Information Management software

Enhancing SAP by Using DB2 9 for z/OS

- Understand the new indexing capabilities of DB2 9
- Streamline efforts to meet compliance obligations
- Examine new utility features and functions

Lydia Parziale
Hyun Baek
Javier Chavez
Vince Crose
Giovanni Grita
Theodor Kuebler
Georg Mayer
Heinz Ruflair
Mary Siart
Johannes Schuetzner

Redbooks

ibm.com/redbooks

International Technical Support Organization

Enhancing SAP by Using DB2 9 for z/OS

July 2007

SG24-7239-00

Note: Before using this information and the product it supports, read the information in "Notices" on page ix.

First Edition (July 2007)

This edition applies to IBM z/OS Version 1.7, DB2 Version 9, DB2 Connect Version 9.1, and SAP ECC 5.00.

Contents

Notices

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

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

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

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

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

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

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

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

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

COPYRIGHT LICENSE:

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

Trademarks

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

AIX®
CICS®
Distributed Relational Database Architecture™
DB2 Connect™
DB2®
DFSMS™
DFSMShsm™
DRDA®
Enterprise Storage Server®
ESCON®
eServer™
FlashCopy®
FICON®
Geographically Dispersed Parallel Sysplex™
GDPS®
Informix®
IBM®
IMS™
MVS™
MVS/ESA™
OMEGAMON®
OS/390®
Parallel Sysplex®
PR/SM™
pSeries®
QMF™
Redbooks®
Redbooks (logo) ®
RACF®
RETAIN®
REXX™
RMF™
S/390®
System z™
System z9™
Tivoli®
TotalStorage®
WebSphere®
z/OS®
z9™

The following terms are trademarks of other companies:

ABAP, SAP NetWeaver, mySAP, SAP, and SAP logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries.

Oracle, JD Edwards, PeopleSoft, Siebel, and TopLink are registered trademarks of Oracle Corporation and/or its affiliates.

Snapshot, and the Network Appliance logo are trademarks or registered trademarks of Network Appliance, Inc. in the U.S. and other countries.

IT Infrastructure Library, IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

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

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

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

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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

Preface

This redbook presents many of the new and improved features and functions of DB2® version 9.1 for z/OS® and DB2 Connect™ v9.1 and how they complement and benefit your SAP NetWeaver® environment.

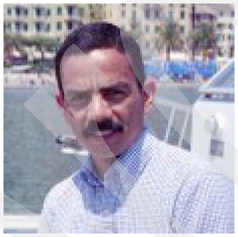
This redbook also shares some of our experiences in migrating our DB2 v8 SAP data sharing environment to DB2 9 for z/OS with a minimal amount of outage.

The team that wrote this redbook

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

Lydia Parziale is a Project Leader for the ITSO team in Poughkeepsie, New York with domestic and international experience in technology management including software development, project leadership and strategic planning. Her areas of expertise include e-business development and database management technologies. Lydia is a Certified IT Specialist with an MBA in Technology Management and has been employed by IBM for 23 years in various technology areas.

Hyun Baek is an IT Specialist for DB2 and DB2 Tools in the IBM Software Group in Korea. He has worked as a DB2 database administrator and system programmer in a telecommucation company for 6 years and joined IBM in 2005. He has an experience in DB2 installation, migration and performance tuning.



Javier Chavez is a Senior I/T Specialist working out of Phoenix, Arizona as part of the Lab Services group of STG supporting the Z brand. He has 28 years of experience in the I/T field related to data base management systems. He joined IBM in 2002 and has been working in the areas of DB2 installation, migration and performance and tuning.

Vince Crose is a DB2 Data Base System Programmer / DBA IT professional with over 17 years of experience in a wide variety of computer disciplines, which include Programmable Controllers, Series 1's, MVS™ on S/390® Mainframes, PCs and the current Internet e-business environment of Unix Systems Services running under z/OS on System z®. He is currently working in IBM's design center working with clients for a better IT solution within their product.

Giovanni Grita is an Infrastructure IT Architect working in SAP Technical Sales in IBM Italy. He joined IBM in 1988, working as a C++ Object Oriented developer, system programmer and branch office IT Specialist. He started working with SAP products in 1998 and has earned Basis Consultant certification. Giovanni is now involved in SAP capacity planning, sizing, and integration in IBM environments. He holds a degree in mathematics from “La Sapienza” University of Rome.

Theodor Kuebler is a consultant and trainer for SAP and Data Management products. He is founder of the Logos Informations Syteme (LIS) AG in Switzerland (www.lis-ag.ch), an IBM enabled Education Center for IBM Software (ECIS), and an IBM Business Partner, providing classroom and onsite courses in Data Management and, Tivoli, He has 12 years experience in implementation and tuning of SAP products on database management systems, such as DB2 UDB, Adabas-D, and Oracle on different platforms. He teaches SAP classes for Z/OS and DB2.

Georg Mayer is a Senior Development Engineer for the Joint SAP IBM Porting Team in Rot, Germany, focussing on the SAP NetWeaver Business Intelligence component for DB2 on z/OS. He joined IBM Data Management in 2001 and has over 18 years of experience with ERP software, databases, OLTP and performance tuning. The last 10 years Georg works as development expert for a variety of SAP database porting teams in Walldorf. Prior to working on IBM, he worked on Informix and Atos. He holds a masters degree in Computer Science and Mathematics from the University of Stuttgart.

Heinz Ruflair is one of the founders of InfoDesign GmbH, which is based in Hamburg and develop and support automated administration tools especially for z/OS and DB2. He holds a diploma in information technology and his areas of expertise include recovery and cloning databases in data sharing environments since he is working with DB2 from 1986.

Mary Siart is a Senior Technical Sales Specialist working out of Phoenix, Arizona in IBM Sales & Distribution, System z - SAP - Solutions Technical Sales (STS), for the Americas. She has 25 years of experience in the I/T field related to data base management systems. She rejoined IBM in 2002 after working for Motorola for nine years, supporting their SAP software on z/OS implementations. Most recently, she has been providing support to SAP customers.

Johannes Schuetzner is is a software engineer in the IBM Boeblingen Lab in Germany. He is a member of the joint SAP/IBM software development team that is in charge of the System z platform at SAP. He has published and presented at conferences on different aspects of DB2. He holds a Master's degree in Computer Science, studying at the University of Stuttgart in Germany and at the University of Connecticut, Storrs.

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

Bob Haimowitz, Rich Conway
International Technical Support Organization, Poughkeepsie Center

Michael Sheets,
IBM® Advanced Technical Support - SAP & Oracle® Solutions

Mike Zazulak,
IBM Technical Sales Specialist - System z™ - SAP - Solutions Technical Sales (STS), Americas

Ben Budiman,
DB2 Utilities Development

Seewah Chan, Veng K. Ly, Mai N. Nguyen, Howard E. Poole, Michael R. Sheets, Akira Shibamiya, Joachim Rees, Terence P Purcell, Chan-hua Liu, Roger Miller, Ludgar Quatmann, Laura Kunioka-Weis, Willie Favero and Martin Dvorsky.

Become a published author

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

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

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

- Use the online **Contact us** review redbook 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

1

DB2 9 for z/OS new and improved functions overview

This chapter is for SAP and DB2 database administrators, application programmers, and other DB2 database users. In this chapter, we provide tables that enable you to have a quick overview about the new and enhanced features that are available in DB2 9 for z/OS for the SAP NetWeaver® environment. In addition, we include the features in DB2 9 for z/OS that have been improved and benefit SAP environments.

The tables are grouped according to the following categories:

- Storage-related enhancements
- DB2 objects
  - Tables
  - Indexes
  - LOBs
  - Data types
- DB2 processes, functions, and procedures
  - Optimizer
  - Real-time statistics (RTS)

- DB2 Connect 9
- DB2 enhancements for backup and recovery

For additional and more detailed information, consult the references that are provided.

1.1 Storage-related enhancements

Some of the enhancements in DB2 9 for z/OS relate to space management. The database management system (DBMS) should use the amount of storage for the full life cycle of the application in the most effective way. DB2 9 for z/OS supports new kinds of table spaces, such as universal table spaces and partition-by-growth table space to improve and control the usage of the available storage space.

Table 1-1 gives an overview about the new and improved functions for the *Storage* category.

Table 1-1 Storage-related new and enhanced features and functions

Element or function	Short description	Effect on SAP
Bootstrap dataset (BSDS) format	In Version 9.1, the BSDS supports up to 10,000 data sets per copy for archive logs and 93 data sets per copy for active logs. In previous releases, the BSDS format allowed only 1000 data sets per copy for archive logs and 31 data sets per copy for active logs.	More efficient rollbacks and recoveries
Alter STOGROUP using a new volume catalog (VCAT)	Through the CATMAINT utility as a system-wide function, DB2 9 for z/OS provides three new options: SCHEMA, OWNER, and VCAT. These options are available only in New Function Mode and require installation SYSADM authority.	Simplifies homogenous system copy
Workfile table spaces	In Version 9.1, the installation process creates workfile table spaces on DB2-managed storage. The default storage group is SYSDEFLT.	Star join optimization
Universal table space	Universal table space is a combination of a partitioned and segmented table space; it includes range-partitioned and partition-by-growth table spaces.	Performance improvement
Partition-by-growth table space	Use partition-by-growth table space when a table is expected to exceed 64 GB and does not have a suitable partitioning key for the table. DB2 automatically adds a new data set when the database needs more space. Partition-by-growth table spaces can grow up to 128 TB.	Simplifies systems management; also for BI/BW

1.2 DB2 objects

DB2 9 for z/OS provides many new improvements for the DB2 objects, such as tables, indexes, and data types. Table 1-2 gives an overview of the new and improved functions for DB2 objects.

Table 1-2 New features and functions related to DB2 objects

Element or function	Short description	Effect on SAP
Work file database	In DB2 Version 9.1, the workfile database is used for all temporary tables. The TEMP database is no longer used by DB2.	Simplifies systems management
Schema name change	Through the CATMAINT utility as a *system-wide* function, DB2 9 for z/OS provides three new options: SCHEMA, OWNER, and VCAT. Using the new SCHEMA option, the CREATOR and OWNER can be changed. System-wide means, that all objects are altered from one schema name to another. These options are available only in New Function Mode and require installation SYSADM authority.	Simplifies homogenous system copy
Implicit object creation ► Tablespace ► Database ► Large objects (LOB): – Table space – Auxiliary table – Auxiliary index	**Prerequisite**: On a CREATE TABLE statement, if you do not specify a database name, DB2 9 for z/OSDB2 9 for z/OS implicitly creates a database. If the table space is implicitly created, DB2 creates all of the system required objects for you, including: ► Enforcing the primary key and unique key index ► Creating the ROWID index ► Creating the LOB table spaces, auxiliary tables, and auxiliary indexes	► Simplifies development ► Simplifies database administration
Mass deletion of the table content or of the partition	Using the TRUNCATE TABLE statement, you can perform fast mass deletes of data rows for tables. The table can belong to a segmented table space, a partitioned table space, or a universal table space. If the table contains LOB or XML columns, its corresponding table spaces and indexes are also emptied.	► Simplifies BI/BW administration ► Improves BI/BW performance
Row compression	Row compression support is available from DB2 8 and is supported in DB2 9 for z/OS. It includes a new dictionary-based row compression feature that you can use to compress data objects. Large tables with rows that contain repetitive patterns will benefit from this feature. Query performance might improve for tables that use row compression.	► Reduces needed resources ► Performance improvement

Element or function	Short description	Effect on SAP
Index compression	DB2 9 for z/OS provides index compression support by creating an index or altering an index with the COMPRESS YES parameter. The DSN1COMP utility helps you to estimate the saved space by compressing the indexes. In BI/BW, indexes are often larger than tables.	Reduces needed BI/BW resources and workload
Index on expression	Index on expression allows you to create an index on a general expression. You can enhance your query performance if the optimizer chooses the index created on the expression. Use index on expression when you want an efficient evaluation of queries that involve a column expression.	► Performance improvement ► Simplifies tuning
Index rename	In DB2 9 for z/OS, the new keyword, INDEX, is added to the RENAME statement. In New Function Mode (NFM), an existing index can be renamed by using the new keyword on the RENAME statement.	Simplifies database administration and BI/BW administration
Dynamic index ANDing	Ensure optimal performance for all star schema queries.	Improves performance for BI/BW queries
Unused index	In DB2 9 for z/OS, the real-time statistics (RTS) provide the information about the last usage of an Index.	Simplifies database administration
FETCH FIRST n ROWS	The DELETE statement is enhanced to delete a limited number of rows only. Alternatively you can define the FETCH FIRST n ROWS subselect.	► Improves SAP BI/BW performance ► Reduces locks
Accessing LOBs	DB2 9 for z/OS includes improvements in network handling for small LOBs. Using the FETCH statement with the new WITH CONTINUE clause allow an optimal balance between performance and storage usage when fetching LOBs.	Improves LOB performance
LOB DML	See *LOBs with DB2 for z/OS: Stronger and Faster*, SG24-7270, for more information about LOBs.	Simplifies database administration
LOB locks	Release LOB LOCKS as soon as possible.	Improves LOB performance

