD
                               SSID  . . . . : DBOD

1   DB2 HLQS
2   STOGROUPS
3   DDF
4   WLM ENVIRONMENT MASKS
5   DATACLAS-MASKS (DB2 V9)
6   MGMTCLAS-MASKS (DB2 V9)
7   STORCLAS-MASKS (DB2 V9)
8   DB2FIX command
9   DB2SQL command
10  DB2START command
11  DB2STOP command
12  DB2UPDATE command
```

Figure 9-23 Setting target cloning values

The high-level qualifiers (HLQs) of the DB2 source and target subsystems must be provided. See Figure 9-24. Verify that they have already been defined when the subsystems were configured under the Administrative options, or provide the qualifiers by using the Enter DB2 HLQs panel.

For DB2 data sharing, there might be more HLQs that need to be specified here. All VCATS in use in the DB2 system should be specified here. If you do not specify all the VCATS that are in use on the source system, the source DB2 system might become corrupted. Add any additional HLQs here.

```
CKZ1HLQS                       Enter DB2 HLQs
Command ===>                                          Scroll ===> PAGE

Commands:      A - Add Line
Line commands: D - Delete Line

Creator  . . . : ADMR1         Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE        Description . : DBOC - DBOD
                               SSID  . . . . : DBOD

                                                          Row 1 of 3
Cmd Source HLQ Target HLQ
    DBOCD      DBODD
```

Figure 9-24 Enter DB2 HLQs

Enter any masking that is needed for DB2 STOGROUP. If renaming a DB2 STOGROUP, the target STOGROUP name must be the same length as the source STOGROUP name. If they are not the same length, when the DB2UPDATE job is run, you get the following message:

```
CKZ22460E TARGET STORAGE GROUP NAME MUST BE THE SAME LENGTH AS THE SOURCE STORAGE
```

See Figure 9-25.

```
CKZ1STRG                    Enter DB2 STOGROUPs
Command ===>                                              Scroll ===> PAGE

Commands:      A - Add Line
Line commands: D - Delete Line

Creator  . . . : ADMR1          Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE         Description . : DBOC - DBOD
                                SSID  . . . . : DBOD

                                                                Row 1 of 3
Cmd Source Stogroup               Target Stogroup
    DBOCLRG                       DBODLRG
```

Figure 9-25 After selecting option 2 Stogroups on the Editing DB2 cloning values menu

Note: In DB2 Cloning Tool V3.1, APAR PM76860 (PTF UK83725) has resolved the following error that relates to STOGROUPS:

```
CKZ24631I   UPDATE SYSIBM.SYSSTOGROUP
 CKZ24631I     SET NAME = 'DBODLRG' WHERE NAME = 'DBOCLRG' ;
 CKZ24621E SQL EXECUTE  FAILED, SQLCODE:     -531 SQLSTATE: 23504
  DSNT408I SQLCODE = -531, ERROR:  PARENT KEY IN A PARENT ROW CANNOT BE UPDATED
  BECAUSE IT HAS ONE OR MORE DEPENDENT ROWS IN RELATIONSHIP DSNSS@SV
```

This required not specifying STOGROUPs, if SMS-managed, and making the changes post cloning using SQL.

On Figure 9-26 on page 430, after selecting option 3 DDF on the Editing DB2 cloning values menu, we review the settings that were used when the subsystems were defined.

```
CKZ1EDDF                    Edit DB2 DDF values
Command ===>

Commands: A - ALIAS values

Creator  . . . : ADMR1          Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE         Description . : DBOC - DBOD
                                SSID  . . . . : DBOD

GENERIC  . . . .
GRPIPV4 (V9) . .
GRPIPV6 (V9) . .
IPNAME  (V9) . .
IPV4    (V9) . .
IPV6    (V9) . .
LOCATION . . . . DBOD
LUNAME . . . . . SCPDBOD
PASSWORD . . . .
PORT . . . . . 38390  (1-65535)
RESPORT  . . . . 38391  (1-65535)
SECPORT (V9) . . 38392  (1-65535)
```

Figure 9-26 After selecting option 3 DDF values

After we select option 4 WLM ENVIRONMENT MASKS on the Editing DB2 cloning values menu, Figure 9-27 is displayed. The Workload Manager (WLM) environments should already be set up and defined on the target. In Figure 9-27, we define the mapping for updating the catalog on the target for all of the stored procedures.

```
CKZ1WLEM              Enter DB2 WLM Environment masks
Command ===>                                              Scroll ===> PAGE

Commands:      A - Add Line
Line commands: D - Delete Line

Creator  . . . : ADMR1          Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE         Description . : DBOC - DBOD
                                SSID  . . . . : DBOD

                                                                Row 1 of 3
Cmd Source WLM environment mask      Target WLM environment mask
    DBOC*                            DBOD*
```

Figure 9-27 Option 4 allows us to define the WLM environments

Options 5 to 7 on the Editing DB2 cloning values menu allow us to rename the SMS rules when they are specified within the stogroup definition. Generally, and in our test environment, system-wide SMS rules are defined. These options provide an alternative to renaming a DB2 STOGROUP. See Figure 9-28.

```
5   DATACLAS-MASKS (DB2 V9)
6   MGMTCLAS-MASKS (DB2 V9)
7   STORCLAS-MASKS (DB2 V9)
```

Figure 9-28 SMS masking

The DB2FIX command is only required when performing an online clone. Online cloning is when a DB2 SET LOG SUSPEND command is issued on the source while the FlashCopy is taken or a consistent flash or a consistent mirror break. An SLB (DB2 BACKUP SYSTEM or Recovery Expert system level backup) does not necessitate a log suspend, but it will still require the DB2FIX command.

Some page spaces might be left with LPL or GRECP status. The DB2FIX command will fix target DB2 page spaces that have LPL or GRECP status by issuing a DB2 START DATABASE command against them. See Figure 9-29.

```
CKZ1PFIX                Edit DB2FIX command settings
Command ===>

Creator  . . . : ADMR1         Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE        Description . : DBOC - DBOD
                               SSID  . . . . : DBOD

DSNDBO1-DBDO1-STARTED RC . . . 16         (0-4095)
MAX-CONCURRENT-CMDS  . . . . . 1          (1-99)
MEMBERS-AND-DBDO1 RC . . . . . 16         (0-4095)
MEMBERS-NEED-STARTING RC . . . 8          (0-4095)
START-SCOPE  . . . . . . . . . PAGESPACE  (DATABASE, PAGESPACE)
WAIT ACTION  . . . . . . . . . QUIT       (QUIT, CONTINUE)
WAIT RC  . . . . . . . . . . . 8          (0-4095)
WAIT TIME  . . . . . . . . . . 5          (0-999)
WAIT-AND-DBDO1 RC  . . . . . . 16         (0-4095)
```

Figure 9-29 Option 8 DB2FIX command on the Editing DB2 cloning values menu

The DB2SQL command generates and executes the SQL statements that are necessary to update the DB2 catalog. The values represent the return code to be used if there is a WLM ENVIRONMENT value in SYSIBM.SYSROUTINES that is not updated. See Figure 9-30.

The most likely cause of this condition is that the definition for WLM masking needs updating. It is feasible that WLM environments can be the same but it is not recommended.

```
CKZ1PSQL                Edit DB2SQL command settings
Command ===>

Creator  . . . : ADMR1         Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE        Description . : DBOC - DBOD
                               SSID  . . . . : DBOD

LISTSQL . . . . . . . . . . . YES   (Yes/No)
WLM-ENV-NOT-UPDATED RC  . . . 4     (0-4095)
DATACLAS-NOT-UPDATED RC . . . 4     (0-4095)
MGMTCLAS-NOT-UPDATED RC . . . 4     (0-4095)
STORCLAS-NOT-UPDATED RC . . . 4     (0-4095)
```

Figure 9-30 Option 9 DB2SQL command on the Editing DB2 cloning values menu

The DB2 Cloning Tool needs to stop and start the target DB2 during the process, and options 10 DB2START command and 11 DB2STOP command define the behavior of the jobs. Again, here the defaults are used as set in the user option. See Figure 9-31 on page 432.

```
CKZ1PSTT               Edit DB2START command settings
Command ===>

Creator  . . . : ADMR1         Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE        Description . : DBOC - DBOD
                               SSID  . . . . : DBOD

DB2-ALREADY-RUNNING RC . . . 8     (0-4095)
WAIT time  . . . . . . . . . . 5     (0-999)
WAIT RC  . . . . . . . . . . . 8     (0-4095)
```

Figure 9-31 Option 10 DB2START command on the Editing DB2 cloning values menu

With both the stop and start of the target DB2, we want to ensure that the DB2 Cloning Tool is in control because there are critical steps in which DB2 Cloning Tool needs exclusive use of DB2. We therefore ensure that the processing stops if DB2 has been stopped or started outside of the cloning process.

With the DB2STOP command settings, we can use a Force option but only if we fully understand what is currently running that is preventing DB2 from shutting down. See Figure 9-32.

```
CKZ1PSTP                Edit DB2STOP command settings
Command ===>

Creator  . . . : ADMR1         Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE        Description . : DBOC - DBOD
                               SSID  . . . . : DBOD

CASTOUT  . . . . . . . . . . . YES      (Yes/No)
DB2-ALREADY-STOPPED RC . . . 8        (0-4095)
MODE . . . . . . . . . . . . . QUIESCE  (FORCE, QUIESCE)
WAIT time  . . . . . . . . . . 5        (0-999)
WAIT RC  . . . . . . . . . . . 8        (0-4095)
```

Figure 9-32 Option 11 DB2STOP command on the Editing DB2 cloning values menu

We have not considered the archives with our clone, but if the DB2 archive logs are on the source DASD volumes and are then copied to the target volumes, you will need to update the BSDS with the renamed archives.

On Figure 9-33 on page 433, on the Edit DB2UPDATE command settings panel, specify YES to “Update Archive log data set names in BSDS”.

```
CKZ1PDAT               Edit DB2UPDATE command settings
Command ===>

Creator  . . . : ADMR1         Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE        Description . : DBOC - DBOD
                               SSID  . . . . : DBOD

DB2-XCFCLEAN . . . . . . YES   (Yes/No)
DDF-NOT-UPDATED RC . . . 4     (0-4095)
HLQ-NOT-UPDATED RC . . . 4     (0-4095)

Update Archive log data set names in BSDS . . . NO   (Yes/No)
```

Figure 9-33 Option 12 DB2UPDATE command on the Editing DB2 cloning values menu

The Edit DB2 Subsystem Clone Profile panel is displayed (Figure 9-34).

```
CKZ1ESCP                 Edit DB2 Subsystem Clone Profile
Option ===> 2

Creator  . . . : ADMR1        Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE       Description . . DBOC - DBOD

1  Select Source and Target DB2 subsystems
2  Select Source and Target Volume Pairing
3  Select Source and Target ICF catalogs
4  Select Rename masks
5  Select other parameters
```

Figure 9-34 Option 2 Select Source and Target Volume Pairing

When using the DB2 SLB as the source, on the Select Source and Target Volume Pairing panel, you select option 3 Source System Level Backup, as shown in Figure 9-35. If SLB is used, any settings that are specified by using the Source Storage Group or Source Volume options will be ignored.

```
CKZ1SSTV            Select Source and Target Volume Pairing
Option ===> 3

Creator  . . . : ADMR1          Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE         Description . : DBOC - DBOD

1  Source Storage Group
2  Source Volume
3  Source System Level Backup
4  Target Storage Group
5  Target Volume
```

Figure 9-35 Select Source and Target Volume Pairing panel

From the Set Source System Level Backup panel (Figure 9-36), for System Level Backup Type, you need to enter DB2SLB. You can list the SLB by using the Recovery Expert Tool or looking at the output of the DSNJU004 utility.

Use the Token keyword LAST to use the last successful SLB. Any other value is the specific token value as reported by hierarchical storage management (HSM), Recovery Expert, or the output of the DSNJU004 utility. This is covered in “Backups” on page 463. See Figure 9-36.

The Location field identifies the specific DB2 source system whose system level backup is to be used.

```
CKZ1SSLB            Set Source System Level Backup
Option ===>

Creator  . . . : ADMR1          Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE         Description . : DBOC - DBOD

System Level Backup Type . . DB2SLB  (DB2SLB or blank)

Token  . . . LAST                                  (token or LAST)
Location . . DBOC
```

Figure 9-36 Set Source System Level Backup

You now need to exit this panel, and on the Select Source and Target Volume Pairing panel, select option 4 Enter Target Storage Groups. See Figure 9-37.

We need to define the target volumes to be used by naming the storage groups for the target environment. Determining these groups can be done by using the options already covered, such as the FINDUCAT, ADMSMS, or DB2 Recovery Expert. There must be as many target volumes as there are source volumes (as provided by HSM or DB2 Recovery Expert).

```
CKZ1TSGR                Enter Target Storage Groups
Command ===>                                              Scroll ===> PAGE

Commands:      A - Add Line
Line commands: D - Delete Line

Creator  . . . : ADMR1          Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE         Description . : DBOC - DBOD

                                                                  Row 1 of 3
Cmd Target Storage Group Name or Mask
    DBODDATA
    DBODLOG1
    DBODLOG2
```

Figure 9-37 Target DB2 Storage Groups

You should now return to the Edit DB2 Subsystem Clone Profile panel by pressing PF3. From the Edit DB2 Subsystem Clone Profile panel, we select option 3 Select Source and Target ICF catalogs.

You specify the source catalogs in which the source data sets are cataloged, and the corresponding target catalogs in which the renamed volume data sets are to be cataloged. See Figure 9-38.

We have described the ICF catalogs and their physical locations in “ICF catalog” on page 419.

In our environment, we have two ICF catalogs in scope for cloning. One ICF catalog is for the logs and the otherICF catalog is for the data.

Note: It is important that the target volumes do not contain the target ICF catalogs during cloning.

```
CKZ1SSTC            Select Source and Target ICF catalogs
Command ===>                                              Scroll ===> PAGE

Commands:      A - Add Line
Line commands: D - Delete Line

Creator  . . . : ADMR1          Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE         Description . : DBOC - DBOD

                                                                Row 1 of 3
Cmd Source ICF Catalog                 Target ICF Catalog
    UCAT.DBOCD                         UCAT.DBODD
    UCAT.DBOCL                         UCAT.DBODL
```

Figure 9-38 Select Source and Target ICF catalogs

When you exit from pairing the ICF catalogs, from the Edit DB2 Subsystem Clone Profile menu, select option 4 Select Rename Masks, and then, select option 1 Rename Masks. There is an option to manage exclude masks, but we have no requirement for this option in our environment.

On the Rename Masks panel (Figure 9-39 on page 436), we determine the names of our source data sets on the target. Using the SLB volumes, you hopefully find only the logs and DB2 object data sets. Therefore, complying with your target naming convention can be covered with a couple of simple masks as shown in Figure 9-39 on page 436. You can provide a mask for other data sets, if present, with a generic name that you can clean up after the clone.

The rename masks are processed in the order that you define them.

```
CKZ1RNMS                          Rename Masks
Command ===>                                                Scroll ===> PAGE

Commands:      A - Add Line
Line commands: D - Delete Line

Creator  . . . : ADMR1          Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE         Description . : DBOC - DBOD

                                                                  Row 1 of 3
Cmd Source Mask                         Target Mask
    DBOCD.**                            DBODD.**
    DBOC%.**                            DBOD%.**
```

Figure 9-39 Rename Masks

We are nearly finished. After we exit the rename panels, from the Edit DB2 Subsystem Clone Profile menu, we click option 5 Select other parameters. Many of these parameters were set as defaults in the administration panels but can be overwritten here and saved with the profile.

We set the prefix (HLQ) for the work data sets to be used for this profile. See Figure 9-40.

```
CKZ1SOPR                    Select other parameters
Option ===>

Creator  . . . : ADMR1        Name  . . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE       Description . . : DBOC - DBOD

Prefix for work data sets . . . DB2TOOLS.CLONE.SLB2
Work data sets unit device  . . SYSDA     (SYSDA, DISK, or etc.)

1  COPY command
2  COPYCHECK command
3  RENAME command
4  Specify SMS classes for RENAME command
```

Figure 9-40 Option 5 Select other parameters

We step through these options to set how the cloning process should react to conditions shown by the cloning process: what conditions are acceptable and how to deal with them.

We start with the COPY command that is shown in Figure 9-41 on page 437. You can specify the data mover program. If you are using EMCSNAP, enter it here on the DATA-MOVER PGM field to instruct the DB2 Cloning Tool.

```
CKZ1DSCC                  DB2 subsystem COPY command
Option ===>

Commands: P - DATA-MOVER PGM values

Creator  . . . : ADMR1        Name  . . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE       Description . . : DBOC - DBOD

TARGET-VOLS-SHOULD-BE-EMPTY . . NO   (Yes/No)
TARGETSUONLINE  . . . . . . . . YES  (Yes/No)

CATWORK-DSN Mask  . . . . . . . DB2TOOLS.CLONE.SLB2.*
CATWORK-ATTR  . . . . . . . . . UNIT(SYSALLDA) SPACE(10 50) CYLINDERS      >

DATA-MOVER PGM  . . . . . . . . ADRDSSU  (ADRDSSU/EMCSNAP)
```

Figure 9-41 DB2 subsystem COPY command

On Figure 9-41, there is also an option to configure values for the Data-Mover Program by selecting option P for use with the ADRDSSU program. Enter P and Figure 9-42 is displayed.

Generally, the defaults are acceptable. Support for cloning volumes, where the clone's target volumes are sources for Peer-to-Peer Remote Copy (PPRC) mirroring, can be defined here using FCTOPPRCPRIMARY.

```
CKZ1ADRD          DB2 Subsystem COPY via ADRDSSU parameters
Option ===>

Creator . . . . : ADMR1       Name  . . . . . : SOURCE DBOC BACKUP SY
Share Option  . : UPDATE      Description . . : DBOC - DBOD

DATA-MOVER PGM(ADRDSSU) parameters:
  CONSISTENT  . . . . . . . . . . NO             (Yes/No)
  COPYCMDLIMIT  . . . . . . . . . 24             (1-250)
  DSSPARM . . . . . . . . . . . .
  FASTREP . . . . . . . . . . . . REQ            (PREF, REQ, or NONE)
  INCREMENTAL . . . . . . . . . . NO             (Yes/No)

Add parameter to ADRDSSU COPY command:
  CHECKVTOC . . . . . . . . . . . NO             (Yes/No)
  FCNOCOPY  . . . . . . . . . . . NO             (Yes/No)
  FCSETGTOK . . . . . . . . . . . NO             (Yes/No)
  FCTOPPRCPRIMARY . . . . . . . . NO             (Yes, No, PRESMIRREQ,
                                                  PRESMIRRPREF, or PRESMIRNONE)
  NOCONCURRENT  . . . . . . . . . NO             (Yes/No)
```

Figure 9-42 Option P Data-Mover program parameters for ADRDSSU

With the Rename parameters, we only focused on the NOTRENAMED RC option. See Figure 9-43 on page 438.

You may find that there are data sets that came from the source volumes that are not interesting because they do not belong to the logs or the DB2 data. This data may not match a rename mask that you specified. The default setting causes the RENAME step to fail with a

return code (RC) of 8. If you find it acceptable to have orphaned data sets on the target, you can change the job to accept this condition and finish with an RC of 4.

Leave the RC as 8 and if the RENAME job fails, you can analyze the output and change the setting within the step's JCL and then rerun the step.

The RENAME TYPE is set to SAFE in our run. By using SAFE, you can rerun the job if it fails by editing the step's JCL and replacing SAFE with RERUN and changing the DISP of the cataloged data sets to OLD. You also must remove the DELETE step.

An alternative to running SAFE and having the step fail is to use the SIMULATE option. This allows you to run without actually making the changes but still highlighting potential issues. Using the simulate feature enables you to fix the issues before making the changes.

One other option to note here is RECATALOG. By setting this option to NO, it is assumed that the target catalog is empty and therefore no duplicates or already cataloged messages exist. If you are cloning to a dirty catalog, set this option to YES and the DB2 Cloning Tool will overwrite the old entry.

```
CKZ1DRNC                DB2 subsystem RENAME command
Option ===>

Creator  . . . : ADMR1        Name  . . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE       Description . . : DBOC - DBOD
                                                              More:
EXCLUDE-SRCNAME  . . . . . . . NOTRENAMED-RC   (NOTRENAMED-RC, 0)
GDG-ALL-MIGRATED . . . . . . . SKIP            (SKIP, RETAIN)
GDG-ALL-MIGRATED RETAIN RC . . 4               (0, 4)
GDG-EMPTY  . . . . . . . . . . SKIP            (SKIP, RETAIN)
GDG-EMPTY RETAIN RC  . . . . . 4               (0, 4)
GDG-MIGRATED . . . . . . . . . ERROR           (ERROR, RETAIN)
GDG-MIGRATED RETAIN RC . . . . 4               (0, 4)
GDG-TAPE . . . . . . . . . . . ERROR           (ERROR, RETAIN)
GDG-TAPE RETAIN RC . . . . . . 4               (0, 4)
ISSUE-VCLOSE . . . . . . . . . YES             (NO, YES, BEFORE, or AFTER)
ISSUE-VCLOSE SCOPE . . . . . . LOCAL           (LOCAL, SYSPLEX)
MAX-TASKS  . . . . . . . . . . 5               (1-255)
MISSINGUCAT  . . . . . . . . . KEEP            (DELETE, KEEP)
MISSINGUCAT RC . . . . . . . . 4               (0, 4, or 8)
NOTRENAMED . . . . . . . . . . KEEP            (DELETE, KEEP)
NOTRENAMED RC  . . . . . . . . 8               (0, 4, or 8)
ORPHANCATENTRY . . . . . . . . KEEP            (DELETE, KEEP)
ORPHANCATENTRY RC  . . . . . . 8               (0, 4, or 8)
RECATALOG  . . . . . . . . . . NO              (Yes/No)
RENAME-AUDIT-LOG . . .   . . . NO              (No, SMF)
RENAME-AUDIT-LOG SMF . . . . . 130             (128-255)
RENAME-ERROR . . . . . . . . . ABORT           (ABORT, CONTINUE)
RENAME-ERROR CONTINUE RC . . . 8               (0, 4, or 8)
RENAME-LIST  . . . . . . . . . NO              (Yes/No)
RENAME TYPE  . . . . . . . . . SAFE            (SAFE, SPEED)
TEMPDSN  . . . . . . . . . . . DELETE          (DELETE, KEEP)
TEMPDSN RC . . . . . . . . . . 4               (0, 4, or 8)
UPDATE-IAM-ASSOCIATIONS. . . . NO              (Yes/No)
VALIDATE-SMS-CLASSES . . . . . YES             (Yes/No)
VOLBKUP-DDN  . . . . . . . . . VOLBKUP
```

Figure 9-43 RENAME Command parameters

The SMS rules option refers to the SMS DATACLAS, MGMTCLAS, and STORCLAS that the renamed (target) data sets should be given in their ICF catalog entries to be consistent with the SMS rules as defined by the user's installation. If not changed by this option, the target data sets will potentially not have the correct SMS DATACLAS, MGMTCLAS, and STORCLAS in their ICF catalog entries. This will not affect use of the data sets, but a site utility that reports on them will show an incorrect (source) value.

Without specifying any rules, the target data sets will take on the source values in the ICF catalog entries (Figure 9-44). For example, we will set the changes for the DATACLAS. There is no masking for the names. Explicit translations need to be specified. You can verify the SMS rules for a data set by using the ISMF dialog in ISPF.

```
CKZ1SRCV                Specify SMS STORCLAS values
Command ===>                                                Scroll ===> PAGE

Commands:      A - Add Line
Line commands: D - Delete Line

Creator  . . . : ADMR1          Name  . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE         Description . : DBOC - DBOD

Enter either one STORCLAS or one or more STORCLAS pairs:
  STORCLAS . . .            (storage class/SOURCE)
                                                                  Row 1 of 4
Cmd Source STORCLAS Target STORCLAS
    DBOCDATA        DBODDATA
    DBOCLOG1        DBODLOG1
    DBOCLOG2        DBODLOG2
    DBOADATA        DBODDATA
```

Figure 9-44 Specify SMS STORCLAS

We can exit to the profile lists. We are now ready to build the jobs that we need for our clone. See Figure 9-45 on page 440.

```
CKZ1DSCL            DB2 Subsystem Clone Profile Display
Command ===>                                                Scroll ===> PAGE

Commands:      C - Create
Line Commands: B - Build  D - Delete  E - Edit  R - Rename  V - View  C - Copy

Profile Like . . . *
Creator Like . . . *
                                                                  Row 1 of 3
Cmd Name                           Creator  Share Option Description
    SOURCE DBOA                    ADMR1    UPDATE       DBOA - DBOC
B   SOURCE DBOC BACKUP SYSTEM      ADMR1    UPDATE       DBOC - DBOD
    SSID CLONE                     ADMR1    UPDATE



              EsssssssssssssssssssssssssssssssssssssssssssssN
              e CKZ011I - Operation completed successfully e
              DsssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 9-45 DB2 Subsystem Clone Profile Display after updating

The next panel as shown in Figure 9-46 provides the place to specify the PDS in which to save the JCL. It is important that you give the PDS a unique name for each profile as you might overwrite another clone job set.

```
CKZ1DSCJ               Build DB2 Subsystem Clone jobs
Option ===>

Creator  . . . : ADMR1        Name  . . . . . : SOURCE DBOC BACKUP SY
Share Option . : UPDATE       Description . . : DBOC - DBOD

The jobs to do the DB2 Subsystem Clone will be placed in the provided data set
  Data Set Name . . . ADMR1.CKZSLB3.JCLLIB

Processing options
  Enter "/" to select option
  /  Warn if any members in output data set already exist
  /  Edit JCL data set
```

Figure 9-46 Build DB2 Subsystem Clone jobs

We exit from this panel, and our jobs are generated in ADMR1.CKZSLB3.JCLLIB. See Figure 9-47 on page 441.

```
CKZ1DSML             Edit ADMR1.CKZSLB3.JCLLIB
Command ===>                                                Scroll ===> PAGE

Line Commands: E - Edit  V - View
                                                                  Row 1 of 20
Cmd Name     Created    Changed             ID
    ST01     2012/10/01 2012/10/01 07:31:58 ADMR1
    ST02     2012/10/01 2012/10/01 07:31:58 ADMR1
    ST03     2012/10/01 2012/10/01 07:31:58 ADMR1
    ST04     2012/10/01 2012/10/01 07:31:58 ADMR1
    ST05     2012/10/01 2012/10/01 07:31:58 ADMR1
    ST06     2012/10/01 2012/10/01 07:31:58 ADMR1
    ST07     2012/10/01 2012/10/01 07:31:58 ADMR1
    ST08     2012/10/01 2012/10/01 07:31:59 ADMR1
    ST09DB0D 2012/10/01 2012/10/01 07:31:59 ADMR1
    ST11DB0D 2012/10/01 2012/10/01 07:31:59 ADMR1
    ST13DB0D 2012/10/01 2012/10/01 07:31:59 ADMR1
    ST14DB0D 2012/10/01 2012/10/01 07:31:59 ADMR1
    ST15DB0D 2012/10/01 2012/10/01 07:31:59 ADMR1
    ST16DB0D 2012/10/01 2012/10/01 07:31:59 ADMR1
    ST17DB0D 2012/10/01 2012/10/01 07:31:59 ADMR1
    ST18DB0D 2012/10/01 2012/10/01 07:32:00 ADMR1
    ST19DB0D 2012/10/01 2012/10/01 07:32:00 ADMR1
    ST23DB0D 2012/10/01 2012/10/01 07:32:00 ADMR1
    ST24DB0D 2012/10/01 2012/10/01 07:32:00 ADMR1
    ST25     2012/10/01 2012/10/01 07:32:00 ADMR1
```

Figure 9-47 Clone jobs generated

We have already verified that we have a valid SLB. In addition, we must check that we have valid volumes on the target. You can use either FINDUCAT and ADMSMS. From this point, we step through the jobs that we run.

9.5.3 System level backup jobs

We will summarize the jobs that are run and highlight any issues or tips that might be useful. A full description of the function of each step is in Chapter 26, “Cloning scenarios” in the *DB2 Cloning Tool for z/OS, V3.1, User’s Guide*, SC19-3493-01.

Table 9-2 shows the jobs and the DB2 Cloning Tool command that we use.

Table 9-2 Cloning jobs member reference

Member	Command name	Special parameters
ST01	DB2GETBACKINFO	
ST02	BACKINFO-REFORMAT	
ST03	COPY	DATA-MOVER(PGM(NONE))
ST04	COPY	DATA-MOVER(PGM(ADRDSSU)
ST05	COPYCHECK	
ST06	CKZRNTGT	

Member	Command name	Special parameters
ST07	VOLOPTIONS UPDATE	
ST08	RENAME	SAFE
ST09DB0D	DB2UPDATE	BSDS
ST11DB0D	DB2START	DSNZPARM(DSNZSPEC)
ST13DB0D	DB2FIX	DATABASES(DB2)
ST14DB0D	DB2STOP	
ST15DB0D	DB2UPDATE	DBD01ONLY
ST16DB0D	DB2START	DSNZPARM(DSNZSPEC)
ST17DB0D	DB2SQL	
ST18DB0D	DB2FIX	DATABASES(APPLICATION)
ST19DB0D	DB2STOP	MODE(QUIESCE)
ST23DB0D	DB2START	NORMAL
ST24DB0D	DB2STOP	For cleanup
ST25	BCSCLEAN	Clean up for rerun

ST01 DB2GETBACKINFO

This step (ST01) retrieves the source to back up volume pairing (SRC*xxx* to BKP*xxx*) and identifies the source ICF catalogs from the *LAST* DB2 BACKUP SYSTEM taken for location DB0C.

Note: Ensure that the token of the backup, which you can check, matches what you requested. Information is in the BSDS and it can be reviewed by running DSNJU004 to print.

The ICF catalogs are copied to the COPYPOOL with the DB2 BACKUP SYSTEM command. The backup volumes refer to the COPYPOOL from the SLB. Sample syntax and output are in Example 9-10.

Example 9-10 DB2GETBACKINFO SLB output

```
//CKZIN    DD  *
   DB2GETBACKINFO   -
     BACKINFO-DDN(BACKINFO) -
     WORK-DDN(HSMLIST) -
     LAST-
     LOCATION(DB0C) -
     USERCATALOGS( -
            UCAT.DB0CD -
            UCAT.DB0CL -
                 )
//*
.............
CKZ21501I 20.55.27 DB2GETBACKINFO STARTED - PROGRAM REV=1
CKZ21586I VALIDATING KEYWORD: BACKINFO-DDN
CKZ21586I VALIDATING KEYWORD: WORK-DDN
CKZ21586I VALIDATING KEYWORD: LOCATION
```

```
CKZ21586I VALIDATING KEYWORD: USERCATALOGS

CKZ21581I DSNS FOR KEYWORD: USERCATALOGS
          UCAT.DBOCD
          UCAT.DBOCL

CKZ21535I SELECTED BACKUP
          DATABASE                LOG          TOKEN                            NUM
       DATE      TIME        DATE      TIME                                   VOL PAIRS
    2012/05/08 15:56:20   2012/05/08 15:56:43   C4C2F0C3C989C1090B27D7F00018C13AA868 21
       SOURCE  BACKUP
       SBOXJK  X5DAOD
       SBOXJL  SBOXKP
       SBOXJM  X8DOO2
       SBOXJN  SBOXKQ
      X66125  X66525
.....................................................

CKZ21545I USERCATALOG: UCAT.DBOCD WAS FOUND ON BACKUP VOLUME: X66725
CKZ21545I USERCATALOG: UCAT.DBOCL WAS FOUND ON BACKUP VOLUME: X8D400

CKZ21530I       21 VOLMAP RECORDS WRITTEN
CKZ21530I        2 UCAT   RECORDS WRITTEN


CKZ21501I 20.55.28 DB2GETBACKINFO COMPLETED; RETURN CODE=0

CKZ01009I DB2 CLONING TOOL EXECUTION COMPLETE. HIGHEST RETURN CODE WAS 0
```

ST02 BACKINFO-REFORMAT

This step is to reformat the output of step 1 (BACKINFO data set) for use in the COPY in steps 3 and 4. See Example 9-11.

The user catalog pairs are also specified here.

Example 9-11 BACKINFO-REFORMAT output

```
CKZ01020I PROGRAM: CKZ01CIO 20110309 10.44 VERS=1.0 REV=35

   BACKINFO-REFORMAT           -
     BACKINFO-DDN(BACKINFO)    -
     VOLPAIRS-DDN(VOLPAIRS)    -
     FROM-VOLSER-DDN(FRVOLSER) -
     USERCATALOGS-DDN(UCATS)   -
     USERCATALOGS( -
          UCAT.DBOCD UCAT.DBODD -
          UCAT.DBOCL UCAT.DBODL -
                 )

CKZ06081I DSNS FOR KEYWORD: USERCATALOGS
          UCAT.DBOCD                                          UCAT.DBODD
          UCAT.DBOCL                                          UCAT.DBODL


CKZ06042I VOLUME PAIRS BEING USED:
          SOURCE  BACKUP
          SBOXJI  X76724
          SBOXJJ  X6642A
          SBOXJK  X5DAOD
```

```
         SBOXJL  SBOXKP
.................................................
CKZ06001I 20.55.41 BACKINFO REFORMAT COMPLETED; RETURN CODE=0
```

ST03 COPY DATA-MOVER(PGM(NONE))

This step is the first of two COPY steps to set the pairing between the source (SRC*xxx*) and backup (BKP*xxx*) volumes in the journal and to back up the source ICF catalogs from the backup volumes.

The first COPY command, shown in Example 9-12, performs these functions:

- Uses the DATA-MOVER(PGM(NONE)) keyword and does *not* perform any volume copies because the copies (set A to set B) were performed by the DB2 BACKUP SYSTEM command.
- Identifies the source volume (set A) to backup volume (set B) pairing by reading the VOLPAIRS-DDN created in the previous step and creates a new data set to which the VOLPLIST DD HLQ?.WRK.VOLPLIST points. Another VOLPLIST is created in the second COPY. These VOLPISTs will be used later.
- Backs up the source ICF catalog copies from the BACKUP SYSTEM backup volumes.
- Creates a journal data set. This is the journal data set that is used throughout the RENAME and DB2 conditioning steps and passes information between steps.

Example 9-12 COPY DATA-MOVER(PGM(NONE)) output

```
    COPY                          -
      DATA-MOVER(PGM(NONE))       -
      VOLPAIRS-DDN(VOLPAIRS)      -
      USERCATALOGS-DDN(UCATS)     -
      CATWORK-DSN(DB2TOOLS.CLONE.SLB2.*) -
      JOURNAL-DDN(JOURNAL)

 CKZ02001I 20.55.47 COPY PROCESS STARTED - PROGRAM REV=66

 CKZ02030I DSS LEVEL=X'03011300'
CKZ02043I ANTRQST LEVEL=12; ESSRVCS LEVEL=106

CKZ02087I     21 VOLUME PAIRS     FOR KEYWORD: VOLPAIRS-DDN
          SBOXJI X76724
          SBOXJJ X6642A
          SBOXJK X5DA0D
          SBOXJL SBOXKP
.....................................
 CKZ02088I VOLUME SERIAL(S) TO BE USED FOR SOURCE
           SBOXJI SBOXJJ SBOXJK SBOXJL SBOXJM SBOXJN SBOXJQ SBOXJR SBOXJU
SBOXJV
           X66526 X66628 X66727 X76029
           TOTAL NUMBER:      21

 CKZ02088I VOLUME SERIAL(S) TO BE USED FOR TARGET
           SBOXKP SBOXKQ X5DA0D X66228 X66229 X6642A X66425 X66426 X66427
X66525
           X76724 X8D002 X8D400 X8D80D
           TOTAL NUMBER:      21

 CKZ02091I VALIDATING KEYWORD: USERCATALOGS-DDN
```

```
 CKZ02085I DSNS FOR KEYWORD: USERCATALOGS-DDN
           UCAT.DBOCD                                    UCAT.DBODD
           UCAT.DBOCL                                    UCAT.DBODL


 CKZ03501I 20.55.47 CHECK USERCATALOGS STARTED - PROGRAM REV=6

CKZ02089I TARGET VOLUMES WILL NOT BE CHECKED FOR EMPTY

CKZ02048I OPTIONS IN EFFECT FOR THIS EXECUTION:
           CONCURRENT_EXECUTIONS:           N


CKZ02101I 20.55.47 VOLUME COPY STARTED - PROGRAM REV=55
CKZ02131I PRE-COPIED VOLUME PAIRS ACCEPTED:
           SOURCE  TARGET
           SBOXJI  X76724
           SBOXJJ  X6642A
.................................................
CKZ02161I SOURCE USERCATALOG WILL BE READ FROM TARGET VOLUME; VOLSER=X66725
BCS=UCAT.DBOCD
 CKZ02161I SOURCE USERCATALOG WILL BE READ FROM TARGET VOLUME; VOLSER=X8D400
BCS=UCAT.DBOCL
 CKZ02101I 20.55.47 VOLUME COPY COMPLETED; RETURN CODE=0

 CKZ02201I 20.55.47 BCS BACKUP STARTED - PROGRAM REV=31
 CKZ02241I BCS=UCAT.DBOCD WILL BE BACKED UP TO
DSN=DB2TOOLS.CLONE.SLB3.UCATBKUP.BKP00001
 CKZ02241I BCS=UCAT.DBOCL WILL BE BACKED UP TO
DSN=DB2TOOLS.CLONE.SLB3.UCATBKUP.BKP00002
 CKZ02230I BCS BACKUP TASK COMPLETED; RETURN CODE=0   SYSOUT DD=BKP00001
 CKZ02230I BCS BACKUP TASK COMPLETED; RETURN CODE=0   SYSOUT DD=BKP00002
 CKZ02201I 20.55.49 BCS BACKUP COMPLETED; RETURN CODE=0

 CKZ02001I 20.55.49 COPY PROCESS COMPLETED; RETURN CODE=0
.................................
```

ST04 COPY DATA-MOVER(PGM(ADRDSSU))

The second COPY command, which is shown in Example 9-13, performs these tasks:

- Copies the backup volumes (set B) that are contained in the DD that is pointed to by the FROM-VOLSER-DDN(FRVOLSER) to the target volumes (set C) that are pointed to by the TO-VOLSER.
- Volume sets (set B) and (set C) volume pairs are added to a new VOLPLIST data set, HLQ?.NUC.WRK.VOLPLIST.
- The source ICF catalogs are *not* backed up again.
- A journal data set is created in this step but it will *not* be used again. The journal from the first copy will be used.

This step copies the data to the target volumes.