Element or function	Short description	Effect on SAP
Autonomic buffer pool	Using the adaptive, self-tuning memory allocation, you can reduce or eliminate the task of configuring your DB2 server by continuously updating configuration parameters and resizing buffer pools.	Simplifies database administration
Buffer pool 64-bit architecture	From DB2 8 and later releases, storage areas, such as buffer pools, reside above the 2 GB bar in the *ssnm*DBM1 (Database Services) address space. With 64-bit virtual addressing to access these storage areas, buffer pools can scale to extremely large sizes.	► Simplifies database administration ► Improves scalability
Data type BIGINT	DB2 9 for z/OS provides the new data type BIGINT. BIGINT is a string of numbers that does not contain a decimal point nor an exponent identifier. The first character can be a plus (+) or minus (-) sign. It is used, for example, for data types in the SYSIBM.SYSTABLESPACESTATS.	Enhances SQL
Data types: ► VARBINARY ► BINARY	The data types BINARY and VARBINARY have been added to DB2 9 for z/OS. In addition, support for the types has been added to DB2 CLI and DB2.NET Data Provider.	Enhances SQL
VARCHAR enhancements	When reorganizing a table space, DB2 9 for z/OS automatically reorders the varying length columns.	Improves performance
New catalog table	DB2 9 for z/OS provides new catalog tables, SYSTABLESPACESTATS and SYSINDEXSPACESTATS, which contain RTS for table and index spaces.	Simplifies database administration

1.3 DB2 processes, functions, and procedures

In DB2 9 for z/OS, existing functions and routines are expanded, including:

- The DB2 optimizer
- The real-time statistics
- The stored procedures
- The usage of commands

Table 1-3 Overview about new functions and procedures

Element or function	**Short description**	**Effect on SAP**
Reoptimize SQL statements	Using the new DB2 system configuration parameter (ZPARM) REOPT(AUTO), the dynamic SQL statements are reoptimized automatically.	► Improves performance ► Simplifies database administration
Log latch contention	DB2 generates the log record sequence number (LRSN) and associates it with each log record. The log latch is held while incrementing and setting the LRSN value for a record. To reduce contention, an updater passes in the current LRSN value of the updated page when a log record reflecting the change is created. The spin needs to be done only if the LRSN value of the page and the current LRSN match. In addition, if LRSN spin is necessary, it is done without holding the log latch so that other work is not delayed.	Performance improvement
Thread identifier	In Version 9, the thread identifier (token) is shown in more messages that refer to threads.	Simplifies database administration
Streaming for retrieval of LOBs and XML data	DB2 Version 9.1 for z/OS supports progressive streaming. With progressive streaming, the database server dynamically determines the most efficient mode in which to return LOB or XML data, based on the size of the LOBs or XML objects.	LOB performance improvements
RTS objects in catalog	The new catalog tables SYSIBM.SYSTABLESPACESTATS and SYSIBM.SYSINDEXSPACESTATS are provided, which contain RTS for table and index spaces.	Simplifies database administration
RTS: No initialization	DB2 9 for z/OS uses the result of RUNSTATS to avoid RTS initialization.	Simplifies database administration
Java™	This option is fully compliant with JBDC 3.0.	Simplifies development

Element or function	Short description	Effect on SAP
Java	This new option controls the insert algorithm and skips uncommitted inserts.	Improves performance

1.4 DB2 utilities

Table 1-4 gives an overview of the new and improved functions for the utilities of DB2 9 for z/OS.

Table 1-4 New features and functions overview

Element or function	Short description	Effect on SAP
Online REORG	DB2 9 for z/OS Online REORG eliminates the BUILD2 phase for all types of secondary indexes.	Improves availability
Online REORG for LOBs	DB2 9 for z/OS provides a new design for LOB REORG. After stopping write access, the LOB is extracted from the original data set and inserted into the shadow data set. When this is complete, all access to the LOB table space is stopped for a short period while the original data set is switched with the shadow. At this point, full access to the new data set is allowed. The allocation of the shadow and deallocation of the old original data set follows the same rules as exist today for REORG SHRLEVEL CHANGE. Instead of SHRLEVEL NONE, REORG runs SHRLEVEL REFERENCE. Full read access is allowed to LOB data for the duration of the REORG with the exception of the switch phase, during which the readers are drained. After the switch occurs, full access is restored.	► Improves performance ► Simplifies database administration
Online CHECK DATA and CHECK LOB	You can run the CHECK DATA or the CHECK LOB utilities in DB2 9 for z/OS using the SHRLEVEL CHANGE option (read/write mode).	Improves availability
CHECK INDEX online and parallel	In DB2 9 for z/OS, checking indexes can run using the SHRLEVEL CHANGE option. Running the utility in *parallel mode* reduces the runtime of the utility and improves the availability of the SAP objects.	Improves performance and availability
Online REBUILD INDEX	In DB2 9 for z/OS, you can run the REBUILD INDEX utility in online mode (SHRLEVEL CHANGE) and access the SAP objects in change mode.	Improves availability

Element or function	Short description	Effect on SAP
Non-padded indexes for KPROCS	CPU time is reduced when extracting non-padded keys from data rows for the following utilities: ► REORG ► REBUILD INDEX ► CHECK INDEX ► LOAD	Improves performance

1.5 DB2 Connect 9

Table 1-5 gives an overview of the new and improved functions that are included in DB2 Connect V9.1.

Table 1-5 New features and functions in DB2 Connect V9

Element or function	Short description	Effect on SAP
DB2 Driver for ODBC and CLI	This element includes the CLI/ODBC runtime libraries to connect directly to DB2 z/OS.	Reduces time for installation and upgrades
Easy installation	ODBC and CLI applications can now run without a DB2 client. The IBM DB2 Driver for ODBC and CLI provides runtime support for ODBC and CLI applications, as well as connectivity for those applications. You can install the IBM DB2 CLI driver as a stand-alone product, without DB2 Connect installed.	Simplifies installation of the UNIX and Microsoft® Windows® application server
Fixpack installation upgrade	It is easy to upgrade IBM DB2 CLI driver on UNIX, Windows, and Linux® operating systems. An entire installation of DB2 Connect is no longer needed when upgrading. You simply replace the CLI libraries on the fly.	Reduces the upgrade time
IPv6	IPv6 communication protocol support has been added. Support for connections to server using IPv4 and IPv6 addresses is available.	Enhances network security and scalability
DRDA® LOB streaming	DB2 Connect 9 provides Distributed Relational Database Architecture™ (DRDA) LOB streaming, which allows immediate access to the LOB.	Improves performance
CLI API	Full Unicode support is available.	

Element or function	Short description	Effect on SAP
Trusted connections through DB2 Connect	DB2 CLI and IBM DB2 Driver for JDBC™ and SQLJ now support making trusted connections to DB2 database servers that support trusted contexts. If the database server is configured to allow it to do so, a client can create trusted connections using ODBC, XA, or new Java methods. The user name associated with the trusted connection can then be switched without the database server having to fully authenticate the new name.	Security improvement
DB2 Connect Trace	► Automated stop after recognizing particular errors ► Trace file wrapping ► Added CLI diagnostic information	Simplifies administration and problem analysis

1.6 DB2 enhancements for backup and recovery

Table 1-6 gives an overview of the new and improved functions related to backup and recovery.

Table 1-6 Backup and recovery features and functions overview

Element or function	Short description	Effect on SAP
Copy to tape	In DB2 9 for z/OS, the system-level utilities and the RECOVER utility are enhanced and include support for new features provided by the z/OS Version 1 Release 8 DFSMShsm™ to further facilitate management of your backup and recovery needs. The BACKUP SYSTEM and the RESTORE SYSTEM utilities are extended to allow for the management and the restoration of dumps on tape volumes of the system-level backups.	► Simplifies backup and recovery ► Improves availability
Restore individual objects	The RECOVER utility is enhanced to have the ability to use a system-level backup on disk or a system-level backup that has been dumped to tape as the recovery base for a single object or a list of objects.	Simplifies backup and recovery
Incremental FlashCopy®	The BACKUP SYSTEM utility supports the incremental IBM TotalStorage® Enterprise Storage Server® FlashCopy feature.	Simplifies backup and recovery

2

Facilitating DB2 subsystem cloning

In this chapter, we describe the new DB2 9 for z/OS CATMAINT utility extensions. Usually the existing schema of database objects, plans, and packages does not change. In the case of migrating a whole database system or even a release change in SAP, it depends on how the installer wants to do the renaming.

The schema of these database objects is set up at installation or when they are created. You can only change it later by dropping and recreating whole objects. In this case, you have to unload the contents and then reload it into the renamed objects. Either you unload the contents of the whole table space or table with specific utilities, such as UNLOAD and DSNTIAUL. Or if a similar renamed object still exists, you use the following SQL statement:

```
CREATE TABLE newtable LIKE oldtable...
```

Then you reload the object using this SQL statement:

```
INSERT INTO TABLE newtable (SELECT * FROM oldtable...)
```

Both methods require consideration of the entire environment with respect to creation of the outer shells such as STOGROUPs, databases, and table spaces.

The disadvantage of this method is the time it takes to do the amount of work required to unload, create, and reload. The amount of work required depends on the task to be done:

- Change the schema of the database objects
- Change the owner of the database objects
- Change the volume catalog (VCAT) names

You can make these changes with the DB2 online utility, CATMAINT. In previous DB2 releases, the use of CATMAINT is reduced to support the internal change of the DB2 catalog during migration from one DB2 release to the next. With DB2 9 for z/OS, the CATMAINT utility also changes the schema, owner, and VCAT names.

Because of serious and restricted operations, you must meet the following prerequisites:

- Only those who have the highest authority can execute CATMAINT; therefore, you need INSTALL SYSADM authority.
- DB2 system must be started using the parameters ACCESS(MAINT); the job will fail if it is not started in this way.
- In case of data sharing environments, all other members must be stopped.
- The source schema name cannot be referenced in a check condition in any check constraints. In most cases, this has no meaning for SAP database systems because referential integrity with explicit checking constraints is not used in SAP database systems.

After the CATMAINT utility completes, all the existing objects whose owners, schema, or VCATNAME changed function in the same way as they did before.

2.1 Reasons for cloning SAP DB2 systems

One type of routine work that is done during the life of an SAP database is to clone the whole database. You may clone an entire database for the following reasons:

- Create a shadow system for training people with no impact on production systems
- Create a test system for new applications or to customize new functions with no impact on production systems
- Create a revision for legal purposes
- Gain experience in the migration process

The DB2 system cloning process, which is used as part of the SAP homogenous system copy, is described more in detail in the SAP publication *System Copy for SAP Systems Based on SAP NetWeaver 2004s SR1 Java* or see SAP Note 870381.

In most instances, the cloned system resides on the same physical system as the source system, using shared disks. Several entries must be adjusted to connect to the new system. This procedure, which should only be done by experienced database administrators, consists of dumping and restoring the source DB2 system as well as postprocessing activities to modify the DB2 catalog of the target system. The postprocessing activities are:

- Change the bootstrap data set
- Create temporary data sets
- Modify the DB2 STOGROUPs to reflect the new high-level qualifier (HLQ)

  This modification is done by assigning objects of a STOGROUP to user-managed data sets, drop and recreate the STOGROUP with the new HLQ, assign objects back to the STOGROUP, and then restart the databases. The new data sets have to be renamed. To do this, the user is responsible for renaming the underlying Virtual Storage Access Method (VSAM) data sets.

With the enhancements of DB2 9 for z/OS, it is easier and faster to perform the cloning process. The need to free STOGROUPs, while creating dummy STOGROUPs, and dropping and reassigning the objects, is no longer necessary.

2.2 Changing the schema of the database objects

Within SAP databases, the default of the schema name was SAPR3, in addition to the system schema name of SYSIBM. In the last few years, the use of Web applications has increased. The new Web applications have been developed mostly with Java workbench and need the new extension Java stack.

Cloning the whole database system is one of the reasons why schema names must be changed.

2.2.1 Naming convention for the Java stack

During the installation of the Java stack, the user is asked for the triple-digit SAP system identification (SAPSID). This SAPSID is one part of the user-defined schema name, which forms the naming convention:

```
SAP<SID>DB
```

2.2.2 Schema switch with CATMAINT

To find the different schemas that are used to identify the tables to be changed, you can use the SQL in Example 2-1.

Example 2-1 SQL to obtain creator names

```
SELECT DISTINCT CREATOR FROM SYSIBM.SYSTABLES
UNION
SELECT DISTINCT CREATOR FROM SYSIBM.SYSVIEWS
```

Example 2-2 shows the result set from the SQL statement in Example 2-1.

Example 2-2 Result set from the SQL statement

```
---------+---------+---------+---------+-------
CREATOR
---------+---------+---------+---------+-------
DSN8810
DSN8910
DSNACC
DSNRGCOL
SAPDB9
SAPR3
SYSIBM
```

Note: You can never change the schema named SYSIBM.

If you want to switch from the formerly used schema SAPR3 to something else, for example SAPDB9, run the job control language (JCL) as shown in Example 2-3.

Example 2-3 CATMAINT with the schema switch

```
//CATM  EXEC PGM=DSNUTILB,PARM='D2S1,JOB2'
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD  *
      CATMAINT UPDATE
      SCHEMA SWITCH(SAPR3,SAPDB9)
```

Then you receive the output shown in Example 2-4.

Example 2-4 Output from the CATMAINT utility

```
DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = HRUFC
DSNUGTIS - PROCESSING SYSIN AS EBCDIC
DSNUGUTC -  CATMAINT UPDATE SCHEMA SWITCH(SAPR3, SAPDB9)
DSNUECM0 - CATMAINT UPDATE PHASE 1 STARTED
DSNUECM0 - CATMAINT UPDATE STATUS - VALIDATING SCHEMA/OWNER/VCAT
OPTIONS
DSNUECM0 - CATMAINT UPDATE STATUS - UPDATING SCHEMA/OWNER/VCAT NAMES
DSNUECM0 - CATMAINT UPDATE PHASE 1 COMPLETED
DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
```

After you restart DB2 in normal mode (CATMAINT use is only allowed when in MAINT mode), you must modify all user-written application programs or formerly used SQL that used the old schema names.

If the changes are completed, you see the result of any SQL using the old schema names as shown in Example 2-5.

Example 2-5 SQL after the schema switch

```
SELECT * FROM SAPR3.TABLETST;

---------+---------+---------+---------+---------+---------+---------+-
-
 DSNT408I SQLCODE = -204, ERROR:  SAPR3.TABLETST IS AN UNDEFINED NAME
 DSNT418I SQLSTATE   = 42704 SQLSTATE RETURN CODE
 DSNT415I SQLERRP    = DSNXOTL SQL PROCEDURE DETECTING ERROR
 DSNT416I SQLERRD    = -500 0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
 DSNT416I SQLERRD     = X'FFFFFE0C'  X'00000000'  X'00000000'
X'FFFFFFFF'
         X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION

---------+---------+---------+---------+---------+---------+---------+-
-
 DSNE618I ROLLBACK PERFORMED, SQLCODE IS 0
```

2.2.3 Catalog columns updated by the SCHEMA option

Table 2-1 shows the columns and their corresponding tables that are updated by the SCHEMA option.

Table 2-1 Columns and their corresponding tables that are updated

Tables	Columns
SYSAUXRELS	TBOWNER,AUXTBOWNER
SYSCHECKDEP	TBOWNER
SYSCHECKS	TBOWNER,CREATOR
SYSCHECKS2	TBOWNER,PATHSCHEMAS
SYSCOLAUTH	CREATOR,GRANTEE,GRANTOR
SYSCOLDIST	TBOWNER
SYSCOLDISTSTATS	TBOWNER
SYSCOLDIST_HIST	TBOWNER
SYSCOLSTATS	TBOWNER
SYSCOLUMNS	TBCREATOR,TYPESCHEMA
SYSCOLUMNS_HIST	TBCREATOR
SYSCONSTDEP	BSCHEMA,DTBCREATOR,DTBOWNER
SYSDATABASE	CREATOR,CREATEDBY
SYSDATATYPES	SCHEMA,OWNER,CREATEDBY,SOURCESCHEMA
SYSDBAUTH	GRANTEE,GRANTOR
SYSDBRM	PLCREATOR
SYSDEPENDENCIES	BSCHEMA,BOWNER,DSCHEMA,DOWNER
SYSENVIRONMENT	CURRENT_SCHEMA,PATHSCHEMASs
SYSFIELDS	TBCREATOR
SYSFOREIGNKEYS	CREATOR
SYSINDEXES	CREATOR,TBCREATOR,CREATEDBY,OWNER
SYSINDEXES_HIST	CREATOR,TBCREATOR

Tables	Columns
SYSINDEXPART	IXCREATOR
SYSINDEXPARTSTATS	CREATOR
SYSINDEXPART_HIST	IXCREATOR
SYSINDEXSTATS	OWNER
SYSINDEXSTATS_HIST	OWNER
SYSJARCONTENTS	JARSCHEMA
SYSJAROBJECTS	JARSCHEMA,OWNER
SYSJAVAOPTS	JARSCHEMA,BUILDSCHEMA,BUILDOWNER
SYSJAVAPATHS	JARSCHEMA,OWNER,PE_JARSCHEMA,JARSCHEMA
SYSKEYCOLUSE	TBCREATOR
SYSKEYS	IXCREATOR
SYSKEYTARGETS	IXSCHEMA,TYPESCHEMA
SYSKEYTARGETS_HIST	IXSCHEMA,TYPESCHEMA
SYSKEYTGTDIST	IXSCHEMA
SYSKEYTGTDIST_HIST	IXSCHEMA
SYSKEYTGTDISTSTATS	IXSCHEMA
SYSKEYTARGETSTATS	IXSCHEMA
SYSOBJROLEDEP	DEFINER,DSCHEMA
SYSPACKAGE	OWNER,CREATOR,QUALIFIER,PATHSCHEMAS
SYSPACKAUTH	GRANTOR,GRANTEE
SYSPACKDEP	BQUALIFIER,DOWNER
SYSPARMS	SCHEMA,OWNER,TYPESCHEMA
SYSPENDINGDDL	OBJSCHEMA
SYSPLAN	CREATOR,QUALIFIER,PATHSCHEMAS
SYSPLANAUTH	GRANTOR,GRANTEE
SYSPLANDEP	BCREATOR

Tables	Columns
SYSRELS	CREATOR,REFTBCREATOR,IXOWNER
SYSRESAUTH	QUALIFIER,GRANTOR,GRANTEE
SYSROLES	DEFINER,DEFINERTYPE
SYSROUTINEAUTH	SCHEMA,GRANTOR,GRANTEE
SYSROUTINES	SCHEMA,OWNER,CREATEDBY
SYSROUTINES	SOURCESCHEMA,JARSCHEMA
SYSROUTINES_OPTS	SCHEMA,BUILDSCHEMA,BUILDOWNER
SYSROUTINES_SRC	SCHEMA
SYSSCHEMAAUTH	SCHEMANAME,GRANTOR,GRANTEE
SYSSEQUENCEAUTH	SCHEMA,GRANTOR,GRANTEE
SYSSEQUENCES	SCHEMA,OWNER,CREATEDBY
SYSSEQUENCESDEP	DCREATOR,BSCHEMA,DSCHEMA
SYSSTMT	PLCREATOR
SYSSTOGROUP	CREATOR,VCATNAME,CREATEDBY
SYSSYNONYMS	CREATOR,TBCREATOR,CREATEDBY
SYSTABAUTH	SCREATOR,TCREATOR,GRANTOR,GRANTEE
SYSTABCONST	TBCREATOR,CREATOR,IXOWNER
SYSTABLEPART	IXCREATOR
SYSTABLES	CREATOR,CREATEDBY,TBCREATOR,OWNER
SYSTABLESPACE	CREATOR,CREATEDBY
SYSTABLES_HIST	CREATOR
SYSTABSTATS	OWNER
SYSTABSTATS_HIST	OWNER
SYSTRIGGERS	SCHEMA,OWNER,CREATEDBY,TBOWNER
SYSUSERAUTH	GRANTOR,GRANTEE
SYSVIEWDEP	BCREATOR,DCREATOR,BSCHEMA,DOWNER

Tables	Columns
SYSVIEWS	CREATOR,OWNER,PATHSCHEMAS
SYSVOLUMNS	SGCREATOR
SYSXMLRELS	TBOWNER,XMLTBOWNER

2.3 Changing the owner of the database objects

The CATMAINT utility changes the ownership of objects from a user to a role. A user who runs this utility must run it under a trusted context with a role, which becomes the owner. Privileges held on the object are transferred from the original owner to the role. The original user can be the grantor or grantee. The original owners no longer have any privileges to the object after the utility completes.

Information about available roles are given in the catalog table named SYSIBM.SYSROLES. To see the available roles, enter the following SQL statement:

```
SELECT * FROM SYSIBM.SYSROLES;
```

Example 2-6 shows the result set in our case.

Example 2-6 Roles available in our system

```
---------+---------+---------+---------+---------+---------+
NAME
---------+---------+---------+---------+---------+---------+
CTX1ROLE
SAP1ROLE
DSNE610I NUMBER OF ROWS DISPLAYED IS 2
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+
```

2.3.1 Owner change with CATMAINT

You must be running under a trusted context with a role to run the CATMAINT utility. In Example 2-6, the trusted context name is CTX1ROLE. A trusted context establishes a trusted relationship between DB2 and an external entity, such as a middleware server. A series of trust attributes is evaluated to determine if a specific context can be trusted. SYSIBM is not allowed as an owner name. The ownership of roles is changed like other objects. Example 2-7 shows an example of this, where CTX1ROLE is the name of the role.

Example 2-7 CATMAINT to change the role

```
//CATM  EXEC PGM=DSNUTILB,PARM='D2S1,JOB3'
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD  *
      CATMAINT UPDATE
      OWNER FROM(JOHNDOE) TO CTX1ROLE
```

2.3.2 Catalog columns updated by the OWNER option

Table 2-2 outlines the columns, with their corresponding tables, that are updated after using the OWNER option on the CATMAINT UPDATE command.

Table 2-2 Catalog columns that are updated by the OWNER option

Tables	**Columns**
SYSCOLAUTH	GRANTORTYPE,GRANTOR,GRANTEE,GRANTEETYPE,CREATOR
SYSCONSTDEP	OWNERTYPE,DTBOWNER
SYSCONTEXT	DEFINER,DEFINERTYPE
SYSDATABASE	CREATORTYPE,CREATOR
SYSDATATYPES	OWNERTYPE,OWNER
SYSDBAUTH	GRANTORTYPE,GRANTOR,GRANTEETYPE,GRANTEE
SYSDBRM	PLCREATORTYPE,PLCREATOR
SYSINDEXES	OWNERTYPE,OWNER
SYSJAROBJECTS	OWNERTYPE,OWNER
SYSPACKAGE	OWNERTYPE,OWNER
SYSPACKAUTH	GRANTORTYPE,GRANTOR,GRANTEETYPE,GRANTEE
SYSPACKDEP	DOWNERTYPE,DOWNER
SYSPARMS	OWNERTYPE,OWNER
SYSPLAN	CREATORTYPE,CREATOR

Tables	Columns
SYSPLANAUTH	GRANTORTYPE,GRANTOR,GRANTEETYPE,GRANTEE
SYSRESAUTH	GRANTORTYPE,GRANTOR,GRANTEETYPE,GRANTEE
SYSROLES	DEFINER,DEFINERTYPE
SYSROUTINEAUTH	GRANTORTYPE,GRANTOR,GRANTEETYPE,GRANTEE
SYSROUTINES	OWNERTYPE,OWNER
SYSSCHEMAAUTH	GRANTEETYPE,GRANTORTYPE,GRANTOR,GRANTEE
SYSSEQUENCEAUTH	GRANTORTYPE,GRANTOR,GRANTEETYPE,GRANTEE
SYSSEQUENCES	OWNERTYPE,OWNER
SYSSTOGROUP	CREATORTYPE,CREATOR
SYSSTMT	PLCREATORTYPE,PLCREATOR
SYSSYNONYMS	CREATORTYPE,CREATOR
SYSTABAUTH	GRANTORTYPE,GRANTOR,GRANTEETYPE,GRANTEE
SYSTABCONST	CREATORTYPE,CREATOR
SYSTABLES	OWNERTYPE,OWNER
SYSTABLESPACE	CREATORTYPE,CREATOR
SYSTRIGGERS	OWNERTYPE,OWNER
SYSUSERAUTH(*)	GRANTORTYPE,GRANTOR,GRANTEETYPE,GRANTEE
SYSVIEWDEP	OWNERTYPE,DOWNER
SYSVIEWS	OWNERTYPE,OWNER
SYSOBJROLEDEP	New rows will be created

2.4 Changing the VCAT names

You enter VCAT names in DSNZPARM for the DB2 system data sets (the directory and catalog) at installation. You can retrieve them with DSNJU004 (print log map) or by running stored procedure DSN8ED7, which you can set up with installation job DSNTEJ6Z.

Changing the HLQ of these system data sets can only be done by altering the bootstrap dataset (BSDS) entry with DSNJU003 (change log inventory) and their corresponding DSNZPARM entries (with data sharing for each group member).

You can find other VCAT names, especially for SAP data sets, in the following DB2 catalog tables with the column name of VCATNAME:

- SYSIBM.SYSSTOGROUP
- SYSIBM.SYSTABLEPART
- SYSIBM.SYSINDEXPART

2.4.1 The traditional approach

Prior to DB2 9 for z/OS, the procedure to change the VCAT for the SAP data sets was quite complicated. Before removing the old VCATNAME from any STOGROUP, all objects (table spaces and indexes) had to be changed in the following order:

1. Create a new temporary STOGROUP with new VCAT.
2. Stop all index spaces.
3. Change user-defined indexes in the DB2 catalog.
4. Restart the index spaces.
5. Stop all databases.
6. Alter all table spaces to a temporary STOGROUP.
7. Alter all index spaces to a temporary STOGROUP.
8. Drop the old STOGROUPs with the old VCAT.
9. Create new STOGROUPs with a new VCAT.
10. Alter all table spaces using new STOGROUPs.
11. Alter all indexes using new STOGROUPs.
12. Restart all databases.
13. Clean up; for example, drop temporary objects.

For a more detailed description about altering the VCAT this way, read Section 4.1.4 in the SAP guide *System Copy for SAP Systems Based on NW 7.0 SR2 ABAP+Java*, which is provided in the media library (requires user ID and password) on the Web at:

http://service.sap.com/systemcopy

2.4.2 VCAT switch with CATMAINT

The time-consuming process that is described in 2.4.1, “The traditional approach” to create dummy STOGROUPs, and alter and restore clusters in the new STOGROUPs for the sole purpose of changing the VCATNAME, is now a part of the past. With DB2 9 for z/OS, support for altering the VCATNAME is done automatically by the system using the CATMAINT UPDATE utility, as shown in Example 2-8.

Example 2-8 CATMAINT with the VCAT change

```
//CATM  EXEC PGM=DSNUTILB,PARM='D2S1,JOB4'
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD  *
      CATMAINT UPDATE
      VCAT SWITCH(DB2SAP,SAPDB2)
```

The resulting output is shown in Example 2-9.