Example 9-13 ST04 COPY DATA-MOVER(PGM(ADRDSSU) output

```
COPY                                    -
      DATA-MOVER(PGM(ADRDSSU)
```

```
       CONSISTENT(NO) -
       COPYCMDLIMIT(24) -
       FASTREP(REQ) -
       INCREMENTAL(NO) -
                ) -
      FROM-VOLSER-DDN(FRVOLSER) -
      TO-STORAGEGROUP( DBODDATA DBODLOG1 DBODLOG2)-
      NOUSERCATALOGS -
      JOURNAL-DDN(JOURNAL)

 CKZO2001I 20.56.45 COPY PROCESS STARTED - PROGRAM REV=66

 CKZO2030I DSS LEVEL=X'03011300'
 CKZO2043I ANTRQST LEVEL=12; ESSRVCS LEVEL=106

 CKZO2086I STORAGE GROUPS/MASKS FOR KEYWORD: TO-STORAGEGROUP
           DBODDATA DBODLOG1 DBODLOG2

 CKZO2087I     21 VOLUMES DERIVED  FOR KEYWORD: TO-STORAGEGROUP
           XV6136 XV633F XV643D XV643F XV6435 XV6439 XV673A X3652E X4612F X4622F X4632F X4642F X4652F X5DE61 X5DE62
X5D842 X5D843 X5D844 X5D90C X76624 X9602F

 CKZO2087I     21 VOLUMES RESOLVED FOR KEYWORD: TO-STORAGEGROUP

 CKZO2091I VALIDATING KEYWORD: FROM-VOLSER-DDN
 CKZO2087I     21 VOLUMES OR MASKS FOR KEYWORD: FROM-VOLSER-DDN
           SBOXKP SBOXKQ X5DA0D X66228 X66229 X6642A X66425 X66426 X66427 X66525 X66528 X66529 X6672A X66725 X66726
           X66728 X66729 X76724 X8D002 X8D400 X8D80D

 CKZO2087I     21 VOLUMES RESOLVED FOR KEYWORD: FROM-VOLSER-DDN

 CKZO2088I VOLUME SERIAL(S) TO BE USED FOR SOURCE
           SBOXKP SBOXKQ X5DA0D X66228 X66229 X6642A X66425 X66426 X66427 X66525 X66528 X66529 X6672A X66725 X66726
           X66728 X66729 X76724 X8D002 X8D400 X8D80D
           TOTAL NUMBER:      21

 CKZO2088I VOLUME SERIAL(S) TO BE USED FOR TARGET
           XV6136 XV633F XV643D XV643F XV6435 XV6439 XV673A X3652E X4612F X4622F X4632F X4642F X4652F X5DE61 X5DE62
           X5D842 X5D843 X5D844 X5D90C X76624 X9602F
           TOTAL NUMBER:      21

 CKZO2089I TARGET VOLUMES WILL NOT BE CHECKED FOR EMPTY
CKZO2101I 20.56.46 VOLUME COPY STARTED - PROGRAM REV=55
 CKZO2130I ADRDSSU TASK COMPLETED; RETURN CODE=0  SYSOUT DD=CPY00001
 CKZO2101I 20.56.47 VOLUME COPY COMPLETED; RETURN CODE=0

 CKZO2001I 20.56.47 COPY PROCESS COMPLETED; RETURN CODE=0

 CKZO1009I DB2 CLONING TOOL EXECUTION COMPLETE. HIGHEST RETURN CODE WAS 0

 CKZO3001I 20.56.46 COPY STARTED - PROGRAM REV=30
 CKZO3003I DDNAME=SYS00003 ALLOCATED FOR DSN=**TEMPORARY DSSIN DSN
 CKZO3003I DDNAME=SYS00004 ALLOCATED FOR DSN=**TEMPORARY DSSOUT DSN
  PAGE 0001     5695-DF175  DFSMSDSS V1R13.0 DATA SET SERVICES     2012.275 20:56
   PARALLEL
  ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'PARALLEL '
   COPY FULL INDYNAM(SBOXKP) OUTDYNAM(X5DE61) ALLDATA(*) ALLEXCP       -
        ADMINISTRATOR FASTREP(REQ)                                     -
        DUMPCONDITIONING PURGE            DEBUG(FRMSG(DETAILED))
  ADR101I (R/I)-RI01 (01), TASKID 002 HAS BEEN ASSIGNED TO COMMAND 'COPY '
   COPY FULL INDYNAM(SBOXKQ) OUTDYNAM(X5DE62) ALLDATA(*) ALLEXCP       -
        ADMINISTRATOR FASTREP(REQ)                                     -
        DUMPCONDITIONING PURGE            DEBUG(FRMSG(DETAILED))
.....................................................
ADR101I (R/I)-RI01 (01), TASKID 022 HAS BEEN ASSIGNED TO COMMAND 'COPY '
  ADR109I (R/I)-RI01 (01), 2012.275 20:56:46 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED
```

```
ADR014I (SCH)-DSSU (02), 2012.275 20:56:46 ALL PREVIOUSLY SCHEDULED TASKS COMPLETED. PARALLEL MODE NOW IN EFFECT
.....................................
ADR006I (002)-STEND(01), 2012.275 20:56:46 EXECUTION BEGINS
ADR241I (002)-DDTFP(01), TARGET VTOC BEGINNING AT 000000003:00 AND ENDING AT 000000008:14 IS OVERLAID
ADR806I (002)-TOMI (02), VOLUME SBOXKP WAS COPIED USING A FAST REPLICATION FUNCTION
ADR006I (002)-STEND(02), 2012.275 20:56:46 EXECUTION ENDS
PAGE 0003     5695-DF175  DFSMSDSS V1R13.0 DATA SET SERVICES     2012.275 20:56
ADR050I (011)-PRIME(01), DFSMSDSS INVOKED VIA APPLICATION INTERFACE
ADR016I (011)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
ADR013I (002)-CLTSK(01), 2012.275 20:56:46 TASK COMPLETED WITH RETURN CODE 0000
ADR006I (003)-STEND(01), 2012.275 20:56:46 EXECUTION BEGINS
ADR050I (012)-PRIME(01), DFSMSDSS INVOKED VIA APPLICATION INTERFACE
ADR016I (012)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
.....................................................
```

Note: The message "CKZ02089I TARGET VOLUMES WILL NOT BE CHECKED FOR EMPTY" appears because we chose the option in the setup to ignore data that exists on the target volumes and therefore overwrite the volumes.

ST05 COPYCHECK

COPYCHECK provides a mechanism to either WAIT for copies to complete, or to WITHDRAW or STOPSNAP (terminate) previously established volume relationships.

WAIT is used to keep checking for completion of the copies. The WAIT time that we set is 10 minutes but this can be increased. After 10 minutes, if the copies are still in progress, the job will finish with a return code of 4 based on the parameters that we set.

The option WITHDRAW can be used to stop the copies when FlashCopy is used, and the option STOPSNAP is used to stop EMC SNAP. This allows for rerunning the process or canceling it altogether. For more details about COPYCHECK and WITHDRAW, see "COPYCHECK" on page 410.

Example 9-14 ST05 COPYCHECK

```
CKZ01020I PROGRAM: CKZ01CIO 20110309 10.44 VERS=1.0 REV=35

   COPYCHECK           -
     WAIT(10,RC(4)) -
     JOURNAL-DDN(JOURNAL)

CKZ05501I 20.56.55 VOLUME CHECK STARTED - PROGRAM REV=31
CKZ05586I VALIDATING KEYWORD: WAIT

CKZ05601I 20.56.55 VOLUME STATUS STARTED - PROGRAM REV=34
CKZ05643I ANTRQST LEVEL=12; ESSRVCS LEVEL=106
CKZ05651I 20.56.55 VOLSER PAIR: SBOXKP/X5DE61 COPY STILL IN PROGRESS,  39% COMPLETED
CKZ05651I 20.56.55 VOLSER PAIR: SBOXKQ/X5DE62 COPY STILL IN PROGRESS,  13% COMPLETED
CKZ05651I 20.56.55 VOLSER PAIR: X5DA0D/X5D842 COPY STILL IN PROGRESS,  16% COMPLETED
CKZ05651I 20.56.55 VOLSER PAIR: X66228/XV6136 COPY STILL IN PROGRESS,  59% COMPLETED
CKZ05651I 20.56.55 VOLSER PAIR: X66229/XV633F COPY STILL IN PROGRESS,  21% COMPLETED
CKZ05651I 20.56.55 VOLSER PAIR: X66425/XV643F COPY STILL IN PROGRESS,  99% COMPLETED
CKZ05651I 20.56.55 VOLSER PAIR: X66427/XV6439 COPY STILL IN PROGRESS,  99% COMPLETED
CKZ05651I 20.56.55 VOLSER PAIR: X66525/XV673A COPY STILL IN PROGRESS,  45% COMPLETED
.....................................
.....................................
CKZ05641W TIME LIMIT EXCEEDED

CKZ05630I VOLUME PAIRS STATUS
          SBOXKP/X5DE61 BOTH VOLUME SERIALS ARE AVAILABLE
          SBOXKQ/X5DE62 BOTH VOLUME SERIALS ARE AVAILABLE
```

```
          X5DA0D/X5D842 BOTH VOLUME SERIALS ARE AVAILABLE
          X66228/XV6136 BOTH VOLUME SERIALS ARE AVAILABLE
          X66229/XV633F COPY STILL IN PROGRESS,  44% COMPLETED
          X6642A/XV643D BOTH VOLUME SERIALS ARE AVAILABLE
          X66425/XV643F BOTH VOLUME SERIALS ARE AVAILABLE
          X66426/XV6435 BOTH VOLUME SERIALS ARE AVAILABLE
          X66427/XV6439 BOTH VOLUME SERIALS ARE AVAILABLE
          X66525/XV673A COPY STILL IN PROGRESS,  81% COMPLETED
          X66528/X3652E COPY STILL IN PROGRESS,  44% COMPLETED
          X66529/X4612F COPY STILL IN PROGRESS,  55% COMPLETED
          X6672A/X4622F COPY STILL IN PROGRESS,  80% COMPLETED
          X66725/X4632F COPY STILL IN PROGRESS,  59% COMPLETED
          X66726/X4642F COPY STILL IN PROGRESS,  67% COMPLETED
          X66728/X4652F COPY STILL IN PROGRESS,  56% COMPLETED
          X66729/X76624 COPY STILL IN PROGRESS,  73% COMPLETED
          X76724/X9602F BOTH VOLUME SERIALS ARE AVAILABLE
          X8D002/X5D843 BOTH VOLUME SERIALS ARE AVAILABLE
          X8D400/X5D844 BOTH VOLUME SERIALS ARE AVAILABLE
          X8D80D/X5D90C BOTH VOLUME SERIALS ARE AVAILABLE


CKZ05601I 21.06.59 VOLUME STATUS COMPLETED; RETURN CODE=4

CKZ05501I 21.06.59 VOLUME CHECK COMPLETED; RETURN CODE=4

CKZ01009I DB2 CLONING TOOL EXECUTION COMPLETE. HIGHEST RETURN CODE WAS 4
```

Note that the COPYCHECK finished with a return code of 4. You can see from the example that the COPY is still in progress. You can submit the job again and extend the wait time until all copies complete. See Example 9-15.

A return code of 0 indicates that the copy is complete.

Example 9-15 COPYCHECK completed

```
X8D002/X5D843 BOTH VOLUME SERIALS ARE AVAILABLE
          X8D400/X5D844 BOTH VOLUME SERIALS ARE AVAILABLE
          X8D80D/X5D90C BOTH VOLUME SERIALS ARE AVAILABLE


CKZ05601I 21.30.24 VOLUME STATUS COMPLETED; RETURN CODE=0

CKZ05501I 21.30.24 VOLUME CHECK COMPLETED; RETURN CODE=0

CKZ01009I DB2 CLONING TOOL EXECUTION COMPLETE. HIGHEST RETURN CODE WAS 0
```

ST06 CKZRNTGT

This step runs the CKZRNTGT program with the VOLPLIST data sets from previous steps to build a NEWTGT data set. See Example 9-16.

The two VOLPLIST data sets that were created in each of the COPY commands contain volume set A to set B and volume set B to set C. These are read to create a data set to which DD NEWTGT points. This data set contains the pairing set A to set B to set C.

Example 9-16 ST06 CKZRNTRGT

```
   DEL DB2TOOLS.CLONE.SLB3.NEWTGT
IDC0550I ENTRY (A) DB2TOOLS.CLONE.SLB3.NEWTGT DELETED
```

```
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

   SET MAXCC=0

IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
CKZRX103I CKZRNTGT CKZIN HEADER IS V002 CKZ    2012/10/01 20:55:47
CKZRX104I CKZRNTGT CKZIN FILE HAS 21 VOLUME PAIRS

CKZRX103I CKZRNTGT NUCIN HEADER IS V002 CKZNUC 2012/10/01 20:56:46
CKZRX104I CKZRNTGT NUCIN FILE HAS 21 VOLUME PAIRS
```

NEWTGT contents

```
VIEW       DB2TOOLS.CLONE.SLB2.NEWTGT
Command ===>
****** **************************** T
000001      SBOXJI X76724 X9602F
000002      SBOXJJ X6642A XV643D
000003      SBOXJK X5DA0D X5D842
.............................
```

ST07 VOLOPTIONS UPDATE

This step (Example 9-17) reads the data set to which DD NEWTGT points and updates the journal that was created in the first COPY with the volumes from set A to set C, so that the data sets can be renamed and DB2 can be conditioned correctly.

Example 9-17 ST07 VOLOPTIONS UPDATE

```
   VOLOPTIONS UPDATE -
     NEWTARGETS-DDN(NEWTGT) -
     JOURNAL-DDN(JOURNAL)

CKZ42001I 21.36.58 VOLOPTIONS STARTED - PROGRAM REV=27
CKZ42031I VOLUMES FOR NEW TARGETS:
          SBOXJI X76724 X9602F
          SBOXJJ X6642A XV643D
          SBOXJK X5DA0D X5D842
          SBOXJL SBOXKP X5DE61
          SBOXJM X8D002 X5D843
          SBOXJN SBOXKQ X5DE62
          SBOXJQ X8D400 X5D844
          SBOXJR X8D80D X5D90C
.......................................
CKZ42020I CURRENT JOURNAL VOLUME PAIRS:
          SBOXJI X76724        6724
          SBOXJJ X6642A        642A
          SBOXJK X5DA0D        DA0D
          SBOXJL SBOXKP        D319
          SBOXJM X8D002        D002
          SBOXJN SBOXKQ        D31F
.........................................
          X76029 X66427        6427

CKZ42021I UPDATED USERCATALOGS VOLUME:
          UCAT.DB0CL                                  X5D844
          UCAT.DB0CD                                  X4632F

CKZ42020I UPDATED JOURNAL VOLUME PAIRS:
          SBOXJI X9602F        602F
          SBOXJJ XV643D        643D
```

```
          SBOXJK X5D842        D842
          SBOXJL X5DE61        DE61
          SBOXJM X5D843        D843
          SBOXJN X5DE62        DE62
          SBOXJQ X5D844        D844
.........................................
CKZ42001I 21.36.58 VOLOPTIONS COMPLETED; RETURN CODE=0
```

ST08 RENAME

This step renames and catalogs target volume data sets. The SAFE option, a keyword of the RENAME command, is recommended to enable the rerun of the RENAME step after correcting any issues.

There are many options that will affect the behavior of this step. In the setup, we preferred the use of SAFE because it provided the ability to RERUN the step without having to perform the copy again.

An alternative to only running in SAFE mode is to use the SIMULATE mode where nothing is actually renamed but checking is done by simulating and reporting potential errors. You then resolve the issues and then run with SAFE mode.

In our scenario, we encountered data sets, which are on the source and which were copied to the target, that are not required. We had not set up the appropriate rename mask.

Example 9-18 is the first run made by using the SAFE option with snapshots of the errors in the following examples.

Example 9-18 Rename Job parameters

```
RENAME                                                -
  SAFE -
  VOLBKUP-DDN(VOLBKUP) -
  GDG-ALL-MIGRATED(SKIP) -
  GDG-EMPTY(SKIP) -
  GDG-MIGRATED(ERROR) -
  GDG-TAPE(ERROR) -
  ISSUE-VCLOSE(YES,LOCAL) -
  MAX-TASKS(5) -
  MISSINGUCAT(KEEP,RC(4)) -
  NOTRENAMED(KEEP,RC(8)) -
  ORPHANCATENTRY(KEEP,RC(8)) -
  RECATALOG(Y) -
  RENAME-AUDIT-LOG(NO) -
  RENAME-ERROR(ABORT) -
  RENAME-LIST(N) -
  TEMPDSN(DELETE,RC(4)) -
  UPDATE-IAM-ASSOCIATIONS(N) -
VALIDATE-SMS-CLASSES(Y) -
STORCLAS-PAIRS(DBOATEMP,DBODDATA -
                   DBOCDATA,DBODDATA -
                   DBOADATA,DBODDATA -
                   DBOALOG1,DBODLOG1 -
                   DBOALOG2,DBODLOG2) -
    RENAME-MASKS(DBOCD.** DBODD.** -
                 DBOC%.** DBOD%.**) -
    EXCLUDE-SRCNAME(RC(NOTRENAMED-RC)) -
    EXCLUDE-SRCNAME-MASKS(DBOAT.** -
                          DBOAM.**) -
```

```
     JOURNAL-DDN(JOURNAL)

CKZ10001I 10.57.54 RENAME PROCESS STARTED - PROGRAM REV=52
CKZ10085I DSNS FOR KEYWORD: RENAME-MASKS
          DBOC%.**                                    DBOD%.**
          DBOCD.**                                    DBODD.**
```

We can see that the masks for the rename were kept fairly simple but the ability is there for some quite complex masks. For more information, see the manuals and PF1 Help on the Rename Masks ISPF panel.

Note the STORCLAS pairing, which will update the ICF catalog for matched data sets with SMS details.

In Example 9-19, we see the job failing with a return code 8 as requested by the parameter NOTRENAMED(KEEP,RC(8)).

Example 9-19 Rename SAFE mode

```
CKZ11032I VOLUME BACKUP STARTED FOR X5DE62
CKZ11032I VOLUME BACKUP COMPLETED FOR X5DE62
CKZ11030I VOLUME CONVERSION STARTED FOR SBOXJN/X5DE62
CKZ11030I VOLUME CONVERSION COMPLETED FOR SBOXJJ/XV643D; RETURN CODE=0  DATA SET=3

CKZ11032I VOLUME BACKUP STARTED FOR X5D844
CKZ11032I VOLUME BACKUP COMPLETED FOR X5D844
CKZ11030I VOLUME CONVERSION STARTED FOR SBOXJQ/X5D844
CKZ11030I VOLUME CONVERSION FAILED    FOR SBOXJQ/X5D844; RETURN CODE=8  DATA SET=9
CKZ11330E RETURN CODE 8 SET FOR NOT RENAMED DATA SET(S)
CKZ11130E RETURN CODE 8 SET FOR NOT RENAMED DATA SET(S)
CKZ11030I VOLUME CONVERSION FAILED    FOR SBOXJK/X5D842; RETURN CODE=8  DATA SET=889
CKZ11330E RETURN CODE 8 SET FOR NOT RENAMED DATA SET(S)
```

Example 9-20 shows the rename errors.

Example 9-20 Rename errors

```
CKZ11301I 21.38.17 VVDS UPDATE STARTED FOR VOLUME: X5DE62 - PROGRAM REV=49
 CKZ11303I DDNAME=VDX5DE62 ALLOCATED FOR DSN=SYS1.VVDS.VX5DE62
 CKZ11312W DATA SET MATCHES NO RENAME MASK: ADMRX.IC.GLW.GLWSEMP.P00001FC.D
 CKZ11341W RENAMES NOT DONE FOR VVDS ENTRY - COMPONENT NAME=ADMRX.IC.GLW.GLWSEMP.P00001FC.D
.........................................
 CKZ11342W USER CATALOG NOT IN CATALOG LIST - COMPONENT NAME=DBOCI.DSN8D10A.DSN8S10D.N00001.C6K43QFY.D
 CKZ11342W BCS=UCAT.DBOCI
 CKZ11342W USER CATALOG NOT IN CATALOG LIST - COMPONENT NAME=DBOCI.DSN8D10A.DSN8S10E.N00002.C6K43RDM.D
 CKZ11342W BCS=UCAT.DBOCI
 CKZ11342W USER CATALOG NOT IN CATALOG LIST - COMPONENT NAME=DBOCI.DSN8D10A.DSN8S10P.N00001.C6K43R6D.D
 CKZ11342W BCS=UCAT.DBOCI
 CKZ11342W USER CATALOG NOT IN CATALOG LIST - COMPONENT NAME=DBOCI.DSN8D10P.DSN8S10C.N00001.C6K43PHA.D
 CKZ11342W BCS=UCAT.DBOCI
.................................
 CKZ11312W DATA SET MATCHES NO RENAME MASK: ADMRX.IC.GLW.GLWSSPL.P00003FC.D
 CKZ11341W RENAMES NOT DONE FOR VVDS ENTRY - COMPONENT NAME=ADMRX.IC.GLW.GLWSSPL.P00003FC.D
 CKZ11342W USER CATALOG NOT IN CATALOG LIST - COMPONENT NAME=DBOCI.DSN8D10A.DSN8S10R.N00001.C6K43812.D
 CKZ11342W BCS=UCAT.DBOCI
 CKZ11342W USER CATALOG NOT IN CATALOG LIST - COMPONENT NAME=DBOCI.DB2PM.ACCS.N00001.C7EQCAMH.D
 CKZ11342W BCS=UCAT.DBOCI
 CKZ11312W DATA SET MATCHES NO RENAME MASK: ADMRX.IC.GLW.TESTTS6.P00002FC.D
 CKZ11341W RENAMES NOT DONE FOR VVDS ENTRY - COMPONENT NAME=ADMRX.IC.GLW.TESTTS6.P00002FC.D
CKZ11342W USER CATALOG NOT IN CATALOG LIST - COMPONENT NAME=DBOCI.DB2PMAUD.AUDIT.N00001.DCF2MBHY.D
```

```
 CKZ11342W BCS=UCAT.DBOCI
 CKZ11342W USER CATALOG NOT IN CATALOG LIST - COMPONENT NAME=DBOCI.DB2PMAUD.AUDIT.NOOOO1.DCF2MDHW.D
 CKZ11342W BCS=UCAT.DBOCI
 CKZ11330E RETURN CODE 8 SET FOR NOT RENAMED DATA SET(S)
 CKZ11301I 21.38.18 VVDS UPDATE COMPLETED; RETURN CODE=8
 CKZ11130E RETURN CODE 8 SET FOR NOT RENAMED DATA SET(S)

 CKZ11101I 21.38.18 VOLUME UPDATE COMPLETED; RETURN CODE=8   F1DSCB COUNT=858

.........................

CKZ11335I SMS STORCLAS COPIED FROM SOURCE FOR VVDS ENTRY - DBOCL.LOGCOPY1.DSO1.DATA
 CKZ11335I SMS STORCLAS COPIED FROM SOURCE FOR VVDS ENTRY - DBOCL.LOGCOPY1.DSO3.DATA
 CKZ11335I SMS STORCLAS COPIED FROM SOURCE FOR VVDS ENTRY - DBOCL.LOGCOPY1.DSO2.DATA
 CKZ11312W DATA SET MATCHES NO RENAME MASK: UCAT.DBOCL
 CKZ11341W RENAMES NOT DONE FOR VVDS ENTRY - COMPONENT NAME=UCAT.DBOCL
 CKZ11330E RETURN CODE 8 SET FOR NOT RENAMED DATA SET(S)
 CKZ11301I 21.38.16 VVDS UPDATE COMPLETED; RETURN CODE=8
 CKZ11130E RETURN CODE 8 SET FOR NOT RENAMED DATA SET(S)
```

In this case, there are rename errors with image copy and ad hoc data sets coming from the source volume. So, the error messages concerning the copy and ad hoc data sets are correct and are not required.

There are three types of errors or warnings in Example 9-20 on page 451:

- `SMS STORCLAS COPIED FROM SOURCE FOR VVDS ENTRY`

  This message indicates that because there is not a pairing for the STORCLAS for the specified entry, the source value is used. For more details, see Figure 9-44 on page 439. This message can generally be bypassed.

- `RENAMES NOT DONE FOR VVDS ENTRY`

  This message occurs when there is no rename mask and the HLQ can exist in a target ICF catalog.

- `USER CATALOG NOT IN CATALOG LIST`

  This message occurs when the data set name matches a rename mask, but the catalog back-pointer is not one of the “source” catalogs specified in the corresponding COPY command. That is, this data set was not in the ICF catalogs associated with the BACKUP taken on the source system.

  In the second case, there were image copy data sets residing in error on the source data volumes.

It might also be the case that the archive logs were on the source volumes and not renamed. In general, it is possible however to rename archive logs on DASD and in the BSDS with the DB2 Cloning Tool. You can also remove the archives entries; see 9.5.4, “Post-cloning steps” on page 463.

To resolve these issues and allow the rename to continue, we can perform these tasks:

- Add additional rename masks to the step.
- Include an EXCLUDE-SRCNAME-MASKS and an EXCLUDE-SRCNAME(RC(0)) to ignore the data sets.
- Allow the data sets to remain “as is” or uncataloged without failing the step by modifying NOTRENAMED(KEEP,RC(8)) to NOTRENAMED(KEEP,RC(4)).

- Remove the data sets by modifying the NOTRENAMED(KEEP,RC(8)) to NOTRENAMED(DELETE,RC(8)) and MISSINGUCAT(KEEP,RC(4)) to MISSINGUCAT(DELETE,RC(4)).

In all cases, this step can only be rerun if either the SIMULATE or SAFE option was used in the first run; otherwise, we must perform the COPY step again.

After reviewing the errors, we decided to use the option to delete and return code 4.

To RERUN, we follow the instructions in the generated job and remove the delete step and change the DISP of the BCSRECS and VOLBKUP data sets to SHR. See Example 9-21.

We can rerun this step because we specified the SAFE option for RENAME.

Example 9-21 RENAME RERUN JCL

```
//BCSRECS  DD  DSN=DB2TOOLS.CLONE.SLB2.BCSRECS,
//             UNIT=SYSALLDA,DISP=OLD
//VOLBKUP  DD  DSN=DB2TOOLS.CLONE.SLB2.VOLBKUP,
//             UNIT=SYSALLDA,DISP=OLD
//CKZIN    DD  *
RERUN -
     VOLBKUP-DDN(VOLBKUP) -
     GDG-ALL-MIGRATED(SKIP) -
     GDG-EMPTY(SKIP) -
     GDG-MIGRATED(ERROR) -
     GDG-TAPE(ERROR) -
     ISSUE-VCLOSE(YES,LOCAL) -
     MAX-TASKS(5) -
     MISSINGUCAT(DELETE,RC(4)) -
     NOTRENAMED(DELETE,RC(4)) -
     ORPHANCATENTRY(KEEP,RC(8)) -
     RECATALOG(Y) -
     RENAME-AUDIT-LOG(NO) -
     RENAME-ERROR(ABORT) -
     RENAME-LIST(N) -
     TEMPDSN(DELETE,RC(4)) -
     UPDATE-IAM-ASSOCIATIONS(N) -
     VALIDATE-SMS-CLASSES(Y) -
     STORCLAS-PAIRS(DBOATEMP,DBODDATA -
                    DBOCDATA,DBODDATA -
                    DBOADATA,DBODDATA -
                    DBOALOG1,DBODLOG1 -
                    DBOALOG2,DBODLOG2) -
     RENAME-MASKS(DBOCD.** DBODD.** -
                  DBOC%.** DBOD%.**) -
     EXCLUDE-SRCNAME(RC(NOTRENAMED-RC)) -
     EXCLUDE-SRCNAME-MASKS(DBOAT.** -
                           DBOAM.**) -
     JOURNAL-DDN(JOURNAL)
//*
```

The resulting job output will complete now with a return code of 4 and clean up the orphaned data sets. See Example 9-22.

Example 9-22 RENAME successful

```
CKZ11312W DATA SET MATCHES NO RENAME MASK: ADMRX.IC.GLW.TESTTS6.P00001FC.D
..................
CKZ11342W USER CATALOG NOT IN CATALOG LIST - COMPONENT NAME=DBOCI.GLWOA.GLWSSPL.
```

```
CKZ11342W BCS=UCAT.DBOCI
CKZ11301I 21.52.49 VVDS UPDATE COMPLETED; RETURN CODE=4

CKZ11101I 21.52.49 VOLUME UPDATE COMPLETED; RETURN CODE=4   F1DSCB COUNT=858
```

ST09 DB2UPDATE

The DB2UPDATE command makes the necessary DB2 changes to reflect the renamed data sets.

DB2UPDATE updates the DB2 directory and the DB2 BSDSs:

- DB2 directory updates: The VCATNAME and, optionally, the DB2 storage group names are updated.
- BSDS updates: The DB2 catalog name, distributed data facility (DDF) location details, and the “active” log data set names are updated.
- Optionally, the ARCHIVE data set names and volume serial numbers in the BSDS can be updated.

This step passes information to the DB2SQL step, which updates the DB2 catalog. See Example 9-23.

Example 9-23 ST09 DB2UPDATE

```
DB2UPDATE        -
        DB2-NAME(SOO1) -
        DB2-HLQS(DBOCD, DBODD) -
          DDF -
             LOCATION(DBOD) -
             LUNAME(SCPDBOD) -
             PORT(38390) -
             RESPORT(38391) -
             SECPORT(38392) -
           ) -
        DDF-NOT-UPDATED(RC(4)) -
        HLQ-NOT-UPDATED(RC(4)) -
        JOURNAL-DDN(JOURNAL)

CKZ22001I 21.56.38 DB2UPDATE STARTED - PROGRAM REV=39
CKZ22085I PAIRS FOR KEYWORD: DB2-HLQS
          DBOCD    DBODD

CKZ22086I VALIDATING KEYWORD: DDF
CKZ22086I VALIDATING KEYWORD: DDF-NOT-UPDATED
CKZ22086I VALIDATING KEYWORD: HLQ-NOT-UPDATED
CKZ22030I OPTIONS IN EFFECT FOR THIS EXECUTION:
          DB2-NAME:                 SOO1
          DB2-XCFCLEAN:             Y
          DDF-NOT-UPDATED:          RC(4)
          HLQ-NOT-UPDATED:          RC(4)
          SIMULATION:               N

CKZ22101I 21.56.38 BSDS UPDATING STARTED - PROGRAM REV=45
CKZ22130I PROCESSING BSDSO1

CKZ22130I PROCESSING BSDSO2

CKZ22101I 21.56.39 BSDS UPDATING COMPLETED; RETURN CODE=0
```

```
CKZ22201I 21.56.39 LINEAR FILE UPDATING STARTED - PROGRAM REV=45
CKZ22230I PROCESSING DDNAME: DBD01
CKZ22231I THE FILE'S ENDING RBA IS: 000023592960 X'00000000_01680000'
CKZ22232I NO DBD INFORMATION FOUND
CKZ22242I DBD01    CHANGED RECORDS: 0        CHANGED FIELDS: 0

CKZ22201I 21.56.39 LINEAR FILE UPDATING COMPLETED; RETURN CODE=0

CKZ22035I NO DBD INFORMATION WAS FOUND IN DBD01; WILL DYNAMICALLY ALLOCATE SYSDBXA TO USE
CKZ22003I DDNAME=SYSDBDXA ALLOCATED FOR DSN=DBODD.DSNDBC.DSNDB01.SYSDBDXA.I0001.A001

CKZ22401I 21.56.39 DIRECTORY UPDATING STARTED - PROGRAM REV=1
CKZ22401I 21.56.39 DIRECTORY UPDATING STARTED - PROGRAM REV=1
CKZ22430I PROCESSING DDNAME: SYSDBDXA
CKZ22431I THE FILE'S ENDING RBA IS: X'00000000_01D34000'
CKZ22442I SYSDBDXA CHANGED RECORDS: 704          CHANGED FIELDS: 27,334

CKZ22401I 21.56.45 DIRECTORY UPDATING COMPLETED; RETURN CODE=0

CKZ22001I 21.56.45 DB2UPDATE COMPLETED; RETURN CODE=0

CKZ01009I DB2 CLONING TOOL EXECUTION COMPLETE. HIGHEST RETURN CODE WAS 0
```

To verify the changes and to confirm that the active logs are renamed, run the DSNJU004 utility. See Example 9-24.

We have not cleaned up the archives from the BSDS. To do this, see “BSDS cleanup to remove source archives” on page 463.

Example 9-24 DSNJU004 extract

```
********************************************************************************* *
*
* LOG MAP OF THE BSDS DATA SET BELONGING TO MEMBER 'NO NAME ' OF GROUP ‘NO NAME ‘. *
*                                                                                *
**********************************************************************************
DSNJCNVB CONVERSION PROGRAM HAS RUN   DDNAME=SYSUT1
   LOG MAP OF BSDS DATA SET COPY 1, DSN=DB0DB.BSDS01
   LTIME INDICATES LOCAL TIME, ALL OTHER TIMES ARE GMT.
         DATA SHARING MODE IS OFF
         SYSTEM TIMESTAMP   - DATE=2012.275  LTIME=22:00:42.84
         UTILITY TIMESTAMP  - DATE=2012.127  LTIME=14:23:32.52
         VSAM CATALOG NAME=DBODD
         HIGHEST RBA WRITTEN       0018C16E7A56  0000.000  00:00:00.0
         HIGHEST RBA OFFLOADED     0018BC437FFF
         RBA WHEN CONVERTED TO V4  000000000000
       THIS BSDS HAS MEMBER RECORDS FOR THE FOLLOWING MEMBERS:
         HOST MEMBER NAME:
           MEMBER ID:                0
           GROUP NAME:
           BSDS COPY 1 DATA SET NAME:
           BSDS COPY 2 DATA SET NAME:
           ENFM START RBA/LRSN:      000000000000
                 **** DISTRIBUTED DATA FACILITY ****
                        COMMUNICATION RECORD
                      07:27:45 OCTOBER 02, 2012
LOCATION=DBOD IPNAME=(NULL) PORT=38390 SPORT=38392 RPORT=38391
ALIAS=(NULL)
IPV4=NULL IPV6=NULL
```

```
GRPIPV4=NULL GRPIPV6=NULL
LUNAME=SCPDBOD PASSWORD=(NULL) GENERICLU=(NULL)

ACTIVE LOG COPY 1 DATA SETS
    START RBA/TIME         END RBA/TIME        DATE    LTIME DATA SET INFORMA
 --------------------  --------------------  --------  ----- ----------------
   0018B5F9D000           0018B5FABFFF       2012.127  13:56 DSN=DBODL.LOGCOPY1.DS01
 2012.127  18:03:38.1  2012.127  18:17:29.3  PASSWORD=(NULL) STATUS=TRUNCATED, REUSABLE
   0018B5FAC000           0018BC437FFF       2012.127  14:15 DSN=DBODL.LOGCOPY1.DS03
 2012.127  18:17:29.3  2012.129  18:59:35.7  PASSWORD=(NULL) STATUS=REUSABLE
   0018BC438000           0018C28C3FFF       2012.127  14:23 DSN=DBODL.LOGCOPY1.DS02
 2012.129  18:59:35.7  ........  ..........  PASSWORD=(NULL) STATUS=NOTREUSABLE
ARCHIVE LOG COPY 1 DATA SETS
    START RBA/TIME         END RBA/TIME        DATE    LTIME DATA SET INFORMATION
 --------------------  --------------------  --------  ----- ----------------
   000000000000           0000021BFFFF       2011.037  19:20 DSN=DBOAA.ARCHLOG1.A0000001
 2011.038  00:09:36.3  2011.038  00:20:51.0                  PASSWORD=(NULL) VOL=SBOX8B ...
                                                            CATALOGUED
```

ST11DB0D DB2START special DSNZPARM

This step starts the target DB2s in maintenance mode with changed DSNZPARMs (DEFER YES & SPRMCTU SETC 'I').

This job starts DB2 on the target and will complete when DB2 is up within the WAIT time. If DB2 is already up, the job completes with a return code of 8. See Example 9-25.

Example 9-25 ST11 DB2START SPECIAL

```
      DB2START                    -
        DB2-SSID(DBOD)       -
        SPECIAL -
        DB2-ALREADY-RUNNING(RC(8)) -
        DSNZPARM(DSNZSPEC) -
        WAIT(5,RC(8))
  ....................................................
CKZ25570I STARTING DB2 SUBSYSTEM: DBOD
CKZ25540I START COMMAND: -DBOD START DB2 PARM(DSNZSPEC) ACCESS(MAINT)
CKZ25538I 21.56.57 WAITING FOR DB2 TO START
CKZ25537I 21.57.05 DB2 HAS STARTED

CKZ25501I 21.57.05 DB2START COMPLETED; RETURN CODE=0
```

ST13DB0D DB2FIX

This command is only used for "online" cloning of a DB2 subsystem with DB2 SET LOG SUSPEND or DB2 SLB. DB2FIX will fix the target DB2 page spaces that have LPL or GRECP status by issuing a DB2 START DATABASE command against them. If the DB2 system is data sharing, only one DB2 member should be running when DB2FIX is run.

The DB2FIX command should be run twice. The first run will fix any DB2 CATALOG (DSNDB06) or DB2 Directory (DSNDB01) page spaces by using DATABASES(DB2). The second run will fix all other page spaces by using DATABASES(APPLICATION). The second run with DATABASES(APPLICATION) must only happen after the DB2 Catalog has been updated with the DB2SQL command.

In our run, there were no restricted statuses that needed recovery.