Example 2-9 CATMAINT output

```
DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = HRUF2C
DSNUGTIS - PROCESSING SYSIN AS EBCDIC
DSNUGUTC -  CATMAINT UPDATE VCAT SWITCH(DB2SAP, SAPDB2)
DSNUECM0 - CATMAINT UPDATE PHASE 1 STARTED
DSNUECM0 - CATMAINT UPDATE STATUS - VALIDATING SCHEMA/OWNER/VCAT
OPTIONS
DSNUECM0 - CATMAINT UPDATE STATUS - UPDATING SCHEMA/OWNER/VCAT NAMES
DSNUECM0 - CATMAINT UPDATE PHASE 1 COMPLETED
DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
```

The CATMAINT UPDATE function changes the catalog name that is used by storage groups, user indexes, and table spaces but only in the catalog tables. The success of altering the VCATNAME can be checked against the related catalog tables. Changing VCAT names has no effect on the system indexes and table spaces in DSNDB01 and DSNDB06. The VCATNAME for the system data sets in SYSIBM.SYSTABLEPART and SYSIBM.SYSINDEXPART are reported as ‘00000001’.

You must rename the underlying cluster names or copy them in a separate step. If the renaming process is not done or fails, you see the message shown in Example 2-10.

*Example 2-10 JESMSGLG of *MSTR*

```
DSNP012I  -D2S1 DSNPCNP0 - ERROR IN ICF CATALOG 132
LOCATE FUNCTION FOR DB2SAP.DSNDBC.DBTEST.TSTEST1.I0001.A001
           CTLGRC=AAAAAA08
           CTLGRSN=AAAAAA08
           CONNECTION-ID=TSO, CORRELATION-ID=USRHRUF,
           LUW-ID=*
```

To rename all existing clusters in a database, use the IDCAMS ALTER utility as shown in Example 2-11.

Example 2-11 Using IDCAMS to rename the clusters in a database

```
//IDCRENAM EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD *
 ALTER  SAPR3.DSNDBC.DBTEST.*.I0001.A001 -
        NEWNAME(SAPDB9.DSNDBC.DBTEST.*.I0001.A001)
 ALTER  SAPR3.DSNDBC.DBTEST.*.I0001.A001 -
        NEWNAME(SAPDB9.DSNDBC.DBTEST.*.I0001.A001)
```

When using generic names (indicated by the asterisk (*) in Example 2-11) to simplify the process, remember that access method services only allows one per dataset name string. This example also assumes that the new qualifier and the old qualifier reside in the same user catalog.

See the *DB2 Version 9.1 for z/OS Administration Guide*, SC18-9840, for procedures to move DB2 data sets.

2.4.3 Catalog columns updated by the VCAT option

Table 2-3 lists the catalog columns, along with their corresponding tables, that are updated by the VCAT option.

Table 2-3 Catalog columns updated by the VCAT option

Tables	Columns
SYSSTOGROUP	VCATNAME
SYSINDEXPART	VCATNAME
SYSTABLEPART	VCATNAME

3

New and enhanced data types

New and enhanced data types are exploited by DB2 9 for z/OS to better alignment with ABAP™ and Java data types used by SAP applications. This chapter discusses the following new and enhanced data types:

- Big integers (BIGINT)
- Binary strings
  - Fixed length binary strings
  - Variable length binary strings
  - Binary large objects or BLOBs
- Decimal float (DECFLOAT)

3.1 BIGINT

A big integer (BIGINT) is a binary integer with a precision of 63 bits. The range of big integers is -9223372036854775808 to +9223372036854775807. BIGINT extends a set of currently supported exact numeric data types, SMALLINT and INTEGER. It is compatible with all numeric data types.

Table 3-1 shows the precedence list that is used to determine the data types to which the data type can be promoted.

Table 3-1 Precedence of numeric data types

Data type	Data type precedence list (in best-to-worst order)
SMALLINT	SMALLINT, INTEGER, BIGINT, decimal, real, double
INTEGER	INTEGER, BIGINT, decimal, real, double
BIGINT	BIGINT, decimal, real, double

In Example 3-1, we create a table with various data types and insert their corresponding values.

Example 3-1 Creating and populating a data type reference table

```
CREATE TABLE NUM_DATATYPE
  (NUM_SMALLINT SMALLINT,NUM_INTEGER INTEGER,NUM_BIGINT BIGINT,
  NUM_DECIMAL DECIMAL(7,5),NUM_REAL REAL,NUM_DOUBLE DOUBLE) ;

  INSERT INTO NUM_DATATYPE (NUM_SMALLINT,NUM_INTEGER,NUM_BIGINT,
  NUM_DECIMAL,NUM_REAL,NUM_DOUBLE)
  VALUES(1,1,1,1,1,2) ;
```

The data type table created in Example 3-1 is then queried using the SQL statement in Example 3-2. The results can assist you in finding the length of the data types with numeric operands.

Example 3-2 BIGINT numeric operands

```
SELECT
  NUM_BIGINT+NUM_SMALLINT AS BIGINT_SMALLINT_OPERATION,
  LENGTH(NUM_BIGINT+NUM_SMALLINT) AS LENGTH
  FROM NUM_DATATYPE ;
---------+---------+---------+---------+---------+---------+
BIGINT_SMALLINT_OPERATION       LENGTH
---------+---------+---------+---------+---------+---------+
                        2            8
```

```
SELECT
  NUM_BIGINT+NUM_INTEGER  AS BIGINT_INTEGER_OPERATION,
  LENGTH(NUM_BIGINT+NUM_INTEGER) AS LENGTH
  FROM NUM_DATATYPE ;
---------+---------+---------+---------+---------+---------+
BIGINT_INTEGER_OPERATION       LENGTH
---------+---------+---------+---------+---------+---------+
                       2            8

SELECT
  NUM_BIGINT+NUM_BIGINT AS BIGINT_BIGINT_OPERATION,
  LENGTH(NUM_BIGINT+NUM_BIGINT) AS LENGTH
  FROM NUM_DATATYPE ;
---------+---------+---------+---------+---------+---------+
BIGINT_BIGINT_OPERATION       LENGTH
---------+---------+---------+---------+---------+---------+
                      2            8

SELECT
  NUM_DECIMAL+NUM_BIGINT AS DECIMAL_BIGINT_OPERATION,
  LENGTH(NUM_DECIMAL+NUM_BIGINT) AS LENGTH,
  LENGTH(NUM_DECIMAL) AS DECIMAL_LENGTH
  FROM NUM_DATATYPE ;
---------+---------+---------+---------+---------+---------+
   DECIMAL_BIGINT_OPERATION       LENGTH  DECIMAL_LENGTH
---------+---------+---------+---------+---------+---------+
                    2.00000           13               4

SELECT
  NUM_REAL+NUM_BIGINT AS REAL_BIGINT_OPERATION,
  LENGTH(NUM_REAL+NUM_BIGINT) AS LENGTH,
  NUM_DOUBLE AS DOUBE_COLUMN
  FROM NUM_DATATYPE ;
---------+---------+---------+---------+---------+---------+--
  REAL_BIGINT_OPERATION       LENGTH             DOUBE_COLUMN
---------+---------+---------+---------+---------+---------+--
+0.2000000000000000E+01            8  +0.2000000000000000E+01
```

Table 3-2 shows the result data types with numeric operands.

Table 3-2 Result data types with numeric operands

If one operand is...	And the other operand is...	The data type of the result is...
BIGINT	SMALLINT	BIGINT
BIGINT	INTEGER	BIGINT
BIGINT	BIGINT	BIGINT
DECIMAL(w,x)	BIGINT	DECIMAL(p,x) where p=x+max(w-x,19)
REAL	BIGINT	DOUBLE

If you want to use CAST with BIGINT, refer to Table 3-3.

Table 3-3 Supported CASTs between built-in data types

CAST from data type	CAST to data type
SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE, CHAR, VARCHAR, GRAPHIC, VARGRAPHIC	BIGINT
BIGINT	SMALLINT, INTEGER, DECIMAL, REAL, DOUBLE, CHAR, VARCHAR

3.2 BINARY strings

A *binary string* is a sequence of bytes. The length of a binary string is the number of bytes in the sequence. Binary strings are not associated with any CCSID. There are three binary string data types:

- BINARY (introduced in V9)
- VARBINARY (BINARY VARYING, introduced in V9)
- BLOB (BINARY LARGE OBJECT)

The BINARY and VARBINARY data types extend the current support of binary strings (BLOB) and are compatible with BLOB data type. They are not compatible with character string data types.

A binary string column is useful for storing non-character data, such as encoded or compressed data, pictures, voice, and mixed media. Another use is to hold structured data for exploitation by distinct types, user-defined functions, and stored procedures.

Note: Although binary strings and FOR BIT DATA character strings might be used for similar purposes, the two data types are not compatible. The BINARY, BLOB, VARBINARY built-in functions, and CAST specification can be used to change a FOR BIT DATA character string into a binary string.

BINARY (fixed-length binary strings)

The type of fixed-length binary strings is BINARY. When fixed-length binary string distinct types, columns, and variables are defined, the length attribute is specified. Data types and all values have the same length. For a fixed-length binary string, the length attribute must be between 1 and 255 inclusive.

VARBINARY (varying-length binary strings)

When varying-length binary strings, distinct types, columns, and variables are defined, the maximum length is specified. This length becomes the length attribute. Actual length values might have a smaller value than the length attribute value. For varying-length binary strings, the actual length specifies the number of bytes in the string. The maximum length of VARBINARY is 32704 bytes.

Table 3-4 Precedence of string data types

Data type	Data type precedence list (in best-to-worst order)
BINARY	BINARY, VARBINARY, BLOB
VARBINARY	VARBINARY, BLOB

Binary string comparisons are always performed according to the binary values. Additionally, two binary strings are equal only if the lengths of the two strings are identical. If the strings are equal up to the length of the shorter string length, the shorter string is considered less than the longer string even when the remaining bytes in the longer string are hexadecimal zeros.

3.3 DECFLOAT (decimal floating point)

Decimal floating point (DECFLOAT) is similar to both packed decimal (or binary coded decimal), and floating point (IEEE or Hex). The main advantages that decimal floating point has over packed decimal or binary floating point (IEEE) is that it can contain a larger number, both in terms of digits of significance, and in terms of exponent. Also the rules for manipulation of DECFLOAT more closely follow the rules for manipulation of packed decimal. That is DECFLOAT processing deals with exact numbers, not numerical approximations that IEEE floating point deals with.

Binary integer includes a small integer, large integer, and big integer. Binary numbers are exact representations of integers. Decimal numbers are exact representations of real numbers. Binary and decimal numbers are considered exact numeric types.

Decimal floating point numbers include DECFLOAT(16) and DECFLOAT(34), which are capable of representing either 16 or 34 significant digits. Floating point includes single precision and double precision. Floating-point numbers are approximations of real numbers and are considered approximate numeric types. DECFLOAT is compatible with all numeric data types. You can also check the result data types with numeric operands. Table 3-5 summarizes these operands with DECFLOAT.

Table 3-5 Operands with DECFLOAT

One operand	Other operand	Data type of the result
DECFLOAT(n)	SAMLLINT	DECFLOAT(n)
DECFLOAT(n)	INTEGER	DECFLOAT(n)
DECFLOAT(n)	BIGINT	DECFLOAT(34)
DECFLOAT(n)	DECIMAL(y,z)	DECFLOAT(n) where y <= 16 or DECFLOAT (34) where y > 16
DECFLOAT(n)	REAL	DECFLOAT(n)
DECFLOAT(n)	DOUBLE	DECFLOAT(n)
DECFLOAT(n)	DECFLOAT(m)	DECFLOAT(max(n,m))

DECFLOAT is not exploited by SAP because DB2 does not support the definition of indexes on DECFLOAT columns.

4

New and enhanced index capabilities

In this chapter, we discuss some of the new or changed index capabilities with DB2 9 on z/OS and how they might affect SAP applications. We present examples of the new or changed index, explain how to invoke them, and discuss the implications they have on SAP.

4.1 RENAME INDEX

The RENAME statement has been changed with the addition of a new clause, INDEX. To execute this statement, you need one of the following privileges:

- Be the owner of the table, for which the index was defined
- Have ownership of the index that is to be renamed
- Have DBADM, DBCTRL, or DBMAINT authority on the database where the rename will occur
- SYSADM or SYSCTRL authority with the particular DB2 system on which you are working

The syntax for renaming an index is:

```
RENAME INDEX source-index-name TO new-index-name
```

The privileges that you had on the source-index-name are retained on the renamed index.

When you execute the RENAME INDEX statement, the contents of the plan table do not change. You must run an EXPLAIN statement for the plan table to be updated with the renamed index. Another occurrence of the RENAME INDEX statement is the object identifier for that index, which is maintained as the original index. When you use the rename index statement, the associated catalog tables for indexes are updated. You can find a list of these tables in *DB2 Version 9.1 for z/OS SQL Reference*, SC18-9854.

The capability to rename objects is critical in order for SAP to repackage existing tables and indexes in its applications. For example, when objects from SAP banking applications are moved into the namespace of the SAP basis infrastructure, it is important to rename them quickly as online operations. The approach to drop and recreate indexes is too disruptive.

Another usage case for RENAME INDEX is to consolidate the different index naming schemes that were used in past SAP releases and that are still in effect when SAP systems are moved to later SAP releases.

4.1.1 Example of RENAME INDEX

The following example shows how to change the name of an index to something that may be more meaningful:

```
RENAME INDEX PAY.INDX01 TO PAY.DEPARTMENT_INDX01
```

In this example, we change the name of the index on the PAY table to further identify it as the index on the PAY table by DEPARTMENT, where perhaps DEPARTMENT is the column that the index is built on.

4.1.2 New tables and indexes

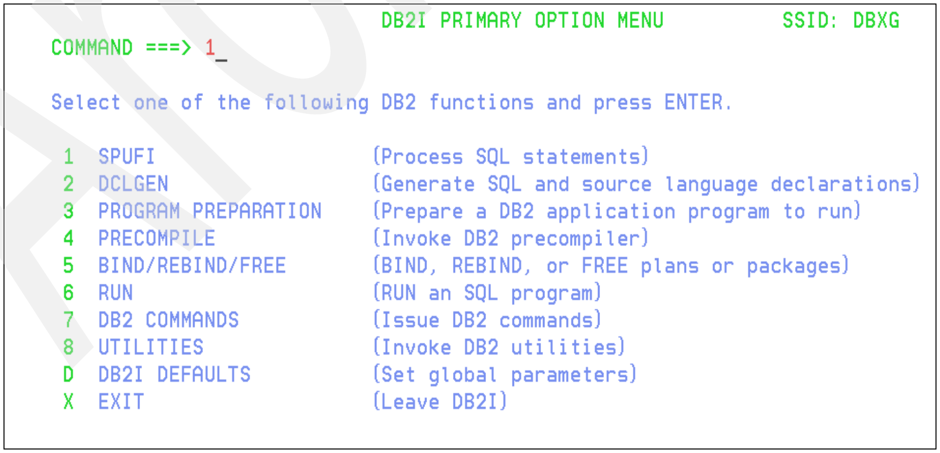
New tables and indexes have been created (see Table 4-1) for the RENAME INDEX statement.

Table 4-1 Table of new tables and indexes

New table name	New index name
SYSDEPENDENCIES	DSNONX01
SYSENVIRONMENT	DSNONX02
SYSJAVAPATHS	DSNRTXX03
SYSRELS	DSNDLX03
SYSTABLESPACESTATS	DSNDPX04

4.1.3 Testing the RENAME INDEX

To test the RENAME INDEX statement, we rename an index on a DB2 9 for z/OS system using the SQL Processing Using File Input (SPUFI) application. We log on to the system where DB2 9 for z/OS is installed. Then we navigate to the DB2I Primary Option Menu (Figure 4-1). On this menu, we type 1 to select SPUFI so we can process our work.



```
                         DB2I PRIMARY OPTION MENU          SSID: DBXG
COMMAND ===> 1_

Select one of the following DB2 functions and press ENTER.

 1  SPUFI                 (Process SQL statements)
 2  DCLGEN                (Generate SQL and source language declarations)
 3  PROGRAM PREPARATION   (Prepare a DB2 application program to run)
 4  PRECOMPILE            (Invoke DB2 precompiler)
 5  BIND/REBIND/FREE      (BIND, REBIND, or FREE plans or packages)
 6  RUN                   (RUN an SQL program)
 7  DB2 COMMANDS          (Issue DB2 commands)
 8  UTILITIES             (Invoke DB2 utilities)
 D  DB2I DEFAULTS         (Set global parameters)
 X  EXIT                  (Leave DB2I)
```

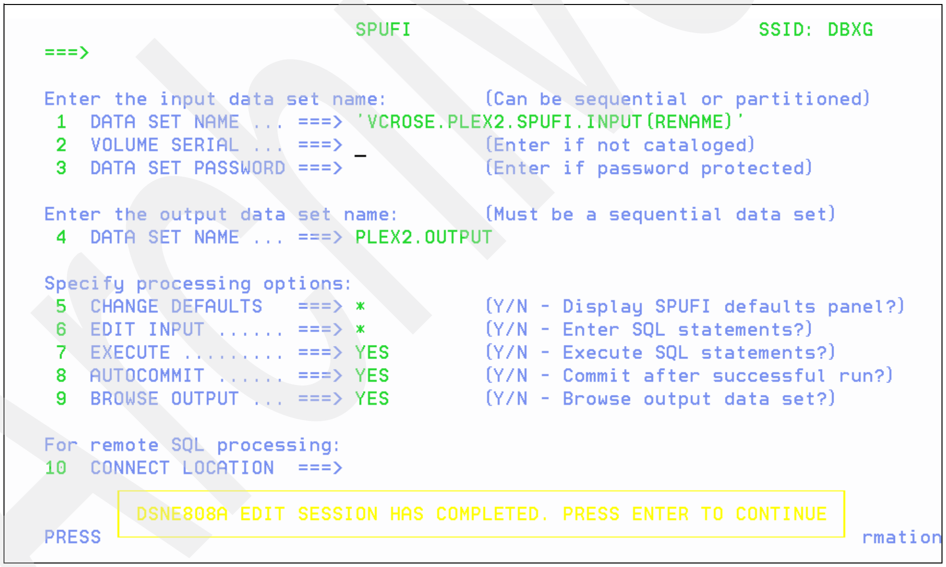
Figure 4-1 Selecting SPUFI

In SPUFI, we navigate to a file where we will create some SQL statements for querying the catalog tables, SQL used to rename the index, and then another query to show the final output (see Figure 4-2).

```
SELECT SUBSTR(NAME,1,35),SUBSTR(TBNAME,1,20),INDEXSPACE
FROM SYSIBM.SYSINDEXES
WHERE NAME LIKE 'MYCUT%';
--
RENAME INDEX MYCUT_CID_XMLIDX TO VINCE_CID_XMLIDX01;
--
SELECT SUBSTR(NAME,1,35),SUBSTR(TBNAME,1,20),INDEXSPACE
FROM SYSIBM.SYSINDEXES
WHERE NAME LIKE 'VINCE%';
```

Figure 4-2 SQL statements to query catalog tables, rename the index, view final output

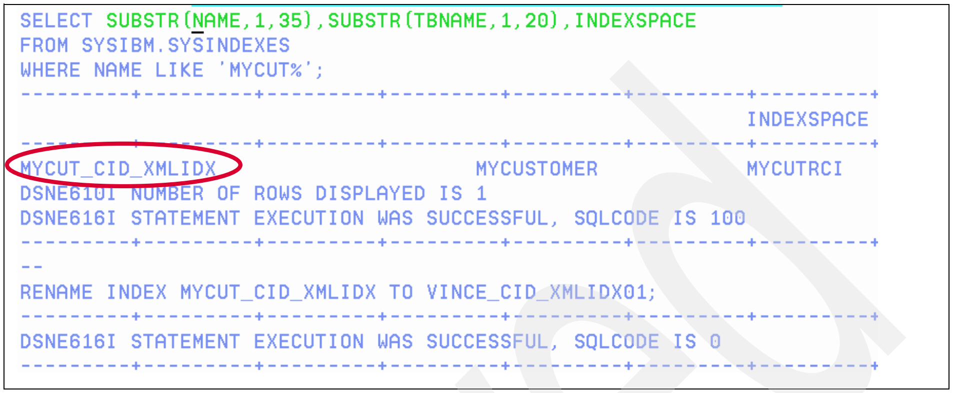
After we create our queries and the RENAME statement, we exit the data set. We reach the SPUFI screen shown in Figure 4-3, so we can run the command.



```
                         SPUFI                             SSID: DBXG
===>

Enter the input data set name:        (Can be sequential or partitioned)
 1  DATA SET NAME ... ===> 'VCROSE.PLEX2.SPUFI.INPUT(RENAME)'
 2  VOLUME SERIAL ... ===> _          (Enter if not cataloged)
 3  DATA SET PASSWORD ===>            (Enter if password protected)

Enter the output data set name:       (Must be a sequential data set)
 4  DATA SET NAME ... ===> PLEX2.OUTPUT

Specify processing options:
 5  CHANGE DEFAULTS   ===> *          (Y/N - Display SPUFI defaults panel?)
 6  EDIT INPUT ...... ===> *          (Y/N - Enter SQL statements?)
 7  EXECUTE ......... ===> YES        (Y/N - Execute SQL statements?)
 8  AUTOCOMMIT ...... ===> YES        (Y/N - Commit after successful run?)
 9  BROWSE OUTPUT ... ===> YES        (Y/N - Browse output data set?)

For remote SQL processing:
10  CONNECT LOCATION  ===>
      DSNE808A EDIT SESSION HAS COMPLETED. PRESS ENTER TO CONTINUE
PRESS                                                               rmation
```

Figure 4-3 SPUFI edit screen

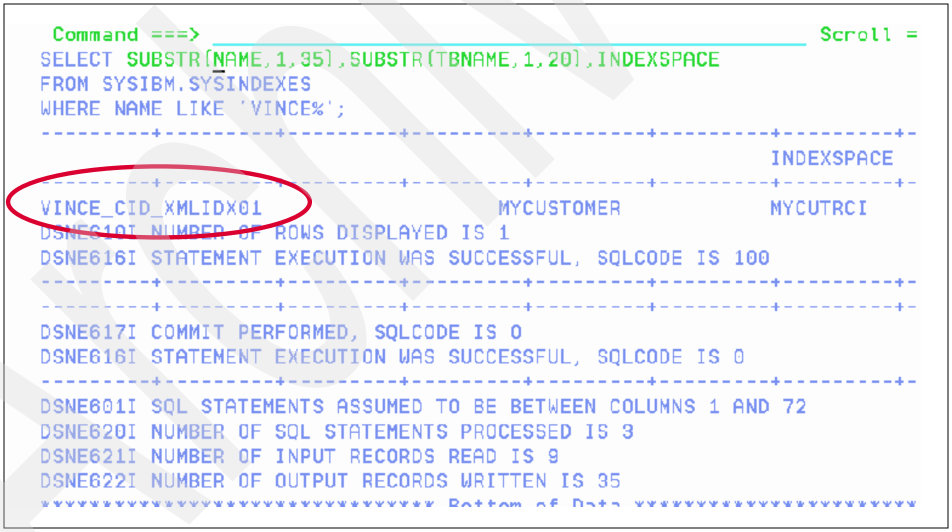
Figure 4-4 shows the first part of the results from the first query and the RENAME statement. Notice that the index name before the RENAME is MYCUT_CID_XMLIDX.



```
SELECT SUBSTR(NAME,1,35),SUBSTR(TBNAME,1,20),INDEXSPACE
FROM SYSIBM.SYSINDEXES
WHERE NAME LIKE 'MYCUT%';
---------+---------+---------+---------+---------+---------+---------+
                                                           INDEXSPACE
---------+---------+---------+---------+---------+---------+---------+
MYCUT_CID_XMLIDX                     MYCUSTOMER            MYCUTRCI
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+
--
RENAME INDEX MYCUT_CID_XMLIDX TO VINCE_CID_XMLIDX01;
---------+---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+
```

Figure 4-4 Results from the query rename

In Figure 4-5, you see the results of the RENAME after the query is repeated.



```
 Command ===>                                                 Scroll =
SELECT SUBSTR(NAME,1,35),SUBSTR(TBNAME,1,20),INDEXSPACE
FROM SYSIBM.SYSINDEXES
WHERE NAME LIKE 'VINCE%';
---------+---------+---------+---------+---------+---------+---------+-
                                                           INDEXSPACE
---------+---------+---------+---------+---------+---------+---------+-
VINCE_CID_XMLIDX01                   MYCUSTOMER            MYCUTRCI
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+-
---------+---------+---------+---------+---------+---------+---------+-
DSNE617I COMMIT PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+-
DSNE601I SQL STATEMENTS ASSUMED TO BE BETWEEN COLUMNS 1 AND 72
DSNE620I NUMBER OF SQL STATEMENTS PROCESSED IS 3
DSNE621I NUMBER OF INPUT RECORDS READ IS 9
DSNE622I NUMBER OF OUTPUT RECORDS WRITTEN IS 35
******************************** Bottom of Data ************************
```

Figure 4-5 Results from the RENAME after the query is repeated

As you can see in Figure 4-5, the index name has been successfully changed to VINCE_CID_XMLIDX01 and the name of the index space did not change.

4.1.4 Index compression

You use index compression for applications that do sequential inserts or that have few deletions run against the DB2 tables. Doing inserts and deletions randomly can also hurt performance of the compression routine. Ideally you should use index compression for applications where the indexes are created primarily for reading data.

You can turn on index compression by creating an index or altering an index with the COMPRESS YES parameter. When you create an index to use the compression routine, the buffer pool for that index must be either 8K or 16K. The compression for the index takes place immediately.

If you decide to alter your index using ALTER and specify COMPRESS YES, then index compression does not take place until a REORG INDEX, REBUILD INDEX, or a REORG TABLESPACE occurs. The reorg is placed in ADVISORY REORG PENDING status. If your index is also partitioned, it takes place on all partitions.

If you decide to compress your indexes, an added bonus is the saving of disk space. The utilities that you use run faster with COMPRESS YES as an added bonus.

To find whether your current index is compressed, query the SYSIBM.SYSINDEXES table. There is a one-character column named COMPRESS. There is either an N for “not compressed”, which is the default, or a Y for “compressed”.

SAP compresses all DB2 table spaces in the case of Unicode. Because the performance overhead of index compression can be extremely high, compress only selected indexes, preferably those that are more buffer-pool resistant.

4.1.5 Data compression in SAP

In many cases, using data compression can significantly reduce the amount of DASD space necessary to store the data. This is on your table space and now with DB2 9 for z/OS, in your index space. Previous to this, you could only compress your table space. Implicitly created table spaces, which are new with DB2 9 for z/OS, are automatically compressed by DB2 if the ZPARM IMPTSCMP is set to YES.

In order to compress data, you specify COMPRESS YES on CREATE or ALTER TABLESPACE and COMPRESS YES on the CREATE or ALTER INDEX statements.

In addition to the savings on DASD, you gain a higher buffer pool hit ratio and fewer I/Os and getpage operations.