Example 9-26 DB2FIX DATABASES(DB2)

```
DB2FIX           -
       DB2-SSID(DBOD) -
       DATABASES(DB2) -
       DSNDB01-DBD01-STARTED(RC(16)) -
       MEMBERS-AND-DBD01(RC(16)) -
       MEMBERS-NEED-STARTING(RC(8)) -
       WAIT(5,RC(8),ACTION(QUIT)) -
       WAIT-AND-DBD01(RC(16))

.............................................

CKZ23523I CONNECTED TO DB2 SUBSYSTEM: DBOD RELEASE: 1010

CKZ23540I DB2 COMMAND: -DIS THD(*)

 DSNV401I  -DBOD DISPLAY THREAD REPORT FOLLOWS -
 DSNV402I  -DBOD ACTIVE THREADS -
 NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
 DB2CALL  T  *     3 ADMR1X       ADMR1             0097     2
  V437-WORKSTATION=DB2CALL, USERID=ADMR1,
       APPLICATION NAME=ADMR1X
 DISPLAY ACTIVE REPORT COMPLETE
 DSN9022I  -DBOD DSNVDT '-DIS THD' NORMAL COMPLETION

CKZ23540I DB2 COMMAND: -DIS UTIL(*)

 DSNU112I  -DBOD DSNUGDIS - NO AUTHORIZED UTILITY FOUND FOR UTILID = *
 DSN9022I  -DBOD DSNUGCCC '-DIS UTIL' NORMAL COMPLETION

CKZ23540I DB2 COMMAND: -DIS GROUP

 DSN7100I  -DBOD DSN7GCMD
 *** BEGIN DISPLAY OF GROUP(........) CATALOG LEVEL(101) MODE(NFM )
                  PROTOCOL LEVEL(3)  GROUP ATTACH NAME(....)
 ------------------------------------------------------------------------
 DB2                                    DB2 SYSTEM    IRLM
 MEMBER   ID  SUBSYS CMDPREF   STATUS   LVL NAME      SUBSYS IRLMPROC
 -------- --- ----   --------  -------- --- --------  ----   --------
 ........   0 DBOD   -DBOD     ACTIVE   101 SC63      IDOD   DBODIRLM
 ------------------------------------------------------------------------
 SPT01 INLINE LENGTH:        32138
 *** END DISPLAY OF GROUP(........)
 DSN9022I  -DBOD DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION

CKZ23540I DB2 COMMAND: -DIS DATABASE(DSNDB01 ) SPACENAM(*) LOCKS LIMIT(*)

 DSNT360I  -DBOD **********************************
 DSNT361I  -DBOD *  DISPLAY DATABASE SUMMARY
                 *    GLOBAL LOCKS
 DSNT360I  -DBOD **********************************
 DSNT362I  -DBOD     DATABASE = DSNDB01  STATUS = RW
                    DBD LENGTH = 108200
 DSNT397I  -DBOD
 NAME     TYPE PART  STATUS            CONNID   CORRID       LOCKINFO
 -------- ---- ----- ----------------- -------- ------------ ---------
 DBD01    TS    0001 RW
 DBD01    TS         RW
 SPT01    TS    0001 RW
```

```
 SPT01    TS         RW
 SCT02    TS         RW
 SYSUTILX TS         RW
 SYSLGRNX TS         RW
.............................................................
******* DISPLAY OF DATABASE DSNDB06  ENDED      **********************
 DSN9022I  -DB0D DSNTDDIS 'DISPLAY DATABASE' NORMAL COMPLETION

CKZ23524I NO DB2 DIRECTORY OR CATALOG DATA OR INDEX SPACES NEED TO BE STARTED

CKZ23501I 21.57.30 DB2FIX COMPLETED; RETURN CODE=0

CKZ01009I DB2 CLONING TOOL EXECUTION COMPLETE. HIGHEST RETURN CODE WAS 0
```

ST14DB0D DB2STOP

You need to perform steps 14, 15, and 16 only if the table space DBD01 in database DSNDB01 was restricted when the first DB2FIX ran. The changes made to DBD01 by DB2UPDATE might have been regressed and might need to be redone.

We ran the steps in Example 9-27 to illustrate the tasks.

Example 9-27 ST14DB0D DB2STOP

```
DB2STOP                    -
        DB2-SSID(DB0D)     -
        CASTOUT(YES) -
        DB2-ALREADY-STOPPED(RC(8)) -
        MODE(QUIESCE) -
        WAIT(5,RC(8))

.........................................

CKZ26523I CONNECTED TO DB2 SUBSYSTEM: DB0D RELEASE: 1010

CKZ26540I DB2 COMMAND: -DIS THD(*)

 DSNV401I  -DB0D DISPLAY THREAD REPORT FOLLOWS -
 DSNV402I  -DB0D ACTIVE THREADS -
 NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
 DB2CALL  T  *     3 ADMR1X       ADMR1             0097    13
  V437-WORKSTATION=DB2CALL, USERID=ADMR1,
       APPLICATION NAME=ADMR1X
 DISPLAY ACTIVE REPORT COMPLETE
 DSN9022I  -DB0D DSNVDT '-DIS THD' NORMAL COMPLETION

CKZ26540I DB2 COMMAND: -DIS GROUP

 DSN7100I  -DB0D DSN7GCMD
 *** BEGIN DISPLAY OF GROUP(........) CATALOG LEVEL(101) MODE(NFM )
                   PROTOCOL LEVEL(3)  GROUP ATTACH NAME(....)
 ------------------------------------------------------------------
DB2                                  DB2 SYSTEM    IRLM
MEMBER   ID  SUBSYS CMDPREF   STATUS   LVL NAME      SUBSYS IRLMPROC
-------- --- ----   --------  -------- --- --------  ----   --------
........   0 DB0D   -DB0D     ACTIVE   101 SC63      IDOD   DB0DIRLM
------------------------------------------------------------------
SPT01 INLINE LENGTH:        32138
*** END DISPLAY OF GROUP(........)
DSN9022I  -DB0D DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```

```
KZ26570I STOPPING DB2 SUBSYSTEM: DB0D
KZ26540I DB2 COMMAND: -STOP DB2 MODE(QUIESCE) CASTOUT(YES)

DSNY002I  -DB0D SUBSYSTEM STOPPING

KZ26538I 21.57.45 WAITING FOR DB2 TO TERMINATE
KZ26537I 21.58.04 DB2 HAS TERMINATED
```

ST15DB0D DB2UPDATE DBD01ONLY

This optional DB2UPDATE with the DBD01ONLY keyword updates only the DB2 directory. The VCATNAME and, optionally, the DB2 storage group names are updated.

Example 9-28 DB2UPDATE DBD01ONLY

```
      DB2UPDATE        -
        DBD01ONLY -
        DB2-NAME(SO01) -
        DB2-HLQS(DB0CD, DB0DD) -
        DDF( -
            LOCATION(DB0D) -
            LUNAME(SCPDB0D) -
            PORT(38390) -
            RESPORT(38391) -
            SECPORT(38392) -
           ) -
        DDF-NOT-UPDATED(RC(4)) -
        HLQ-NOT-UPDATED(RC(4)) -
        JOURNAL-DDN(JOURNAL)

.................................................
CKZ22201I 21.58.27 LINEAR FILE UPDATING STARTED - PROGRAM REV=45
CKZ22230I PROCESSING DDNAME: DBD01
CKZ22231I THE FILE'S ENDING RBA IS: 000023592960 X'00000000_01680000'
CKZ22232I NO DBD INFORMATION FOUND
CKZ22242I DBD01    CHANGED RECORDS: 0        CHANGED FIELDS: 0

CKZ22201I 21.58.28 LINEAR FILE UPDATING COMPLETED; RETURN CODE=0

CKZ22035I NO DBD INFORMATION WAS FOUND IN DBD01; WILL DYNAMICALLY ALLOCATE SYSDBDXA TO USE
CKZ22003I DDNAME=SYSDBDXA ALLOCATED FOR DSN=DB0DD.DSNDBC.DSNDB01.SYSDBDXA.I0001.A001

CKZ22401I 21.58.28 DIRECTORY UPDATING STARTED - PROGRAM REV=1
CKZ22430I PROCESSING DDNAME: SYSDBDXA
CKZ22431I THE FILE'S ENDING RBA IS: X'00000000_01D34000'
CKZ22442I SYSDBDXA CHANGED RECORDS: 0            CHANGED FIELDS: 0

CKZ22401I 21.58.29 DIRECTORY UPDATING COMPLETED; RETURN CODE=0

CKZ22001I 21.58.29 DB2UPDATE COMPLETED; RETURN CODE=0
```

ST16DB0D DB2START

This step is the same as for ST11 where DB2 is started with the special DSNZPARM.

ST17DB0D DB2SQL

The DB2SQL command makes the necessary changes to the DB2 catalog. See Example 9-29. The changes include the VCATNAME, storage group names, and volumes.

Example 9-29 ST17 DB2 SQL output

```
CKZ01020I PROGRAM: CKZ01CIO 20110309 10.44 VERS=1.0 REV=35

      DB2SQL           -
        DB2-SSID(DB0D) -
        DB2-NAME(SO01) -
        LISTSQL(Y) -
        WLM-ENVIRONMENT-MASKS( -
          DB0C* DB0D* -
                          ) -
        WLM-ENV-NOT-UPDATED(RC(4)) -
        DATACLAS-NOT-UPDATED(RC(4)) -
        MGMTCLAS-NOT-UPDATED(RC(4)) -
        STORCLAS-NOT-UPDATED(RC(4)) -
        JOURNAL-DDN(JOURNAL)
.........................................
CKZ24523I CONNECTED TO DB2 SUBSYSTEM: DB0D RELEASE: 1010


 CKZ24601I 21.59.41 SQL PROCESSOR STARTED - PROGRAM REV=12

 CKZ24631I   UPDATE SYSIBM.SYSSTOGROUP
 CKZ24631I     SET VCATNAME     = 'DB0DD   ' WHERE VCATNAME     = 'DB0CD   ' ;
 CKZ24630I VCATNAME INFORMATION UPDATED IN TABLE: SYSIBM.SYSSTOGROUP CHANGED ROWS: 16
 CKZ24631I   UPDATE SYSIBM.SYSTABLEPART
 CKZ24631I     SET VCATNAME     = 'DB0DD   ' WHERE VCATNAME     = 'DB0CD   ' ;
 CKZ24630I VCATNAME INFORMATION UPDATED IN TABLE: SYSIBM.SYSTABLEPART CHANGED ROWS: 719
 CKZ24631I   UPDATE SYSIBM.SYSINDEXPART
 CKZ24631I     SET VCATNAME     = 'DB0DD   ' WHERE VCATNAME     = 'DB0CD   ' ;
 CKZ24630I VCATNAME INFORMATION UPDATED IN TABLE: SYSIBM.SYSINDEXPART CHANGED ROWS: 19,566

 CKZ24631I   UPDATE SYSIBM.SYSVOLUMES
 CKZ24631I     SET VOLID        = 'X9602F' WHERE VOLID        = 'SB0XJI' ;
 CKZ24631I   UPDATE SYSIBM.SYSVOLUMES
 CKZ24631I     SET VOLID        = 'XV643D' WHERE VOLID        = 'SB0XJJ' ;
 CKZ24631I   UPDATE SYSIBM.SYSVOLUMES
 CKZ24631I     SET VOLID        = 'X5D842' WHERE VOLID        = 'SB0XJK'
;.........................................
CKZ24630I VOLUME   INFORMATION UPDATED IN TABLE: SYSIBM.SYSVOLUMES      CHANGED ROWS: 4

 CKZ24650W WLM ENVIRONMENT DSNWLMDB0C_GENERAL NOT CHANGED, IT DOES NOT MATCH KEYWORD
 CKZ24650W WLM ENVIRONMENT DSNWLM_DEBUGGER NOT CHANGED, IT DOES NOT MATCH KEYWORD
 CKZ24650W WLM ENVIRONMENT DSNWLM_DEBUGGER NOT CHANGED, IT DOES NOT MATCH KEYWORD
.............................................
CKZ24630I WLM ENVIRONMENTINFORMATION UPDATED IN TABLE:SYSIBM.SYSROUTINES CHANGED 4

CKZ24601I 11.33.44 SQL PROCESSOR COMPLETED; RETURN CODE=4

CKZ24501I 11.33.44 DB2SQL COMPLETED; RETURN CODE=4
```

ST18DB0D DB2FIX DATABASES(APPLICATION)

This time, we run the DB2FIX command to correct any of the application page spaces that are restricted. See Example 9-30 on page 461. The DB2FIX command is run using the keyword DATABASES(APPLICATION) on the target subsystem. This will start any application page spaces that have LPL or GRECP status.

Example 9-30 DB2FIX DATABASES(APPLICATION)

```
      DB2FIX           -
        DB2-SSID(DBOD) -
        DATABASES(APPLICATION) -
        DSNDB01-DBD01-STARTED(RC(16)) -
        MAX-CONCURRENT-CMDS(1) -
        MEMBERS-AND-DBD01(RC(16)) -
        MEMBERS-NEED-STARTING(RC(8)) -
        START-SCOPE(PAGESPACE) -
        WAIT(5,RC(8),ACTION(QUIT)) -
        WAIT-AND-DBD01(RC(16))
.............................................
CKZ23523I CONNECTED TO DB2 SUBSYSTEM: DBOD RELEASE: 1010

CKZ23540I DB2 COMMAND: -DIS THD(*)

 DSNV401I  -DBOD DISPLAY THREAD REPORT FOLLOWS -
 DSNV402I  -DBOD ACTIVE THREADS -
 NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
 DB2CALL  T  *     3 ADMR1T       ADMR1             0097     3
  V437-WORKSTATION=DB2CALL, USERID=ADMR1,
       APPLICATION NAME=ADMR1T
 DISPLAY ACTIVE REPORT COMPLETE
 DSN9022I  -DBOD DSNVDT '-DIS THD' NORMAL COMPLETION

CKZ23540I DB2 COMMAND: -DIS UTIL(*)

 DSNU112I  -DBOD DSNUGDIS - NO AUTHORIZED UTILITY FOUND FOR UTILID = *
 DSN9022I  -DBOD DSNUGCCC '-DIS UTIL' NORMAL COMPLETION

CKZ23540I DB2 COMMAND: -DIS GROUP
.............................................................
CKZ23540I DB2 COMMAND: -DIS DATABASE(* ) SPACENAM(* ) LOCKS RESTRICT(LPL, GRECP) LIMIT(*)

 DSNT367I  -DBOD    NO INFORMATION AVAILABLE
 DSN9022I  -DBOD DSNTDDIS 'DISPLAY DATABASE' NORMAL COMPLETION

CKZ23524I NO APPLICATION DATA OR INDEX SPACES NEED TO BE STARTED

CKZ23501I 22.00.03 DB2FIX COMPLETED; RETURN CODE=0
```

ST19DB0D DB2STOP

The updates to DB2 have now completed successfully. We stop DB2 and then start it again with the normal DSNZPARM and normal mode.

This step ensures that buffers have been flushed, all data has been committed to disk, and no transactions are in-flight. See Example 9-31.

Example 9-31 ST19DB0D DB2STOP output

```
      DB2STOP                     -
        DB2-SSID(DBOD)       -
        CASTOUT(YES) -
        DB2-ALREADY-STOPPED(RC(8)) -
        MODE(QUIESCE) -
        WAIT(5,RC(8))

CKZ26523I CONNECTED TO DB2 SUBSYSTEM: DBOD RELEASE: 1010
```

```
CKZ26540I DB2 COMMAND: -DIS THD(*)

DSNV401I  -DB0D DISPLAY THREAD REPORT FOLLOWS -
DSNV402I  -DB0D ACTIVE THREADS -
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
DB2CALL  T  *     3 ADMR1X       ADMR1             0097    12
 V437-WORKSTATION=DB2CALL, USERID=ADMR1,
      APPLICATION NAME=ADMR1X
DISPLAY ACTIVE REPORT COMPLETE
DSN9022I  -DB0D DSNVDT '-DIS THD' NORMAL COMPLETION

KZ26540I DB2 COMMAND: -DIS GROUP

DSN7100I  -DB0D DSN7GCMD
*** BEGIN DISPLAY OF GROUP(........) CATALOG LEVEL(101) MODE(NFM )
                  PROTOCOL LEVEL(3)  GROUP ATTACH NAME(....)
------------------------------------------------------------------------
DB2                                    DB2 SYSTEM    IRLM
MEMBER   ID  SUBSYS CMDPREF   STATUS   LVL NAME      SUBSYS IRLMPROC
 -------- --- ----   --------  -------- --- --------  ----   --------
 ........   0 DB0D   -DB0D     ACTIVE   101 SC63      IDOD   DB0DIRLM
 ------------------------------------------------------------------------
 SPT01 INLINE LENGTH:        32138
 *** END DISPLAY OF GROUP(........)
 DSN9022I  -DB0D DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION

CKZ26570I STOPPING DB2 SUBSYSTEM: DB0D
CKZ26540I DB2 COMMAND: -STOP DB2 MODE(QUIESCE) CASTOUT(YES)

 DSNY002I  -DB0D SUBSYSTEM STOPPING

CKZ26538I 22.00.22 WAITING FOR DB2 TO TERMINATE
CKZ26537I 22.00.41 DB2 HAS TERMINATED

CKZ26501I 22.00.41 DB2STOP COMPLETED; RETURN CODE=0
```

ST23DB0D DB2START

DB2 is now started with the normal DSNZPARM and in normal mode.

We now have a cloned DB2 subsystem. We can verify the clone by running SQL and utilities. See Example 9-32.

Example 9-32 ST23DB0D DB2START

```
      DB2START                    -
        DB2-SSID(DB0D)       -
        NORMAL -
        DB2-ALREADY-RUNNING(RC(8)) -
        WAIT(5,RC(8))

CKZ25570I STARTING DB2 SUBSYSTEM: DB0D
CKZ25540I START COMMAND: -DB0D START DB2
CKZ25538I 22.00.41 WAITING FOR DB2 TO START
CKZ25537I 22.00.46 DB2 HAS STARTED

CKZ25501I 22.00.46 DB2START COMPLETED; RETURN CODE=0
```

ST24DB0D DB2STOP

This additional stop DB2 job is not required but it can be used for rerunning the clone jobs. Follow up running this job with BCSCLEAN to rerun the clone.

ST25 BCSCLEAN

After we complete this cloning, we can rerun the clone jobs by running the BCSCLEAN job, which deletes (with no scratch) all catalog entries created in a target catalog by the RENAME step. BCSCLEAN is intended to delete target catalog entries created from a previous run of the DB2 Cloning Tool process that may be orphaned as a result of target volume contents being replaced.

9.5.4 Post-cloning steps

After we clone a subsystem, we want to run a few checks and cleanup tasks.

ICF catalog

Move the target ICF catalog back onto the respective SMS storage pools for the logs and data. This task is essential if you want to perform a full BACKUP SYSTEM on the target and use it for further cloning to other systems. This is a step for your system programmers.

Backups

When the cloning is complete, there are currently no archives (assuming they were not copied) and no image copies. Therefore, before any further processing, a log switch will be useful to verify the archive processing and initiate a backup of your objects.

BSDS cleanup to remove source archives

If the archives are not copied and renamed from the source subsystem, we should clean up the target BSDS by removing them. We use the DB2 Cloning Tool command DB2ALTERBSDS. See Example 9-33.

Example 9-33 DB2ALTERBSDS job

```
      DB2ALTERBSDS     -
        DB2-NAME(SO01) -
        REMOVE-ARCHIVE-LOGS( NOTRENAMED ) -
        JOURNAL-DDN(JOURNAL)

CKZ27001I 10.11.46 DB2ALTERBSDS STARTED - PROGRAM REV=9
CKZ27086I VALIDATING KEYWORD: REMOVE-ARCHIVE-LOGS
CKZ27030I OPTIONS IN EFFECT FOR THIS EXECUTION:
          DB2-NAME:         SO01
          SIMULATION:       N

CKZ27101I 10.11.46 BSDS ALTER STARTED - PROGRAM REV=15
CKZ27103I DDNAME=SYSIN    ALLOCATED FOR DSN=**TEMPORARY SYSIN DSN
CKZ27103I DDNAME=SYSPRINT ALLOCATED FOR DSN=**TEMPORARY SYSPRINT DSN
CKZ27103I DDNAME=SYSUT1   ALLOCATED FOR DSN=DBODB.BSDS01
CKZ27103I DDNAME=SYSUT2   ALLOCATED FOR DSN=DBODB.BSDS02

DSNJCNVB CONVERSION PROGRAM HAS RUN   DDNAME=SYSUT1
DSNJCNVB CONVERSION PROGRAM HAS RUN   DDNAME=SYSUT2
DELETE DSNAME=DBOAA.ARCHLOG1.A0000001,COPY1VOL=SBOX8B
DSNJ225I  DELETE OPERATION COMPLETED SUCCESSFULLY
.........................
DELETE DSNAME=DBOAA.ARCHLOG1.A0000002,COPY1VOL=SBOX8A
```

```
DELETE DSNAME=DBOCA.ARCHLOG2.A0003001,COPY2VOL=SBOXJJ
DSNJ225I  DELETE OPERATION COMPLETED SUCCESSFULLY
DELETE DSNAME=DBOCA.ARCHLOG2.A0003002,COPY2VOL=X76029
DSNJ225I  DELETE OPERATION COMPLETED SUCCESSFULLY
DSNJ200I  DSNJU003 CHANGE LOG INVENTORY UTILITY PROCESSING COMPLETED SUCCESSFULLY

CKZ27101I 10.14.45 BSDS ALTER COMPLETED; RETURN CODE=0

CKZ27001I 10.14.45 DB2ALTERBSDS COMPLETED; RETURN CODE=0
```

Runstats

Cloning at a subsystem level takes all the catalog statistics, including real-time statistics (RTS) from the source to the target.

9.6 Subsystem level cloning using the stored procedure

When you are comfortable with cloning a subsystem and you have your infrastructure set up, it might be a good time to consider using the supplied Stored Procedure CLONE_SS to build and run your jobs.

The stored procedure is defined as part of the Tools Customizer customization. It requires the DB2 administrative task scheduler to be active for the jobs that are created to be run.

The stored procedure will perform these tasks:

- Generate the necessary jobs to perform the subsystem cloning
- Schedule the jobs in the DB2 administrative task scheduler and cause the jobs to be submitted
- Monitor the execution of the jobs
- Return to the caller, when the requested cloning has ended, either in success or failure.

If the jobs fail, you either need to clean up your clone and rerun the stored procedure or manually modify the jobs and submit the jobs manually, as needed.

9.6.1 Stored procedure process

In summary, the process consists of these steps:

1. Set up the product parameter file.
2. Set up the DB2 systems parameter file.
3. Set up the cloning parameter file.
4. Invoke the stored procedure.
5. Verify the cloning.
6. Run the verified cloning.

9.6.2 Stored procedure parameters

Input to the stored procedure is provided with the three files listed. Samples of these files are available in the SCKZPARM[1] software library and members CKZPPARM, CKZSPARM, and CKZCPARM:

- Product parameter file:
  - PRODUCT_PARMS_DSN
  - PRODUCT_PARMS_MEM:
    - DSName of the product load library
    - DSName of the CKZINI file
- DB2 systems parameter file:
  - DB2_SYSTEM_PARMS_DSN
  - DB2_SYSTEM_PARMS_MEM:
    - Subsystem identifier (SSID) of each DB2 subsystem
    - DSName of the SDSNLOAD library
    - SYSAFFINITY for the LPAR where the jobs will execute
    - VCATNAME for the DB2 catalog
    - DSNames of the BSDSs
- Cloning parameter file:
  - CLONING_PARMS_DSN
  - CLONING_PARMS_MEM:
    - JOBCARD, USERID, and PASSWORD for the executing jobs
    - DSName of the output JCL library
    - HLQ of the work data sets
    - Source and target volumes, ICF catalogs, and DB2 subsystems
    - Rename masks

Run type

The TYPE parameter identifies the function that the stored procedure is to perform. It must be one of the following parameters:

BUILD	Builds JCL and adds tasks to the DB2 Administrative Task Scheduler. It sets up the environment for CLONE, RECLONE, and CLEAN.
BUILDJCL	Builds JCL only.
CLONE	Runs the initial cloning.
RECLONE	Stops the target DB2 systems, runs BCSCLEAN to clean up from the previous cloning, and then runs the cloning.
REMOVE	Deletes all JCL and removes tasks from the DB2 Administrative Task Scheduler. If CLONE or RECLONE have been done, a CLEAN should be done before REMOVE.
CLEAN	Stops the target DB2 systems and runs BCSCLEAN to clean up from the previous cloning. Can be used when the clone will no longer be used.

Run a combination of REMOVE, BUILD, and CLONE. Running with BUILD sets up all the jobs and the task scheduler for reviewing. When you are satisfied that the jobs that are generated are the correct jobs, you can call the stored procedure (SP) again with the CLONE run type.

[1] APAR PM85930 (PTF UK93214) resolves an issue due to incorrect DB value.

CLONE will pass control of the jobs to the task scheduler to run. If any job fails, you will need to either start again by using run type REMOVE and then BUILD and CLONE or by manually running the remainder of the jobs.

Output of the stored procedure

The stored procedure produces the following output:

- Output JCL PDS

  The jobs created by the stored procedure are written to the JCL-DSN PDS defined in the cloning parameter file.

- Output status file

  This file is used by the stored procedure to record and monitor the jobs submitted by the DB2 Administrative Task Scheduler.

- Jobs

  The jobs are defined in the task scheduler.

Example 9-34 shows the product parameter files that were used to set up and run the System Level Backup, which we built and ran through the ISPF dialog.

Example 9-34 Product parameter file

```
* EXAMPLE OF THE SUBSYSTEM CLONING STORED PROCEDURE PRODUCT
* PARAMETER FILE
*
CKZINI   = DBTLSP.SCKZPARM(CKZINI)
SCKZLOAD = DBTLSP.SCKZLOAD
SCKZPARM = DBTLSP.SCKZPARM
*SCKZDBRM = DBTLSP.V310.SCKZDBRM
```

The system parameter files are shown in Example 9-35.

Example 9-35 System parameter file

```
* EXAMPLE OF THE SUBSYSTEM CLONING STORED PROCEDURE DB2 SYSTEMS
* PARAMETER FILE
*
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* SOURCE DB2 DB2P (NON DATA SHARING)
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*
SSID            =  DBOC
SDSNLOAD        =  DBOCT.SDSNLOAD
SDSNEXIT        =  DBOCT.SDSNEXIT
DDF-LOCATION    =  DBOC
EXEC-SYSTEM     =  SC63
NORMAL-DSNZPARM =  DSNZPARM
*
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* TARGET DB2 DB2T (NON DATA SHARING)
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*
SSID            =  DBOD
SDSNLOAD        =  DBODT.SDSNLOAD
SDSNEXIT        =  DBODT.SDSNEXIT
BSDS01          =  DBODB.BSDS01
BSDS02          =  DBODB.BSDS02
DDF-LOCATION    =  DBOD
```

```
SYSVCAT          = DB0DD
SPECIAL-DSNZPARM = DSNZSPEC
NORMAL-DSNZPARM  = DSNZPARM
EXEC-SYSTEM      = SC63
DDF-LOCATION     = DB0D
DDF-LUNAME       = SCPDB0D
*DDF-PASSWORD    = ABCDEFG
DDF-PORT         = 38390
DDF-RESPORT      = 38391
DDF-SECPORT      = 38392
*
```

The Cloning parm file sets the location to which the JCL is to be generated (JCL-DSN). An authorized user ID and password are required to run the jobs at this point. You might want to protect the file in which the parameters are stored, which is required at this stage. See Example 9-36.

The following key parameters define the setup of a system level backup clone:

- SOURCE-VOLUMES = DB2SLB
- SOURCE-TOKEN = LAST
- SOURCE-LOCATION = DB0C

Example 9-36 Cloning parm file

```
* EXAMPLE OF THE SUBSYSTEM CLONING STORED PROCEDURE CLONING
* PARAMETER FILE
*
JCL-DSN      = ADMR1.CLONE2.SP.JCL
STATUS-DSN   = ADMR1.CLONE2.SP.STATUS
WORK-PREFIX  = ADMR1.CLONE2.WRK
TASK-PREFIX  = CKZ_CLONE2_SP
CLONING-TYPE = ONLINE
*
USERID   = ADMR1
PASSWORD = xxxxxxx
*
JOBCARD1 = //ADMR1Q JOB (999,POK),'SP CLONE',CLASS=A,
JOBCARD2 = // MSGCLASS=X,NOTIFY=ADMR1,TIME=NOLIMIT,REGION=0M
JOBCARD3 = /*JOBPARM SYSAFF=SC63,L=9999
JOBCARD4 = // JCLLIB ORDER=(DB0AM.PROCLIB)
JOBCARD5 = //*************************************************
*JOBCARD6 = //*
*JOBCARD7 = //*
*
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* COPY PARAMETERS FOR CLONING DDOC TO DDOD
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
SOURCE-VOLUMES   = DB2SLB
SOURCE-TOKEN     = LAST
SOURCE-LOCATION  = DB0C
*
TO-STORAGEGROUP   = DB0DDATA DB0DLOG1 DB0DLOG2 DB0DTEMP
*
USERCATALOGS = UCAT.DB0CD UCAT.DB0DD
               UCAT.DB0CL UCAT.DB0DL
*
DM-PGM             = ADRDSSU
DM-CONSISTENT      = N
DM-COPYCMDLIMIT    = 24
```

```
DM-FASTREP         = REQ
DM-INCREMENTAL     = N
*
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* RENAME PARAMETERS FOR CLONING DB2P TO DB2T
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
RENAME-MASKS             = DBOCD.**   DBODD.**
                           DBOCB.**   DBODB.**
                           DBOCL.**   DBODL.**
                           DBOCX.**   DBODX.**
MAX-TASKS                = 10
MISSINGUCAT              = DELETE,RC(4)
NOTRENAMED               = DELETE,RC(4)
ORPHANCATENTRY           = KEEP,RC(8)
RECATALOG                = Y
RENAME-ERROR             = ABORT
TEMPDSN                  = DELETE,RC(4)
*
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* DB2 PARAMETERS FOR CLONING SUBSYSTEM DB2P TO DB2T
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
DB2-SUBSYSTEM-PAIR     = DBOC DBOD
DB2-HLQS               = DBOCD DBODD
                         DBOCX DBODX
ARCHIVE                = N
LISTSQL                = Y
WLM-ENVIRONMENT-MASKS  =  DSNWLMDBOC*  DSNWLMDBOD*
*
```

Quick checks

Perform these quick checks:

1. Check that the stored procedure CLONE_SS exists. Verify that it has an associated WLM-managed address space where the DB2 Cloning Tool load libraries are in STEPLIB.
2. The SP can run on the source system unless the source DB2 is to be shut down.
3. The SP cannot be run on the target subsystem because it will need to be started and stopped.
4. The Admin Scheduler started task (STC) on the target will need to be shut down manually when DB2 on the target is shut down if the STOPONDB2STOP parameter is not used for the ssidADMT STC.

9.6.3 Invoke the stored procedure

Without writing a customized wrapper for the stored procedure, there are two generally available ways that you can invoke it: DB2 Administration Tool and Data Studio.

Likely, the use of the stored procedure can be integrated into application processes and other tools.

A CALL for the stored procedure might look like this example:

```
EXEC SQL CALL CKZTOOLS.CLONE_SS( :TYPE,
:PPARMDSN, :PPARMMEM,
:SPARMDSN, :SPARMMEM,
:CPARMDSN, :CPARMMEM,
:SP_RC, :SP_MSGS)
```

DB2 Administration Tool

We can call the SP from the DB2 Administration Tool. After displaying the SP, use the CALL line command, as shown in Figure 9-48.

```
DB2 Admin ---------------- DBOC Stored Procedures ----------- Row 258 from 310
Command ===>                                                  Scroll ===> PAGE

Commands: GRANT
Line commands:
 AH - Schema Auth  A - Auth  DROP - Drop  AL - Alter  K - Package  PA - Parms
 DIS - Display  STO - Stop  STA - Start  GR - Grant  COM - Comment  CALL - Call
 ? - Show all line commands
                                                                 S
                                                        Res      Q S P C External
Sel  Schema   Name               Version  A Lang Parms  Set O L R T R Name
     CKZ*     *                  *        * *         *   * * * * * * *
---- -------- ------------------ -------- - ---- ------ --- - - - - - ---------
CALL CKZTOOLS CLONE_SS                     ASSE      9   1 E M N M N CKZ00200
****************************** END OF DB2 DATA ******************************
```

Figure 9-48 Calling the clone stored procedure from the DB2 Admin Tool

See Figure 9-49.

```
DB2 Admin ---------- DBOC Call Procedure Input Parameters ---- Row 1 to 7 of 7
Command ===>                                                   Scroll ===> PAGE

Commands:  CALL

Stored procedure . : CKZTOOLS.CLONE_SS
Version  . . . . . : n/a
Invocations  . . . . 1     (number of times to call the procedure)
Honor max rows . . . YES   (Yes or No to restrict the number of rows to fetch)

Line commands:
<value> - Parameter value

Parameter value                 Parameter name     Type          Length  Scale
                                *                  *                  *      *
-------------------------------> ------------------ -------- ----------- ------
CLONE                           RUN_TYPE           CHAR               10      0
ADMR1.CLONE3.SP.JCL             PRODUCT_PARMS_DSN  CHAR               44      0
PPARM                           PRODUCT_PARMS_MEM  CHAR                8      0
ADMR1.CLONE3.SP.JCL             DB2_SYSTEM_PARMS_D CHAR               44      0
SPARM                           DB2_SYSTEM_PARMS_M CHAR                8      0
ADMR1.CLONE3.SP.JCL             CLONING_PARMS_DSN  CHAR               44      0
CPARM                           CLONING_PARMS_MEM  CHAR                8      0
```

Figure 9-49 Setting the input parameters for the clone SP from the DB2 Admin Tool

The drawback with using this method is that your Parm values are not saved and you must type them each time. The result sets are not as easy to view as they are with Data Studio.

The other alternative is to use Data Studio.

IBM Data Studio 3.1 or 3.2

The method that we used more often is Data Studio. This method provides more flexibility with setting up and monitoring the SP. This functionality is not currently supported with Data Studio 3.1.1. It is resolved in Data Studio 3.2.

Use the Administration Explorer Perspective and display the CLONE_SS SP as shown in Figure 9-50.

Use Run Settings to set the input parameters.

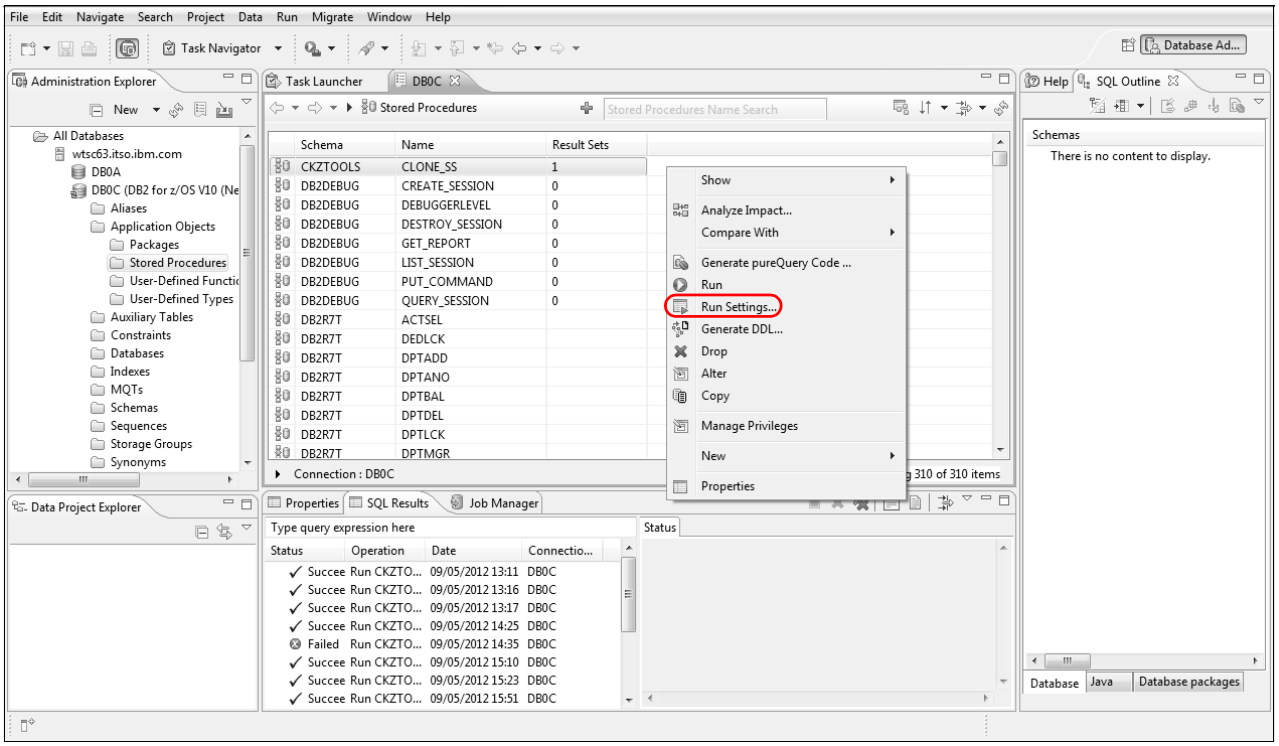

Figure 9-50 Data Studio Run Settings to set the input parameters for the stored procedure

See Figure 9-51 on page 471.

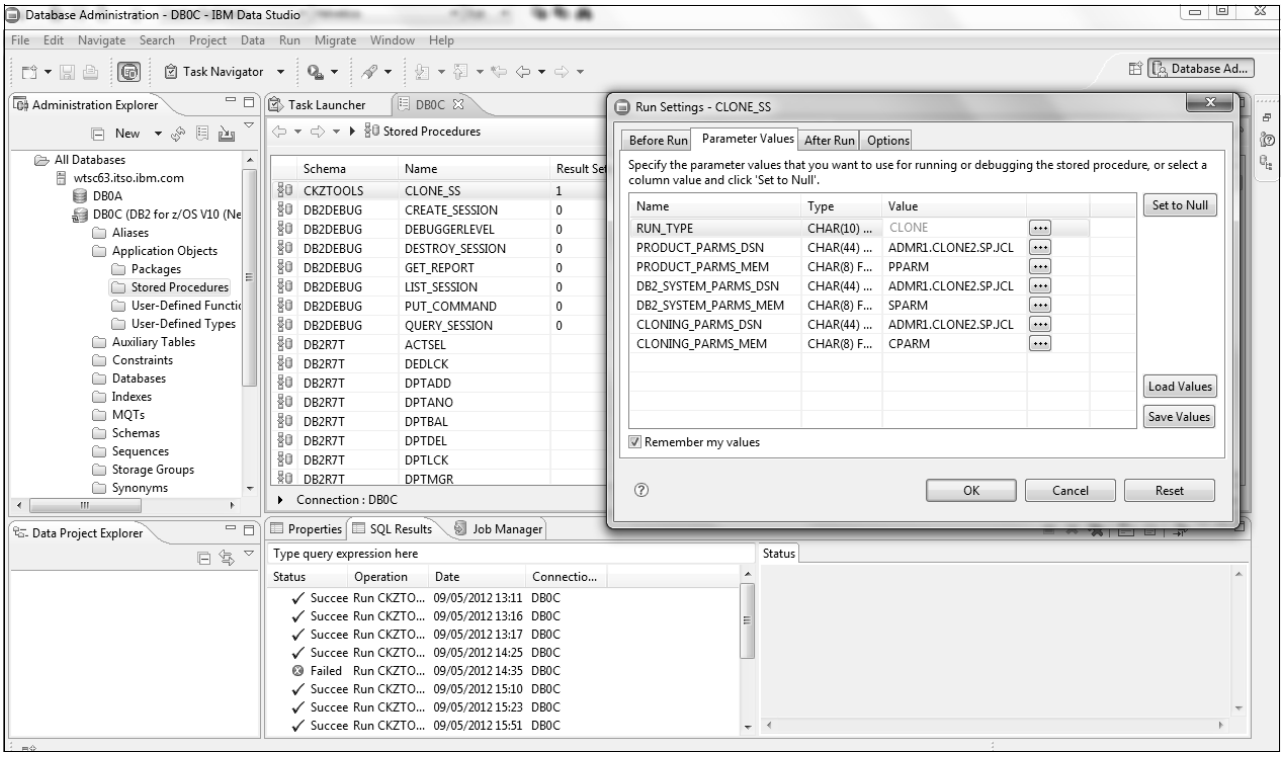

Figure 9-51 Setting parameters within Data Studio

When the settings are defined, the SP is run. The results are shown in the Status area at the bottom of Figure 9-52.

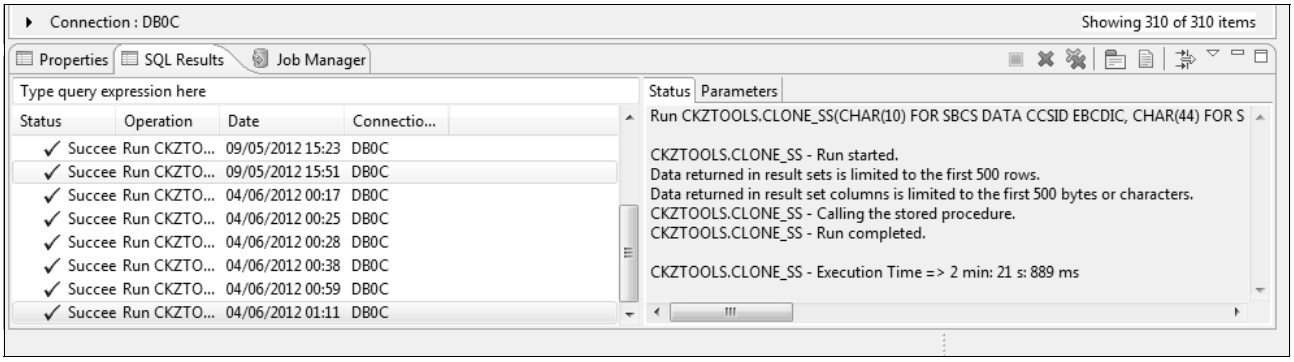

Figure 9-52 Results from the execution of the stored procedure

The stored procedure builds the jobs into a PDS and updates the DB2 Administrative Task Scheduler. The Scheduler manages the job executions from here. We will need to check the output of the jobs from our TSO session. If there is a failure, a decision is made whether the job can be fixed and continue or whether we need to start the clone again.

To continue from a failed job involves manually submitting the jobs, because there is no simple restart setup for the DB2 Administrative Task Scheduler. We need a way to see the jobs that are scheduled.

How to view the scheduled jobs

There are several ways to view the scheduled jobs:

- DB2 Administrative Task Scheduler
- IBM DB2 Automation Tool for z/OS, which contains an interface to these DB2 administrative task scheduler stored procedures. You can also view any task within the admin task scheduler DB2 tables.
- SQL
- Data Studio

DB2 administrative task scheduler

The scheduler is installed with user-defined functions (UDFs) and stored procedures for management. See 6.4, “DB2 administrative task scheduler” on page 237.

SQL

We can run queries, such as these examples:

- `SELECT * FROM TABLE (DSNADM.ADMIN_TASK_STATUS()) AS T`

  `WHERE T.TASK_NAME LIKE 'CKZ_CLONE3%' ;`
- `SELECT * FROM TABLE (DSNADM.ADMIN_TASK_LIST()) AS L;`

These queries list the jobs and their status, as shown in Example 9-37.

Example 9-37 SQL queries for Admin_task_status

```
DB2 Admin -- Browse Result of SQL Select              ----- Row 1 to 17 of 17
Command ===>                                                 Scroll ===> PAGE

L TASK_NAME                       STATUS    NUM_INVOCATIONS START_TIMESTAMP
  *                               *                       * *
- ------------------------------- --------- --------------- ------------------
  CKZ_CLONE3_SP_ST001             COMPLETED               2 2013-02-26-18.29.1
  CKZ_CLONE3_SP_ST002             COMPLETED               2 2013-02-26-18.29.2
  CKZ_CLONE3_SP_ST003             COMPLETED               3 2013-02-26-18.29.5
  CKZ_CLONE3_SP_ST004             COMPLETED               3 2013-02-26-18.29.5
  CKZ_CLONE3_SP_ST005             COMPLETED               3 2013-02-26-18.29.5
  CKZ_CLONE3_SP_ST006             COMPLETED               3 2013-02-26-18.30.0
  CKZ_CLONE3_SP_ST007             COMPLETED               3 2013-02-26-18.30.0
  CKZ_CLONE3_SP_ST008             COMPLETED               3 2013-02-26-18.30.0
  CKZ_CLONE3_SP_ST009             COMPLETED               3 2013-02-26-18.30.0
  CKZ_CLONE3_SP_ST010             COMPLETED               3 2013-02-26-18.31.1
  CKZ_CLONE3_SP_ST011             COMPLETED               2 2013-02-26-16.52.5
  CKZ_CLONE3_SP_ST012             COMPLETED               2 2013-02-26-16.52.5
  CKZ_CLONE3_SP_ST013             COMPLETED               2 2013-02-26-16.53.0
  CKZ_CLONE3_SP_ST014             COMPLETED               2 2013-02-26-16.53.1
  CKZ_CLONE3_SP_ST015             COMPLETED               2 2013-02-26-16.53.1
  CKZ_CLONE3_SP_ST016             COMPLETED               2 2013-02-26-16.53.2
  CKZ_CLONE3_SP_ST017             COMPLETED               1 2013-02-26-16.53.4
```

Data Studio

Data Studio is useful in this situation. You can run the UDF and display the output. The output can be exported to a number of formats, as well. See Figure 9-53 on page 473.

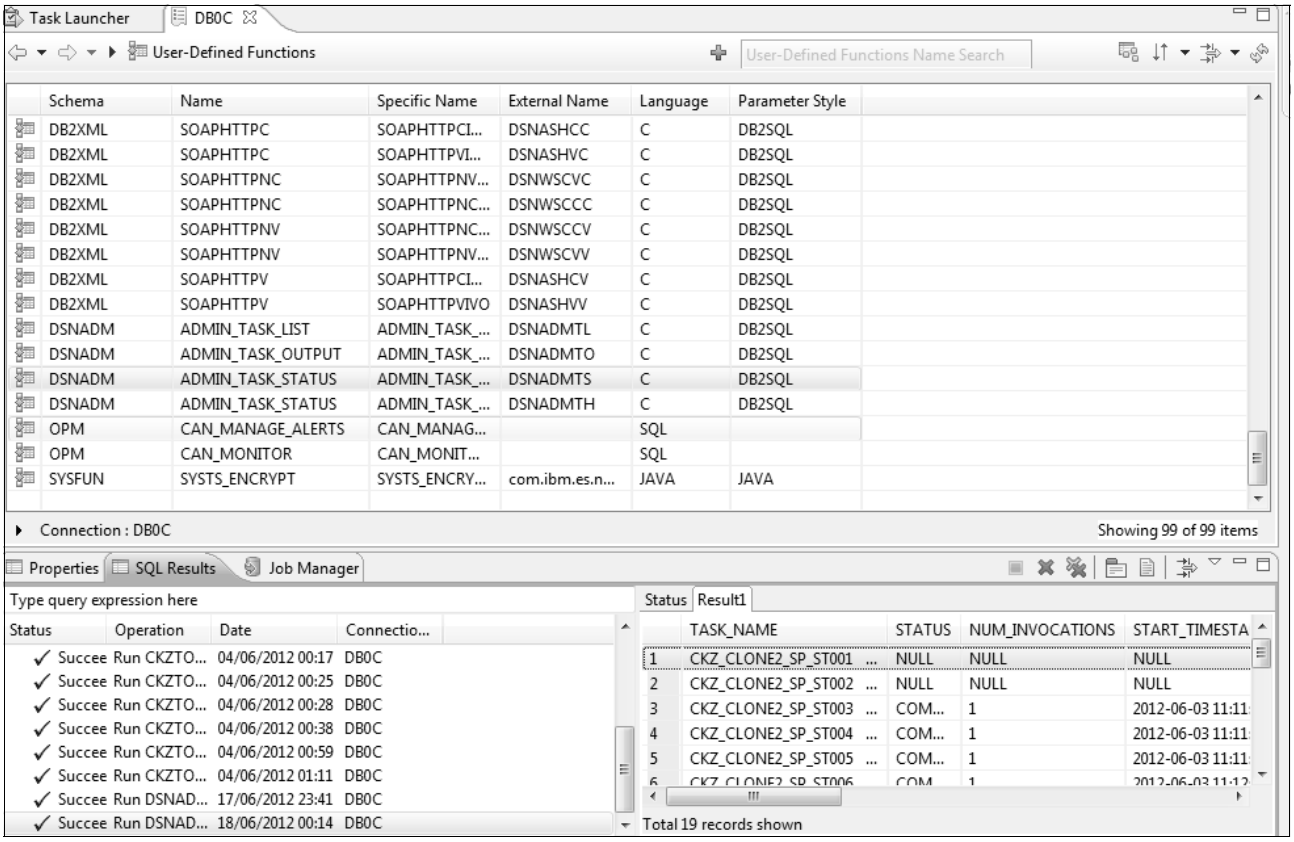

Figure 9-53 Data Studio used to display DB2 administrative task scheduler

IBM DB2 Automation Tool for z/OS

DB2 Automation Tool for z/OS provides a dialog (option 12 DB2 Admin Scheduler) to manage the DB2 Administrative Task Scheduler. See Figure 9-54. We used DB2 Automation Tool for z/OS to manage the DB2 Administrative Task Scheduler.

```
AUTOTOOL V4R1 --- IBM DB2 Automation Tool for z/OS    --- 2013/04/22  23:27:28
Option  ===>
--------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Disaster Recovery
                3 - Exception Profiles      11 - Stand Alone Utilities
                4 - Job Profiles            12 - DB2 Admin Scheduler
                5 - Quick Build             13 - DB2 Autonomic Statistics
                6 - Execution Reports        X - Exit
                7 - DB2 Command Processor

--------------------------------------------------------------------------------
 DB2 Subsystem ID: DB0C         (1-4 Character Subsystem ID or ? for list)
 Current SQLID:    DB2AUTH      User: ADMR1   - HAA
--------------------------------------------------------------------------------
```

Figure 9-54 DB2 Automation Tool option 12

Selecting option 12 displays Figure 9-55 on page 474.

```
AUTOTOOL V4R1   -------- DB2 Admin Task Scheduler ------- 2013/04/22  23:28:57
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  V - View  S - Status  U - Update
--------------------------------------------------------------------------------
 Task Name Like     *CLONE3*                              DB2 Subsystem: DBOC
 Task Creator Like  *                                  Row 1 of 17            >
--------------------------------------------------------------------------------
    Task                                Task
Cmd Name                                Description         Last Modified
    CKZ_CLONE3_SP_ST001                 DB2STOP RCO for DB2 2013-02-26-18.29
    CKZ_CLONE3_SP_ST002                 BCSCLEAN for DB2 CL 2013-02-26-13.35
    CKZ_CLONE3_SP_ST003                 DB2GETBACKINFO for  2013-02-26-18.29
    CKZ_CLONE3_SP_ST004                 BACKINFO-REFORMAT f 2013-02-26-13.35
    CKZ_CLONE3_SP_ST005                 COPY from SLB for D 2013-02-26-13.35
    CKZ_CLONE3_SP_ST006                 COPY for DB2 CLONIN 2013-02-26-13.35
    CKZ_CLONE3_SP_ST007                 CKZRNTGT SLB for DB 2013-02-26-13.35
    CKZ_CLONE3_SP_ST008                 VOLOPTIONS SLB for  2013-02-26-13.35
    CKZ_CLONE3_SP_ST009                 RENAME for DB2 CLON 2013-02-26-13.35
    CKZ_CLONE3_SP_ST010                 DB2UPDATE for DB2 C 2013-02-26-13.35
    CKZ_CLONE3_SP_ST011                 DB2ALTERBSDS SLB fo 2013-02-26-13.35
    CKZ_CLONE3_SP_ST012                 DB2START SPECIAL WT 2013-02-26-13.35
    CKZ_CLONE3_SP_ST013                 DB2FIX DB2 for DB2  2013-02-26-13.35
    CKZ_CLONE3_SP_ST014                 DB2SQL for DB2 CLON 2013-02-26-13.35
    CKZ_CLONE3_SP_ST015                 DB2FIX APPL for DB2 2013-02-26-13.35
    CKZ_CLONE3_SP_ST016                 DB2STOP for DB2 CLO 2013-02-26-13.35
    CKZ_CLONE3_SP_ST017                 DB2START NORMAL for 2013-02-26-13.35
```

Figure 9-55 DB2 Automation Tool showing DB2 Cloning Tool job schedule

Verifying the cloning jobs

After we run the SP with the BUILD option, we need to check the JCL that is generated. The jobs that are listed by the task scheduler are similar to the jobs that are generated when the ISPF dialog is used. However, there are a few differences:

- There is no COPYCHECK job. This job is not required unless you want to verify the status of the volume copies. Remember that the copies are flash copies and processing of your other jobs can continue while the background copy is still running. If you want to verify the status or cancel the copy process, build a job to run this command. See “COPYCHECK” on page 410.
- DB2FIX for DB2 and APPLICATION. These jobs will check the status of the catalog objects and the application for the status of LPL and GRECP. It will then issue a start database to perform a recovery, if required. These jobs are only needed when performing an online clone, but the jobs still are generated for other clones when using the SP.
- DB2ALTERBSDS. This job specifies that an SLB start conditional restart record is to be added to the BSDS. The SLB start conditional restart record that is created will have an ENDLRSN value that comes from the system backup record in the BSDS that was extracted by the prior DB2UPDATE. This job is unnecessary in our scenario, but it is generated, by default, when using the SP.

As part of verifying the jobs, you manually step through the job suite and run the jobs outside of the task scheduler. Remember that with each of the jobs, you can add the parameter SIMULATE. This parameter allows you to run the job without updating anything. If the job performs as expected, remove the parameter and resubmit the job.

Job results

We can use the System Display and Search Facility (SDSF) output to review the job results after we run the SP. The other alternative is to use the DB2 Automation Tool Admin Task Scheduler.

After listing the tasks, we can use the S option (Status Detail) to select the task that we want to view (Figure 9-56). This option lists all executions of this task. From this panel, we can view the status detail of the job and the job output.

```
AUTOTOOL V4R1   --------- DB2 Admin Task Status --------- 2013/04/24  08:45:02
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: S - Status Detail  O - View Output
--------------------------------------------------------------------------------
 Task Name    CKZ_CLONE3_SP_ST015
 Task Creator ADMR1                            DB2 Subsystem: DBOC
 Max History  0010                             Row 1 of 2
--------------------------------------------------------------------------------
Cmd Userid  SSID  Status      Start Timestamp             End Timestamp
    ADMR1   DBOC  COMPLETED   2013-02-26-16.53.15.000000  2013-02-26-16.53.22
    ADMR1   DBOC  COMPLETED   2013-02-26-13.38.50.000000  2013-02-26-13.38.57
****************************** Bottom of Data ***********************************
```

Figure 9-56 DB2 Automation Tool task list

Select option S to view the status detail of a job, as shown in Figure 9-57 on page 476.

```
AUTOTOOL V4R1   --------- DB2 Admin Task Status --------- 2013/04/24  08:45:02
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: S - Status Detail  O - View Output
--------------------------------------------------------------------------------
 Task Name    CKZ_CLONE3_SP_STO15
 Task Creator ADMR1                          DB2 Subsystem: DBOC
 Max History  0010                           Row 1 of 2
--- Esssssssssssssssssss DB2 Admin Task Status Detail sssssssssssssssssssN
Cmd e                                                                    e
 S  e Task Creator . . . ADMR1                                        >  e
    e Task Name  . . . . CKZ_CLONE3_SP_STO15                          >  e
*** e Task Description   DB2FIX APPL for DB2 CLONING TOOL             >  e
    e                                                                    e
    e Task Userid  . . . ADMR1                                       >  e
    e Begin Timestamp  . 2013-02-26-16.53.15.000000                      e
    e End Timestamp  . . 2013-02-26-16.53.22.000000                      e
    e SSID . . . . . . . DBOC                                            e
    e                                                                    e
    e Status   . . . . . COMPLETED                                       e
    e Message. . . . . .                                             >  e
    e Job ID . . . . . . JOB18021                                        e
    e Max RC . . . . . . 000000                                          e
    e System Abend . . . N/A                                             e
    e User Abend . . . . N/A                                             e
    e Completion Type    Job ended normally                          >  e
    e                                                                    e
    e                                                                    e
    DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

Figure 9-57 DB2 Automation Tool Admin Task Scheduler option S for status detail

We can select option O (View Output) to view the job output. See Figure 9-58.

```
AUTOTOOL V4R1   --------- DB2 Admin Task SYSOUT --------- 2013/04/24  08:47:14
Option  ===>                                                Scroll ===> PAGE
--------------------------------------------------------------------------------
 Task Name    CKZ_CLONE3_SP_STO15
 Task Creator ADMR1                          DB2 Subsystem: DBOC
 JobID        JOB18021                       Row 1 of 818         +>
--------------------------------------------------------------------------------
                  J E S 2  J O B  L O G  --  S Y S T E M  S C 6 3  --  N O D E

16.53.15 JOB18021 ---- TUESDAY,   26 FEB 2013 ----
16.53.15 JOB18021  IRR010I  USERID ADMR1    IS ASSIGNED TO THIS JOB.
16.53.15 JOB18021  ICH70001I ADMR1    LAST ACCESS AT 16:53:15 ON TUESDAY, FEBRU
16.53.15 JOB18021  $HASP373 ADMR1Q   STARTED - INIT 1    - CLASS A - SYS SC63
16.53.15 JOB18021  IEF403I ADMR1Q - STARTED - TIME=16.53.15 - ASID=00A8 - SC63
16.53.21 JOB18021  -                                          --TIMINGS (MINS.)-
16.53.21 JOB18021  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOC
16.53.21 JOB18021  -ADMR1Q            DB2FIXAP    00   1364    .00    .00    .1
16.53.21 JOB18021  IEF404I ADMR1Q - ENDED - TIME=16.53.21 - ASID=00A8 - SC63
16.53.21 JOB18021  -ADMR1Q   ENDED.  NAME-SP CLONE             TOTAL CPU TIME=
16.53.21 JOB18021  $HASP395 ADMR1Q   ENDED
```

Figure 9-58 DB2 Automation Tool Admin Task Scheduler option O to view output

9.7 Cloning at the table space level

DB2 Cloning Tool Table Space Cloning simplifies and automates the refresh of DB2 table spaces and index spaces. When paired with data set level fast data replication tools, DB2 Cloning Tool Table Space Cloning can refresh data easily in minutes, instead of hours.

It is an offline utility that uses data set level fast replication tools, so it causes minimal disruption and fits into tight maintenance windows.

DB2 Cloning Tool Table Space Cloning makes it fast and easy for you to refresh DB2 test or quality assurance environments, troubleshoot production problems, and aid in development.

9.7.1 Key features

Table space cloning offers these key features:

- Uses high speed data set fast replication utilities, instead of traditional slow utilities.
- Automates manual processes, such as object ID translation between source and target subsystems.
- Uses TCP/IP to copy to subsystems that are not connected via call attachment facility (CAF).
- Uses an interface similar to the IBM LISTDEF facility to drastically reduce the learning curve.
- Allows you to select individual table spaces or an entire database.
- Provides the capability to exclude undesired table spaces and indexes.
- Allows you to select all table spaces in a referential integrity (RI) relationship.
- Select and migrates LOB tables and clone tables (for DB2 Version 9.1) easily.
- Copies tables that have identity columns to another subsystem.
- Copies tables containing XML columns (beginning with DB2 9).
- Includes the capability to mask column data. The changes are made based on masking rules that are enabled during the copy.
- Allows you to create the necessary jobs by using ISPF interactive panels, if you want.
- Provides integrated support with DB2 Admin Tools 10.2.

9.7.2 Refreshing DB2 objects without DB2 Cloning Tool

DB2 table spaces and index spaces can be refreshed without using DB2 Cloning Tool Table Space Cloning by using DSN1COPY or UNLOAD/LOAD utilities. However, there are several considerations:

- DSN1COPY:
  - Requires static JCL and control parameters. It does not allow for adding new DB2 extents, adding new table spaces or index spaces, or dropping existing ones.
  - Object ID translation parameters require you to perform painstaking manual research and maintenance.
  - Cannot manage partition-by-growth (PBG) table spaces where the number of partitions differs.

- DB2 UNLOAD/LOAD:
  - Can require a significant amount of time before large cloned data sets are available for use.
  - The VSAM objects on the target side can require more space than on the source side due to FREESPACE and FREEPAGE assignments. Therefore, the LOAD utility might abend and require a manual increase of space for a target table space or index space. In addition, extra time is needed to rerun the LOAD process.
  - A PBG cannot be loaded in parallel and, therefore, might take considerable more time.

Using DB2 Cloning Tool Table Space Cloning, DB2 table spaces and index spaces can be refreshed quickly if using a fast replication utility. DB2 Cloning Tool Table Space Cloning dynamically adjusts to new or dropped table spaces and index spaces, performs automatic object ID translation, and there are no unanticipated size changes between the source and target table spaces and index spaces.

9.7.3 Our table space cloning scenario

We are using our test environment database, which has several object types that illustrate the capability of the table space clone:

- PBG
- PBR
- LOBS
- XML
- Identity columns

9.7.4 Target Data Definition Language (DDL)

DB2 Cloning Tool Table Space Cloning will optionally generate and execute DDL to use to create non-existent target objects.

CREATE DDL is generated for the following objects:

- Databases
- Table spaces
- Tables
- Indexes

Both LOB and XML spaces are supported. All referenced STOGROUPs, distinct types, and other supporting objects must exist on the target to be able to execute the DDL that is generated.

In addition to supporting missing target objects, DB2 Cloning Tool Table Space Cloning also will generate source object DDL to save to a data set or it can execute DDL from an input data set.

Because LISTDEF statements select objects to be processed, you can use this function to generate the DDL. This LISTDEF is not the same as the LISTDEF that is used by the DB2 utilities and, therefore, behaves slightly differently.

When target DB2 table spaces and index spaces are missing, DDL might be generated to create those *missing* DB2 objects along with their tables and indexes.

Optionally, DDL for *all* target table spaces, index spaces, tables, and indexes can be generated. In this case, the LISTDEF is used differently to select objects that require DDL.

All table spaces that are referenced in LISTDEF statements, either directly or indirectly, are referred to as the *object set*.

All databases that contain table spaces in the object set can have DDL generated. However, due to LISTDEF INCLUDE/EXCLUDE and other selection parameters, not all table spaces in a database can be in an object set and have DDL generated.

Note: While the Cloning Tool can generate the DDL, you might prefer to use the DB2 Compare Tool and align your source and target structures before cloning. Using this method will ensure that you have the source and target in sync down to the column specification.

The other clean method is to drop the target object and re-create them with the source DDL.

Remember that the Cloning Tool only generates a limited set of object types, so objects, such as views, will not be generated. Complete DDL can be generated by using the DB2 Admin Tool GEN function.

General rule: Align your environments before you clone.

We do not cover the DDL and assume that the source and target are aligned with a few exceptions, such as managing PBG table spaces, partitions, and numbers.

9.7.5 Renaming table spaces and index spaces

Table spaces and index spaces cannot be renamed because DB2 does not support it. You can only rename a DB2 table space or index space by using the DB2 DROP and CREATE commands. A DB2 DROP deletes both the VSAM data set and the DB2 catalog information that concern the DB2 table space or index space.

However, a source table space or index space can be copied to an existing target table space or index space with a different name.

The process is much smoother if you spend the effort to align the source and target structures outside of the Cloning Tool. You can then set up Object Translation where the object names are different. Our example demonstrates this scenario because we clone to a different database and schema.

9.7.6 Object status

Without setting the copy step to use a FUZZY-COPY, the source and target objects will be stopped during the cloning process in order to maintain data integrity. The COPY command controls when the source objects are started with the AUTO-START-SOURCE-SPACE parameter. Y starts the source table spaces and index spaces in RW mode after the copy is complete.

If the COPY command keyword FUZZY-COPY(Y) is specified, DB2 Cloning Tool Table Space Cloning will not stop the source table spaces and index spaces. This is not recommended because if the table spaces and index spaces are in RW status, there can be data integrity issues. A QUIESCE before starting the clone is recommended.

By using a FUZZY copy, the cloning tool process can be supplemented with a Log Apply, which will use either a Quiesce point-in-time copy that is taken or a consistent point when the target job is run.

9.7.7 Prerequisite to table space cloning

In the Tools Customizer Customization library from the tool installation, there is a member B4TS2 that allocates the table space cloning data sets. One job will be generated. The JCL will allocate the libraries that can be used by the table space cloning jobs.

Member B5TS3 in the customization library provides XML support for the data sets.

DDL can be used by Cloning Tool to create a table with an XML column. This DDL must be available to the target job if there are copied XML strings on the source subsystem that are not on the target subsystem.

9.7.8 Tablespace cloning with the ISPF dialog

Setting up the default setting for a table space clone is the same as for the subsystem clone. On the DB2 Cloning Tool for z/OS Primary Option Menu, you select User settings. You type the details on the User settings option and select the User DB2 tablespace clone settings option.

The default settings can be overwritten for each clone profile that is generated. The example here reviews the settings as we build the profile. We will build the jobs on the source environment (Figure 9-59). We start from option 1 on Figure 9-7 on page 422 and then option 2 on Figure 9-8 on page 422. Then, on the DB2 Tablespace Clone Profile Display, we create a new profile.

```
CKZ1DTCL             DB2 Tablespace Clone Profile Display
C EsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssN => PAGE
  e CKZ1DTCN                                                            e
C e Enter New DB2 Tablespace Clone Profile Options:                     e
L e   Creator . . . . . : ADMR1                                         e  - Copy
  e   Profile Name  . . . TABLESPACE 6                                  e
P e   Description . . . . GLWSAMP TARGET-JOB-INDEX-REBUILD-DDN         e
C e   Share Option  . . . UPDATE     (Update, View only, No access)    e
  e                                                                     e 14    >
  e  Press ENTER or F3 to process                                       e
C e  Press F12 to cancel                                                e
  e                                                                     e
  e                                                                     e
  DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM  OPTION
    SIMPLE TS 1                    ADMR1    UPDATE    SINGLE CLONE
```

Figure 9-59 Create a tablespace clone profile

We can now start building our profile. See Figure 9-60 on page 481.

```
CKZ1ETCP              Edit DB2 Tablespace Clone Profile
Option ===>

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . . GLWSAMP TARGET-JOB-INDEX-REBUI >

1  Source job
2  Target job
3  Report job
4  TCPIP Server job
```

Figure 9-60 Tablespace clone profile

9.7.9 Setting up the target job

There is little to do to configure the target and report job, primarily the job card. Although it is covered first in this chapter, it is best to do this configuration after the source job has completed because the DD specifications can be promoted from the source job to the target job. Type 2 to set up the target.

```
CKZ1SFTJ                      Setup Target Job
Option  ===>

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

1  Job card
2  DD Specification
```

Figure 9-61 Target job setup

The Setup TCP/IP Server Job panel displays a list of options to be set when creating the TCP/IP server job. The TCP/IP server job is optional. It facilitates communication between the source job and a target DB2 subsystem on a different z/OS system.

You will need to review the job card to ensure that you are running the target job at the right location and with a valid job card.

The DD specifications default from the user settings and are mapped from the source setup.

9.7.10 Setting up the source job

The majority of the setup is within the source job, as can be seen in Figure 9-62 on page 482.

```
CKZ1FTSJ                       Setup Source Job
Option  ===>

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI

1  Job card and qualifiers
2  DD Specification
3  SET Command
4  COPY Command
5  HLQDDDF Command
6  XML Object Definition
7  LISTDEF Commands
8  Data Masking Commands
```

Figure 9-62 Tablespace cloning source setup

DD specifications

This panel allows you to enter DD specifications for all the necessary DDs. The default DD specifications are provided from the setup in the User Settings. You must allocate any data sets that do not exist before you attempt to execute the table space cloning jobs. See 9.7.7, “Prerequisite to table space cloning” on page 480. See Figure 9-63 on page 483 for the default DD cards that are used.

```
CKZ1DDSP            DB2 tablespace clone DD Specification
Command ===>                                                Scroll ===> PAGE

Commands:      D - Set Defaults  C - Clear Defaults  U - User DD Specification
Line commands: S - Select/Unselect

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

Control DD:
  HLQ  . . . . . ADMR1.TS2                          (control HLQ)
  Member . . . . GLWSAMP6                           (control member)

                                                           Row 1 of 19   +
      DD Name  DD
  SEL CKZIN    *
  SEL CKZPRINT SYSOUT=*
  SEL CKZINI   DISP=SHR,DSN=DBTLSP.SCKZPARM(CKZINI)
  SEL CKZLOG   SYSOUT=*
  SEL CKZLSTDF DISP=SHR,DSN=ADMR1.TS2.LISTDEF(GLWSAMP6)
      CKZMSKDF DISP=SHR,DSN=ADMR1.TS2.MASKDEF(GLWSAMP6)
  SEL CKZCRXML DISP=SHR,DSN=ADMR1.TS2.XMLCRDDL(GLWSAMP6)
  SEL CKZSDBOD DISP=OLD,DSN=ADMR1.TS2.SYNCDB2(GLWSAMP6)
      CKZCDBOD DISP=OLD,DSN=ADMR1.TS2.COPYDSNS(GLWSAMP6)
  SEL CKZMDBOD DISP=OLD,DSN=ADMR1.TS2.XMLSTR(GLWSAMP6)
  SEL CKZQDBOD DISP=OLD,DSN=ADMR1.TS2.SQLOUT(GLWSAMP6)
      CKZWDBOD DISP=OLD,DSN=ADMR1.TS2.CMDSSTPT(GLWSAMP6)
      CKZXDBOD DISP=OLD,DSN=ADMR1.TS2.CMDSSTPS(GLWSAMP6)
      CKZYDBOD DISP=OLD,DSN=ADMR1.TS2.CMDSSTRS(GLWSAMP6)
      CKZZDBOD DISP=OLD,DSN=ADMR1.TS2.IDCAMS(GLWSAMP6)
      CKZDDBOD DISP=OLD,DSN=ADMR1.TS2.DDLOUT(GLWSAMP6)
   SEL CKZRRJOB DISP=OLD,DSN=ADMR1.TS2.RRJOB
   SEL CKZRRDSN DISP=OLD,DSN=ADMR1.TS2.RRDSN
   SEL CKZERROR SYSOUT=*
*
```

Figure 9-63 Tablespace clone DD specifications

To make a global change to the HLQ and member names of the allocated data sets used, set your preferences in the Control DD fields, and then, select option D (Set Defaults). You can reuse the same HLQ for different profiles, but it is best to set the Member field to your profile name.

After you leave this panel, a panel appears that provides the option to promote the DD names to the target DD specifications. Generally, you do not allow this to happen. See Figure 9-64 on page 484.

```
CKZ1DDSP            DB2 tablespace clone DD Specification
C EsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssN croll ===> PAGE
  e CKZ1WPMT                                                        e
C e  Do you want to promote source DD cards to target and           e D Specification
L e  report cards?                                                  e
  e                                                                 e
C e  Press ENTER or F3 to continue                                  e
S e  Press F12 to cancel                                            e B-INDEX-REBUI >
  e                                                                 e
C e                                                                 e
  DssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM  HLQ)
  Member . . . . GLWSAMP6                                (control member)

                                                                    Row 17 of 19  -
      DD Name  DD
  SEL CKZRRJOB DISP=OLD,DSN=ADMR1.TS2.RRJOB
  SEL CKZRRDSN DISP=OLD,DSN=ADMR1.TS2.RRDSN
  SEL CKZERROR SYSOUT=*
```

Figure 9-64 Promote source DD to target

We can now edit the SET command.

SET command

The main purpose of the SET command is to specify the source DB2 subsystem and the default SQLID. However, it can override several parameters that were set in the Cloning Tool PARMLIB CKZINI.

```
CKZ1SETC              DB2 tablespace clone SET Command
Command ===>

Commands: A - Set ADVISORY-STATUS-VALUES  R - Set RESTRICT-STATUS-VALUES
          T - Set TEMPLATE-VARIABLES  I - Edit DB2 SSID

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

LOCAL-SSID  . . . . . . . . . . . DBOC      (asterisk to select from list)
ADVISORY-STATUS-VALUES  . . . . . YES       (Yes/No)
DEFAULT-SQLID . . . . . . . . . . DB2AUTH
DB2-COMMAND-RESPONSE-WAIT(secs) . . 60      (0-99999)
DB2-PLAN  . . . . . . . . . . . . CKZPLAN
IP-VERSION6 . . . . . . . . . . . NO        (Yes/No)
MAX-COPYRC  . . . . . . . . . . . 8         (0, 4, or 8)
MAX-RC  . . . . . . . . . . . . . 4         (0, 4)
MAX-SUBTASKS  . . . . . . . . . . 5         (1-99)
SUBTASK-TERMINATION-WAIT  . . . . 60        (0-9999)
MERGE-PRINT . . . . . . . . . . . NO        (Yes/No)
RESTRICT-STATUS-VALUES  . . . . . YES       (Yes/No)
TCPIP-SERVER-PORT . . . . . . . . 5099      (1-65535)
TCPIP-STC-NAME  . . . . . . . . . TCPIP
TEMPLATE-VARIABLE . . . . . . . . NO        (Yes/No)
REMOTE-CONNECT-TYPE . . . . . . .           (CAF, DDF, TCPIP, or blank)
USE-RUNTIME-REPOSITORY  . . . . . YES       (Yes/No)
```

Figure 9-65 Tablespace clone SET Command

In Figure 9-65 on page 484, there are two options here that affect the selection of the source object:

- ADVISORY-STATUS-VALUES
- RESTRICT-STATUS-VALUES

Enter YES in these fields to check the status of table spaces and index spaces before copies are performed. During cloning, if a specified advisory or restricted status is detected, the space is marked mismatched and a warning message is issued.

As long as ALLOW-COPY-ON-MISMATCH(YES) and MAX-RC(4) are in effect, the copy may proceed; otherwise, the copy is not allowed for all affected data sets. To specify values for the ADVISORY-STATUS-VALUES or RESTRICT-STATUS-VALUES parameter, enter A for Set ADVISORY-STATUS-VALUES or R for Set RESTRICT-STATUS-VALUES on theCommands line.

Figure 9-66 shows the restrictive status that we have used to warn us. It will be included in the error report.

```
CKZ1PRSV               Specify Restrict Status Values
Command ===>                                                Scroll ===> PAGE

Commands:      D - Select Defaults  A - Select All  C - Clear All Selections
Line commands: S - Select/Unselect

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

                                                            Row 1 of 16
     Name     Description
  SEL ACHKP    Auxiliary warning advisory
  SEL CHKP     CHECK-pending
      COPY     COPY-pending
  SEL GRECP    Group buffer pool RECOVER-pending
  SEL LPL      Logical page list entries
  SEL RBDP     Index objects that are in REBUILD- or RECOVER-pending
  SEL RECP     RECOVER-pending
  SEL REORP    REORG-pending
      RO       Read-only mode
      STOP     Stopped including STOP, STOPE, STOPP, and LSTOP
      UT       Utility access mode
      UTRO     Utility access and available for read-only access
      UTRW     Utility access and available for read-write access
      UTUT     Utility access and unavailable
      UT*      Any utility access mode
      WEPR     Write error page range
```

Figure 9-66 Restrictive status checked on source

COPY command

There are many options available with the COPY command. The detailed description can be viewed by using the ISPF Help within the dialog.

The COPY command is where we set the target environment. The target must have been defined within the Administrative Function of the dialog. If we use an asterisk (*****) for the TARGET-DB2_SSID, we can select a valid target environment. The other related fields are then populated.

In setting up a clone where we know we might encounter a range of different object types, we need to be able to tolerate the copying of all the objects and then work on the exceptions at the target end. The following object clone types cause mismatches between source and target:

- Identity columns settings
- Number of partitions different
- Define NO

We describe how to manage these object clone types later.

Hopefully, we only get a few of these types of object clones, but we will need to manage them when we have copied the objects. To assist with this situation, we set the option ALLOW-COPY-ON-MISMATCH to YES.

Key settings

The following settings are important:

- TARGET-DB2-SSID. Use an asterisk (*) to select and populate.
- PROCESS-DDL DDL-ENABLE = NO. We will assume that the target is aligned with the source already.
- ALWAYS-COPY-INDEXSPACES = YES. The default is NO, which means that the clone only copies the table space and you will need to rebuild ALL the indexes. When set to YES, for every table space included in a LISTDEF, all index spaces are also included. No LISTDEF INCLUDE INDEXSPACES syntax is required. Our preference is to select YES and use the TARGET-JOB-INDEX-REBUILD-DDN parameter if you are creating a FUZZY clone.
- TARGET-JOB-INDEX-REBUILD-DDN can be used to rebuild all indexes whose tables were affected by data masking or log apply page changes. Any table in the target job that has a page changed via data masking or log apply requires an index to be rebuilt. DDname can only be seven characters because two DD cards will be generated with I and O appended. You can use use job templates for this function.
- AUTO-START-SOURCE-SPACE = YES. This option is ignored in our case because we planning to create a FUZZY clone. That is, we do not want to stop the source system for our clone.
- AUTO-START-TARGET-SPACE = NO. The default is YES. The reason that we recommend leaving the target object stopped after the clone is to have time to verify the integrity of your clone. You will need to manage (start) the target database and fix any exceptions before making it available. By all means, if you are confident that you will consistently have a clean clone, set it to YES. This is obviously the case if the clone is part of the job schedule.
- DATA-MASKING = NO. This option allows for the data to be masked by scrambling or obscuring it from being seen in raw form at the target. This function is not used in our example.
- FUZZY-COPY = YES. If we do not want to stop and start the source objects, setting this option to YES allows an online clone. If you specify YES, ADRDSSU is invoked with TOLERATE(ENQF); IBM RACF® authority for TOLERATE(ENQF) will be required.

Important: This procedure can cause data integrity issues. To assist with this issue, we will use the LOG APPLY option to find a consistent point for the objects. See 9.7.15, “LOG APPLY and FUZZY-COPY” on page 505.

- LONGVAR-COMPATIBILITY = NO. The cloning tool caters for tables created prior to DB2 V9 where LONGVAR was used. Any new columns created are VARCHAR columns. If running a source job where the source objects have LONGVAR and the target objects have corresponding VARCHAR columns, or vice versa, a mismatch will be reported. Setting this value to YES will not report this as a mismatch. The lengths of the corresponding columns must be the same. If not, data might be truncated or a DB2 abend might occur. To be cautious, we set this option to NO and review the mismatches.
- REPLACE-TARGET-DSN = YES. Without this option set to YES, we really are not cloning, so we are not sure when you would use NO, but it is an option. If set to NO, the data sets are copied but the 5th level qualifier is changed to F001 as opposed to I001 or J001.
- RESET-LOGRBA = YES. The LOGRBA will always be reset if there are object ID (OBID) changes to be made. The level IDs in the target VSAM objects are always reset to prevent DB2 down-level rejection of the target VSAM objects. If you specify NO, the DB2 table space or index space may be unusable after the completion of the target job.
- SIM = NO. We described the use of Simulation mode previously. It is an extremely useful option to use when setting up and verifying your clones. We have used this option already to run this clone and will leave it at NO for now. It can be modified when the JCL is generated. Accompanying the use of simulation is setting the parameter for the Data Mover program. Use A to specify one of the following values:
  - If PGM(ADRDSSU), stop target and then source spaces, call ADRDSSU in NORUN mode, start source and target spaces and write out SYNCDB2 commands for the target.
  - If PGM(NONE), validate target table spaces and index spaces and write out SYNCDB2 commands for the target.
- CHECK-DATASET-COMPATIBILITY= NO. Several VSAM attributes should be checked for compatibility between source and target subsystems. In initial runs, it is recommended to run with this as YES and use PGM(NONE) to check data set compatibility. These attributes must be the same between the source and target subsystems. When one or more data set incompatibilities exist, no copies are attempted and the source job ends with RC = 8, regardless of MAX-RC.
- ENABLE Prefetch. Initially set to NO, but if we find the jobs are slow, it is worth exploring this option. The option supports prefetching the clone objects into a cache for faster processing.
- INCLUDE OBJECT-TRANSLATE = YES. The option allows for renaming the target table spaces and index spaces with supplied names. This feature allows you to copy table space and index spaces to the same subsystem. To specify values for the OBJECT-TRANSLATE parameter, enter O for Set OBJECT-TRANSLATE on the Commands line. We are cloning to a different database than the source, and therefore, we will set up the object translation.
- INCLUDE JOB-TEMPLATE = YES. To specify job template data set and DD names, enter J for Set JOB-TEMPLATE on the Commands line. More details about this option are provided in “Option J - Set JOB-TEMPLATE” on page 494. We use the job template to generate Runstats jobs.
- DDNAMES - Simply take the defaults.