While SAP compresses all the table spaces that are used for Unicode systems by default, turning on index compression is done on a case-by-case decision. Depending on the characteristics of the workload, the benefits of the reduced disk space outweighs potential performance overheads.

4.2 Functional indexes

Functional indexes are described better as *index on expression*. It is used by the optimizer in such a way that when the user creates the index on a general expression, it improves the performance of your query.

4.2.1 Index on expression

Index on expression is a new type of index in DB2 9 for z/OS. You can create the index when you want a thorough scan of your query, which involves a column expression. The key values are not the same as a simple-index where the index keys are made up of one or more concatenated table columns. The keys for this index are changed by the expressions that you give.

When creating this type of index, some general rules have changed for index on expression. You still use CREATE INDEX. However if you want this index to be clustered, it is not supported at this time.

Now because of this new type of index, there are new catalog tables, as well as updated existing tables.

4.2.2 Creating an index on expression

When you create an index on expression, the index keys are made up of the concatenation of the result or results. You specify the expressions in the ON clause. The expression can be a reference to a column, the invoking of a built-in function, or a general expression.

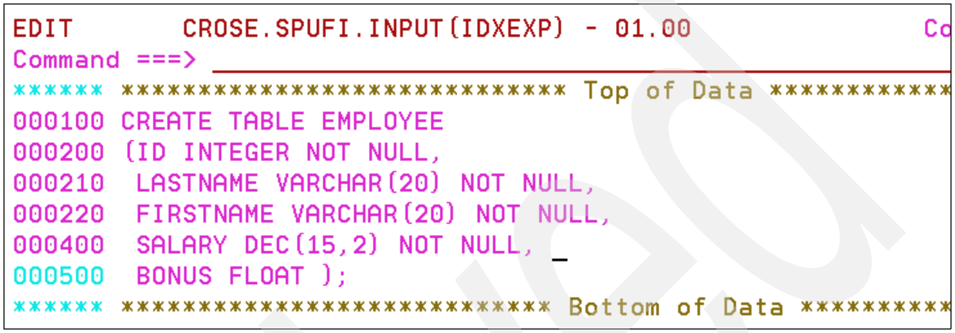
The following example shows an index on expression. First you create a table as indicated by the CREATE TABLE statement:

```
CREATE TABLE employee ( id INTEGER NOT NULL, lastname VARCHAR(20) NOT
NULL, firstname VARCHAR(20) NOT NULL, salary DEC(15,2) NOT NULL,
bonus FLOAT );
```

Then you create an index on expression, as shown in the following example:

```
CREATE INDEX upper_empname ON employee(UPPER(lastname,
'EN_US'), UPPER(firstname, 'EN_US'), id);
```

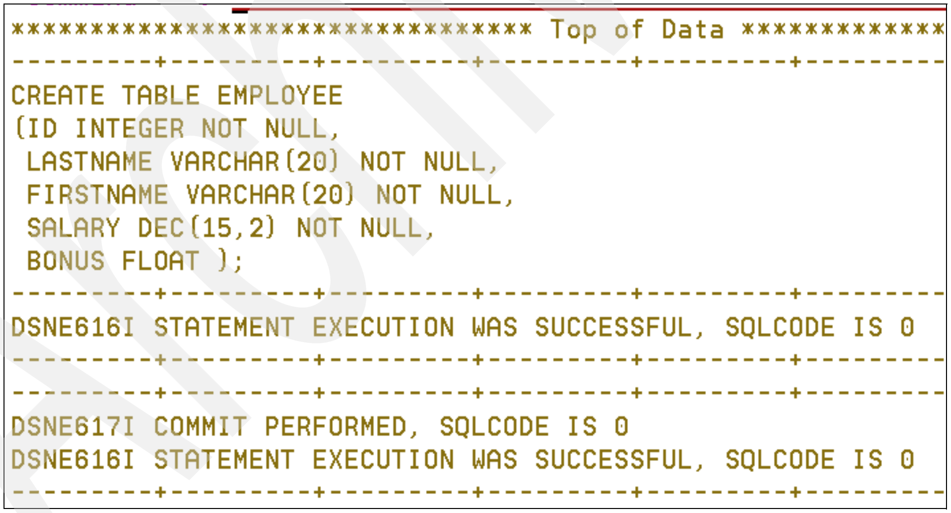
We created the table example in the SPUFI tool on our DB2 9 for z/OS system as shown in Figure 4-6.



```
EDIT       CROSE.SPUFI.INPUT(IDXEXP) - 01.00                   Co
Command ===> 
****** ***************************** Top of Data ************
000100 CREATE TABLE EMPLOYEE
000200 (ID INTEGER NOT NULL,
000210  LASTNAME VARCHAR(20) NOT NULL,
000220  FIRSTNAME VARCHAR(20) NOT NULL,
000400  SALARY DEC(15,2) NOT NULL, _
000500  BONUS FLOAT );
****** **************************** Bottom of Data **********
```

Figure 4-6   Creating a table to use in the test of index on expression

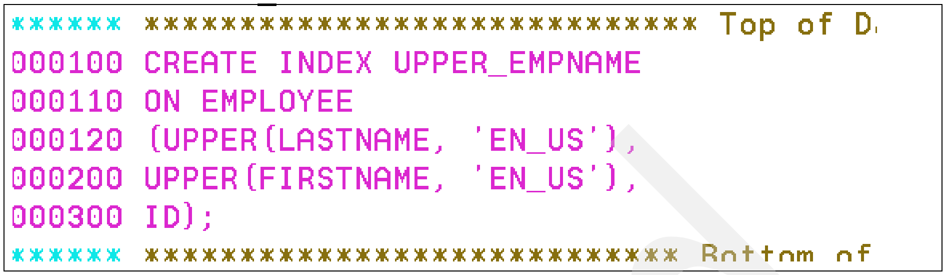
We successfully created the table as shown by the results in Figure 4-7.



```
********************************* Top of Data *************
---------+---------+---------+---------+---------+---------
CREATE TABLE EMPLOYEE
(ID INTEGER NOT NULL,
 LASTNAME VARCHAR(20) NOT NULL,
 FIRSTNAME VARCHAR(20) NOT NULL,
 SALARY DEC(15,2) NOT NULL,
 BONUS FLOAT );
---------+---------+---------+---------+---------+---------
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------
---------+---------+---------+---------+---------+---------
DSNE617I COMMIT PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------
```

Figure 4-7   Results of CREATE TABLE to test the index on expression

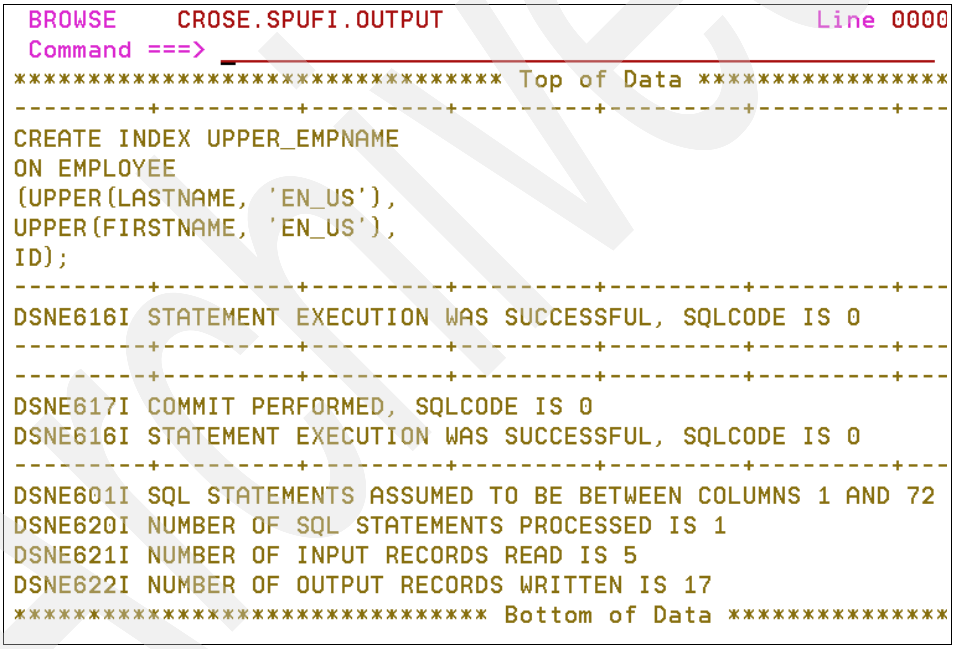
Then we tested the index on expression with the statement shown in Figure 4-8.



```
****** ****************************** Top of D
000100 CREATE INDEX UPPER_EMPNAME
000110 ON EMPLOYEE
000120 (UPPER(LASTNAME, 'EN_US'),
000200 UPPER(FIRSTNAME, 'EN_US'),
000300 ID);
****** ***************************** Bottom of
```

Figure 4-8 Index on expression example

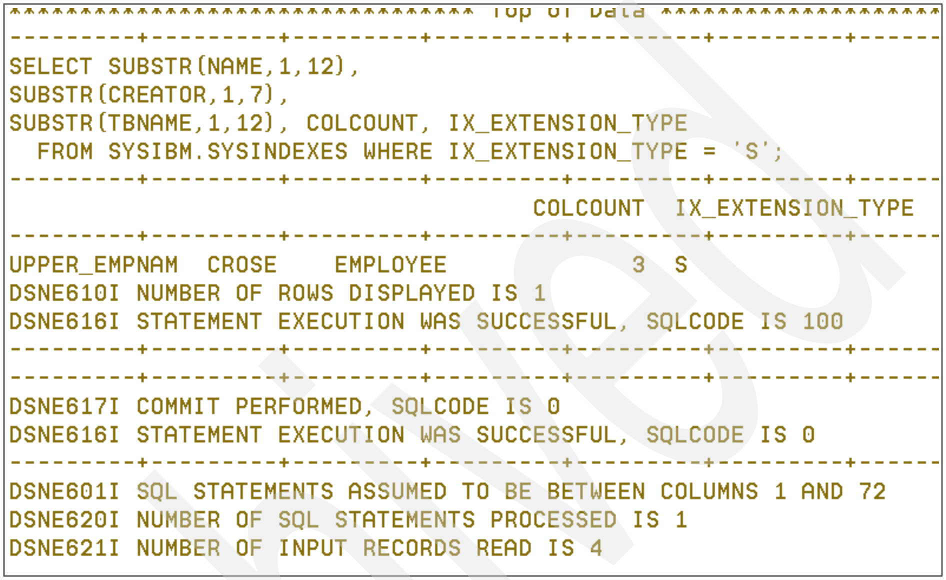
We ran the create index statement, and it was successful as you can see in Figure 4-9.



```
 BROWSE    CROSE.SPUFI.OUTPUT                       Line 0000
 Command ===> _
********************************* Top of Data ****************
---------+---------+---------+---------+---------+---------+---
CREATE INDEX UPPER_EMPNAME
ON EMPLOYEE
(UPPER(LASTNAME, 'EN_US'),
UPPER(FIRSTNAME, 'EN_US'),
ID);
---------+---------+---------+---------+---------+---------+---
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---
---------+---------+---------+---------+---------+---------+---
DSNE617I COMMIT PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---
DSNE601I SQL STATEMENTS ASSUMED TO BE BETWEEN COLUMNS 1 AND 72
DSNE620I NUMBER OF SQL STATEMENTS PROCESSED IS 1
DSNE621I NUMBER OF INPUT RECORDS READ IS 5
DSNE622I NUMBER OF OUTPUT RECORDS WRITTEN IS 17
******************************** Bottom of Data **************
```

Figure 4-9 Successful completion of the create index on expression

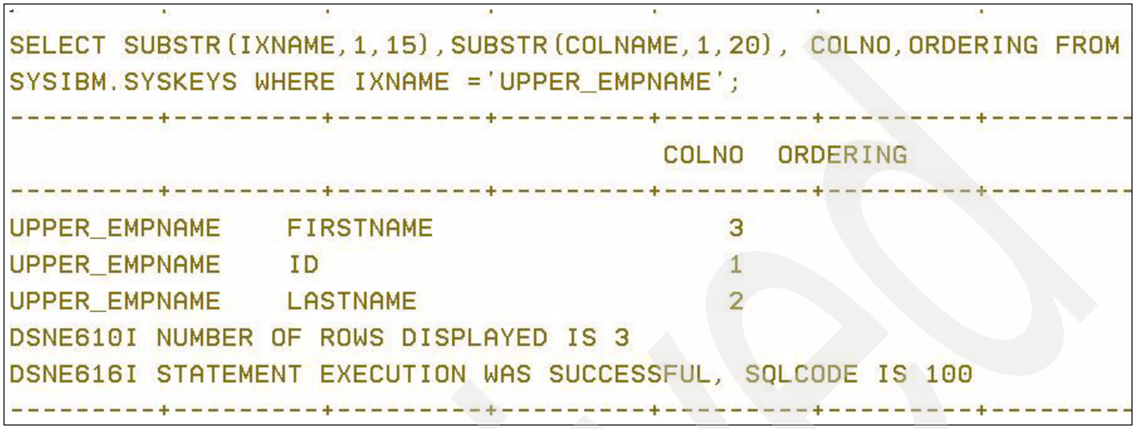
Now that we have created the index, we can see from Figure 4-10 that we truly created an index on expression. Looking at the table SYSIBM.SYSINDEXES, column IX_EXTENSION_TYPE has an ‘S’ in it, indicating an index on a scalar expression extended index type. SYSIBM.SYSINDEXES is one of the catalog tables that had to be changed because of this new feature.



```
********************************* Top of Data ********************
---------+---------+---------+---------+---------+---------+------
SELECT SUBSTR(NAME,1,12),
SUBSTR(CREATOR,1,7),
SUBSTR(TBNAME,1,12), COLCOUNT, IX_EXTENSION_TYPE
  FROM SYSIBM.SYSINDEXES WHERE IX_EXTENSION_TYPE = 'S';
---------+---------+---------+---------+---------+---------+------
                                     COLCOUNT  IX_EXTENSION_TYPE
---------+---------+---------+---------+---------+---------+------
UPPER_EMPNAM  CROSE    EMPLOYEE             3  S
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+------
---------+---------+---------+---------+---------+---------+------
DSNE617I COMMIT PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+------
DSNE601I SQL STATEMENTS ASSUMED TO BE BETWEEN COLUMNS 1 AND 72
DSNE620I NUMBER OF SQL STATEMENTS PROCESSED IS 1
DSNE621I NUMBER OF INPUT RECORDS READ IS 4
```