In summary, the settings that we use are shown in Figure 9-67 on page 488 and Figure 9-68 on page 489.

```
CKZ1COPC              DB2 tablespace clone COPY Command
Command ===>

Commands: S - Set SOURCE-PREFETCH-DATABASE-LIST  O - Set OBJECT-TRANSLATE
          T - Set TARGET-PREFETCH-DATABASE-LIST  J - Set JOB-TEMPLATE
          D - Set DDL-ATTRIBUTE-CHANGE  L - Set LOG-APPLY  I - Edit DB2 SSID

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >
                                                          More:     +
TARGET-DB2 SSID  . . . . . . . . . . . DBOD      (asterisk to select from list
  LOCATION . . . . . . . . . . . . . . DBOD
  USERID . . . . . . . . . . . . . . .
  PASSWORD . . . . . . . . . . . . . .
  SERVER-IP  . . . . . . . . . . . . .
  SERVER-PORT  . . . . . . . . . . . . 38390
  DEFVCAT  . . . . . . . . . . . . . .
DATA-MOVER PGM . . . . . . . . . . . . ADRDSSU       (ADRDSSU, EMCAPI, or NONE
  FASTREP  . . . . . . . . . . . . . . PREF          (PREF, REQ, or NONE)
  FCTOPPRCPRIMARY  . . . . . . . . . . NO            (Yes, No, PRESMIRREQ,
                                                      PRESMIRPREF, or
                                                      PRESMIRNONE)
  CMDDDNAME  . . . . . . . . . . . . .
PROCESS-DDL DDL-ENABLE . . . . . . . . NO          (Yes/No)
  PROCESS-TYPE . . . . . . . . . . . . YES         (YES, NO, GEN, EXEC, or ALL
  PROCESS-DDL-DDN  . . . . . . . . . . CKZDDBOD
  IGNORE-CREATE-OBJECT-EXISTS  . . . . YES         (Yes/No)
  GENERATE-DDL-DEFAULTS  . . . . . . . NO          (Yes/No)
  INCLUDE DDL-ATTRIBUTE-CHANGE . . . . NO          (Yes/No)
ALLOW-COPY-ON-MISMATCH . . . . . . . . YES         (Yes/No)
ALWAYS-COPY-INDEXSPACES  . . . . . . . YES         (Yes/No)
```

Figure 9-67 COPY command (Part 1 of 2)

```
TARGET-JOB-INDEX-REBUILD-DDN . . . . . IXRBLD
AUTO-START-SOURCE-SPACE  . . . . . YES         (YES, NO, or RESTORE)
AUTO-START-TARGET-SPACE  . . . . . NO          (Yes/No)
AUTO-STOP-TARGET-SPACE . . . . . YES         (Yes/No)
CHECK-INDEX-KEYS . . . . . NO          (Yes/No)
COPY-IF-NO-DB2-TARGET-OBJECTS  . . . . NO          (Yes/No)
DATA-MASKING . . . . . NO          (Yes/No)
DSNS-PER-COPY  . . . . . 255         (1-255)
DSS-COPY-COMMANDS  . . . . . 24          (1-256)
FUZZY-COPY . . . . . YES         (Yes/No)
INCLUDE-ALL-RI . . . . . YES         (Yes/No)
LONGVAR-COMPATIBILITY  . . . . . NO          (Yes/No)
REPLACE-TARGET-DSN . . . . . YES         (Yes/No)
RESET-LOGRBA . . . . . YES         (Yes/No)
SIM  . . . . . NO          (YES, NO, or ALLOC)
V7-MIGRATED-OBJECTS-PRESENT  . . . . . NO          (Yes/No)
CHECK-DATASET-COMPATIBILITY  . . . . . NO          (Yes/No)
IGNORE-RF-MISMATCH-IF-NO-VAR-COLS  . . NO          (Yes/No)
WARN-IF-OBJECT-NOT-TRANSLATED  . . . . YES         (Yes/No)
WARN-ON-INCOMPLETE-RI  . . . . . NO          (Yes/No)
WARN-ON-SIMPLE-TABLESPACE  . . . . . NO          (Yes/No)
CATALOG-PREFETCH ENABLE-PREFETCH . . . NO          (Yes/No)
   ENABLE-SOURCE-PREFETCH . . . . . NO           (Yes/No)
   ENABLE-TARGET-PREFETCH . . . . . NO           (Yes/No)
SYNCDB2-DDN  . . . . . CKZSDBOD
DATASETS-TO-COPY-DDN . . . . . CKZCDBOD
XMLSTRING-DDN  . . . . . CKZMDBOD
SQLOUT-DDN . . . . . CKZQDBOD
STOP-TARGET-DDN  . . . . . CKZWDBOD
STOP-SOURCE-DDN  . . . . . CKZXDBOD
START-SOURCE-DDN . . . . . CKZYDBOD
IDCAMS-DDN . . . . . CKZZDBOD

INCLUDE OBJECT-TRANSLATE . . . . . YES         (Yes/No)
INCLUDE JOB-TEMPLATE . . . . . YES         (Yes/No)
USE CURRENT DDNAMES  . . . . . NO          (Yes/No)
```

Figure 9-68 COPY command (Part 2 of 2)

Option S - Set SOURCE-PREFETCH-DATABASE-LIST

On Figure 9-67 on page 488, when entered, the list of target databases to be cached is generated from the source data set names mapped to target names using object translate. When copying a large number of table spaces, compare source job run times with and without this database list to determine which one gives the best performance. See Figure 9-69 on page 490.

```
CKZ1PDBL             Specify Source Prefetch Databases
Command ===>                                              Scroll ===> PAGE

Commands:      A - Add Line
Line commands: D - Delete Line

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

                                                                Row 1 of 3
Cmd Source Database
    GLWSAMP
```

Figure 9-69 Specify Source Prefetch Databases

Option T - Set TARGET-PREFETCH-DATABASE-LIST

On Figure 9-67 on page 488, target databases can be left empty because they will use the source databases generated from the LISTDEF after the object translation. We are renaming the database; therefore, the target database is listed. See Figure 9-70.

```
CKZ1PDBL             Specify Target Prefetch Databases
Command ===>                                              Scroll ===> PAGE

Commands:      A - Add Line
Line commands: D - Delete Line

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

                                                                Row 1 of 3
Cmd Target Database
    GLWSAMP2
```

Figure 9-70 Specify Target Prefetch Databases

Option O - Set OBJECT-TRANSLATE

On Figure 9-67 on page 488, if the object names are different between environments, object translation is entered here. This makes it possible to clone within the same subsystem, effectively a migration. See Figure 9-71 on page 491.

Wildcarding is supported:

- The percent sign (%) or asterisk (*) represents *n* characters.
- The underscore (_) or question mark (?) represents a single character.
- Use the question mark (?) rather than the underscore (_) for creator, table, and index names, because the underscore is a valid character for these three object names.

Note: Be careful with your translation so that you do not rename to existing objects if you intend to build the target DDL. You might see messages similar to the messages that are in Example 9-38 on page 491. This is a good case for running in Simulation or PGM (NONE) to verify the DDL creation and copy. You can also use PROCESS-TYPE(G) and PROCESS-TYPE (A) with PGM(NONE). Our preference is to sort out the source and target structures before you create the clone.

Example 9-38 Error with source job due to object translation

```
CKZ99213E Tablespace TESTDB3.TESTTS3 already exists
CKZ9904E Nodes initialization failed!
CKZ76703E DDL Generator Completed, with Error(s), RC=X'0000000
CKZ50004E Discovery Phase has Failed
CKZ50012E Completed, with Errors, RC=X'00000008, RS=X'00000000
```

```
CKZ1FJTC            DB2 tablespace OBJECT-TRANSLATE Command
Command ===>                                                 Scroll ===> PAGE

Commands:      A - Add Line
Line commands: D - Delete Line

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

Object Types : CR - Creator  DB - Database  TS - Tablespace  IX - Index
               IS - Indexspace  TB - Table  VC - VCAT
                                                              Row 1 of 3
Cmd OBJTYPE SOURCE NAME                      TARGET NAME
    DB      GLWSAMP                          GLWSAMP2
    CR      GLWSAMP                          GLWSAMPC
    CR      DB2R2                            DB2AUTH
```

Figure 9-71 Object Translate command

Option D - Set DDL-ATTRIBUTE-CHANGE

On Figure 9-67 on page 488, select option D - Set DDL-ATTRIBUTE-CHANGE to set up default DDL attributes for the target. Because we are not managing the DDL by using the Cloning Tool this time, it will not be used. For illustration only, Figure 9-72 on page 492 has an example.

```
CKZ1PDAC          DB2 tablespace DDL-ATTRIBUTE-CHANGE Command
Command ===>                                              Scroll == => PAGE

Commands:      C - Create
Line Commands: D - Delete  E - Edit  V - View  R - Repeat

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP

                                                                  Row 1 of 4
  Attribute Name Source Value       Target Value       ATT Apply To Object
  PRIQTY                            7200               TP
  SECQTY                            7200               TP
  PRIQTY                            720                IP
  SECQTY                            7200               IP
```

Figure 9-72 DDL attribute changes

In Figure 9-73, using the Edit line command shows the scope of the attributes that can be managed. A full list of options is provided in F1 Help or in the *DB2 Cloning Tool for z/OS, V3.1, User's Guide*, SC19-3493-01.

```
CKZ1EDAC               Edit DDL-ATTRIBUTE-CHANGE Command
Option ===>

Creator . . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option  . : UPDATE       Description . : GLWSAMP

Attribute Name  . . SECQTY        (STOGROUP, GBPCACHE, LOG, PRIQTY, SECQTY,
                                   TRACKMOD, CLOSE, DATACAPTURE, or BUFFERPOOL
Source Value  . . .                                                      >
Target Value  . . . -1                                                   >
Apply To Type . . .               (DATABASE, TABLESPACE, TABLEPART, TABLE,
                                   INDEX, INDEXPART, or blank)
Apply To Object . .                                                      >
```

Figure 9-73 DDL attributes

Option L - Set LOG-APPLY

On Figure 9-67 on page 488, we now set up logging options. Logging options allow us to clone to a consistent point, primarily to be used with FUZZY- COPY where the source objects are not stopped when the copy is taken.

This panel allows you to enter settings for the LOG-APPLY command. This feature allows log records written by DB2 from before the copies in the source job until the target job is run to be applied to DB2 pages being updated in the target job. See Figure 9-74 on page 493.

We will use the option to obtain a consistent point and then update the Mini Log details.

If you expect to work with a large number of logs, you can manage the size of the files through these panels. Select option S, M, and W. For more information about the LOG-APPLY command setup, see 9.7.15, "LOG APPLY and FUZZY-COPY" on page 505.

```
CKZ1PLAP               DB2 tablespace LOG-APPLY Command
Command ===>                                                Scroll ===> PAGE

Commands: M - Set MINILOG  S - Set SORTFILE  W - Set WORKFILE

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

LA-ENABLE . . . . . . . . . . YES       (Yes/No)
QUIESCE-POINT . . . . . . . . NO        (Yes/No)
COMMON-CONSISTENT-POINT . . . YES       (Yes/No)
WARN-IF-SKIP-QUIESCE  . . . . NO        (Yes/No)
NUMBER-OF-BUFFERS . . . . . . 5         (0-99)
NUMBER-OF-CHANNEL-PROGRAM . . 1         (0-99)
SORT-PROGRAM  . . . . . . . . DFSORT    (DFSORT, SYNCSORT)
ZPARM-MEMBER  . . . . . . . . : DSNZPARM

DATA-SHARING-MEMBERS:
SSID ZPARM    BSDS01 DSN      BSDS02 DSN
```

Figure 9-74 Log Apply

Option S - Set SORTFILE

On Figure 9-74, set LARGE-FILE-TYPE to YES to specify that dynamic allocation of the sort file data set should include the LARGE attribute. This allows for data sets to exceed 65,535 tracks. See Figure 9-75.

```
CKZ1PSRT                    Set SORTFILE options
Option ===>

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

SORTFILE-LARGE-FILE-TYPE  . . . . YES       (Yes/No)
SORTFILE-UNIT-TYPE  . . . . . . . SYSALLDA
SORTFILE-QUANTITY-IN-TRACKS . . . NO        (Yes/No)
SORTFILE-PRIMARY-QUANTITY . . . . 250       (1-9999)
SORTFILE-SECONDARY-QUANTITY . . . 250       (1-9999)
SORTFILE-VOLUME-COUNT . . . . . . 1         (1-255)
SORTFILE-DATACLAS . . . . . . . .
SORTFILE-STORCLAS . . . . . . . .
SORTFILE-MGMTCLAS . . . . . . . .
```

Figure 9-75 SORTFILE

The same applies to the Workfiles and the Mini Logs.

Option M - Set MINILOG

After selecting M on Figure 9-74, the Set MINILOG options panel (Figure 9-76 on page 494) is displayed. On the Set MINILOG options panel, we need to define the HLQ that will be used to store the log extracts that are applied during log processing on the target system.

```
CKZ1PMLG                    Set MINILOG options
Option ===>

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

MINILOG HLQ  . . . . . . . . . . ADMR1.TS2.GLW.MINILOG
SPACES-PER-MINILOG . . . . . . . 30        (1-999)
MINILOG-LARGE-FILE-TYPE  . . . . YES       (Yes/No)
MINILOG-UNIT-TYPE  . . . . . . . SYSALLDA
MINILOG-QUANTITY-IN-TRACKS . . . NO        (Yes/No)
MINILOG-PRIMARY-QUANTITY . . . . 250       (1-9999)
MINILOG-SECONDARY-QUANTITY . . . 250       (1-9999)
MINILOG-VOLUME-COUNT . . . . . . 1         (1-255)
MINILOG-DATACLAS . . . . . . . .
MINILOG-STORCLAS . . . . . . . .
MINILOG-MGMTCLAS . . . . . . . .
```

Figure 9-76 MINILOG options

Option J - Set JOB-TEMPLATE

After selecting J on Figure 9-67 on page 488, Figure 9-77, the Specify Job Template Data Sets and Members panel, is displayed. We set up the use of a job template that will generate JCL for running Runstats. The templates and JCL are stored in existing PDSs with the specified member names. More than one template can be used. The DDNAMES need to be eight characters in length.

For more details about job templates, see 9.8.9, “Job templates” on page 515.

```
CKZ1PJTS          Specify Job Template Data Sets and Members
Command ===>                                                Scroll ===> PAGE

Commands:      A - Add Line
Line commands: D - Delete Line

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

TEMPLATE INPUT DATA SET  . . ADMR1.CLONE.SKELJCL
TEMPLATE OUTPUT DATA SET . . ADMR1.CLONE.JOBS

INPUT DD DISP . . SHR    OUTPUT DD DISP . . OLD
                                                             Row 1 of 3
Cmd INPUT MEMBER INPUT DDNAME OUTPUT MEMBER OUTPUT DDNAME
    CKZJOBRS     RSTATIA6     RSTAT1J       RSTATOA6
```

Figure 9-77 Specify Job Template Data Sets and Members panel

TARGET-JOB-INDEX-REBUILD-DDN: There is no panel to define templates for the index rebuild at this stage. After the JCL is generated, you will need to supply the appropriate template input and output dataset.

This completes the COPY command options that we need. We now end these panels to return to the DB2 Setup Source panel.

Option 5 HLQDDDF Command

This option is generally only required where non-SMS data sets are used. Our case does not require this option.

The panel in Figure 9-78 allows you to enter parameters for the HLQDDDF command. The HLQDDDF command is optional. It is used in table space cloning to pass input and output DDs to ADRDSSU. This option might be useful to pass VOLSERs to ADRDSSU for non-SMS managed volumes.

```
CKZ1HLQC               DB2 tablespace HLQDDDF Command
Command ===>                                                Scroll ===> PAGE

Commands:      A - Add Line
Line commands: D - Delete Line  E - Edit HLQ DD lines  V - View HLQ DD lines

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 1
Share Option . : UPDATE       Description . : SMALL CLONE

                                                                    Row 1 of 3
Cmd HLQNAME  DIR DD Names
             OUT
             OUT
             OUT
```

Figure 9-78 HLQDDDF command

Option 7 LISTDEF command

We set the scope of our clone by using the LISTDEF command. We have chosen to clone a single database, but the scope can certainly be expanded, as needed.

As the LISTDEF statements select objects to be processed, they also select DDL to be generated if the DDL generation is enabled. See Figure 9-79.

```
CKZ1PLDC           DB2 Tablespace Clone LISTDEF Commands
Command ===>                                              Scroll ===> PAGE

Commands:      C - Create
Line Commands: D - Delete  E - Edit  V - View  R - Repeat

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

List Name  . . . TS
                                                          Row 1 of 1    >

  Incl Type      Obj  Object Specification     Object Specification
  Excl Spec Copy Type Qualifier 1              Qualifier 2              PLevel
  I    TS        DB   GLWSAMP
```

Figure 9-79 Clone LISTDEF commands

In our examples, we chose to include the whole database. Another option that might be useful is to include objects within a STOGROUP.

When you use STOGROUP, all objects within a DB2 storage group can be selected with a single LISTDEF STOGROUP statement, for example:

```
INCLUDE TABLESPACES STOGROUP STOXYZ ALL
```

Internally, this LISTDEF statement is replaced by one statement for each database in the DB2 STOGROUP named STOXYZ.

The object selection options that we used are shown in Figure 9-80.

```
CKZ1ELDC                    Edit LISTDEF Command
Option ===>

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI

Include/Exclude  . . . . . . . . . . INCLUDE     (INCLUDE, EXCLUDE)
Type Specification . . . . . . . . . TABLESPACE  (TABLESPACE, INDEXSPACE)
Copy . . . . . . . . . . . . . . . . : NO        (Yes/No)
Object Type  . . . . . . . . . . . . DATABASE    (DATABASE, TABLESPACE,
                                                  INDEXSPACE, TABLE, INDEX, o
                                                  STOGROUP)
Object Specification Qualifier 1 . . GLWSAMP                              >
Object Specification Qualifier 2 . .                                      >
Partlevel  . . . . . . . . . . . . .
RI . . . . . . . . . . . . . . . . . NO          (Yes/No)
LOB Indicator Keywords . . . . . . . ALL         (ALL, LOB, BASE, XML, or
                                                  blank)
Cloned . . . . . . . . . . . . . . .             (Yes, No, or blank)
```

Figure 9-80 Tablespace cloning LISTDEF

In Figure 9-80, Type Specification = INDEXSPACE will cause indexes to be included in the COPY operation. ALWAYS-COPY-INDEXSPACES Y causes all indexes to be included whether referenced in the LISTDEF or not. It also causes indexes specifically excluded via the LISTDEF to be included. We chose to use ALWAYS-COPY-INDEXSPACES Y.

The cloning tool manages LOBs and clone tables. Their exclusion or inclusion is set up here.

LOB indicator keywords

In Figure 9-80, specify one of the following LOB indicator keywords:

- Enter ALL to select base, LOB, and XML table spaces.
- Enter BASE to select only the base table.
- Enter LOB to select only LOB table spaces.
- Enter XML to select only XML table spaces.

Cloned tables

In Figure 9-80, for Cloned, specify Yes to indicate that ONLY table spaces and index spaces that contain cloned tables are to be selected. The base table is always included with the clone table.

Specify No to indicate that only table spaces and index spaces that do not contain clone tables are to be selected.

Leave the field blank and all objects are included.

Masking

The DB2 Tablespace Clone MASKDEF Commands panel allows you to specify how data masking is applied during the copy. Data copied from a source object to a target object can be modified so that the target data in one or more columns might be different from the source data. The changes are made based on masking rules that are enabled during the copy.

This panel offers some basic masking. After building a test database environment with sanitized data, maybe the use of an alternative product, such as Optim Test Database Generator, is worth considering for this scenario.

An example of setting a mask is shown in Figure 9-81. This example is using the keyword `scramble`, which will scramble the data in column REQUIREMENT. A full description and detailed examples are in the *DB2 Cloning Tool for z/OS, V3.1, User's Guide*, SC19-3493-01.

```
CKZ1PMDC            DB2 Tablespace Clone MASKDEF Commands
Command ===>                                              Scroll ===> PAGE

Commands:      C - Create
Line Commands: D - Delete  E - Edit  V - View  R - Repeat

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 3
Share Option . : UPDATE       Description . : GLWSAMP

                                                                  Row 1 of 1
  Table Creator Table Name         Column Name        Mask Rule
  H89OPAY       PS_AA_OVRD_WHERE   REQUIREMENT        scramble
```

Figure 9-81 An example of data masking

9.7.11 Running the clone

We have now provided all the detail, and we are ready to build our table space clone job, as shown in Figure 9-82.

```
CKZ1DTCL             DB2 Tablespace Clone Profile Display
Command ===>                                              Scroll ===> PAGE

Commands:      C - Create
Line Commands: B - Build  D - Delete  E - Edit  R - Rename  V - View  C - Copy

Profile Like . . . *
Creator Like . . . *
                                                          Row 1 of 15    >
                                        Share
Cmd Name                          Creator  Option    Description
    TABLESPACE 3                  ADMR1    UPDATE    GLWSAMP
    TABLESPACE 4                  ADMR1    UPDATE    GLWSAMPTS3 AFTER FIXES
    TABLESPACE 5 FUZZY            ADMR1    UPDATE    GLWSAMP
b   TABLESPACE 6                  ADMR1    UPDATE    GLWSAMP TARGET-JOB-INDEX
    TEST1                         ADMR1    UPDATE    TESTDB
    TEST2 CL                      ADMR1    UPDATE    TESTDB WITH CLONE TABLE
    TS PS 1                       ADMR1    UPDATE    CLONE MANY TS
    XXXXXXXXXXXX                  ADMR1    UPDATE    SMALL CLONE
```

Figure 9-82 Build clone JCL

We select option 1 Generate Source and Target Jobs, as shown in Figure 9-83.

```
CKZ1BDCJ               Build DB2 tablespace clone jobs
Option ===> 1

1  Generate Source and Target Jobs
2  Generate Report Job
3  Generate TCPIP Server Job
```

Figure 9-83 Build DB2 tablespace clone jobs option 1

On the next panel (Figure 9-84), we provide the library and member names to which to save our JCL. Be sure to provide unique member names if you are sharing the PDS with other cloning profiles.

```
CKZ1GFSJ              Generate Source and Target Jobs
Option  ===>

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

Data set name  . . . . . . ADMR1.CKZ.TS2JCLLB
Source member name . . . . CLRCE
Target member name . . . . CLTRGT

Processing options
  Enter "/" to select option
  /  Review Source Job
  /  Review Target Job
     Warn if jobs, LISTDEF, or MASKDEF already exist
     Warn if jobs, LISTDEF, or MASKDEF were edited outside the panels
```

Figure 9-84 Generate Source and Target Jobs

9.7.12 Source job

Example 9-39 shows the JCL with the COPY and SET command syntax that we generated. Example 9-40 on page 499 shows the SET command for the source table space clone. Example 9-41 on page 499 shows the COPY command for the table space clone.

Before we run this job, we need to ensure that we have created the data sets provided in the Tools Customizer customization library.

Running the source job will require the objects to be stopped if we had not chosen FUZZY-COPY. Even though we do not need to stop the objects, it makes sense to still look for a quiet time to run these jobs. See 9.7.15, “LOG APPLY and FUZZY-COPY” on page 505.

Example 9-39 Source JCL for COPY and SET

```
//ADMR1I JOB (999,POK),'TS CLONE 6',
//         REGION=0M,NOTIFY=&SYSUID,MSGCLASS=X,CLASS=T
//PROCLIB JCLLIB ORDER=DBOAM.PROCLIB
/*JOBPARM SYSAFF=SC63
//*************************************************************
/*JOBPARM S=SC63
```

```
//S1      EXEC PGM=CKZ00500,REGION=0M
//STEPLIB  DD  DISP=SHR,DSN=DBTLSP.SCKZLOAD
//         DD  DISP=SHR,DSN=DBOCT.SDSNLOAD
//CKZINI   DD  DISP=SHR,DSN=DBTLSP.SCKZPARM(CKZINI)
//CKZLOG   DD  SYSOUT=*
//CKZPRINT DD  SYSOUT=*
//CKZLSTDF DD  DISP=SHR,DSN=ADMR1.TS2.LISTDEF(GLWSAMP6)
//CKZCRXML DD  DISP=SHR,DSN=ADMR1.TS2.XMLCRDDL(GLWSAMP6)
//CKZSDBOD DD  DISP=OLD,DSN=ADMR1.TS2.SYNCDB2(GLWSAMP6)
//CKZMDBOD DD  DISP=OLD,DSN=ADMR1.TS2.XMLSTR(GLWSAMP6)
//CKZQDBOD DD  DISP=OLD,DSN=ADMR1.TS2.SQLOUT(GLWSAMP6)
//CKZRRJOB DD  DISP=OLD,DSN=ADMR1.TS2.RRJOB
//CKZRRDSN DD  DISP=OLD,DSN=ADMR1.TS2.RRDSN
//CKZERROR DD  SYSOUT=*
//BSDS01   DD  DISP=SHR,DSN=DBOCB.BSDS01
//BSDS02   DD  DISP=SHR,DSN=DBOCB.BSDS02
//RSTATIA6 DD  DISP=SHR,DSN=ADMR1.CLONE.SKELJCL(CKZJOBRS)
//RSTATOA6 DD  DISP=OLD,DSN=ADMR1.CLONE.JOBS(RSTAT1J)
//CKZIN    DD  *
```

Example 9-40 SET command for the source table space clone

```
SET                 -
          LOCAL-SSID(DBOC) -
          DEFAULT-SQLID(DB2AUTH) -
          TCPIP-SERVER-PORT(5099) -
          TCPIP-STC-NAME(TCPIP) -
          MAX-RC(4) -
          MAX-COPY-RC(8) -
          DB2-COMMAND-RESPONSE-WAIT(60) -
          DB2-PLAN(CKZPLAN) -
          IP-VERSION6(N) -
          MAX-SUBTASKS(5) -
          SUBTASK-TERMINATION-WAIT(60) -
          ADVISORY-STATUS-VALUES(AUXW,ARBDP,AREO*,AREOR) -
          RESTRICT-STATUS-VALUES(ACHKP,CHKP,GRECP,LPL,RBDP,RECP,REORP) -
          MERGE-PRINT(N) -
          USE-RUNTIME-REPOSITORY(Y)
```

Example 9-41 COPY command for the table space clone

```
COPY              -
         TARGET-DB2(SSID(DBOD)  -
              LOCATION(DBOD) -
              SERVER-PORT(38390) -
         ) -
         LOG-APPLY -
         ( -
              LA-ENABLE(Y) -
              SPACES-PER-MINILOG(30) -
              MINILOG-HLQ(ADMR1.TS2.GLW.MINILOG) -
              QUIESCE-POINT(N) -
              COMMON-CONSISTENT-POINT(Y) -
              WARN-IF-SKIP-QUIESCE(N) -
              WORKFILE-LARGE-FILE-TYPE(Y) -
              WORKFILE-UNIT-TYPE(SYSALLDA) -
```

```
                    WORKFILE-QUANTITY-IN-TRACKS(N) -
                    WORKFILE-PRIMARY-QUANTITY(250) -
                    WORKFILE-SECONDARY-QUANTITY(250) -
                    WORKFILE-VOLUME-COUNT(1) -
                    SORTFILE-LARGE-FILE-TYPE(Y) -
                    SORTFILE-UNIT-TYPE(SYSALLDA) -
                    SORTFILE-QUANTITY-IN-TRACKS(N) -
                    SORTFILE-PRIMARY-QUANTITY(250) -
                    SORTFILE-SECONDARY-QUANTITY(250) -
                    SORTFILE-VOLUME-COUNT(1) -
                    MINILOG-LARGE-FILE-TYPE(N) -
                    MINILOG-UNIT-TYPE(SYSALLDA) -
                    MINILOG-QUANTITY-IN-TRACKS(N) -
                    MINILOG-PRIMARY-QUANTITY(250) -
                    MINILOG-SECONDARY-QUANTITY(250) -
                    MINILOG-VOLUME-COUNT(1) -
                    NUMBER-OF-BUFFERS(5) -
                    NUMBER-OF-CHANNEL-PROGRAMS(1) -
                    SORT-PROGRAM(DFSORT) -
                    ZPARM-MEMBER(DSNZPARM) -
                ) -
                DATA-MOVER(PGM(ADRDSSU) -
                    FASTREP(PREF) -
                ) -
                ALLOW-COPY-ON-MISMATCH(Y) -
                ALWAYS-COPY-INDEXSPACES(Y) -
                TARGET-JOB-INDEX-REBUILD-DDN(IXRBLD) -
                AUTO-START-SOURCE-SPACE(Y) -
                AUTO-START-TARGET-SPACE(N) -
                AUTO-STOP-TARGET-SPACE(Y) -
                CHECK-INDEX-KEYS(N) -
                COPY-IF-NO-DB2-TARGET-OBJECTS(N) -
                DATA-MASKING(N) -
                DSNS-PER-COPY(255) -
                DSS-COPY-COMMANDS(24) -
                FUZZY-COPY(Y) -
                INCLUDE-ALL-RI(Y) -
                REPLACE-TARGET-DSN(Y) -
                LONGVAR-COMPATIBILITY(N) -
                RESET-LOGRBA(Y) -
                SIM(N) -
                V7-MIGRATED-OBJECTS-PRESENT(N) -
                CHECK-DATASET-COMPATIBILITY(N) -
                IGNORE-RF-MISMATCH-IF-NO-VAR-COLS(N) -
                WARN-IF-OBJECT-NOT-TRANSLATED(Y) -
                WARN-ON-INCOMPLETE-RI(N) -
                WARN-ON-SIMPLE-TABLESPACE(N) -
                SYNCDB2-DDN(CKZSDBOD) -
                SQLOUT-DDN(CKZQDBOD) -
                XMLSTRING-DDN(CKZMDBOD) -
                OBJECT-TRANSLATE( -
                    DATABASE,GLWSAMP,GLWSAMP2-
                    CREATOR,GLWSAMP,GLWSAMPC-
                    CREATOR,DB2R2,DB2AUTH-
                                )  -
                JOB-TEMPLATE( -
                    RSTATIA6,RSTATOA6 -
                            )
//*
```

Verify the source job

Verify the source job by changing SIM(N) to SIM(Y) and DATA-MOVER(PGM(NONE)). Using this option with PGM(NONE) will validate target table spaces and index spaces and write out SYNCDB2 commands for the target. The LISTDEF will get source table spaces and index spaces and then stop.

Snapshots of the job output

If we look at the output of the source job, we have three reports:

1. CKZLOG displays LISTDEF processing, DB2 commands issued by DB2 Cloning Tool Table Space Cloning, and responses/results of the commands.
2. CKZPRINT displays CKZINI tokens, control parameters, data set names and associated DB2 table spaces and index spaces, DB2 start and stop space command status, and DFSMSdss program ADRDSSU commands and status. Example 9-42 shows the Copy Completion Status Report. The **DBODD.DSNDBC.GLWSAMP2.GLWSEPA.F0001.AO05** target data set uses F0001 because there was no target partition for the source dataset. See 9.8.2, “Partition-by-growth (PBG) table spaces” on page 508.

Example 9-42 CKZPRINT extract

```
CKZ54825I - Begin COPY Completion Status Report
```

SOURCE DATASET	TARGET DATASET	RC	SPTAYCPEE	CLONE	REPLDSN	TRGOBJ	XLATE DTIXXBSSCN
DBOCD.DSNDBC.GLWSAMP.GLWSDPT.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWSDPT.IO001.AO01	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWSEMP.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.AO01	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWSEMP.IO001.AO02	DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.AO02	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWSEMP.IO001.AO03	DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.AO03	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWSEMP.IO001.AO04	DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.AO04	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWSEPA.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.AO01	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWSEPA.IO001.AO02	DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.AO02	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWSEPA.IO001.AO03	DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.AO03	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWSEPA.IO001.AO04	DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.AO04	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWSEPA.IO001.AO05	**DBODD.DSNDBC.GLWSAMP2.GLWSEPA.FO001.AO05**	**0**	**TS**		**N**	**Y**	+
DBOCD.DSNDBC.GLWSAMP.GLWSPJA.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWSPJA.IO001.AO01	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWSPRJ.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWSPRJ.IO001.AO01	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWSSPL.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.AO01	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWSSPL.IO001.AO02	DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.AO02	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWSSPL.IO001.AO03	DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.AO03	0	TS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.GLWXLNM1.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWXLNM1.IO001.AO01	0	IS		Y	Y	+ +
DBOCD.DSNDBC.GLWSAMP.GLWXPGW1.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWXPGW1.IO001.AO01	0	IS		Y	Y	+ +
DBOCD.DSNDBC.GLWSAMP.GLWXPJA1.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWXPJA1.IO001.AO01	0	IS		Y	Y	+ +
DBOCD.DSNDBC.GLWSAMP.GLWXPNG1.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWXPNG1.IO001.AO01	0	IS		Y	Y	+ +
DBOCD.DSNDBC.GLWSAMP.GLWXPRJ1.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWXPRJ1.IO001.AO01	0	IS		Y	Y	+ +
DBOCD.DSNDBC.GLWSAMP.GLWXSFN1.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWXSFN1.IO001.AO01	0	IS		Y	Y	+ +
DBOCD.DSNDBC.GLWSAMP.GLWXSQL1.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWXSQL1.IO001.AO01	0	IS		Y	Y	+ +
DBOCD.DSNDBC.GLWSAMP.GLWXSTR1.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWXSTR1.IO001.AO01	0	IS		Y	Y	+ +
DBOCD.DSNDBC.GLWSAMP.GLWXTWN1.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWXTWN1.IO001.AO01	0	IS		Y	Y	+ +
DBOCD.DSNDBC.GLWSAMP.GLWXVRN1.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.GLWXVRN1.IO001.AO01	0	IS		Y	Y	+ +
DBOCD.DSNDBC.GLWSAMP.IRDOCIDG.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.IRDOCIDG.IO001.AO01	0	IS		Y	Y	+ +
DBOCD.DSNDBC.GLWSAMP.IRNODEID.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.IRNODEID.IO001.AO01	0	IS		Y	Y	+ +
DBOCD.DSNDBC.GLWSAMP.XGLWOOOO.IO001.AO01	DBODD.DSNDBC.GLWSAMP2.XGLWOOOO.IO001.AO01	0	XS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.XGLWOOOO.IO001.AO02	DBODD.DSNDBC.GLWSAMP2.XGLWOOOO.IO001.AO02	0	XS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.XGLWOOOO.IO001.AO03	DBODD.DSNDBC.GLWSAMP2.XGLWOOOO.IO001.AO03	0	XS		Y	Y	+
DBOCD.DSNDBC.GLWSAMP.XGLWOOOO.IO001.AO04	DBODD.DSNDBC.GLWSAMP2.XGLWOOOO.IO001.AO04	0	XS		Y	Y	+

```
CKZ54826I - End COPY Completion Status Report
```

3. CKZERROR. When CKZERROR is included in source, target, and TCP/IP server jobs, all warning and error messages are output to this DD, as well as to CKZPRINT. See Example 9-43.

Example 9-43 CKZERROR report

```
13066 18:28:23.09 CKZ54714W TS Dataset DBODD.DSNDBC.GLWSAMP2.GLWSEPA.*OOO1.AOO5 is Not Cataloged on the Target System and the Object
Exists, will be Copied to the Target Subsystem using CKZ 5th Level Qualifier, FOOO1
 13066 18:28:23.62 CKZ54719W IX GLWSAMP.GLWXEPA1 Status is PSRBD,UTRW, Subsystem DBOC
 13066 18:28:23.62 CKZ54719W IX GLWSAMP.GLWXEPA1 Status is PSRBD,UTRW, Subsystem DBOC
 13066 18:28:23.62 CKZ54773W TABLE COLUMN Mismatch, Attribute START, DBOC.GLWSAMP.GLWTLCN.LOC_NO = 1 and DBOD.GLWSAMPC.GLWTLCN.LOC_NO =
-32768
 13066 18:28:23.62 CKZ54773W TABLE COLUMN Mismatch, Attribute START, DBOC.GLWSAMP.GLWTLNM.LN_NO = 1 and DBOD.GLWSAMPC.GLWTLNM.LN_NO =
-32768
 13066 18:28:23.62 CKZ54773W TABLE COLUMN Mismatch, Attribute START, DBOC.GLWSAMP.GLWTSFN.SFN_NO = 1 and DBOD.GLWSAMPC.GLWTSFN.SFN_NO =
-32768
 13066 18:28:23.62 CKZ54773W TABLE COLUMN Mismatch, Attribute START, DBOC.GLWSAMP.GLWTSTR.STRT_NO = 1 and DBOD.GLWSAMPC.GLWTSTR.STRT_NO =
-32768
 13066 18:28:23.62 CKZ54773W TABLE COLUMN Mismatch, Attribute START, DBOC.GLWSAMP.GLWTTWN.TOWN_NO = 1 and DBOD.GLWSAMPC.GLWTTWN.TOWN_NO =
-32768
 13066 18:28:23.62 CKZ54712W Dataset DBOCD.DSNDBC.GLWSAMP.GLWSOO1.IOOO1.AOO1 has Failed Object Match Checking, Will Be Copied Per
ALLOW-COPY-ON-MISMATCH(Y) to Subsystem DBOD
 13066 18:28:23.62 CKZ54763W TABLESPACE Mismatch, Attribute PARTITIONS, DBOC.GLWSAMP.GLWSEPA = 5 and DBOD.GLWSAMP2.GLWSEPA = 4
 13066 18:28:23.62 CKZ54724W Dataset DBOCD.DSNDBC.GLWSAMP.GLWXEPA1.IOOO1.AOO1 has Failed Status Checking, Will Be Copied Per
ALLOW-COPY-ON-MISMATCH(Y) to Subsystem DBOD
 13066 18:35:02.51 CKZ54557W Target Table DBOD.GLWSAMPC.GLWTDNG has an Identity Column DEPT_NO with a non Zero Cache Value.  This MAY Result
in Insert or Update Failures due to Duplicate Identity Column Values, as the MAXASSIGNEDVALs are the Same on the Source and Target Tables
 13066 18:35:02.51 CKZ54557W Target Table DBOD.GLWSAMPC.GLWTENG has an Identity Column EMP_NO with a non Zero Cache Value.  This MAY Result
in Insert or Update Failures due to Duplicate Identity Column Values, as the MAXASSIGNEDVALs are the Same on the Source and Target Tables
 13066 18:35:02.51 CKZ54557W Target Table DBOD.GLWSAMPC.GLWTLNG has an Identity Column TRAN_NO with a non Zero Cache Value.  This MAY Result
in Insert or Update Failures due to Duplicate Identity Column Values, as the MAXASSIGNEDVALs are the Same on the Source and Target Tables
 13066 18:35:02.51 CKZ54557W Target Table DBOD.GLWSAMPC.GLWTPNG has an Identity Column PROJ_NO with a non Zero Cache Value.  This MAY Result
in Insert or Update Failures due to Duplicate Identity Column Values, as the MAXASSIGNEDVALs are the Same on the Source and Target Tables
 13066 18:35:02.52 CKZ50005W Discovery Phase Ended with Warning(s)
 13066 18:35:03.52 CKZ50013W Completed, with Warnings,
RC=X'00000004',RS=X'00000000
```

Object mismatch

Mismatch messages are shown in the CKZERROR report in Example 9-43. Review this report after the copy job is completed for every clone. Ensure that you interpret the severity of the messages.

We look at a mismatch message with identity columns. See Example 9-44.

Example 9-44 Example mismatch message

```
CKZ54773W TABLE COLUMN Mismatch, Attribute START, DBOC.GLWSAMP.GLWTLCN.LOC_NO = 1 and
DBOD.GLWSAMPC.GLWTLCN.LOC_NO = -32768
CKZ54712W Dataset DBOCD.DSNDBC.GLWSAMP.GLWSOO1.IOOO1.AOO1 has Failed Object Match Checking,
Will Be Copied Per ALLOW-COPY-ON-MISMATCH(Y) to Subsystem DBOD
```

These messages are associated with columns that use identity columns. In these cases, the message indicates that the start values are different. If we look at the target catalog entry of one of these messages in Example 9-45, we see there is a difference with the start values. This difference occurred at creation of the column and when the Minvalue was set.

Example 9-45 Identity column details on the target

```
Details for Identity Column: LOC_NO
Schema name       : GLWSAMPC              Type of entry : I - Identity column
Owner name        : GLWSAMPC              Attributes    : GENERATED BY DEFAULT
Created in DB2 Ver: 0                     Owner type    : Auth ID
Table name        : GLWTLCN               Data type ID  : 500 - SMALLINT
DB2 name generated: SEQDPB1DXWHNWBM       Data source ID: 0
Sequence ID       : 486                   Created by    : ADMR1
Start value       : -32768
Increment by      : 1
Restart with      : 61
Cache             : 20                  Created TS: 2012-05-07-15.06.29.018142
Order             : NO                  Altered TS: 2012-07-16-01.03.14.851729
```

```
Max assigned value: ? - Not specified
Max value allowed : 32767
Min value allowed : -32768
Cycle             : NO
```

In all cases, unless Cycle is used, these messages can be ignored because the RESTART value is modified by the target job. The target job is passed SQL to process via the SQLOUT file. Included in this are the ALTERs for the identity columns.

The results of the SQL can be viewed in the CKZPRINT of the target job. See Example 9-46.

Example 9-46 SQLOUT resetting identity sequence

```
CKZ76031I The following SQL Statement submitted for execution
CKZ76032I    ALTER TABLE GLWSAMPC.GLWTLCN
CKZ76032I          ALTER COLUMN LOC_NO
CKZ76032I          RESTART WITH 61
```

Other error messages

After addressing the mismatches in the CKZERROR report, we look at the remaining messages. See Example 9-47.

Example 9-47 Partition mismatch

```
CKZ54714W TS Dataset DBODD.DSNDBC.GLWSAMP2.GLWSEPA.*OOO1.AOO5 is Not Cataloged on the
Target System and the Ob ject Exists, will be Copied to the Target Subsystem using CKZ 5th
Level Qualifier, FOOO1

CKZ54763W TABLESPACE Mismatch, Attribute PARTITIONS, DBOC.GLWSAMP.GLWSEPA = 5 and
DBOD.GLWSAMP2.GLWSEPA = 4
```

The issue illustrates the problems that can be encountered with PBG TS. The methods to address this issue are documented at 9.8.2, “Partition-by-growth (PBG) table spaces” on page 508.

The source object status is shown in Example 9-48.

Example 9-48 Source object status

```
CKZ54719W IX GLWSAMP.GLWXEPA1 Status is PSRBD, Subsystem DBOC
CKZ54724W Dataset DBOCD.DSNDBC.GLWSAMP.GLWXEPA1.IOOO1.AOO1 has Failed Status Checking, Will
Be Copied Per ALLOW-COPY-ON-MISMATCH(Y) to Subsystem DBOD
```

The source index was in this status. It will require the index to be rebuilt on the target. It is probably not a bad idea to fix the source, as well. The object is still copied because ALLOW-COPY-ON-MISMATCH is set to Y.

9.7.13 Target job

There are no parameters to be changed in the target job. Processing guidance is provided through input files that are updated by the source job. Note the inclusion of the DD cards IXRBLD*x* that are used for the rebuild index jobs as a result of processing the mini logs. See Example 9-49 on page 504.

Example 9-49 Target Job

```
//ADMR1I JOB (999,POK),'DB2 CLONE TS',
//         REGION=0M,NOTIFY=&SYSUID,MSGCLASS=X,CLASS=T
//PROCLIB JCLLIB ORDER=DBOAM.PROCLIB
/*JOBPARM SYSAFF=SC63
//*
//*************************************************************
/*JOBPARM S=SC63
//DELETE   EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
     DELETE ADMR1.TS2.GLW.MINILOG.*
     SET MAXCC=0
//*
//S1      EXEC PGM=CKZ00500,REGION=0M
//STEPLIB  DD  DISP=SHR,DSN=DBTLSP.SCKZLOAD
//         DD  DISP=SHR,DSN=DBODT.SDSNLOAD
//CKZINI   DD  DISP=SHR,DSN=DBTLSP.SCKZPARM(CKZINI)
//CKZLOG   DD  SYSOUT=*
//CKZPRINT DD  SYSOUT=*
//CKZIN    DD  DISP=OLD,DSN=ADMR1.TS2.SYNCDB2(GLWSAMP6)
//CKZMDBOD DD  DISP=OLD,DSN=ADMR1.TS2.XMLSTR(GLWSAMP6)
//CKZQDBOD DD  DISP=OLD,DSN=ADMR1.TS2.SQLOUT(GLWSAMP6)
//SYSINCKZ DD  DISP=SHR,DSN=ADMR1.TS2.LOGAPCTL(GLWSAMP6)
//SYSOUT   DD  SYSOUT=*
//INFOM    DD  SYSOUT=*
//CKZRRJOB DD  DISP=OLD,DSN=ADMR1.TS2.RRJOB
//CKZRRDSN DD  DISP=OLD,DSN=ADMR1.TS2.RRDSN
//CKZERROR DD  SYSOUT=*
//IXRBLDI  DD  DISP=SHR,DSN=ADMR1.CLONE.SKELJCL(CKZRBLDX)
//IXRBLDO  DD  DISP=OLD,DSN=ADMR1.CLONE.JOBS(GLWSAMP6)
//*
```

Target job output

The target produces the following reports:

1. CKZLOG: In the target job, CKZLOG displays the DB2 commands issued and the responses or results of the commands. It also displays detailed information about each DB2 page access.
2. CKZPRINT: In the target job, CKZPRINT displays CKZINI tokens, CKZIN control parameters, DB2 SQL execution status, and SYNCDB2 status and START DB2 command status for each data set processed.
3. SYSOUT.
4. INFOM.

9.7.14 Summary of the table space clone process

The following steps summarize the process to clone the table space:

1. Align the target structures with the source by using one of these tools:
   - DB2 Object Compare Tool
   - Administration Tool - GEN, then drop, and re-create target.

– Cloning Tool Generate dynamic link library (DLL). Only use to create and re-create the target. The scope of the objects is restricted.

2. Set up the cloning jobs.
3. Submit the source job.
4. Submit the target job.
5. Rebuild indexes, if required.
6. Run any additional jobs generated using job templates.
7. Verify the target environment:
   a. Review the mismatch report and take action, if needed.
   b. Check the data.
   c. Verify the statistics.
   d. Ensure that all objects are in RW status.

9.7.15 LOG APPLY and FUZZY-COPY

LOG APPLY refreshes target table spaces and index spaces without stopping the source objects with consistent data from the source objects.

This feature applies to DB2 pages that are updated by the target job and log records written by DB2, before and after the completion of the data set copies in the source job.

It is provided to support the FUZZY-COPY option where the objects are not stopped at the time that the copy is taken. This feature enables a clone without causing an outage to your source system.

The user can select one of two options in specifying a point-in-time:

- Cause DB2 Cloning Tool to issue a QUIESCE after the data set copies are complete.
- Select a consistent point common to the source DB2 objects: COMMON-CONSISTENT-POINT

If a QUIESCE point is not taken after the copies, a consistent point needs to be found. Currently, the option here is for log apply to find a common consistent point TO CURRENT (Change Accumulation option).

With either option, the preference is to perform the copy of the source at a period of low activity, which provides the best chance of finding a consistent point across all cloned objects and fewer log records to process. Log apply does not currently apply to indexes, which is another good reason to keep the amount of activity on the source to a minimum. This may reduce the number of indexes needing to be rebuilt for which the jobs can be generated by the process.

When the consistent point option is taken, the consistent point in time is not negotiated until the time that the target job is run. Therefore, the longer you take to run the target jobs, the more log records might need to be processed. It also means that the clone syncpoint time is not at the time of the copy but around the time of the target job.

Taking a Quiesce point also might not be possible or too disruptive to the source system.

Our example was taken when there was little activity at the time of the copy, but for illustration purposes, we used the common-consistent-point option to show the processing of log records after the copy.

LOG APPLY can create many files for the mini logs. These files are created and used by the target job by using the parameters provided. A delete step is the first step in the target job to clean from previous runs.

At this point in time, there are some limitations with applying logs, which future upgrades for the Cloning Tool and Change Accumulation may resolve.

Restrictions:

- Logs cannot be applied for LOBs, XML spaces, and index spaces.
- The source and target DB2 must be on the same LPAR, but an enhancement is planned.
- Minilog processing can only be used locally on the same LPAR at this point in time.

There is some danger when using this option, considering that you need to decide whether to set ALWAYS-COPY-INDEXSPACES to YES or leave it to the default NO. The problem is that it if you do not include all the indexes, you will have to rebuild all the indexes. Now, depending on the size and number of your indexes, this task can add considerable time and effort to your clone.

If you do include the indexes, you can reduce the number of indexes that need rebuilding by reviewing the target job log status report.

In Example 9-50, we look for log pages changed greater than zero. These are the table spaces that have been updated since the copy was taken and should have the indexes rebuilt.

Example 9-50 Log Apply Status Report sample

```
CKZ54641I - Begin Completion Status Report

                                                                                                                  I   V
                                                                                                                  O   S
                                                  S                                                               E   A
                                                  PT                                                              R   M
                                                  AY                          LOG      DATA                       R   E
                                                  CP                        PAGES  MASKING                      O   R
                TARGET DATASET                  RC EE      PAGES      CHANGED  PAGES CHANGED  VSAM READS  VSAM WRITES  R   SECS
   ------------------------------------------- -- -- -------- -------- -------- -------- -------- -   -   ------
   DBODD.DSNDBC.GLWSAMP2.GLWSDPT.IO001.A001     0 TS        372        0        0      372      251                 0
   DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.A001     0 TS        936        0        0      936      600                 1
   DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.A002     0 TS        936        0        0      936      593                 0
   DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.A003     0 TS        936        0        0      936      595                 0
   DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.A004     0 TS        936        0        0      936      516                 1
   DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.A001     0 TS    1048320        0        0  1048320  1048320               256
   DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.A002     0 TS    1048320        0        0  1048320  1048320               248
   DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.A003     0 TS    1048320        0        0  1048320  1048320               256
   DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.A004     0 TS     715575        0        0   715575   715575               175
   DBODD.DSNDBC.GLWSAMP2.GLWSPJA.IO001.A001     0 TS       5940        0        0     5940     5544                 0
   DBODD.DSNDBC.GLWSAMP2.GLWSPRJ.IO001.A001     0 TS       2556        0        0     2556     2392                 0
   DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.A001     0 TS          2        0        0        2        2                 0
   DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.A002     0 TS          2        0        0        2        2                 0
   DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.A003     0 TS          2        0        0        2        2                 0
   DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.A004     0 TS       1080        0        0     1080      868                 0
   DBODD.DSNDBC.GLWSAMP2.GLWSO01.IO001.A001     0 TS         60        0        0       60       25                 0
   DBODD.DSNDBC.GLWSAMP2.GLWSO02.IO001.A001     0 TS        372        0        0      372        2                 0
   DBODD.DSNDBC.GLWSAMP2.GLWSO03.IO001.A001     0 TS        372        1        0      372        3                 0
   DBODD.DSNDBC.GLWSAMP2.GLWXACT1.IO001.A001    0 IS         12        0        0       12        5                 0
   DBODD.DSNDBC.GLWSAMP2.GLWXDNG1.IO001.A001    0 IS         12        0        0       12        5                 0
   DBODD.DSNDBC.GLWSAMP2.GLWXDPT1.IO001.A001    0 IS         36        0        0       36       33                 1
   DBODD.DSNDBC.GLWSAMP2.GLWXDPT2.IO001.A001    0 IS        396        0        0      396       54                 5
 CKZ54643I - End Completion Status
Report
```

We did not have much activity between the source and target job but we did have one table space that had changes. This is the only table space for which we need to rebuild the indexes.

TARGET-JOB-INDEX-REBUILD-DDN

APAR PM58081 added the option to create the Rebuild Indexes jobs only for those indexes that are needed. The TARGET-JOB-INDEX-REBUILD-DDN keyword was added to the DB2 Cloning Tool Table Space Cloning COPY command. This keyword allows you to generate utility jobs that can rebuild indexes in the target job affected by data masking or log apply changes.

It is implemented by providing the DD name as in the scenario that we just completed. You must create a job template and update your target job with the file name. You will also must create the file to which the job will be written. Our target job cards are shown in Example 9-51.

Example 9-51 Target job extract with TARGET-JOB-INDEX-REBUILD-DDN

```
//IXRBLDI  DD  DISP=SHR,DSN=ADMR1.CLONE.SKELJCL(CKZRBLDX)
//IXRBLDO  DD  DISP=OLD,DSN=ADMR1.CLONE.JOBS(GLWSAMP6)
```

You can make a copy of either the CKZJOBI (recommended) or CKZJOBR template in the SCKZJCL library and modify it for your site.

This template contains the statements used to build the REBUILD INDEX utility job. Instructions for updating the template are contained in the member.

Example 9-52 shows the job template that we used.

Example 9-52 Rebuild index job template

```
//ADMR1G JOB ,'COPY',CLASS=A,MSGCLASS=X,REGION=0M,NOTIFY=&SYSUID
//* &&JOBCARD
//* &&HEADER
//* &&TRGOBJS
//*
//* V9 INDEX REBUILD USING TARGET TABLESPACES
//*
    &&BEGSTEP
//ST.&&STEPNUM EXEC DSNUPROC,UID=' ',
// UTPROC='',SYSTEM='&&TRGSSID',LIB='DBODT.SDSNLOAD',
//SYSUT1     DD DSN=ADMR1.CLONE.&&STEPNUM.SYSUT1,
//     DISP=(MOD,DELETE,CATLG),
//     UNIT=SYSDA,SPACE=(8000,(200,20),,,ROUND)
//SYSERR     DD DSN=AMDR1.CLONE.CHK3.SYSERR,
//     DISP=(MOD,DELETE,CATLG),
//     UNIT=SYSDA,SPACE=(6000,(20,20),,,ROUND)
//SORTOUT     DD DSN=ADMR1.CLONE.CHK3.&&STEPNUM.SORTOUT,
//     DISP=(MOD,DELETE,CATLG),
//     UNIT=SYSDA,SPACE=(6000,(20,20),,,ROUND)
//SYSIN DD *
  REBUILD INDEX (ALL) TABLESPACE &&DATABASE.&&TABLESP
    &&ENDSTEP
/*
```

9.8 Dealing with special objects

There are many different objects and attributes that DB2 manages, and many of them have an impact on cloning behavior. We describe a few of them.

9.8.1 Identity columns

DB2 Cloning Tool Table Space Cloning can update the sequence numbers for identity columns in the DB2 catalog.

The DB2 Cloning Tool Table Space Cloning source job creates the ALTER TABLE SQL to adjust identity column values. The SQL is in the SQLOUT file and processes the target job.

A sample of the messages that are produced in the error report is shown in “Object mismatch” on page 502.

9.8.2 Partition-by-growth (PBG) table spaces

DB2 9 introduced universal table spaces partitioning according to data growth. This enables segmented tables to be partitioned as they grow. DB2 Cloning Tool Table Space Cloning can copy partition-by-growth (PBG) table spaces if the number of source partitions is equal to or less than the number of target partitions.

However, if the number of source partitions is greater than the number of target partitions, DB2 Cloning Tool Table Space Cloning will create a default subsystem name (DSN) on the target for each partition that does not exist, using .F0001 as a data set name qualifier. This data set, and therefore the extra partitions, will not be usable on the target.

If the number of partitions is greater on the source, you must create the extra partitions on the target.

In DB2 10, you can create the extra partitions on the target by using *ALTER TABLE "schema"."tbname" ADD* PARTITION.

In DB2 9, you can create the extra partitions on the target by using the DB2 UNLOAD and LOAD utilities before initiating the cloning process.

Alternatively, you can drop the table space and re-create it specifying the NUMPARTS option on the table space. This will define how many partitions to define initially. NUMPARTS is set to number of partitions defined on the source.

The cloning will be allowed to proceed when there are unequal partitions if the ALLOW_MISMATCH is set to YES. If you review the CKZERROR report from the COPY job, you will be alerted to a mismatch. You will then need to resolve the mismatch by using one of the preceding methods, which can include rerunning the COPY step.

Important: If the indexes are copied, they will be corrupted and unusable if there is a mismatch. *A rebuild of the index will not help.*

An example of the mismatch message is shown in Example 9-53 on page 509.

Example 9-53 Number of partitions mismatch

```
CKZ54763W TABLESPACE Mismatch, Attribute PARTITIONS, DBOC.GLWSAMP.GLWSEPA = 5 and
DBOD.GLWSAMP2.GLWSEPA = 4
```

Note: APAR PM67800 assists with managing a target PBG table space with an uneven number of partitions. It adds the parameter EXTEND-TARGET-PBG-TABLESPACE.

You then need to resolve the mismatch by using one of the preceding methods or by allowing it to be managed with the use of this APAR.

The recent addition of the command (EXTEND-TARGET-PBG-TABLESPACE) enables PBG table spaces on the target to add one or more partitions if the source PBG table space has more partitions than the corresponding target PBG table space.

Partitions are added in one of two ways. If the DB2 version is V9, an UNLOAD and a LOAD are used.

If the DB2 version is higher than V9, an ALTER TABLE is used. After the partitions have been added, the copies are performed if requested in the source job.

So, by adding EXTEND-TARGET-PBG-TABLESPACE(Y) to our copy step, our partition mismatch is resolved. The job will generate and execute the ALTER TABLE before the Copy is run as shown in Example 9-54.

Example 9-54 Add partition in Copy step

```
CKZ76715I ALTER TABLE GLWSAMPC.GLWTEPA ADD PARTITION
CKZ76714I 1 DDL Statement(s) have been Output to the Target Subsystem DBOD
CKZ54775I Extend PBG Successful, Target TS=GLWSAMP2.GLWSEPA
```

9.8.3 Large objects (LOBs)

The example that we used in this chapter contained LOBs that were cloned. When cloning LOBs, you need to ensure that all indexes, base tables, and auxiliary tables are included in the scope.

Log Apply does not currently work with LOBs, so you need to ensure that no updates occur between the source copy job and the target job.

9.8.4 Table space reordered row format

Depending on the value of the reordered row format (RRF) subsystem parameter, newly created table spaces will be either in reordered row format or basic row format. When the value of the RRF parameter is ENABLE, table spaces will be created in reordered row format.

RRF is new with DB2 9 new function mode (NFM). It results in variable-length columns being placed at the end of a row. RRF is not compatible with objects that have been migrated from DB2 V8. Those objects will be in basic row format (BRF) until a REORG or LOAD REPLACE is run on the object under DB2 Version 9.1 NFM.

Newly created table spaces under DB2 Version 9.1 NFM are put in RRF. If a source BRF table space is copied to a target RRF table space (or vice versa), in most cases, the target tables will not be accessible. DB2 Cloning Tool Table Space Cloning will issue a warning if the row formats do not match. Therefore, if running DB2 Cloning Tool Table Space Cloning on DB2

Version 9.1 NFM or a later subsystem, and new objects are added to LISTDEF, run the source job using DATA-MOVER(NONE) and SIM(N). All object incompatibilities will result in warning or error messages.

Run a REORG on the source table spaces in BRF and rerun the DB2 Cloning Tool Table Space Cloning source job to ensure that there are no more object incompatibilities. Another alternative (with less impact) is to run REORG on the target table spaces using ROWFORMAT BRF.

To find tables that are still in BRF format, run a query on the catalog table to look where FORMAT='B' in SYSIBM.SYSTABLEPART.

If there are table spaces, which are being reported as a mismatch, that do not have VARCHAR columns, they can be allowed to copy. To do this, you need to set the option IGNORE-RF-MISMATCH-IF-NO-VAR-COLS = Y (default is N) in the COPY parameters. The option is displayed on the ISPF panels.

This command, if Y, allows table spaces with no variable columns to ignore the mismatch in row format, and to be copied without a warning issued.

9.8.5 LONGVAR versus VARCHAR

When migrating to DB2 9, all LONGVAR columns remain. However, new LONGVAR columns become VARCHAR columns.

If running a source job where the source objects have LONGVAR columns and the target objects have corresponding VARCHAR columns, or vice versa, a mismatch will be reported via a warning message.

Use this new setting to prevent the mismatch message (return code 4):

```
LONGVAR-COMPATIBILITY  (Y | N)
```

You can set the command by using the ISPF dialog on the DDL Attribute change, as shown in Figure 9-85.

```
CKZ1COPC              DB2 tablespace clone COPY Command
Option ===>

Commands: S - Set SOURCE-PREFETCH-DATABASE-LIST  O - Set OBJECT-TRANSLATE
          T - Set TARGET-PREFETCH-DATABASE-LIST  J - Set JOB-TEMPLATE
          D - Set DDL-ATTRIBUTE-CHANGE  L - Set LOG-APPLY

Creator  . . . : ADMR1        Name  . . . . : TEST2 CL
Share Option . : UPDATE       Description . : TESTDB WITH CLONE TABLE
                                                          More:   - +
INCLUDE-ALL-RI . . . . . . . . . . . YES          (Yes/No)
LONGVAR-COMPATIBILITY  . . . . . . . YES          (Yes/No)
REPLACE-TARGET-DSN . . . . . . . . . YES          (Yes/No)
RESET-LOGRBA . . . . . . . . . . . . YES          (Yes/No)
```

Figure 9-85 LONGVAR compatibility example

Important: The lengths of the corresponding columns must be the same. If not, data might be truncated or a DB2 abend might occur. By using the Object compare tool, you can highlight these differences.

9.8.6 Just VARCHAR

Because we just looked at the LONGVAR scenario, it might be useful to consider the case of cloning a table space where VARCHARs are involved.

It is always advisable to run a compare between your source and target environments. This should highlight whether there are differences in the lengths of VARCHARs and then the differences can be corrected before you clone. If you decide to proceed and use the COPY parameter ALLOW-COPY-ON-MISMATCH(Y), you might be in trouble. After you perform the copy and, then, try to access your data on the target, you likely will get a DB2 error, such as the error that is shown in Example 9-55.

Example 9-55 DB2 Inconsistent data due to mismatch in VARCHAR lengths

```
CEE3250C The system or user abend S04E  R=00C90216 was issued.
         From entry point SELECT_STMT at statement 10071 at compile unit offset
+00000744 at entry offset +00000744 at address 24E44CDC.
```

The Cloning Tool will draw your attention to a mismatch in the source job and the mismatch can be discovered before you perform the actual copy. This can be done by using the DATA-MOVER(PGM(NONE)) Copy option. Note that using SIM(Y) will not show the mismatch.

The source job report is shown in Example 9-56.

Example 9-56 Column Mismatch Report

```
13078 00:42:24.06 CKZ54772W TABLE COLUMN Mismatch, Attribute LENGTH, DB0C.ADMR1.ATABLEWITHVARCHAR.VARCHAR2 = 250 and
DB0D.ADMR1.ATABLEWITHVARCHAR.VARCHAR2 = 50
 13078 00:42:24.06 CKZ54712W Dataset DB0CD.DSNDBC.TESTDB2.ATS3.I0001.A001 has Failed Object Match Checking, Will Be
Copied Per ALLOW-COPY-ON-MISMATCH(Y) to Subsystem DB0D
 13078 00:42:30.57 CKZ50005W Discovery Phase Ended with Warning(s)
 13078 00:42:31.57 CKZ50013W Completed, with Warnings, RC=X'00000004, RS=X'00000000
```

9.8.7 Considerations for objects created by using DEFINE NO

The use of DEFINE NO is interesting in relation to cloning table spaces. Initially, running a compare between the source and target will assist you in aligning the object structures and their definitions. The compare does not identify whether a data set exists or not.

There are two key scenarios that you need to consider:

- Source DEFINE NO and no data set - Target DEFINE NO and a data set.
- Source DEFINE NO with a data set - Target DEFINE NO and no data set.

If you do not know whether any target or source objects were created with DEFINE NO and you do not have a data set created by DB2, submit a DB2 Cloning Tool Table Space Cloning source job with COPY-IF-NO-DB2-TARGET-OBJECTS(N) and PGM(NONE). All target table space and index spaces without a data set will be listed in the output with a warning message.

Scenario one

In scenario one, the source has no data set. The objects are automatically excluded, and no copy is made. There are no warning messages written for this scenario, but you can identify the excluded objects by looking at the CKZLOG report. Identify message CKZ61301I (Example 9-57 on page 512).

Example 9-57 CKZLOG with DEFINE NO

```
CKZ61301I Tablespace TESTDB2.ATS5 Partition 0 is Filtered for DEFINE NO Attribute
CKZ61302I Tablespace TESTDB2.ATS5 in List TS.01 Filtered, all Parts Deleted for DEFINE NO Attribute
CKZ54450I Processing LISTDEF TS.01 TS DB INC   ALL              RI   TESTDB2
CKZ54450I Processing LISTDEF TS.01 IS DB INC   ALL              RI   TESTDB2
CKZ54412I Tablespace TESTDB2.ATS in List TS.01 Included
CKZ54444I Table ADMR1.ATABLE in List TS.01 has been Included
CKZ54412I Tablespace TESTDB2.ATS2 in List TS.01 Included
CKZ54444I Table ADMR1.ATABLE2 in List TS.01 has been Included
CKZ54412I Tablespace TESTDB2.ATS3 in List TS.01 Included
CKZ54444I Table ADMR1.ATABLEWITHVARCHAR in List TS.01 has been Included
CKZ54412I Tablespace TESTDB2.ATS4 in List TS.01 Included
CKZ54444I Table ADMR1.ATABLEWITHDEFINENO in List TS.01 has been Included
CKZ54412I Tablespace TESTDB2.TESTTS2 in List TS.01 Included
CKZ54444I Table TEST2.GLWTEPA in List TS.01 has been Included
CKZ54444I Table TESTCM.GLWTEPACL in List TS.01 has been Included
CKZ54423I Index TEST2.GLWXEPA1 in List TS.01 Included
CKZ54423I Index TEST2.GLWXEPA2 in List TS.01 Included
```

After these objects are identified from the source, you will need to check that the target objects have no data.

Scenario two

In the case of scenario two, you have information regarding target objects with DEFINE NO and no data set. These objects are identified in the CKZERR report, as shown in Example 9-58. We describe this situation in detail later.

Example 9-58 CKZERROR with DEFINE NO

```
CKZ54702W Target SS DB0D, Object in DB2 Catalog, TS TESTDB2.ATS4, with DEFINE NO, the CKZ 5th
level qualifier,
                    F0001, will be used for the New Target Dataset
 CKZ50005W Discovery Phase Ended with Warning(s)
 CKZ50013W Completed, with Warnings, RC=X'00000004, RS=X'00000000
```

In Example 9-58, the return code is 4 because we use ALLOW-COPY-ON-MISMATCH and CHECK-DATASET-COMPATIBILITY = N.

Tip: Although it might be useful to run the clone with the parameter CHECK-DATASET-COMPATIBILITY =Y, when used with DEFINE NO objects, it might produce additional irrelevant warning messages. The job will also end with a return code 8.

Run with this option set to Y to identify data set mismatches. Resolve the issues relating to other objects, and then, set this option to N and rerun with COPY-IF-NO-DB2-TARGET-OBJECTS(N) and PGM(NONE).

This issue is resolved by APAR PM86668.

As shown in the CKZ ERROR report, the source job creates a data set on the target but with the 5th level Qualifier using F0001, that is, DB0DD.DSNDBC.TESTDB2.ATS4.F0001.A001. This is done so that when the target DB2 attempts to create an *.I0001.* data set, it will not find the one copied from the source already in existence. If DB2 finds an *.I0001.* data set, it initializes the existing data set and all data becomes inaccessible.

What to do with this data set with F0001

The target job will update object IDs in the *.F0001.* data sets to match those in the target catalog. The job will also automatically start the target objects. To assist you in renaming the

data sets, IDCAMS statements are written to an optional IDCAMS-DDN file. These parameters can be used to rename the *.F0001.* data sets to *.I0001.* data sets.

You need to ensure that you have selected the IDCAMS DD on the DD specifications option on the source if you are using the ISPF panels. Alternatively, add IDCAMS-DDN(CKZZ*xxxx*) to the COPY step and add a corresponding DD card to the JCL. A PDS was created as part of the Tools Customizer installation that you can use.

Example 9-59 shows the IDCAMS that are written for our scenario.

Example 9-59 IDCAMS rename

```
/* 13080 04:23:36.70  JOBNAME=ADMR1E  JOBID=JOB21644             */
/*     SOURCE SUBSYSTEM=DBOC  TARGET SUBSYSTEM=DBOD              */
/* IDCAMS DELETE and ALTER Tablespace Commands */

/*   Tablespace COPY Successful   */

  DELETE DBODD.DSNDBC.TESTDB2.ATS4.I0001.A001 -
     CLUSTER NOERASE PURGE
  ALTER -
     DBODD.DSNDBC.TESTDB2.ATS4.F0001.A001 -
     NEWNAME(DBODD.DSNDBC.TESTDB2.ATS4.I0001.A001)
  ALTER -
     DBODD.DSNDBD.TESTDB2.ATS4.F0001.A001 -
     NEWNAME(DBODD.DSNDBD.TESTDB2.ATS4.I0001.A001)

/* IDCAMS DELETE and ALTER Indexspace Commands */
```

You must then complete the following steps:

1. Insert a row into the target table. If the table is partitioned, a row must be inserted into the desired partitions.
2. Stop the target objects to prevent enqueue conflicts with the IDCAMS job over the *.I0001.* data sets.
3. Submit a job by using the IDCAMS parameters to rename the *.F0001.* data sets to *.I0001.* data sets. This job will delete the *.I0001.* data sets that are created by DB2 and then rename the *.F0001.* data sets to the *.I0001.* data sets.
4. Start the target objects.

So, DEFINE NO makes things interesting, but if you are alert, you can manage the situation. Run several test clones with PGM(NONE) to identify the issues that you need to resolve.

Partitions and DEFINE NO

Partitions are not added if the target PBG table space was created with the DEFINE NO clause and its data sets were not allocated.

If an error occurs because the number of target partitions is greater than the number of source partitions, you need to delete the extra target partitions.

9.8.8 Clone tables

Support is provided, by default. DDL will manage clone tables, and it allows for their inclusion or exclusion. Control of whether to include or exclude tables with clone tables is managed by

the Cloning Tool LISTDEF. Example 9-60 shows a sample LISTDEF, which was set in Figure 9-80 on page 496.

Example 9-60 LISTDEF with CLONED option

```
LISTDEF TS
INCLUDE TABLESPACES DATABASE TESTDB2 CLONED YES
```

Important: The *DB2 Cloning Tool for z/OS, V3.1, User's Guide*, SC19-3493-01, uses the keyword CLONE but CLONED is the correct syntax.

The standard LISTDEF has ALL instead of CLONED. ALL allows for all objects to be cloned within the scope. Using the parameter CLONED restricts the scope to either exclude the base and its clone table from the process (CLONED NO) or to only clone the base and its clone table (CLONED YES). CLONED YES will ignore all table spaces that do not have a clone table.

From the Admin Tool, we can see that one table has a clone table because the type is C. See Figure 9-86.

```
DB2 Admin --------------- DBOD Tables, Views, and Aliases ---- Row 1 to 4 of 4
Command ===>                                                  Scroll ===> PAGE

Commands: GRANT  MIG  ALL
Line commands:
 C - Columns  A - Auth  L - List  X - Indexes  S - Table space  D - Database
 V - Views  T - Tables  P - Plans  Y - Synonyms  SEL - Select prototyping
 ? - Show all line commands

Sel   Name                Schema   T DB Name  TS Name    Cols        Rows Chks C
      *                   *        * *        *             *           *    * *
----- ------------------ -------- - -------- -------- ------ ----------- ---- -
      ATABLE              ADMR1    T TESTDB3  ATS           1          -1    0
      ATABLE2             ADMR1    T TESTDB3  ATS2          1          -1    0
      GLWTEPA             TEST3    T TESTDB3  TESTTS3      10           1    0
      GLWTEPACL           TESTCM   C TESTDB3  TESTTS3      10          -1    0
****************************** END OF DB2 DATA *******************************
```

Figure 9-86 Identifying a CLONE Table with the Admin Tool

The cloning tool process will replace the contents of the target, so if we had EXCHANGED data on the target, the data will be reverted to what it is on the source.

We can see from the output of the source job where the clone table is identified and the relationship that is made. Our example used a PBG table space, therefore, there are the multiple parts. See Figure 9-87 on page 515.

```
CKZ54521I Begin Source Tablespace Dataset Report
CKZ54527I DSN=DBOCD.DSNDBC.TESTDB2.TESTTS2.IO002.A001  List=TS.01  Type=TS  Obj=
CKZ54528I DSN=DBOCD.DSNDBC.TESTDB2.TESTTS2.IO001.A001
CKZ54529I                                  Table C TESTCM.GLWTEPACL
CKZ54529I                                  Table B TEST2.GLWTEPA
CKZ54527I DSN=DBOCD.DSNDBC.TESTDB2.TESTTS2.IO002.A002  List=TS.01  Type=TS  Obj=
CKZ54528I DSN=DBOCD.DSNDBC.TESTDB2.TESTTS2.IO001.A002

CKZ54825I - Begin COPY Completion Status Report

                                                                                                  R
                                                                                                  E   T
                                                                                                  P   R   OBJ
                                                                                              S   L   G   XLATE
                                                                                              PT  L
                                                                                              AY  O   D   O     II
                                                                                              CP  N   S   B   DTIXX
                                                          TARGET DATASET                  RC  EE  E   N   J   BSSCN



             SOURCE DATASET                          TARGET DATASET                       RC  EE  E   N   J   BSSCN
   ------------------------------------------  ------------------------------------------  --  --  -   -   -   -----
   DBOCD.DSNDBC.TESTDB2.GLWXEPA1.IO002.A001     DBODD.DSNDBC.TESTDB3.GLWXEPA1.IO002.A001    0   IS  B   Y   Y   +  +
   DBOCD.DSNDBC.TESTDB2.GLWXEPA1.IO001.A001    DBODD.DSNDBC.TESTDB3.GLWXEPA1.IO001.A001     0   IS   C  Y   Y  + +
   DBOCD.DSNDBC.TESTDB2.GLWXEPA2.IO002.A001     DBODD.DSNDBC.TESTDB3.GLWXEPA2.IO002.A001    0   IS  B   Y   Y   +  +
   DBOCD.DSNDBC.TESTDB2.GLWXEPA2.IO001.A001     DBODD.DSNDBC.TESTDB3.GLWXEPA2.IO001.A001    0   IS  C   Y   Y   +  +
   DBOCD.DSNDBC.TESTDB2.TESTTS2.IO002.A001      DBODD.DSNDBC.TESTDB3.GLWXEPA2.IO001.A001    0   IS  C   Y   Y   +  +
   DBOCD.DSNDBC.TESTDB2.TESTTS2.IO001.A001      DBODD.DSNDBC.TESTDB3.TESTTS3.IO001.A001     0   TS  C   Y   Y   ++
   DBOCD.DSNDBC.TESTDB2.TESTTS2.IO002.A002      DBODD.DSNDBC.TESTDB3.TESTTS3.IO002.A002     0   TS  B   Y   Y   ++
   DBOCD.DSNDBC.TESTDB2.TESTTS2.IO001.A002      DBODD.DSNDBC.TESTDB3.TESTTS3.IO001.A002     0   TS  C   Y   Y   ++
```

Figure 9-87 Source job output managing Clone Tables

It is not possible to clone a base table without the related clone table.

9.8.9 Job templates

Job templates provide the capability to build JCL for running utilities against the scope of objects that are cloned. Possible uses for job templates include Runstats on the target, as used in the scenario we described. This would update the Runstats and also update the real-time statistics (RTS).

Another use is to perform a REORG for objects to reset any versioning or AREOGP status.

The job templates consist of the z/OS JCL statements, DFDSS commands, user variables, and processing variables that CKZ uses for input.

The job statements are generated in the COPY step and are then written to the output DD specified in the JOB-TEMPLATE subcommand.

For the detailed instructions to set up job templates, see the sample members CKZJOBT1 and CKZJOBT2 that are provided in the product sample JCL library, and also in the product user guide.

Table 9-3 lists sample templates in the *.SCKZJCL.