Figure 4-10 Verifying index on expression

Another table that can show us that the index on expression was created is the SYSIBM.SYSKEYS table. In Figure 4-11, you can see the columns that the index uses. When the ORDERING column is blank, this indicates that the index is an index based on expressions.



```
SELECT SUBSTR(IXNAME,1,15),SUBSTR(COLNAME,1,20), COLNO,ORDERING FROM
SYSIBM.SYSKEYS WHERE IXNAME ='UPPER_EMPNAME';
---------+---------+---------+---------+---------+---------+--------
                                            COLNO  ORDERING
---------+---------+---------+---------+---------+---------+--------
UPPER_EMPNAME    FIRSTNAME                      3
UPPER_EMPNAME    ID                             1
UPPER_EMPNAME    LASTNAME                       2
DSNE610I NUMBER OF ROWS DISPLAYED IS 3
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+--------
```

Figure 4-11   Showing the columns that are used by the index

You can create an index on expression using the parameter UNIQUE. The unique function applies to the values stored in the index and not on table columns. You might want to create an index using the following SQL statement:

```
CREATE UNIQUE INDEX EMPNAME ON EMPLOYEE (SUBSTR(FIRSTNAME,1,1) CONCAT
‘.’ CONCAT LASTNAME);
```

You are essentially create an index that has just the first letter of the first name together with the entire last name. For example, if you have names such as “Veronica Cox” and “Virginia Cox” in your table, V Cox is created in your index. Therefore, you may have a duplicate index entry for “Virginia Cox”.

4.2.3 REGENERATE statement

Another changed feature for indexes is in the ALTER INDEX statement. Here you can use the REGENERATE statement. This is a new clause for the ALTER INDEX statement. If you want to use the REGENERATE statement, then this is the only clause of the ALTER INDEX statement.

When you use this clause, the structure is regenerated. The definition of the index is derived from the system catalog tables along with authorizations and dependencies. When this is successful, the catalog is then subsequently updated with the statistics from the index definitions. Other related statistics are deleted from the catalog.

Upon successful completion, the index is placed in a REBUILD PENDING state. Thus all related dependent packages become invalidated. If the regenerate is unsuccessful, you must drop and recreate the index, which you can only use with indexes that have key expressions.

4.3 Index lookaside for data partitioned secondary indexes

Data partitioned secondary indexes (DPSI) are used more and more in SAP environments because they can be taken advantage of frequently. A drawback of DPSIs before DB2 9 for z/OS was that they did not support index lookaside.

The object of index lookaside is to minimize index GETPAGE operations. DB2 checks whether a required entry is in the index leaf page that was accessed by the previous access to the index. If the entry is found there, a further GETPAGE operation is avoided, and no new index traversal is required. Index lookaside is particularly beneficial for repeated index access in a sequence. Examples for this access type are the accesses to the inner table of a nested loop, hybrid join, or an SQL statement that is executed within a program loop.

DB2 9 for z/OS introduces index lookaside for DPSIs within a partition. The advantage of DPSIs can be exploited if only rows from a single DPSI partition qualify, which is often the case.

You can write queries to take advantage of limited partition scans. This is especially useful when a correlation exists between columns that are in a partitioning index and columns that are in a DPSI. For example, suppose that you create table Q2, with the partitioning index DATE_IX and DPSI ORDERNO_IX (see Example 4-1).

Example 4-1 Table Q2 created with partitioning index DATE_IX and DPSI ORDERNO_IX

```
CREATE TABLESPACE TS2 NUMPARTS 3
CREATE TABLE Q2 (DATE DATE, ORDERNO CHAR(8),
STATE CHAR(2),
 PURCH_AMT DECIMAL(9,2))
  IN TS2
 PARTITION BY (DATE)
  (PARTITION 1 ENDING AT ('2004-12-31'),
   PARTITION 2 ENDING AT ('2005-12-31'),
   PARTITION 3 ENDING AT ('2006-12-31'));
CREATE INDEX DATE_IX ON Q2 (DATE) PARTITIONED CLUSTER;
CREATE INDEX ORDERNO_IX ON Q2 (ORDERNO) PARTITIONED;
```

DPSIs have performance advantages in queries, if you write the query following certain criteria:

- Your query has predicates on DPSI columns.
- Your query has added predicates on partitioned columns of the table. This limits the query to a subset of partitions in the table.

The query shown in Example 4-2 with the SELECT statement uses the DPSI. The key values that need to be searched are limited to just the key values of the eligible partitions, hence, your query does not search all partitions of the table.

Example 4-2 Query using DPSI

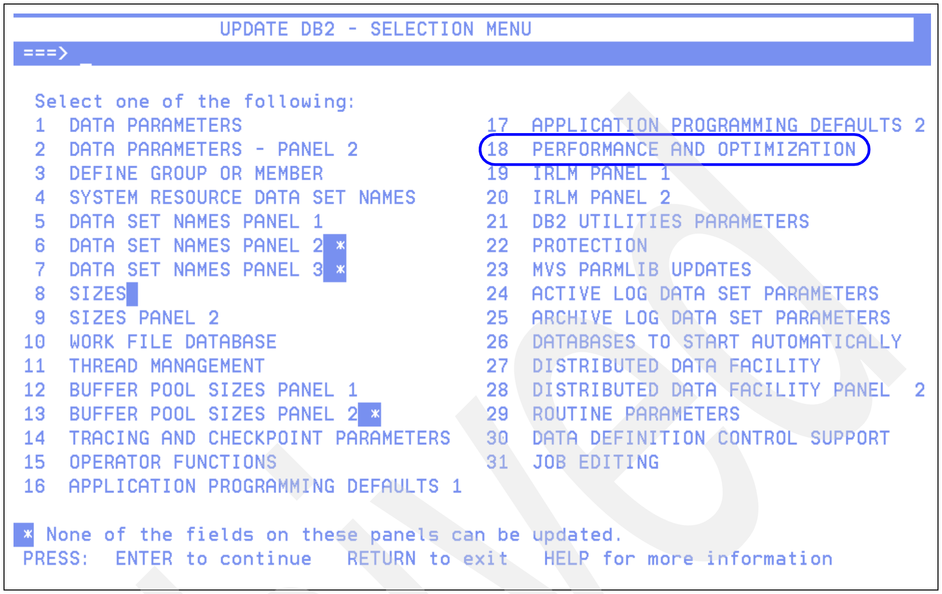
```
Select STATE FROM ARAE_CODES
WHERE AREACODE_NO <= 300 AND STATE = ‘CA’;
```

In SAP, the use of DPSI promotes partition freedom for your applications. It can help you reduce lock contention and improve index availability. This is particularly helpful for utility processing, such as dropping partitions or recovery of an index.

4.4 Generalizing sparse index/in-memory data caching

MXDTCACH is a new installation parameter in DB2 9 for z/OS. It specifies the maximum memory allocated for data caching. The value used for MXDTCACH is in megabytes (MB) or 1,048,576 bytes. The system we use for testing has a value of 128. This is the maximum virtual memory for data caching memory from local above the 2G bar pool.

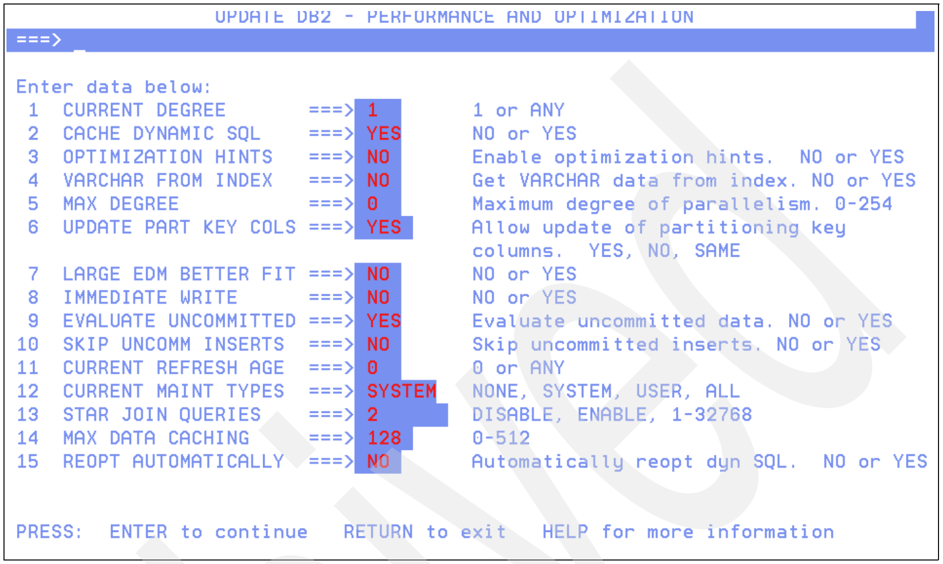
When you start your CLIST installation panels for DB2 9 for z/OS, you can find the parameter in OPTION 18 (Performance and Optimization); see Figure 4-12.



```
                  UPDATE DB2 - SELECTION MENU
===> _

 Select one of the following:
  1  DATA PARAMETERS                        17  APPLICATION PROGRAMMING DEFAULTS 2
  2  DATA PARAMETERS - PANEL 2              18  PERFORMANCE AND OPTIMIZATION
  3  DEFINE GROUP OR MEMBER                 19  IRLM PANEL 1
  4  SYSTEM RESOURCE DATA SET NAMES         20  IRLM PANEL 2
  5  DATA SET NAMES PANEL 1                 21  DB2 UTILITIES PARAMETERS
  6  DATA SET NAMES PANEL 2 *               22  PROTECTION
  7  DATA SET NAMES PANEL 3 *               23  MVS PARMLIB UPDATES
  8  SIZES                                  24  ACTIVE LOG DATA SET PARAMETERS
  9  SIZES PANEL 2                          25  ARCHIVE LOG DATA SET PARAMETERS
 10  WORK FILE DATABASE                     26  DATABASES TO START AUTOMATICALLY
 11  THREAD MANAGEMENT                      27  DISTRIBUTED DATA FACILITY
 12  BUFFER POOL SIZES PANEL 1              28  DISTRIBUTED DATA FACILITY PANEL  2
 13  BUFFER POOL SIZES PANEL 2 *            29  ROUTINE PARAMETERS
 14  TRACING AND CHECKPOINT PARAMETERS      30  DATA DEFINITION CONTROL SUPPORT
 15  OPERATOR FUNCTIONS                     31  JOB EDITING
 16  APPLICATION PROGRAMMING DEFAULTS 1

* None of the fields on these panels can be updated.
 PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 4-12   DB2 Selection Menu showing Option 18 (Performance and Optimization)

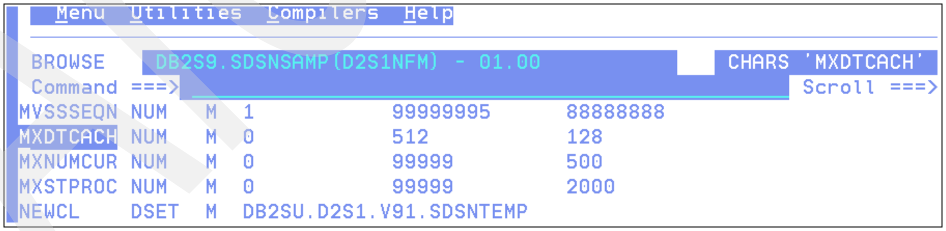
When you select Performance and Optimization, the parameter on the panel is number 14 MAX DATA CACHING (see Figure 4-13). As you can see, we have set it for 128 MB on the system that we are using to test.