Table 9-3 Sample Job templates

Template	Function
CKZJOBCI	Check index
CKZJOBI	Rebuild Indexes for Index space
CKZJOBQ	Quiesce

CKZJOBR	Rebuild Index for table space
CKZJOBRI	Reorg for target Indexes
CKZJOBRO	Reorg for target table spaces
CKZJOBRS	Runstats

Figure 9-88 shows one use for job templates. There is scope for many additional jobs to be created with one clone. This example only shows one utility, which is for Runstats.

```
 CKZ1PJTS          Specify Job Template Data Sets and Members
Command ===>                                                Scroll ===> PAGE

Commands:      A - Add Line
Line commands: D - Delete Line

Creator  . . . : ADMR1        Name  . . . . : TABLESPACE 6
Share Option . : UPDATE       Description . : GLWSAMP TARGET-JOB-INDEX-REBUI >

TEMPLATE INPUT DATA SET  . . ADMR1.CLONE.SKELJCL
TEMPLATE OUTPUT DATA SET . . ADMR1.CLONE.JOBS

INPUT DD DISP . . SHR    OUTPUT DD DISP . . OLD
                                                          Row 1 of 3
Cmd INPUT MEMBER INPUT DDNAME OUTPUT MEMBER OUTPUT DDNAME
    CKZJOBRS     RSTATIA6     RSTAT1J       RSTATOA6
```

Figure 9-88 Specify Job Template Data Sets and Members

Example 9-61 lists the template that we used for the Runstats utility.

Example 9-61 Sample job template

```
//ADMR1G JOB ,'RUNSTATS',CLASS=A,MSGCLASS=X,REGION=0M,NOTIFY=&SYSUID
//* &&JOBCARD
//* &&HEADER
//* &&TRGOBJS
//*
//*  RUNSTATS UTILITY
//*
    &&BEGSTEP
//*
//S.&&STEPNUM EXEC DSNUPROC,UID=' ',
// UTPROC='',SYSTEM='&&TRGSSID',LIB='DBODT.SDSNLOAD',
//*S.&&STEPNUM  EXEC DSNUPROC,UID='USERID',
//*             UTPROC='',SYSTEM='&&TRGSSID'
//*STEPLIB  DD  DISP=SHR,DSN=DSNHLQ.SDSNLOAD
//UTPRINT  DD  SYSOUT=*
//SYSIN    DD  *
    RUNSTATS
      TABLESPACE &&DATABASE.&&TABLESP
//*
    &&ENDSTEP
/* //SYSIN DD *
  REBUILD INDEX (ALL) TABLESPACE &&DATABASE.&&TABLESP
    &&ENDSTEP
/*
```

Note: The option to build Rebuild Indexes jobs where log records exist since the COPY was taken is supported with the TARGET-JOB-INDEX-REBUILD-DDN parameter. This parameter is introduced with APAR PM68129.

9.9 Using DB2 Cloning Tool with the DB2 Administration Tool

It is now possible with recent maintenance to V10.2 of the DB2 Administration Tool to call the DB2 Cloning Tool as an option for migrating your data. The new line command and the new primary command are both CT.

The DB2 Administration Tool requires the customization table that has the appropriate definitions at the specific subsystem and not the NICK:* to enable the DB2 Cloning Tool. The TCZ job should be A0CUSTAA. See Figure 9-89.

```
EDIT       DB2TOOLS.JCL.$SC63$.ADB1020(A0CUSTAA) - 01.04
 Command ===>
 000814 * DB2 SUBSYSTEM DESCRIPTION
 000815 :DESC.
 000816
 000817 * ENABLE DB2 CLONING TOOL (YES/NO)
 000818 :ADBCT.YES
 000819
 000820 * DB2 CLONING TOOL CLIST LIBRARY
 000821 :ACTELIB.'DB2TOOLS.CLIST'
 000822
 000823 * ENABLE DB2 TABLE EDITOR (YES/NO)
 000824 :ADBTE.NO
 000825
```

Figure 9-89 Enable Cloning Tool with the Admin Tool

The line command CT is used at the database or table space level to call the Cloning Tool, as shown in Figure 9-90 on page 518.

```
DB2 Admin ------------------- DBOC Databases ----------------- Row 1 to 4 of 4
Command ===>                                                   Scroll ===> PAGE

Commands: GRANT  MIG  DIS  STA  STO  UTIL  CT
Line commands:
 T - Tables  S - Table spaces  X - Indexes  G - Storage group  ICS - IC status
 DIS - Display database  STA - Start database  STO - Stop database  A - Auth
 ? - Show all line commands
                         Storage  Buffer          Created      Index
Select Name     Owner    Group    Pool       DBID By       T E BPool    I
       *        *        *        *             * *        * * *        *
------ -------- -------- -------- -------- ------ -------- - - -------- -
ct     TESTDB2  DB2AUTH  DBOAXSG  BP15        275 ADMR1      E BP16     N
       TESTDB4  DB2R6    TESTSG4  BP0         310 DB2R6      E BP0      N
       TESTDB5  DB2R6    TESTSG5  BP0         299 DB2R6      E BP0      N
       TESTDB6  DB2R6    TESTSG6  BP0         296 DB2R6      E BP0      N
```

Figure 9-90 Invoking Cloning Tool from Admin Tool

After you enter the line command, you then move to a standard Admin Tool panel where you can add additional table spaces. See Figure 9-91.

```
ADBPCTS n ---------------- DBOC Clone Table Spaces ----------- Row 1 to 4 of 4
Command ===>                                                   Scroll ===> PAGE

Commands: CONTINUE - Generate jobs  ADD - Add table spaces
 ADDRI - Add all RI related table spaces  ADDHIS - Add all history table spaces
Line commands:
 D - Delete  T - Show tables  ADDRI - Add RI related table spaces
 RIT - Show RI related tables  ADDHIS - Add history table spaces
 HIS - History table spaces

       Data     Table            No of           Table RI           VSAM Hist
Select Base     Space      Part Tables LOB   Relations Added     KB Used Added
       *        *             *      * *             * *              * *
------ -------- -------- ------ ------ --- ----------- ----- ---------- -----
       TESTDB2  TESTTS2       1      1 NO            0 NA          58480 NA
       TESTDB2  TESTTS2       2      1 NO            0 NA              8 NA
       TESTDB2  TESTTS2       3      1 NO            0 NA              8 NA
       TESTDB2  TESTTS2       4      1 NO            0 NA              8 NA
****************************** END OF DB2 DATA ******************************
```

Figure 9-91 Clone Table Spaces

Use the command CONTINUE and you are prompted to use an existing profile or to create a new profile. See Figure 9-92 on page 519.

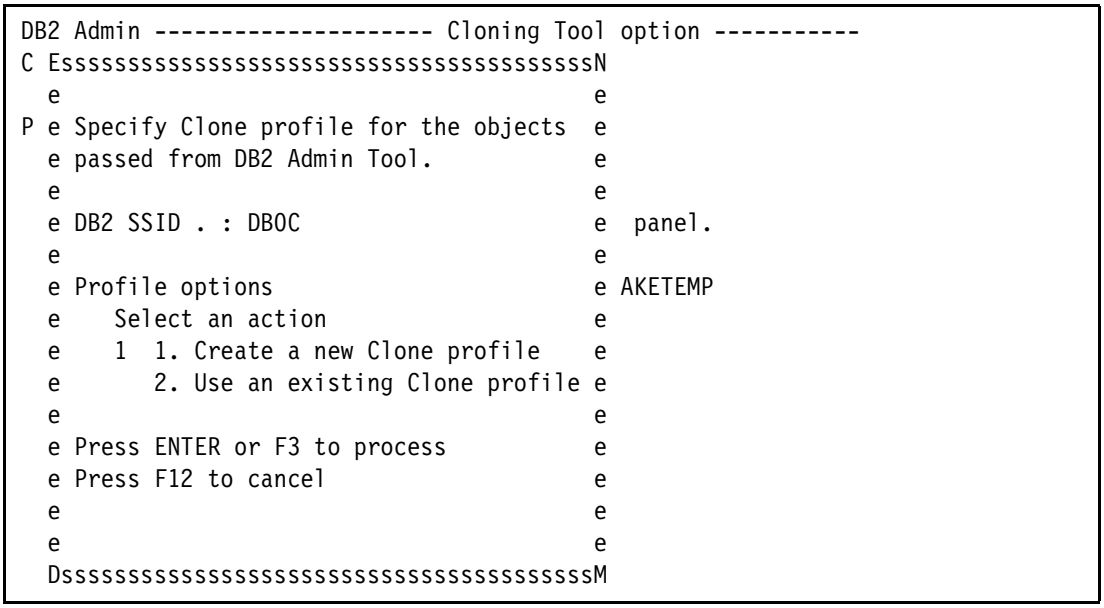

Figure 9-92 DB2 Cloning Tool Option

Build a generic profile from within the DB2 Cloning Tool ISP panels that can be used by the DB2 Administration Tool call. The example that we used for table space level cloning is a good place to start. Do not try to use the same profile, but create a copy and modify it to fit your needs.

For now, we use option 1. Create a new Clone profile on Figure 9-92 to create a new profile. See Figure 9-93.

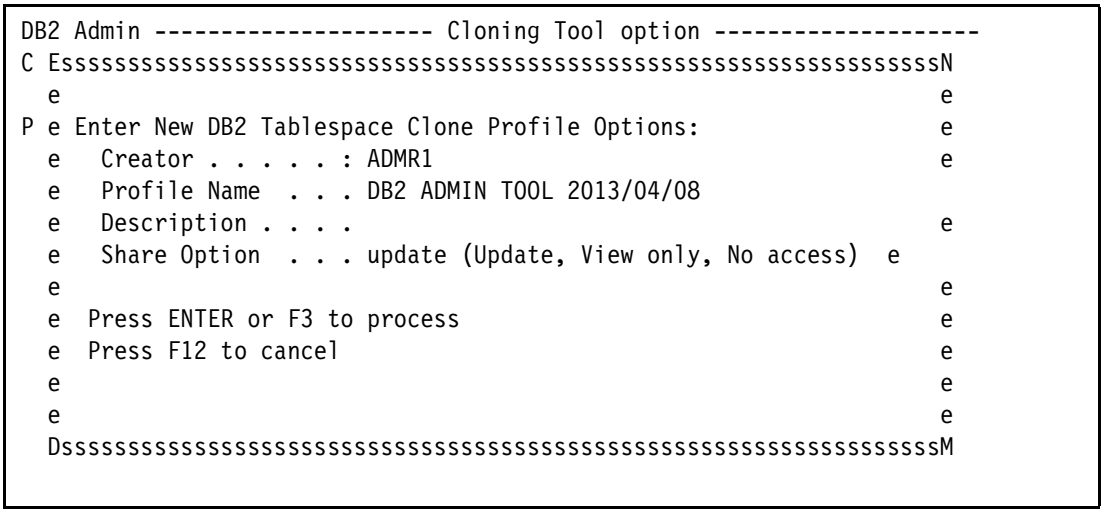

Figure 9-93 Cloning Tool option to create a new DB2 table space clone profile

We are now in the DB2 Cloning Tool. See Figure 9-94 on page 520.

```
CKZ1ETCP               Edit DB2 Tablespace Clone Profile
Option ===>

Creator  . . . : ADMR1        Name  . . . . : DB2 ADMIN TOOL 2013/04/08
Share Option . : UPDATE       Description . .                              >

1  Source job
2  Target job
3  Report job
4  TCPIP Server job
```

Figure 9-94 DB2 Cloning Tool from the DB2 Administration Tool

The new profile takes the DB2 Cloning Tool defaults that were previously set up. Without going through all the options and panels, we describe the key options.

From the source job, review the DD specifications by changing the HLQ and entering command D to reset all the data set names. Most of these data sets already exist. See Example 9-62.

Example 9-62 DD specifications

```
CKZ1DDSP             DB2 tablespace clone DD Specification
Command ===>                                                Scroll ===> PAGE

Commands:      D - Set Defaults  C - Clear Defaults  U - User DD Specification
Line commands: S - Select/Unselect

Creator  . . . : ADMR1        Name  . . . . : DB2 ADMIN TOOL 2013/04/08
Share Option . : UPDATE       Description . :

Control DD:
  HLQ  . . . . . ADMR1.TS1                          (control HLQ)
  Member . . . . ADBCLONE                           (control member)

                                                           Row 1 of 19   +
      DD Name  DD
  SEL CKZIN    *
  SEL CKZPRINT SYSOUT=*
  SEL CKZINI   DISP=SHR,DSN=DBTLSP.SCKZPARM(CKZINI)
  SEL CKZLOG   SYSOUT=*
  SEL CKZLSTDF DISP=SHR,DSN=ADMR1.TS1.LISTDEF(ADBCLONE)
      CKZMSKDF DISP=SHR,DSN=ADMR1.TS1.MASKDEF(ADBCLONE)
      CKZCRXML DISP=SHR,DSN=ADMR1.TS1.XMLCRDDL(ADBCLONE)
  SEL CKZS     DISP=OLD,DSN=ADMR1.TS1.SYNCDB2(ADBCLONE)
      CKZC     DISP=OLD,DSN=ADMR1.TS1.COPYDSNS(ADBCLONE)
      CKZM     DISP=OLD,DSN=ADMR1.TS1.XMLSTR(ADBCLONE)
  SEL CKZQ     DISP=OLD,DSN=ADMR1.TS1.SQLOUT(ADBCLONE)
      CKZW     DISP=OLD,DSN=ADMR1.TS1.CMDSSTPT(ADBCLONE)
      CKZX     DISP=OLD,DSN=ADMR1.TS1.CMDSSTPS(ADBCLONE)
      CKZY     DISP=OLD,DSN=ADMR1.TS1.CMDSSTRS(ADBCLONE)
```

To set the target for your clone, you need to update the copy command. By default, the target DB2 is your local DB2. Type an asterisk in this field for a selection list. See Example 9-63.

Example 9-63 Copy Command options

```
CKZ1COPC              DB2 tablespace clone COPY Command
Command ===>

Commands: S - Set SOURCE-PREFETCH-DATABASE-LIST  O - Set OBJECT-TRANSLATE
          T - Set TARGET-PREFETCH-DATABASE-LIST  J - Set JOB-TEMPLATE
          D - Set DDL-ATTRIBUTE-CHANGE  L - Set LOG-APPLY  I - Edit DB2 SSID

Creator  . . . : ADMR1       Name  . . . . : DB2 ADMIN TOOL 2013/04/08
Share Option . : UPDATE      Description . :
                                                              More:     +
TARGET-DB2 SSID  . . . . . . . . . . . DB0D     (asterisk to select from list)
  LOCATION . . . . . . . . . . . . . . DB0D
  USERID . . . . . . . . . . . . . . .
  PASSWORD . . . . . . . . . . . . . .
  SERVER-IP  . . . . . . . . . . . . .
  SERVER-PORT  . . . . . . . . . . . . 38390
  DEFVCAT  . . . . . . . . . . . . . . DB0DD
DATA-MOVER PGM . . . . . . . . . . . . ADRDSSU       (ADRDSSU, EMCAPI, or NONE)
  FASTREP  . . . . . . . . . . . . . . PREF          (PREF, REQ, or NONE)
  FCTOPPRCPRIMARY  . . . . . . . . . . NO            (Yes, No, PRESMIRREQ,
                                                      PRESMIRPREF, or
                                                      PRESMIRNONE)
  CMDDDNAME  . . . . . . . . . . . . .
PROCESS-DDL DDL-ENABLE . . . . . . . . NO          (Yes/No)
  PROCESS-TYPE . . . . . . . . . . . . NO          (YES, NO, GEN, EXEC, or ALL)
  PROCESS-DDL-DDN  . . . . . . . . . . : CKZDDB0C
  IGNORE-CREATE-OBJECT-EXISTS  . . . . YES         (Yes/No)
  GENERATE-DDL-DEFAULTS  . . . . . . . NO          (Yes/No)
  INCLUDE DDL-ATTRIBUTE-CHANGE . . . . NO          (Yes/No)
```

The final step is to look at the LISTDEF to verify that the scope is what you requested from the DB2 Administration Tool. See Example 9-64.

Example 9-64 LISTDEF from DB2 Administration Tool

```
CKZ1ELDC                    Edit LISTDEF Command
Option ===>

Creator  . . . : ADMR1       Name  . . . . : DB2 ADMIN TOOL 2013/04/08 V2
Share Option . : UPDATE      Description . :

Include/Exclude  . . . . . . . . . . INCLUDE     (INCLUDE, EXCLUDE)
Type Specification . . . . . . . . . TABLESPACE  (TABLESPACE, INDEXSPACE)
Copy . . . . . . . . . . . . . . . . : NO        (Yes/No)
Object Type  . . . . . . . . . . . . DATABASE    (DATABASE, TABLESPACE,
                                                  INDEXSPACE, TABLE, INDEX, or
                                                  STOGROUP)
Object Specification Qualifier 1 . . TESTDB2                              >
Object Specification Qualifier 2 . .                                      >
Partlevel  . . . . . . . . . . . . .
RI . . . . . . . . . . . . . . . . . NO          (Yes/No)
```

```
LOB Indicator Keywords . . . . . .            (ALL, LOB, BASE, XML, or
                                               blank)
Cloned . . . . . . . . . . . . . . . NO       (Yes, No, or blank)
```

Important: In the LISTDEF that is generated, Cloned is defaulted to NO. Change to blank to avoid missing a table space that you intended to clone.

These examples are the key areas. From this point, we can return to generate the jobs based on normal DB2 Cloning Tool procedures.

Before you use the DB2 Administration Tool interface, ensure that you perform a trial with the appropriate profiles from within the DB2 Cloning Tool first.

9.10 Runtime repository

A target job runtime repository is available to allow restarting the target job and to show target job reports.

The repository keeps track of target jobs and all the data sets that are processed by the target job. Furthermore, the repository allows the failed target job to be restarted, skipping any successfully processed target data sets. Examples of the reports follow.

Job report

See Example 9-65 for a shortened form of the RRJREPS Job Report.

Example 9-65 Short form Job report

```
SRC       SRC       SRC                    TRG    TRG       TRG       TRG
           JOB    JOB       JOB      JOB            TRG     JOB    JOB       JOB      JOB     SRC   TRG
   TYPE   DATE   TIME      NAME     NUMBER   NR   SYSTEM    DATE   TIME      NAME     NUMBER   SSID  SSID  TRG JOB RC  TRG JOB RS
   ----  ----- --------  --------  --------  --  --------  ----- --------  --------  --------  ----  ----  ----------  ----------
   RRJB  12197 10:42:05  ADMR1G    B19445 *  01  SC63      12197 10:51:37  ADMR1G    JOB19449  DBOC  DBOD  X'00000000  X'00000000
   RRJB  13063 14:23:53  ADMR1D    B19643 *  01  SC63      13063 14:46:59  ADMR1D    JOB19645  DBOC  DBOD  X'00000000  X'00000000
   RRJB  13064 13:07:16  ADMR1D    B19917 *  01  SC63      13064 13:28:26  ADMR1D    JOB19926  DBOC  DBOD  X'00000000  X'00000000
   RRJB  13064 14:03:50  ADMR1D    B19930 *  01  SC63      13064 14:05:47  ADMR1D    JOB19931  DBOC  DBOD  X'00000000  X'00000000
   RRJB  13064 14:43:04  ADMR1H    B19941 *  01  SC63      13064 14:56:26  ADMR1G    JOB19954  DBOC  DBOD  X'00000000  X'00000000
   RRJB  13064 19:59:03  ADMR1D    B20063 *  01  SC63      13064 20:03:59  ADMR1D    JOB20064  DBOC  DBOD  X'00000000  X'00000000
   RRJB  13065 12:39:25  ADMR1H    B20188 *  01  SC63      13065 12:46:39  ADMR1H    JOB20193  DBOC  DBOD  X'00000000  X'00000000
   RRJB  13065 14:53:15  ADMR1H    B20218 *  01  SC63      13065 14:59:28  ADMR1H    JOB20222  DBOC  DBOD  X'00000000  X'00000000
   RRJB  13065 19:31:52  ADMR1H    B20338 *  01  SC63      13066 12:14:50  ADMR1G    JOB20403  DBOC  DBOD  X'00000008  X'00000000
   RRJB  13066 17:29:46  ADMR1I    B20518 *  01  SC63      13066 17:40:04  ADMR1I    JOB20521  DBOC  DBOD  X'00000000  X'00000000
   RRJB  13066 18:35:02  ADMR1I    B20536 *  01  SC63      13066 18:42:21  ADMR1I    JOB20540  DBOC  DBOD  X'00000000  X'00000000
   RRJB  13080 09:58:17  ADMR1J    B21663 *  01  SC63      13080 10:00:20  ADMR1J    JOB21664  DBOC  DBOD  X'00000000  X'00000000
```

Status report

There are two forms of the status report: a short version and a long version. These reports show the status of the target job activity and what will be restarted.

See Example 9-66 for a Status Report in short form.

Example 9-66 Status report short form

```
S   C
                                                                          PT  L     LOG       DATA
                                                                          AY  O    APPLY    MASKING
                                                                 ELAPSED  CP  N    PAGES     PAGES      VSAM     VSAM
   TYPE   DATE   TIME                TARGET DATASET                RC   SECONDS  EE  E   CHANGED   CHANGED    READS    WRITES
   ----  ----- --------  ------------------------------------------  --  --------  --  -  --------  --------  --------  --------
```

```
   RRDS  12197 10:51:37  DBODD.DSNDBC.GLWSAMP2.GLWSDPT.IO001.A001       0         0  TS            0         0       372       251
   RRDS  12197 10:51:38  DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.A001       0         1  TS            0         0       936       600
   RRDS  12197 10:51:38  DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.A002       0         0  TS            0         0       936       593
   RRDS  12197 10:51:38  DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.A003       0         0  TS            0         0       936       595
   RRDS  12197 10:51:39  DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.A004       0         1  TS            0         0       936       516
   RRDS  12197 10:51:39  DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.A001       0       256  TS            0         0   1048320   1048320
   RRDS  12197 10:56:08  DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.A002       0       248  TS            0         0   1048320   1048320
   RRDS  12197 11:00:28  DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.A003       0       256  TS            0         0   1048320   1048320
   RRDS  12197 11:04:57  DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.A004       0       175  TS            0         0    715575    715575
   RRDS  12197 11:08:00  DBODD.DSNDBC.GLWSAMP2.GLWSPJA.IO001.A001       0         0  TS            0         0      5940      5544
   RRDS  12197 11:08:00  DBODD.DSNDBC.GLWSAMP2.GLWSPRJ.IO001.A001       0         0  TS            0         0      2556      2392
   RRDS  12197 11:08:01  DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.A001       0         0  TS            0         0         2         2
   RRDS  12197 11:08:01  DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.A002       0         0  TS            0         0         2         2
   RRDS  12197 11:08:01  DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.A003       0         0  TS            0         0         2         2
   RRDS  12197 11:08:01  DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.A004       0         0  TS            0         0      1080       868
   RRDS  12197 11:08:01  DBODD.DSNDBC.GLWSAMP2.GLWSO01.IO001.A001       0         0  TS            0         0        60        25
   RRDS  12197 11:08:02  DBODD.DSNDBC.GLWSAMP2.GLWSO02.IO001.A001       0         0  TS            0         0       372         2
   RRDS  12197 11:08:02  DBODD.DSNDBC.GLWSAMP2.GLWSO03.IO001.A001       0         0  TS            1         0       372         3
   RRDS  12197 10:51:38  DBODD.DSNDBC.GLWSAMP2.GLWXACT1.IO001.A001      0         0  IS            0         0        12         5
   RRDS  12197 10:51:38  DBODD.DSNDBC.GLWSAMP2.GLWXDNG1.IO001.A001      0         0  IS            0         0        12         5
   RRDS  12197 10:51:38  DBODD.DSNDBC.GLWSAMP2.GLWXDPT1.IO001.A001      0         1  IS            0         0        36        33
   RRDS  12197 10:51:38  DBODD.DSNDBC.GLWSAMP2.GLWXDPT2.IO001.A001      0         0  IS            0         0       396        54
   RRDS  12197 10:51:38  DBODD.DSNDBC.GLWSAMP2.GLWXEMP1.IO001.A001      0         0  IS            0         0       396       198
   RRDS  12197 10:51:38  DBODD.DSNDBC.GLWSAMP2.GLWXEMP1.IO001.A002      0         0  IS            0         0       396       195
```

A more detailed report is the long status report, which shows the relationship between the source and target objects. In both reports, an RC of 0 indicates that the clone is completed for that relationship. See Example 9-67.

Example 9-67 Status report in long form

```
V

S

I  A

S   C                                      O  M

PT  L     LOG       DATA

AY  O    APPLY    MASKING                   E  E

ELAPSED  CP  N    PAGES     PAGES      VSAM     VSAM    R  R
   TYPE   DATE   TIME                   SOURCE DATASET                          TARGET DATASET                  RC    PAGES
SECONDS  EE  E   CHANGED   CHANGED    READS    WRITES   R  R
   ----  ----- --------  ---------------------------------------------  ----------------------------------------  --  --------
--------  --  -  --------  --------  --------  --------  -  -
   RRDS  12197 10:51:37  DBOCD.DSNDBC.GLWSAMP.GLWSDPT.IO001.A001       DBODD.DSNDBC.GLWSAMP2.GLWSDPT.IO001.A001       0       372
0  TS           0         0       372       251
   RRDS  12197 10:51:38  DBOCD.DSNDBC.GLWSAMP.GLWSEMP.IO001.A001       DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.A001       0       936
1  TS           0         0       936       600
   RRDS  12197 10:51:38  DBOCD.DSNDBC.GLWSAMP.GLWSEMP.IO001.A002       DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.A002       0       936
0  TS           0         0       936       593
   RRDS  12197 10:51:38  DBOCD.DSNDBC.GLWSAMP.GLWSEMP.IO001.A003       DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.A003       0       936
0  TS           0         0       936       595
   RRDS  12197 10:51:39  DBOCD.DSNDBC.GLWSAMP.GLWSEMP.IO001.A004       DBODD.DSNDBC.GLWSAMP2.GLWSEMP.IO001.A004       0       936
1  TS           0         0       936       516
   RRDS  12197 10:51:39  DBOCD.DSNDBC.GLWSAMP.GLWSEPA.IO001.A001       DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.A001       0   1048320
256  TS           0         0   1048320   1048320
   RRDS  12197 10:56:08  DBOCD.DSNDBC.GLWSAMP.GLWSEPA.IO001.A002       DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.A002       0   1048320
248  TS           0         0   1048320   1048320
   RRDS  12197 11:00:28  DBOCD.DSNDBC.GLWSAMP.GLWSEPA.IO001.A003       DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.A003       0   1048320
256  TS           0         0   1048320   1048320
   RRDS  12197 11:04:57  DBOCD.DSNDBC.GLWSAMP.GLWSEPA.IO001.A004       DBODD.DSNDBC.GLWSAMP2.GLWSEPA.IO001.A004       0    715575
175  TS           0         0    715575    715575
   RRDS  12197 11:08:00  DBOCD.DSNDBC.GLWSAMP.GLWSPJA.IO001.A001       DBODD.DSNDBC.GLWSAMP2.GLWSPJA.IO001.A001       0      5940
0  TS           0         0      5940      5544
   RRDS  12197 11:08:00  DBOCD.DSNDBC.GLWSAMP.GLWSPRJ.IO001.A001       DBODD.DSNDBC.GLWSAMP2.GLWSPRJ.IO001.A001       0      2556
0  TS           0         0      2556      2392
   RRDS  12197 11:08:01  DBOCD.DSNDBC.GLWSAMP.GLWSSPL.IO001.A001       DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.A001       0         2
0  TS           0         0         2         2
   RRDS  12197 11:08:01  DBOCD.DSNDBC.GLWSAMP.GLWSSPL.IO001.A002       DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.A002       0         2
0  TS           0         0         2         2
   RRDS  12197 11:08:01  DBOCD.DSNDBC.GLWSAMP.GLWSSPL.IO001.A003       DBODD.DSNDBC.GLWSAMP2.GLWSSPL.IO001.A003       0         2
0  TS           0         0         2         2
```

To use the repository, set up the following information and check it:

- Verify that USE-RUNTIME-REPOSITORY of the SET command is set to Y. See the “SET command” on page 484.

  Specify YES in this field is to indicate that when the target job is executed, it is to only process the data sets that have not been successfully processed. The restart process is enabled by a DB2 Cloning Tool repository. The repository tracks target jobs and all the data sets that are processed by the target job. This allows the failed target job to be restarted, skipping the successfully processed target data sets.

- Ensure that the repository data sets that are needed are defined.

  If you use the repository for rerunning the target job, keeping a history of both, both repository data sets must be in the source job, target job, and report job.

These are the key-sequenced data sets (KSDS) that are required in both the source job and the target job:

```
//CKZRRJOB DD DISP=OLD,DSN=&hlq.RRJOB
//CKZRRDSN DD DISP=OLD,DSN=&hlq.RRDSN
```

The data sets that are required should have been created by the TCZ installation. JCL members B3TS1 and B3TS2 have the JCL for creating these data sets.

The following report output data sets are allocated. They are in B3TS2, which is used by the report job:

```
//CKZJREPS DD  DISP=SHR,DSN=ADMR1.TS2.RRJREPS2(GLWSAMP7)
//CKZDREPL DD  DISP=SHR,DSN=ADMR1.TS2.RRDREPL(GLWSAMP7)
//CKZDREPS DD  DISP=SHR,DSN=ADMR1.TS2.RRDREPS(GLWSAMP7)
```

Dataset LRECL: Check the LRECL of the data sets that are created by TCZ. These are the correct LRECLs:

DD Name	RECFM	LRECL	DESCRIPTION
CKZDREPL	FBA	201	dataset long report
CKZDREPS	FBA	133	dataset short report
CKZJREPL	FBA	133	job long report

The REPORT-JOB is in member CKZREPJB of product library *.SCKZJCL, but it can also be built and run from the ISPF panels, as shown in Example 9-68.

Example 9-68 Generate Report Job

```
CKZ1BDCJ               Build DB2 tablespace clone jobs
Option ===>

1  Generate Source and Target Jobs
2  Generate Report Job
3  Generate TCPIP Server Job
```

The created job looks like the job that is shown in Example 9-69.

Example 9-69 Report repository job

```
//S1      EXEC PGM=CKZ00500,REGION=0M
//STEPLIB  DD  DISP=SHR,DSN=DBTLSP.SCKZLOAD
//CKZINI   DD  DISP=SHR,DSN=DBTLSP.SCKZPARM(CKZINI)
```

```
//CKZLOG   DD  SYSOUT=*
//CKZPRINT DD  SYSOUT=*
//CKZRRJOB DD  DISP=OLD,DSN=ADMR1.TS2.RRJOB
//CKZRRDSN DD  DISP=OLD,DSN=ADMR1.TS2.RRDSN
//CKZJREPS DD  DISP=SHR,DSN=ADMR1.TS2.RRJREPS2(GLWSAMP7)
//CKZDREPL DD  DISP=SHR,DSN=ADMR1.TS2.RRDREPL(GLWSAMP7)
//CKZDREPS DD  DISP=SHR,DSN=ADMR1.TS2.RRDREPS(GLWSAMP7)
//CKZIN    DD  *
   SET REPORT-JOB(Y)
//*
```

With the settings in place, handling a failure in the target job means that you only need to resubmit the target job after you resolve the issue.

Example 9-70 shows the messages that you can see in the target jobs that indicate the use of the repository.

Example 9-70 Target job messages for report repository

```
CKZ55104I VSAM Repository Dataset Open, DD is CKZRRJOB
and
CKZ55101I Repository Use in Effect
```

The repository keeps track of target jobs and all the data sets that are processed by the target job. Furthermore, the repository allows the failed target job to be restarted, skipping any successfully processed target data sets. This provides integrity in the process and visibility to see what your clone has done.

Part 4

Appendixes

In this part, we include the following appendixes:

A

Test environment

This appendix describes the test environment hardware and the software setup for this project. We describe these topics:

- Appendix A.1, “Our test environment” on page 530 describes the components of our test environment.
- Appendix A.2, “Stored procedures workload” on page 537 describes our installation and customization of the DB2 Workload Generator.
- Appendix A.3, “Accessing the tools” on page 539 lists our subsystem parameters.

A.1 Our test environment

This section provides some details about the test environment that we had at our disposal. All DB2 subsystems ran on the same logical partition (LPAR); we did not have a data sharing environment. This section describes the environment in which we performed our tests. Figure A-1 shows our starting configurations.

We are running on z196 2817-M32 software model 716 (with 16 general-purpose CPs).There are three shared regular CPs on our LPAR (SC63). No zIIP or zAAP processors are online to SC63, but two of each type of processor are available to that LPAR. The amount of real storage is 8 GB. The operating system is IBM z/OS 1.13 and we run DB2 10 for z/OS. We have three stand-alone subsystems, DB0A, DB0C, and DB0D, and no data sharing.

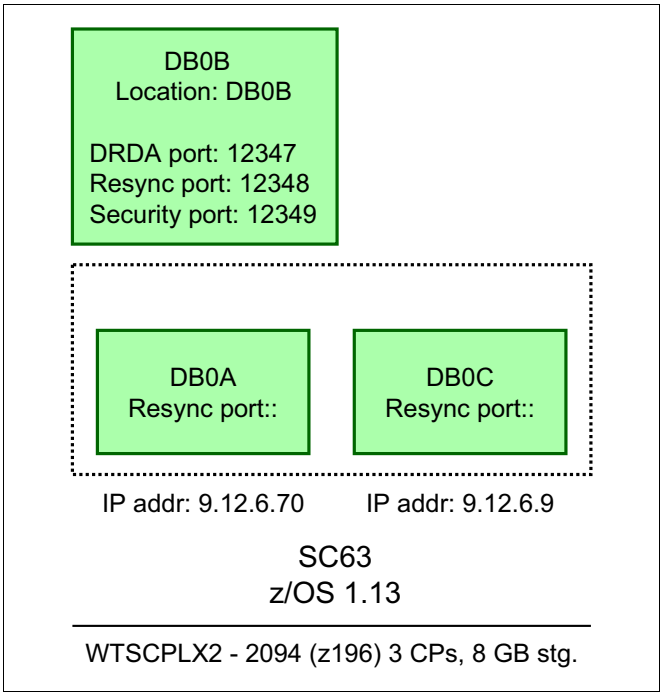

Figure A-1 DB2 configurations

A.1.1 Summary of the hardware configuration

Table A-1 Hardware configuration

CPU	**Engines**	**zIIPs and zAAPs**
z196 2817-M32	2/3 CPUs online	Two of each available
Storage	**Type**	**FlashCopy available**
3390	Model 9 and Model 27	Yes
Tape media	**Type**	**Model**
Virtual Tape Server (VTS)	3494	B20

A.1.2 Software configuration

The z/OS software was z/OS 1.13 running on LPAR SC63 with DB2 10 installed. Initially, there were two DB2 subsystems: DB0A and DB0B. Both were set at DB2 new function mode (NFM). During our scenarios, we used DB0A as our source test environment and cloned to create DB2 subsystem DB0C and even a DB0D. Normal ISPF and TSO services were available, as well as the relevant DB2 Tools software products that we used to generate and test scenarios.

A.1.3 SMS copypools, I/O configurations, and naming conventions

Our test environment had sufficient DASD to enable us to perform the FlashCopy and cloning that were required during our test scenarios. We describe the details of the storage management subsystem (SMS) constructs that were in place for our DB0A environment. These were effectively replicated into the cloned DB0C subsystem.

Storage groups

The following table lists the storage groups used by the DB0B DB2 subsystem.

Table A-2 DB0B storage groups

Storage group	Type	CPB	Use
DB0AARCH	POOL	DB0ACPB	Archive Logs
DB0ACPB	CPB		
DB0ADATA	POOL	DB0ACPB	DB2 Data
DB0ALOG1	POOL	DB0ACPB	DB2 Logs Copy 1
DB0ALOG2	POOL	DB0ACPB	DB2 Logs Copy 2
DB0AMISC	POOL		Data
DB0ATARG	POOL		DB2 Target libraries
DB0AIMAG	POOL		Image copies

Volumes allocated to storage groups

The following figures list the volumes assigned to each storage group. The model type is indicated by the TOTAL_CP field. For example, 24359M is a 3390 model 27, and 8192M is a 3390 model 9.

The volumes that are assigned to storage group DB0AARCH are displayed in Figure A-2 on page 532.

```
1READY
  ADMSMS DB0A S,DB0AARCH
 ################################################################################
 # ADMSYSRX V110     ADMIN_SMS_INFO SP to list SMS details  12/08/24 08:46:58 #
 ################################################################################

 --------------------------------------------------------------------------------
 VOLUME COPY_POOL    STORGRP  TOTAL_CP   FREE_SPACE   LARGEST_FREE PERCENT_FREE
 SB0X8A              DB0AARCH   24359M       16724M          9485M         68.66
 SB0X8B              DB0AARCH   24359M       16863M          9803M         69.23
 SB0X8W              DB0AARCH   24359M       17266M         12522M         70.88
 SB0X8X              DB0AARCH   24359M       15759M         11100M         64.69
 X3642E              DB0AARCH   24359M       23980M         23979M         98.44
 X3662E              DB0AARCH   24359M       23636M         23633M         97.03
 X3672E              DB0AARCH   24359M       23980M         23979M         98.44
 X7603C              DB0AARCH   24359M       22456M         22451M         92.19
 X76124              DB0AARCH   24359M       23361M         23359M         95.90
 X76224              DB0AARCH   24359M       22871M         22866M         93.89
 X76324              DB0AARCH   24359M       22908M         22905M         94.04
```

Figure A-2 Volumes assigned to storage group DB0AARCH

The data storage group, DB0ADATA, is made up of 3390 model 9s. The volumes assigned to storage group DB0ADATA are displayed in Figure A-3.

```
READY
 ADMSMS DB0A S,DB0ADATA
################################################################################
# ADMSYSRX V110     ADMIN_SMS_INFO SP to list SMS details  12/08/24 08:58:03 #
################################################################################

--------------------------------------------------------------------------------
VOLUME COPY_POOL    STORGRP  TOTAL_CP   FREE_SPACE   LARGEST_FREE PERCENT_FREE
SB0X8M              DB0ADATA    8120M        1086M           888M         13.37
SB0X8N              DB0ADATA    8120M        2726M          2706M         33.57
SB0X8O              DB0ADATA    8120M        2375M          1822M         29.25
SB0X8P              DB0ADATA    8120M        2461M          1273M         30.31
READY
END
```

Figure A-3 Volumes assigned to storage group DB0ADATA

The DB2 LOG1 and LOG2 storage groups have only model 3390-9 allocated to each group.

The volumes assigned to storage group DB0ALOG1 and DB0ALOG2 are displayed in Figure A-4 on page 533.

```
READY
 ADMSMS DB0A S,DB0ALOG1
################################################################################
# ADMSYSRX V110     ADMIN_SMS_INFO SP to list SMS details  12/08/24 10:17:33 #
################################################################################

--------------------------------------------------------------------------------
VOLUME COPY_POOL    STORGRP  TOTAL_CP   FREE_SPACE   LARGEST_FREE PERCENT_FREE
SBOX8S              DB0ALOG1    8120M        7660M          7659M        94.33
READY
 ADMSMS DB0A S,DB0ALOG2
################################################################################
# ADMSYSRX V110     ADMIN_SMS_INFO SP to list SMS details  12/08/24 10:17:34 #
################################################################################

--------------------------------------------------------------------------------
VOLUME COPY_POOL    STORGRP  TOTAL_CP   FREE_SPACE   LARGEST_FREE PERCENT_FREE
SBOX8T              DB0ALOG2    8120M        7758M          7756M        95.54
READY
END
```

Figure A-4 Volumes assigned to the DB0ALOG1 and DB0ALOG2 storage groups

The DB0AMISC and DB0ATARG each have one 3390-9 allocated.

The volumes assigned to storage groups DB0AMISC and DB0ATARG are displayed in Figure A-5.

```
READY
 ADMSMS DB0A S,DB0ATARG
################################################################################
# ADMSYSRX V110     ADMIN_SMS_INFO SP to list SMS details  12/08/24 10:22:52 #
################################################################################

--------------------------------------------------------------------------------
VOLUME COPY_POOL    STORGRP  TOTAL_CP   FREE_SPACE   LARGEST_FREE PERCENT_FREE
SBOX8V              DB0ATARG    8120M        4377M          4376M        53.90
READY
 ADMSMS DB0A S,DB0AMISC
################################################################################
# ADMSYSRX V110     ADMIN_SMS_INFO SP to list SMS details  12/08/24 10:22:52 #
################################################################################

--------------------------------------------------------------------------------
VOLUME COPY_POOL    STORGRP  TOTAL_CP   FREE_SPACE   LARGEST_FREE PERCENT_FREE
SBOX8U              DB0AMISC    8120M        4766M          4766M        58.69
READY
END
```

Figure A-5 Volumes assigned to the DB0ATARG and DB0AMISC storage groups

The image copy backup pool consists of a mixture of 3390-27 and 3390-9.

The volumes assigned to storage group DB0AIMAG are displayed in Figure A-6 on page 534.

```
 ADMSMS DB0A S,DB0AIMAG
################################################################################
# ADMSYSRX V110     ADMIN_SMS_INFO SP to list SMS details  12/08/24 10:39:41 #
################################################################################

--------------------------------------------------------------------------------
VOLUME COPY_POOL    STORGRP  TOTAL_CP   FREE_SPACE   LARGEST_FREE PERCENT_FREE
SB0X8Q              DB0AIMAG   24359M           0M             0M            0
SB0X8R              DB0AIMAG   24359M         195M           195M       0.8005
X3D33A              DB0AIMAG    8120M           0M             0M            0
X3D333              DB0AIMAG    8120M           0M             0M            0
X3D334              DB0AIMAG    8120M           0M             0M            0
X3D335              DB0AIMAG    8120M           0M             0M            0
X3D336              DB0AIMAG    8120M           0M             0M            0
X3D337              DB0AIMAG    8120M        1293M          1270M        15.92
X3D338              DB0AIMAG    8120M        4109M          4101M        50.60
X3D339              DB0AIMAG    8120M        4144M          4136M        51.03
X3612B              DB0AIMAG   24359M       19543M         19542M        80.23
X36130              DB0AIMAG   24359M       10082M         10082M        41.39
X3622B              DB0AIMAG   24359M         345M           344M        1.416
X36230              DB0AIMAG   24359M       10163M         10161M        41.72
X3632B              DB0AIMAG   24359M       17962M         13360M        73.74
X36330              DB0AIMAG   24359M       10045M          5439M        41.24
X3642B              DB0AIMAG   24359M       24328M         24328M        99.87
```

Figure A-6 DB2 image copy SMS pool DB0AIMAG

The copypool backup has to have sufficient volume numbers and model types to cater for any FlashCopy invocation, such as BACKUP SYSTEM.

Figure A-7 on page 535 is a volume list of our test environment copypool backup.

```
READY
 ADMSMS DBOA S,DBOACPB
################################################################################
# ADMSYSRX V110     ADMIN_SMS_INFO SP to list SMS details  12/08/24 10:26:58 #
################################################################################

--------------------------------------------------------------------------------
VOLUME COPY_POOL    STORGRP  TOTAL_CP   FREE_SPACE   LARGEST_FREE PERCENT_FREE
SBOX8C              DBOACPB    24359M       24352M         24352M        99.97
SBOX8D              DBOACPB    24359M       24352M         24352M        99.97
SBOX8E              DBOACPB     8120M        8113M          8112M        99.91
SBOX8F              DBOACPB     8120M        8113M          8112M        99.91
SBOX8G              DBOACPB     8120M        8113M          8112M        99.91
SBOX8H              DBOACPB     8120M        8113M          8112M        99.91
SBOX8I              DBOACPB     8120M        8113M          8112M        99.91
SBOX8J              DBOACPB     8120M        8113M          8112M        99.91
SBOX8K              DBOACPB     8120M        8113M          8112M        99.91
SBOX8L              DBOACPB     8120M        8113M          8112M        99.91
XZ6135              DBOACPB    24359M       24346M         24345M        99.95
XZ6138              DBOACPB    24359M       24346M         24345M        99.95
XZ6139              DBOACPB    24359M       24346M         24345M        99.95
XZ6234              DBOACPB    24359M       24346M         24345M        99.95
XZ6237              DBOACPB    24359M       24346M         24345M        99.95
XZ6334              DBOACPB    24359M       24346M         24345M        99.95
XZ6338              DBOACPB    24359M       24346M         24345M        99.95
X0663E              DBOACPB    24359M       24346M         24345M        99.95
X0663F              DBOACPB    24359M       24346M         24345M        99.95
X0673B              DBOACPB    24359M       24346M         24345M        99.95
X0673C              DBOACPB    24359M       24346M         24345M        99.95
X0673D              DBOACPB    24359M       24346M         24345M        99.95
X0673E              DBOACPB    24359M       24346M         24345M        99.95
X0673F              DBOACPB    24359M       24346M         24345M        99.95
X3612E              DBOACPB    24359M       24346M         24345M        99.95
X3622E              DBOACPB    24359M       24346M         24345M        99.95
X3632E              DBOACPB    24359M       24346M         24345M        99.95
..... another 50 3390-27 in the X36*** address range.
YXD01E              DBOACPB     8120M        8107M          8106M        99.84
YXD11E              DBOACPB     8120M        8107M          8106M        99.84
YXD260              DBOACPB     8120M        8107M          8106M        99.84
YXD31C              DBOACPB     8120M        8107M          8106M        99.84
READY
END
```

Figure A-7 3390-27s and 3390-9s making up the copypool backup

The SMS storage classes are associated with the same named SMS group names. The following tables describe the association with the SMS group, class, and high-level qualifiers (HLQs) associated with the SMS classes:

- Copypool names

  Table A-3 on page 536 describes how the DB2 copypool names are associated with the SMS storage group names.

Table A-3 Copypool names

Copypool names	Storage group	Storage group	Storage group
DSN$DB0A$DB	DB0ADATA		
DSN$DB0A$LG	DB0AARCH	DB0ALOG1	DB0ALOG2

- HLQs associated with the SMS class

  Table A-4 shows the SMS class and data set HLQ associations.

Table A-4 SMS class and data set HLQ

SMS class	FILTLIST	HLQs
DB0AARCH	DB0AARCH	DB0AA.*.ARCHLOG*.**
DB0ADATA	DB0ADATA	DB0AD.**
DB0ALOG1	DB0ALOG1	DB0AL.*.LOGCOPY1.**, DB0AB.*.BSDS01
DB0ALOG2	DB0ALOG2	DB0AL.*.LOGCOPY1.**, DB0AB.*.BSDS01
DB0AMISC	DB0AMISC	DB0AM.**
DB0ATARG	DB0ATARG	DB0AT.**
DB0AIMAG	DB0AIMAG	DB0BI.**

- Catalog and alias associations

  Table A-5 shows the catalogs and aliases to support the above SMS constructs and DB0A infrastructure.

Table A-5 Catalog and alias

Catalog	Alias	Example
UCAT.DB0ADATA	DB0AD	DB0AD.DSNDB%.**
UCAT.DB0BLOGS	DB0AL	DB0AL.LOGCOPY%.**
	DB0AA	DB0AA.ARCHLOG*.**
	DB0AB	DB0AB.BSDS0*.**
UCAT.DB0ATARG	DB0AT	DB0AT.**
UCAT.DB0BMISC	DB0AM	Miscellaneous Files
UCAT.DB0BIMAG	DB0AI	Image copy data sets

- Miscellaneous files include some of the following data sets:
  - DB0AM.COPYDDN.DSN8D10P.DSN8S10C
  - DB0AM.DBRMLIB.DATA
  - DB0AM.DB0BWLMJ.JAVAENV
  - DB0AM.DSN8.LISTDEF
  - DB0AM.DSN8D10A.DSN8S10E.PART3
  - DB0AM.DSN8D10A.DSN8S10E.P00001
  - DB0AM.DSN8D10A.DSN8S10E.P00003
  - DB0AM.DSN8D10A.DSN8S10E.P00004

A.2 Stored procedures workload

We have used a sample application based on the stored procedures workload, which provides a simple way of creating and driving a substantial activity on a DB2 for z/OS database.

The database is composed of five principal tables (GLWTDPT, GLWTEMP, GLWTPRJ, GLWTPJA, and GLWTEPA), and 12 supporting tables. The database includes views, referential integrity (RI), partitioned tables, and triggers. The database uses DB2 10 functions.

There are 16 stored procedures:

- DPTADD
- DPTUPD
- DPTUPR
- DPTMGR
- DPTLCK
- DPTBAL
- DPTDEL
- EMPADD
- EMPUPD
- EMPUPR
- EMPDEL
- EMPQRY
- EMPQR2
- EMPFND
- PRJADD
- PRJUPD

There are 12 supporting procedures.

In Figure A-8 on page 538, we show the data schema.

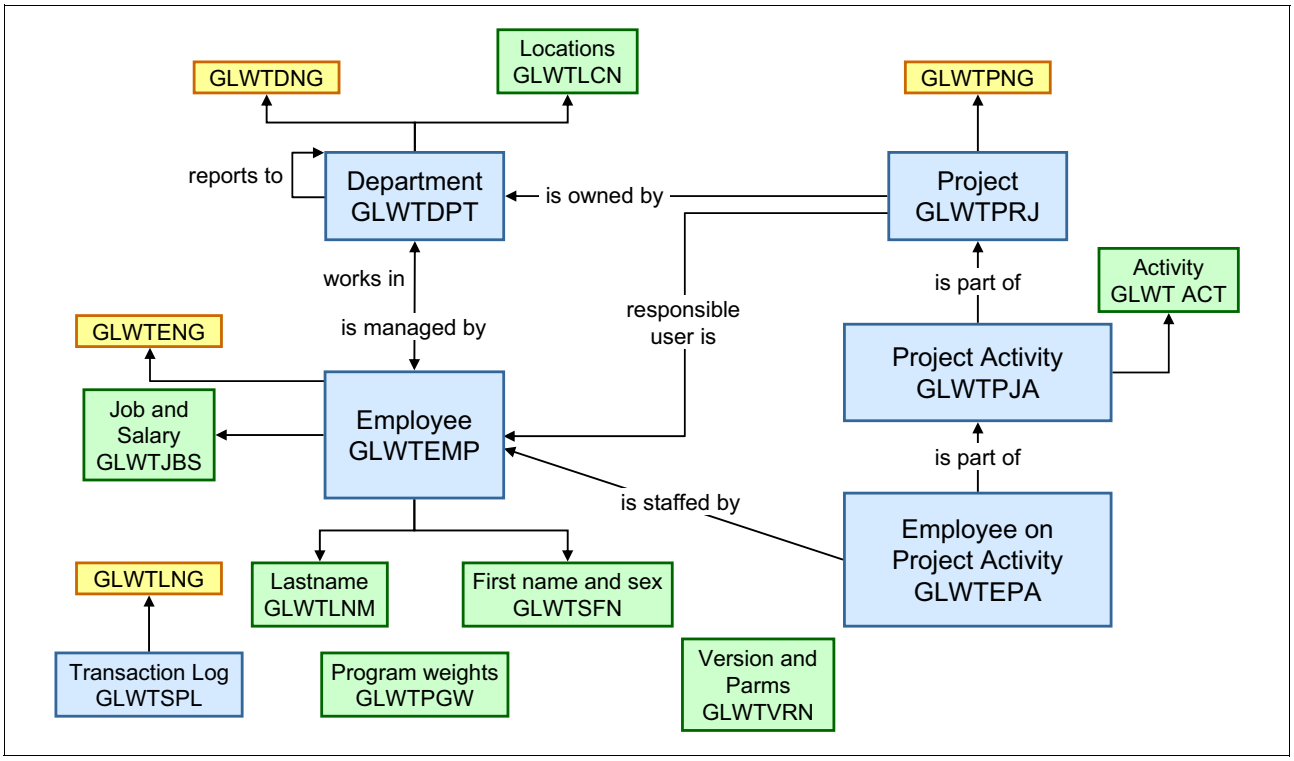

Figure A-8 GLW database

Table A-6 summarizes the characteristics of the tables in the GLW database.

Table A-6 GLW table profiles

Table name	View name	Content	Number of columns	Number of rows (before workload)	Number of rows (after 10-minute workload)
GLWTDPT	WLDEPT	Departments	10	0	588
GLWTEMP	WLEMP	Employees	19	0	7203
GLWTPRJ	WLPROJ	Projects	13	0	1171
GLWTPJA	WLPROJACT	Activities on each project	10	0	21078
GLWTEPA	WLEMPPROJACT	Employees assigned to each activity on each project	10	0	68184
GLWLCN	WLLOCN	Departmental locations	2	130	130
GLWJBS	WLJOBSALARY	Employee jobs and salaries	4	15	15
GLWTLNM	WLLASTNAME	Employee last names	2	351	351
GLWTSFN	WLSEXFIRSTNAME	Employee sex and first names	3	84	84
GLWACT	WLACT	Activities	3	18	18
GLWTDNG	WLDEPT_NO_GEN	Number generator for GLWTDPT	2	0	0

Table name	View name	Content	Number of columns	Number of rows (before workload)	Number of rows (after 10-minute workload)
GLWTENG	WLEMP_NO_GEN	Number generator for GLWTEMP	2	0	0
GLWTPNG	WLPROJ_NO_GEN	Number generator for GLWTPRJ	2	0	0
GLWPNG	WLTRAN_NO_GEN	Number generator for GLWTSPL	2	0	0
GLWTSPL	WLSPLOG	Log of each stored procedure CALL by the driver	6	0	14696
GLWTPGW	WLPROGWT	Runtime weighting for each stored procedure	3	119	119
GLWTVRN	WLVERSN	Database version control table	24	1	1

A.3 Accessing the tools

We invoke the various DB2 Tools used in our scenarios from the DB2TOOLS primary panel, which is accessed by issuing the following command:

```
TSO DB2TOOLS
```

This procedure was implemented during this project in order to facilitate the implementation of scenarios utilizing several tools. It is available for download as described in Appendix C, “Additional material” on page 551.

We are presented with the selection panel in Figure A-9 from where we can access the various tools that have been configured in our environment.

```
-------------------- DB2 Tools Primary Options Menu -------------------
OPTION ===>
                                                          USERID - ADMR2
Select an option from below.                              DATE   - 12/04/24
                                                          TIME   - 17:21
     A Admin.       - Administration Tool
     T Automation   - Automation Tool
     C Cloning      - Cloning Tool
     I HPU          - High Performance Unload
     R RE           - Recovery Expert
     U UET          - Utilities Enhancement Tool

     L TCZ          - DB2 Launchpad
     Z TCZ          - Toolkit Customization

     X EXIT         - Exit
```

Figure A-9 DB2 Tools Primary Menu

B

DSNZPARMs and general settings

The DSNZPARM setup that we used during our testing was configured with certain options. This setup gave us total control of the operations of several of the utilities. This setup helped us to avoid unwanted defaults that might compromise our testing. This appendix lists the description of the DSNZPARMs as they relate to the use of tools and utilities and the areas where we gave special considerations to the DSNZPARM keywords.

We describe the following topics:

- Copy and FlashCopy parameters
- Recover utility
- DB2 Sort
- Backup and restore
- Reorg utility
- Check utility
- Multiple utilities
- Statistics interval
- Administrative scheduler system parameter
- Loading a new DSNZPARM

B.1 DB2 DSNZPARM setup

We describe the following sets of subsystem parameters:

- Copy and FlashCopy parameters
- DB2 Sort
- Statistics interval
- Administrative scheduler system parameter
- Loading a new DSNZPARM

B.1.1 Copy and FlashCopy parameters

The following DSNZPARMs affect the COPY utility:

- FLASHCOPY_COPY

  (Yes/No) Specifies that the COPY utility uses FlashCopy technology when the FLASHCOPY option/keyword is not specified in the control statement

- FLASHCOPY_LOAD

  (Yes/No) Specifies whether LOAD utility uses FlashCopy technology when the FLASHCOPY option is not specified in the control statement – inline copy

- FLASHCOPY_REORG_TS and FLASHCOPY_REORG_IX

  (Yes/No) – Inline copy at load phase for REORG – when not coded on LOAD control statement

• FLASHCOPY_REBUILD_IX

  (Yes/No) – Can Rebuild IX use FlashCopy, by default, when not specified on the REBUILD INDEX utility control statement

The following setting avoided FlashCopy being invoked by default when utilities, such as COPY, LOAD, or REORG, were invoked and the FlashCopy option was not specified:

```
FLASHCOPY_COPY=NO
FLASHCOPY_LOAD=NO
FLASHCOPY_REORG_TS=NO
FLASHCOPY_REBUILD_INDEX=NO
FLASHCOPY_REORG_INDEX=NO
```

So we could test recovering objects from system level backups (SLBs), we had to set the following parameter:

```
SYSTEM-LEVEL BACKUPS=YES
```

By setting these various options, we were able to control some of the DB2 operations from the utilities directly rather than have DFSMSDSS functions invoke actions by default. All of these options are set in the DB2 utilities panel 1: DSNTIP6. See Figure B-1 on page 543.

```
DSNTIP6            INSTALL DB2 - DB2 UTILITIES PARAMETERS 1
===>

Enter system-level backup options for RESTORE SYSTEM and RECOVER below:
 1 SYSTEM-LEVEL BACKUPS  ===> NO        As a recovery base: NO or YES
 2 RESTORE/RECOVER       ===> NO        From dump: NO or YES
 3 DUMP CLASS NAME       ===>           For RESTORE/RECOVER from dump
 4 MAXIMUM TAPE UNITS    ===> NOLIMIT   For RESTORE SYSTEM: NOLIMIT or 1-255
Enter default settings for the DB2 Utilities backup options below:
 5 FAST REPLICATION      ===> REQUIRED  DSScopy replication type for CHECK
                                        utilities: PREFERRED or REQUIRED
 6 FAST RESTORE          ===> PREFERRED For RECOVERY: NONE, PREFERRED, REQUIRED
 7 FLASHCOPY PPRC        ===> REQUIRED  FLASHCOPY peer to peer remote copy:
                                        blank, NONE, PREFERRED, or REQUIRED
 8 DEFAULT TEMPLATE      ===> HLQ.&DB..&SN..N&DSNUM..&UQ.

 9 COPY                  ===> NO        Use FLACHCOPY defaults for COPY
10 LOAD                  ===> NO        Use FLASHCOPY defaults for LOAD
11 REORG TABLESPACE      ===> NO        Use FLASHCOPY defaults for REORG TS
12 REBUILD INDEX         ===> NO        Use FLASHCOPY defaults for REBUILD IX
13 REORG INDEX           ===> NO        Use FLASHCOPY defaults for REORG IX

PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure B-1 DSNTIP6 install panel

B.1.2 Recover utility

The following DSNZPARMs affect the RECOVER utility:

- REC_FASTREPLICATION (Required, Preferred, or None)

  Specifies whether recovery from a FlashCopy image copy should use FlashCopy.

  This setting disables any mirroring processing when a DB2 utility request is made to a primary device that is in a Peer-to-Peer Remote Copy (Metro Mirror) relationship:

  - REQUIRED - The RECOVER utility forces use of FlashCopy when performing recovery from a FlashCopy image copy, to ensure that recovery occurs as quickly as possible. However, this option will cause RECOVERY to fail if FlashCopy cannot be used.
  - PREFERRED - The RECOVER utility uses FlashCopy only if FlashCopy support is available.
  - NONE - The RECOVER utility will use standard input/output to restore the FlashCopy image copy. This setting is not permitted when the FLASHCOPY_PPRC parameter is set to PREFERRED or REQUIRED.

- FLASHCOPY_PPRC (Required, Preferred, or None)

  Specifies the behavior for DFSMSdss FlashCopy requests when the target disk storage volume is the primary device in a Peer-to-Peer Remote Copy (Metro Mirror) relationship:

  - NONE - DFSMSdss does not preserve mirroring while processing a DB2 utilities request, even if all the configuration requirements for preserving the mirror operation are met.
  - PREFERRED - DFSMSdss allows a volume pair to go into duplex pending state while processing a DB2 utilities request only if it is required.

- REQUIRED - DFSMSdss does not permit a PPRC volume pair to go into duplex pending state while processing a DB2 utilities request.
- Blank - No preserve mirror command is passed to DFSMSdss.

FLASHCOPY_PPRC applies to the CHECK DATA, CHECK INDEX, CHECK LOB, COPY, REORG TABLESPACE, REORG INDEX, REBUILD INDEX, LOAD, and RECOVER utilities.

- LOGAPSTG (No longer externalized in DB2 10, default is 510 MB). In previous versions, specify 100 MB for 10 concurrent RECOVER jobs.
- XREC_FASTREPLICATION

  Should the RECOVER utility use fast replication to recover from DB2 10 FCIC:

  - None - Uses standard I/O to restore from FCIC
  - Preferred - Uses FC to recover from FCIC, if available
  - Required - Forces RECOVER to use FC. It will fail if FC is not available.

B.1.3 DB2 Sort

The following DSNZPARMs are available for DB2 Sort utility:

- DB2SORT (Enable/Disable)

  Specifies whether DB2 utilities are to use DB2 Sort instead of DFSORT for utility sort processing when DB2 Sort is installed.

  To ensure that the DB2 utilities used DB2 Sort, the following parameter was specified in our DSNZPARM assemble job:

  ```
  DB2SORT=ENABLE
  ```

  This parameter is also set via option 9 of the DB2 utilities parameters, panel 2: DSNTIP61. See Figure B-2 on page 545 for DB2 Sort enabling and the other subsystem sort parameters.

- UTSORTAL subsystem parameter, when set to YES, the default in DB2 10, allows DB2 for z/OS to use statistics to determine the sort work data set sizes. If you specify NO, you to either specify the sort work data sets explicitly in the JCL or use the SORTNUM clause in the utility's control cards.
- IGNSORTN subsystem parameter determines whether occurrences of the SORTNUM clause in the utility control statements are to be ignored. The value of the IGNORE SORTNUM STAT field (or IGNSORTN) is meaningful only when UTSORTAL is YES.

  You might want set IGNSORTN to NO for those cases, such as large amounts of sorting, where you need to override the SORTNUM.

We set both these parameters to YES.

```
DSNTIP61            INSTALL DB2 - DB2 UTILITIES PARAMETERS 2
===>

Enter other DB2 Utilities options below:
 1 TEMP DS UNIT NAME     ===> SYSDA     Device for temporary utility data sets
 2 UTIL TEMP STORCLAS    ===>           For temporary shadow data sets
                                        by utilities
 3 STATISTICS HISTORY    ===> NONE      Default for collection of stats history
 4 STATISTICS ROLLUP     ===> YES       Allow statistics aggregation: NO or YES
 5 UTILITY TIMEOUT       ===> 6         Utility wait time multiplier
 6 UT SORT DS ALLOCATION ===> YES       Predictable sort disk space allocation
 7 IGNORE SORTNUM STMT   ===> NO        Ignore SORTNUM keywork in UT stmt
 8 SET CHECK PENDING     ===> NO        Set inconsistent objects in CHECK
                                        PENDING status: NO or YES
 9 UT DB2 SORT USE       ===> ENABLE    ENABLE/DISABLE utility use of DB2 Sort

PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure B-2 DSNTIP61 install panel

Other DSNZPARMs affect the temporary data set allocation:

- XVOLTDEFT

  Specifies the device type or unit name that is to be used by DB2 utilities for dynamically allocating temporary data sets. Used for COPY CONCURRENT (DSS) data sets.

- XUTIL_TEMP_STORCLAS

  Specifies the storage management subsystem (SMS) storage class that the CHECK INDEX, CHECK DATA, and CHECK LOB utilities are to use when allocating temporary shadow data sets. These utilities allocate shadow data sets when the SHRLEVEL CHANGE option is used.

B.1.4 Backup and restore

The following DSNZPARMs affect the BACKUP/RESTORE SYSTEM utilities:

- XSYSTEM-LEVEL_BACKUPS

  (Y/N) - Enables the RECOVER utility to use SLB as input for object-level recoveries.

- XRESTORE_RECOVER_FROMDUMP

  (Y/N) - Specifies whether RESTORE SYSTEM and RECOVER can use SLB dump on tape as input for recovery. Yes - Use tape. No - Use only disk.

- XMAXIMUM TAPE UNITS

  RESTORE_TAPEUNIT (1 - 255). The number of tape drives RESTORE SYSTEM allocates for restore from dump.

B.1.5 Reorg utility

The following DSNZPARMs affect the REORG utility:

- XREORG_LIST_PROCESSING (Parallel/Serial)

  This parm controls the PARALLEL option for REORG. It is used during LISTDEF processing (parallel yes or no).

- X REORG_IGNORE_FREESPACE (Y/N)

  Controls whether DB2 uses the PCTFREE and FREEPAGE values for partition-by-growth (PBG):

  - YES - DB2 will set 0 for PCTFREE and FREEPAGE.
  - NO - DB2 will use the PCTFREE and FREEPAGE values.
- XREORG_PART_SORT_NPSI (Y/N)

  For REORG TABLESPACE PART to sort all of the keys of a non-partitioned secondary index (y/n). It is ignored for non-PART REORGs.

  Default of N means only keys for nonpartitioned secondary index (NPSI) are sorted.

B.1.6 Check utility

The following DSNZPARM affects the CHECK utility:

- CHECK_FASTREPLICATION

  Specifies whether the CHECK utilities direct DFSMSdss COPY to use FlashCopy as the preferred method for copying objects to shadow data sets, or as the only method for copying objects to shadow data sets.

B.1.7 Multiple utilities

A DSNZPARM affects multiple utilities:

- XFCCOPYDDN

  Used for the FLASHCOPY option for COPY, LOAD, REBUILD IX, REORG IDEX, and REORG TABLESPACE

  Specifies the template for the FCIC name:

  - Default - HLQ.&DB..&SN..N&DSNUM..&UQ
  - FCIC uses VSAM copies, no generation data group (GDG)

B.1.8 Statistics interval

By default, the real-time statistics (RTS) interval field is set to 30 minutes. This is the time that DB2 waits before writing out the page set statistics to the real-time statistics tables. For testing purposes, we changed this to 1 minute:

```
STATSINT=1
```

This variable is option 13, Operator Functions, of the Update DB2 - Selection Menu panel: DSNTIPB. See Figure B-3 on page 547.

```
DSNTIPB            UPDATE DB2 - SELECTION MENU
===>

 Select one of the following:
 1 CATALOG AND DIRECTORY                  19 PERFORMANCE AND OPTIMIZATION
 2 DATA PARAMETERS PANEL                  20 IRLM PANEL 1
 3 DEFINE GROUP OR MEMBER                 21 IRLM PANEL 2
 4 SYSTEM RESOURCE DATA SET NAMES         22 DB2 UTILITIES PARAMETERS
 5 DATA SET NAMES PANEL 1                 23 DB2 UTILITIES PARAMETERS PANEL 2
 6 DATA SET NAMES PANEL 2 *               24 PROTECTION
 7 DATA SET NAMES PANEL 3 *               25 PROTECTION PANEL 2
 8 SIZES                                  26 MVS PARMLIB UPDATES
 9 SIZES PANEL 2                          27 ACTIVE LOG DATA SET PARAMETERS
10 WORK FILE DATABASE                     28 ACTIVE LOG CHECKPOINT PARAMETERS
11 THREAD MANAGEMENT                      29 ARCHIVE LOG DATA SET PARAMETERS
12 BUFFER POOL SIZES PANEL 1              30 DATABASES TO START AUTOMATICALLY
13 BUFFER POOL SIZES PANEL 2 *            31 DISTRIBUTED DATA FACILITY
14 TRACING AND CHECKPOINT PARAMETERS      32 DISTRIBUTED DATA FACILITY PANEL 2
15 OPERATOR FUNCTIONS                     33 ROUTINE PARAMETERS
16 DEFAULT STARTUP MODULES                34 PARAMETERS FOR DB2-SUPPLIED ROUTINES
17 APPLICATION PROGRAMMING DEFAULTS 1     35 DATA DEFINITION CONTROL SUPPORT
18 APPLICATION PROGRAMMING DEFAULTS 2     36 JOB EDITING
 * None of the fields on these panels can be updated.

PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure B-3 Update selection menu panel: DSNTIPB

B.1.9 Administrative scheduler system parameter

The ADMIN SCHEDULER field (ADMTPROC subsystem parameter) identifies a name for the JCL procedure that is used to start the DB2 administrative task scheduler that is associated with this DB2 subsystem.

The administrative scheduler is a separate address space with the default name, DSNADMT. You can choose a different name if you want. With DSNZPARM ADMTPROC, specify the name that is suitable for you. It is defined in the DSNTIPX installation panel. See Figure B-4.

```
DSNTIPX         INSTALL DB2 - ROUTINE PARAMETERS
===>_
Enter data below:
1 WLM PROC NAME     ===> DSN1WLM    WLM-established stored procedure JCL PROC
2 NUMBER OF TCBS    ===> 8          Number of concurrent TCBs (1-100)
3 MAX ABEND COUNT   ===> 0          Allowable ABENDs for a routine (0-255)
4 TIMEOUT VALUE     ===> 180        Seconds to wait before SQL CALL or function
                                    invocation fails (5-1800,NOLIMIT)
5 WLM ENVIRONMENT   ===> Default    WLM environment name
6 MAX OPEN CURSORS  ===> 500        Maximum open cursors per thread
7 MAX STORED PROCS  ===> 2000       Maximum active stored procs per thread
8 MAXIMUM LE TOKENS ===> 20         Maximum tokens at any time. 0-50
9 BIF COMPATIBILITY ===> CURRENT    Compatibility level for DB2 built-in functions (V9
or CURRENT)
10 ADMIN SCHEDULER  ===> DSNADMT Administrative scheduler task proc name


PRESS: ENTER to continue RETURN to exit HELP for more information
```

Figure B-4 ADMIN SCHEDULER field (DSN6SPRM ADMTPROC subsystem parameter)

It is updated in install panel DSNTIPB, under option 33 (see Figure B-3 on page 547) or in the DSN6SPRM macro. This parameter cannot be updated online. The acceptable values are one to eight alphanumeric characters, with the default: ssnADMT.

When DB2 starts, it starts the administrative task scheduler that is specified in the ADMIN SCHEDULER field, if that scheduler is not yet started. Also, every time that DB2 starts or stops, it posts an event to the administrative task scheduler so that the administrative task scheduler can execute tasks that depend on those events.

To disable the administrative task scheduler, you can enter blanks in the ADMIN SCHEDULER field. If you replace the default value with blanks, DB2 will not start the administrative task scheduler until you update the subsystem parameter.

The commands START, STOP, and MODIFY support admtproc:

- **START admtproc**

  The **START admtproc** command starts the scheduler that is specified in the admtproc parameter.

  This command can be started at the operator's console, at the data sharing member level, or during DB2 startup or initialization. After it is started, the administrative task scheduler is always up, unless it is stopped by a STOP command at the operator's console.

  Each DB2 subsystem has a coordinated administrative task scheduler address space for starting a z/OS started task procedure. Therefore, if there are many DB2 subsystems running on one z/OS, there is a separate administrative task scheduler with a separate name for each DB2 subsystem. Two instances of the same administrative task scheduler cannot run simultaneously.

  To avoid starting up a duplicate administrative task scheduler, at startup, the administrative task scheduler checks all of the address spaces for duplicate names. If another address space with the same name is already running, the administrative task scheduler that is starting up will immediately shut down with a console error message. The administrative

task scheduler can only check the address spaces in the same system, but not the entire sysplex.

The syntax is shown in Figure B-5.

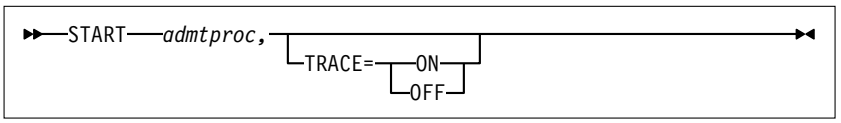

Figure B-5 START admtproc command

- **STOP admtproc**

  The **STOP admtproc** command stops the administrative task scheduler that is specified in the admtproc parameter.

  The command should only be issued to bring down the administrative task scheduler for maintenance or to prepare for an IPL. To stop the administrative task scheduler for other purposes, issue the command: **modify admtproc, appl=shutdown**.

  The syntax is shown in Figure B-6.

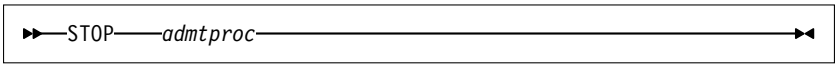

Figure B-6 STOP admtproc command

- **MODIFY admtproc,APPL=SHUTDOWN**

  The **MODIFY admtproc, APPL=SHUTDOWN** command stops the administrative task scheduler from accepting requests and starting new task executions. It also shuts down the administrative task scheduler.

  When the SHUTDOWN option is specified, the administrative task scheduler waits until the execution of all currently running tasks completes. When all running tasks are complete, the administrative task scheduler terminates.

  The syntax is shown in Figure B-7.

Figure B-7 MODIFY admtproc command

You cannot include spaces when you specify options.

B.1.10 Loading a new DSNZPARM

We reassembled the new DSNZPARMs into the SDSNEXIT library, calling it DSNZPRMC, and reloaded it either by stopping and restarting the DB0A DB2 system or by using the following command:

```
-DB0A SET SYSPARM LOAD(DSNZPRMC)
```

C

Additional material

This book refers to additional material that can be downloaded from the Internet as described in the following sections.

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 at:

`ftp://www.redbooks.ibm.com/redbooks/SG248046`

Alternatively, you can go to the IBM Redbooks website at:

`ibm.com/redbooks`

Select the **Additional materials** and open the directory that corresponds with the IBM Redbooks form number, SG248046.

Using the web material

File name	*Description*
`8046ADDMAT.zip`	Zipped code samples

This additional web material zipped file includes the files listed in Table C-1.

Table C-1 Additional material

File name	Description
ADMCOX.JCL	Batch JCL job to call DSNACCOX stored procedure
ADMCOX.REXX	REXX procedure that invokes DSNACCOX
ADMSMS.JCL	Batch JCL job to call REXX SMS stored procedure

File name	Description
ADMSMS.REXX	REXX procedure calls ADMIN_SMS_INFO
AUTOTEST.ASM	Assembler code for DB2 stored procedure AUTOTEST
AUTOTEST.JCL	Assembler and LinkEdit batch job to assemble stored procedure AUTOTEST
CREATESP.JCL	JCL to add DB2 stored procedure AUTOTEST
DB2TOOLS.CLS	CLIST to display DB2TOOLS menu
DB2TOOLS.PNL	DB2TOOLS menu panel
README.TXT	Readme file

System requirements for downloading the web material

The web material requires the following system configuration:

Hard disk space: 2 MB minimum
Operating System: Microsoft Windows
Processor: Intel 386 or higher
Memory: 16 MB

Downloading and extracting the web material

Create a subdirectory (folder) on your workstation, and extract the contents of the web material .zip file into this folder.

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

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- *Disaster Recovery with DB2 UDB for z/OS*, SG24-6370
- *IBM System Storage DS8000 Copy Services for IBM System z,* SG24-6787-06
- *DFSMShsm Fast Replication Technical Guide*, SG24-7069
- *Optimizing Restore and Recovery Solutions with DB2 Recovery Expert for z/OS,* SG24-7606
- *IBM DB2 Recovery Expert for z/OS User Scenarios,* SG24-7226
- *DB2 10 for z/OS Performance Topics,* SG24-7942
- *DB2 10 for z/OS Technical Overview*, SG24-7892
- *DB2 9 for z/OS and Storage Management,* SG24-7823
- *IBM System Storage DS8000: Remote Pair FlashCopy (Preserve Mirror)*, REDP-4504

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

ibm.com/redbooks

Other publications

These publications are also relevant as further information sources:

- *DB2 10 for z/OS Installation and Migration Guide*, GC19-2974
- *IBM DB2 Utilities Enhancement Tool for z/OS Version 2 Release 2 User's Guide*, SC19-3417
- *IBM DB2 Automation Tool for z/OS Version 4 Release 1 User's Guide*, SC19-2752
- *IBM DB2 10 for z/OS Managing Performance*, SC19-2978
- *IBM DB2 Cloning Tool for z/OS Version 3 Release 1 User's Guide,* SC19-3493
- *IBM DB2 Sort for z/OS Version 1 Release 2 User's Guide,* SC19-3418
- *IBM DB2 High Performance Unload for z/OS Version 4 Release 1 User's Guide,* SC19-3169
- *z/OS V1R12.0 Security Server RACF Command Language Reference*, SA22-7687-15
- *DB2 10 for z/OS Administration Guide,* SC19-2968

- *IBM DB2 Administration Tool for z/OS Version 10 Release 1 User's Guide and Reference*, SC19-3033

Online resources

These websites are also relevant as further information sources:

- DB2 Tools for z/OS

  http://www.ibm.com/software/data/db2imstools/products/db2-zos-tools.html
- Updates to DB2 Cloning Tool V3.1 User's Guide

  http://www.ibm.com/support/docview.wss?uid=swg27023876

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Managing DB2 for z/OS Utilities with DB2 Tools Solution Packs

(1.0” spine)
0.875”<->1.498”
460 <-> 788 pages

Managing DB2 for z/OS Utilities with DB2 Tools Solution Packs

Simplify DB2 utilities definition and scheduling

Choose the best recovery and disaster recovery plan

Minimize window of execution for utilities

IBM DB2 Tools for z/OS support and take advantage of the latest versions of DB2 for z/OS. These tools are integral for the administration of the DB2 for z/OS environment and for optimization of data performance. In addition, the IBM portfolio addresses additional client requirements in the areas of data governance and version upgrade acceleration.

Underlying the operation of any database management system are the utilities. With the number of database objects growing exponentially, managing utility jobs, meeting service level agreements (SLAs), and ensuring recoverability can be overwhelming. IBM offers DB2 Tools solution packs that assist in the DB2 utilities management process. Solution packs combine several products into a single consolidated solution providing everything necessary to ensure the execution of a set of database administration functions. The goals are to reduce the operational complexity and reduce cost.

The objective of this IBM Redbooks publication is to document the added value in terms of productivity and performance for database administrators when using the IBM DB2 Utilities Solution Pack and the IBM DB2 Fast Copy Solution Pack. We show the functions of the tools provided by the solution packs as used in real-life scenarios and adopting utilities best practices.

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

SG24-8046-01 ISBN 0738438472