```
                     UPDATE DB2 - PERFORMANCE AND OPTIMIZATION
===> _

Enter data below:
 1  CURRENT DEGREE       ===> 1        1 or ANY
 2  CACHE DYNAMIC SQL    ===> YES      NO or YES
 3  OPTIMIZATION HINTS   ===> NO       Enable optimization hints.  NO or YES
 4  VARCHAR FROM INDEX   ===> NO       Get VARCHAR data from index. NO or YES
 5  MAX DEGREE           ===> 0        Maximum degree of parallelism. 0-254
 6  UPDATE PART KEY COLS ===> YES      Allow update of partitioning key
                                       columns.  YES, NO, SAME
 7  LARGE EDM BETTER FIT ===> NO       NO or YES
 8  IMMEDIATE WRITE      ===> NO       NO or YES
 9  EVALUATE UNCOMMITTED ===> YES      Evaluate uncommitted data. NO or YES
10  SKIP UNCOMM INSERTS  ===> NO       Skip uncommitted inserts. NO or YES
11  CURRENT REFRESH AGE  ===> 0        0 or ANY
12  CURRENT MAINT TYPES  ===> SYSTEM   NONE, SYSTEM, USER, ALL
13  STAR JOIN QUERIES    ===> 2        DISABLE, ENABLE, 1-32768
14  MAX DATA CACHING     ===> 128      0-512
15  REOPT AUTOMATICALLY  ===> NO       Automatically reopt dyn SQL.  NO or YES


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

Figure 4-13 Max Data Caching range and its setting on our system

After you complete the updates or installation, the DSNTIXXX member is updated in the “yourname.SDSNSAMP” library member. XXX stands for the characters that you want the member name to be. We created a member called D2S1NFM and copied DSNTIXXX into it. In this member, the parameter MXDTCACH is set to 128 MB. It has a minimum value of 0 and a maximum value of 512.



```
  Menu  Utilities  Compilers  Help

 BROWSE    DB2S9.SDSNSAMP(D2S1NFM) - 01.00             CHARS 'MXDTCACH'
 Command ===>                                                 Scroll ===>
MVSSSEQN NUM    M   1          99999995      88888888
MXDTCACH NUM    M   0          512           128
MXNUMCUR NUM    M   0          99999         500
MXSTPROC NUM    M   0          99999         2000
NEWCL    DSET   M   DB2SU.D2S1.V91.SDSNTEMP
```

Figure 4-14 Using D2S1NFH

Therefore, when your data becomes cached, the required memory needed is:

```
(number of rows) x ((maximum length of the keys) + total maximum length
of all the relevant columns))
```

Consider this example:

```
20 rows x (80 for keys + 8000 for relevant columns)
```

That is:

```
20 x 8080 = 161,600 bytes which would be in memory
```

Therefore, I/O is reduced, and a sparse index does not need to be created:

```
sparse index = key + rid
```

5

Reporting statistical data

In this chapter, we introduce some of the new features of evaluating statistics and their impact on the SAP database. In general, statistical data is often used to improve the performance of the SAP database. When the waiting time of running periodic transactions or batch jobs increases, it is often a first sign of performance deterioration.

You must run the RUNSTATS utility to provide the optimizer with the facts. However, running this utility costs a lot in CPU resource. One method to avoid unnecessary RUNSTATS is through the stored procedure DSNACCOR, which uses real-time statistics (RTS).

The DSNACCOR stored procedure queries the DB2 RTS tables and makes recommendations to help you maintain your DB2 databases. DSNACCOR is fully integrated into the SAP Computing Center Management System (CCMS). It is used to determine the DB2 objects for which DB2 Utilities should be run.

The Database Planning Calendar, which is part of CCMS (SAP transaction DB13), exploits the recommendations from DSNACCOR and allows you to schedule the adequate utility according to the recommendations. For example, you can use RUSTATS for the objects that have changed considerably. This process can be completely automated with DSNACOR as a cornerstone.

In particular, DSNACCOR performs the following actions:

- Recommends when to reorganize, image copy, or update statistics for table spaces or index spaces
- Indicates table spaces or index spaces that have exceeded their data set
- Indicates whether objects are in a restricted state

DSNACCOR uses data from the SYSIBM.TABLESPACESTATS and SYSIBM.INDEXSPACESTATS RTS tables, which reside in DSNRTSDB.DSNRTSTS to make its recommendations.

With DB2 9 for z/OS, the collection process of useful information has changed.

In this chapter, we discuss the following topics:

- The new DB2 catalog tables SYSIBM.SYSTABLESPACESTATS and SYSIBM.SYSINDEXSPACESTATS residing in DSDB06.SYSRTSTS
- Eliminating unused indexes
- The DB2 Optimization Service Center and the Optimization Expert for z/OS

5.1 New RTS tables

The default time interval to update the statistic tables is 30 minutes. In a data sharing environment, each member has its own interval for writing RTS and can be set by modifying system parameter STATSINT in DSNZPARM.

In DB2 9 for z/OS, the new tables are:

- SYSIBM.SYSTABLESPACESTATS
- SYSIBM.SYSINDEXSPACESTATS

They are created in DSNDB06.SYSRTSTS, but data is moved during the migration process from DB2 8 to DB2 9.

The “old” RTS tables in DSNRTSDB.DSNRTSTS, which are queried by DSNACCOR, are:

- SYSIBM.TABLESPACESTATS
- SYSIBM.INDEXSPACESTATS

When compared to the new tables, most of the columns have kept their order and attributes. Only the new columns (shown in Table 5-1) have been added.

Table 5-1 New columns

Table	Column	Description
Both new tables	IBMREQD	Release dependency indicator
Both new tables	INSTANCE	Indicates whether the object is cloned
SYSTABLESPACESTAT S	DATASIZE	Number of bytes in data rows
SYSTABLESPACESTAT S	UNCOMPRESSED-DATASIZE	Number of bytes if data was not compressed
SYSINDEXSPACESTAT S	LASTUSED	Date when the index is used for SELECT, FETCH, searched UPDATE, and searched DELETE

The switch that is used to determine whether the tables are DB2 8 or DB2 9 for z/OS can be determined by using a query of SYSIBM.SYSTABLESPACE. The IBMREQD column for SYSRTSTS contains an “M” (indicating DB2 9), as shown in Example 5-1.

Example 5-1 Monitoring the use of the RTS tables

```
SELECT IBMREQD FROM SYSIBM.SYSTABLESPACE
 WHERE NAME = 'SYSRTSTS'
```

The results are shown in Example 5-2.

Example 5-2 Results of the query

```
---------+---------+---------+---------+---------+
IBMREQD
---------+---------+---------+---------+---------+
M
```

The time interval for RTS data to be written into statistics tables (externalize the in-memory statistics) is still being set by DSNZPARM STATSINT. Statistics are also externalized at other events, for example, at the beginning of a COPY or RUNSTATS job. All data sharing members externalize their statistics to the RTS tables and reset their in-memory statistics. Stopping the table space or index causes DB2 to write the in-memory statistics to the RTS tables and initialize the in-memory counters.

5.2 Finding unused indexes

Administration of unused indexes influences the performance of the applications in a negative way. Therefore only indexes that are really used by the optimizer should exist in the DB2 system. Unused indexes should be dropped. As we know, an SAP database has an average number of indexes between 30 and 50,000. Unfortunately, up to and including DB2 8, there is no history of index usage, so unused indexes cannot be identified.

In the DB2 catalog, the use of indexes is shown for static SQL only. If an application uses solely dynamic SQL, such as SAP, there is no possibility that a special index is used by the optimizer at all. The exception to this is permanent tracing, which is not practical to monitor. Even specialized monitoring tools do not distinguish between index maintenance operations at one side and usage for access path on the other. Each usage of the index (including index maintenance triggered by insert or delete operations of rows) is shown, but you cannot determine the actual usage of the index by the optimizer.

SAP applications are shipped with a standard set of indexes for the tables. It is crucial not to delete any of these indexes. There may be cases or usage types in which some of these indexes have not been used yet. However, this may change at any time, for example when an SAP support package is applied.

There may be cases in which additional user-defined indexes have been created that are not so beneficial for the query workload of a table. To save space and optimize the performance of INSERT, DELETE and UPDATE statements, these indexes can be deleted.

To identify those that have never been used or that are only sporadically used, DB2 9 for z/OS introduces an addition to the RTS table SYSIBM.SYSINDEXSPACESTATS, which indicates the last time that an index was used to process an SQL statement. This capability allows you to revisit indexes that you have created in addition to the standard indexes created by SAP. If it turns out that an index has never been used or not used for a long time, you may discard it.

Attention: Under no circumstances should you ever drop an SAP index, even if SYSIBM.SYSINDEXSPACESTATS shows that the index has never been used before. A future SAP support package may rely on it.

5.3 The DB2 Optimization Service Center and Optimization Expert for z/OS

The Optimization Service Center (OSC) for DB2 for z/OS is a workstation tool that runs with a GUI on the PC and uses DB2 Connect to access the manageable databases. The OSC is offered as part of the DB2 Accessories Suite for z/OS (product number 5655-R14), an orderable, no-charge feature of DB2 9 for z/OS. The OSC is also available for free on the Web. You can download it from:

https://www14.software.ibm.com/webapp/iwm/web/reg/download.do?source=swg-db2zosc&S_PKG=dl&lang=en_US&cp=UTF-8#

The OSC provides a set of capabilities to improve the performance of individual DB2 SQL queries and entire WQL workloads that run on your DB2 for z/OS subsystem. You can use the OSC to identify and analyze problem SQL statements and receive expert advice about statistics that you might gather to improve the performance of an individual statement.

The OSC for DB2 for z/OS can help you to determine the kind of statistics that should be collected for your objects based on the actual workload. It also allows you to analyze the access path of SQL statements. For example, viewing the statement is transformed by the DB2 optimizer.

In the SAP context, the OSC feature to base the statistics recommendation on a snap of the statement cache statistics is particularly useful. From this, you learn the exact types of statistics that should be collected for your objects based on your real-life production workload. The OSC statistics recommendations, such as, “which statistics should be collected?”, complement the statistics recommendations given by the stored procedure DSNACCOR, which answers the question, “when is it worth collecting new statistics?”.

For individual queries, the OSC can:

- Annotate the query
- Draw an access plan graph
- Generate query reports
- Facilitate generation of optimization hints
- Suggest statistics needed by the SQL optimizer

For SQL workloads, the OSC can gather workloads from various sources such as the dynamic statement cache, catalog tables, files and directories, QMF™ tables, and profile monitoring.

The DB2 Optimization Expert runs as a GUI as well. Along with all of the features of the OSC, it offers a comprehensive set of index and statistics advisors that you can use to improve system performance and lower your total cost of ownership. The features provided by the DB2 Optimization Expert are:

- Advising facilities for individual SQL statements and SQL workloads
- Tools to facilitate performance tuning on individual SQL statements
- Tools for SQL workload management
- Tools for light-weight, policy-based monitoring provided in DB2 9 for z/OS

The DB2 Optimization Expert for z/OS is a priced feature that is ordered separately.

Note: DB2 9 for z/OS must be at the most current level with the most current fix packs applied for OSC and Optimization Expert to perform optimally.

5.3.1 Configuration of the DB2 OSC and Optimization Expert

In this section, we discuss the configuration of the DB2 Optimization Expert. DB2 OSC configures in a similar fashion. To enable Optimization Expert, you must first complete the configuration of the OSC. A JCL job named DSNTIJOS is provided in the sample library DSN910.SDSNSAMP in DB2 9 for z/OS that is used to:

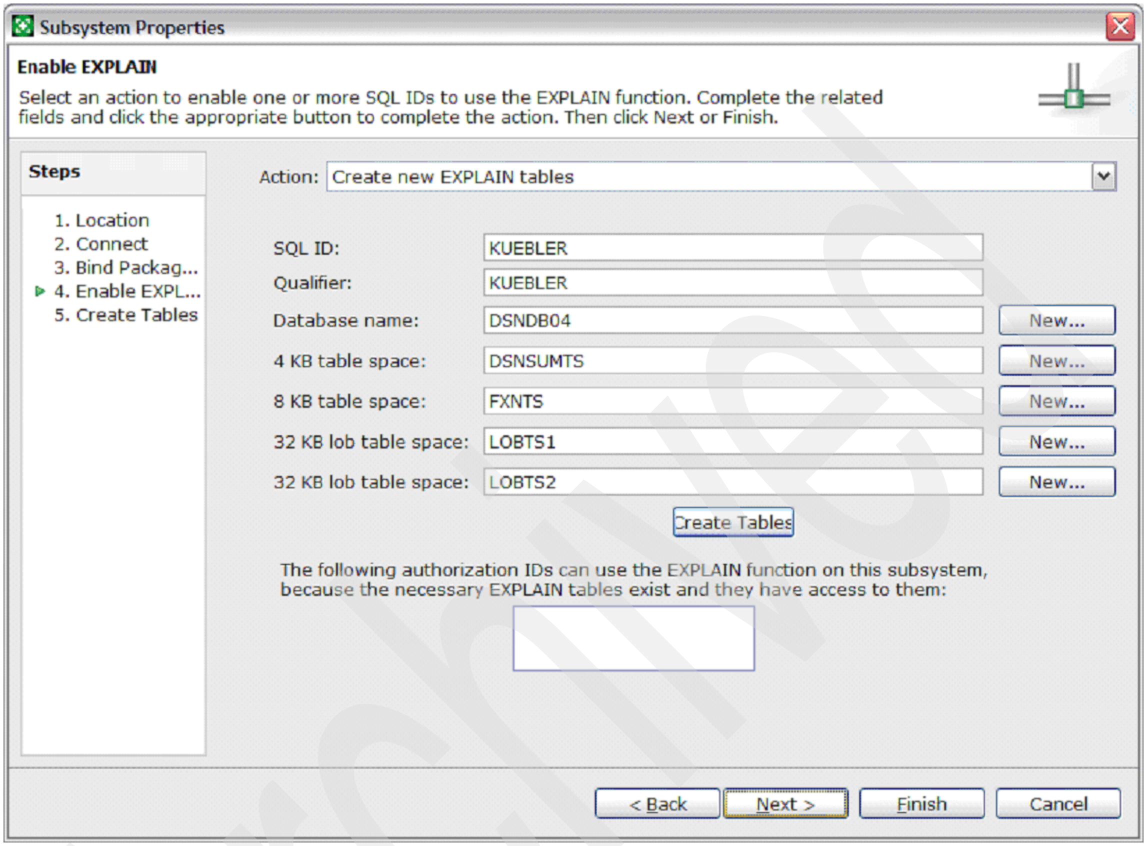
- Create the OSC tables
- Bind packages
- Install and configure stored procedures

Most of the configuration tasks can be done through the Optimization Expert client as well, except the installation and configuration of the stored procedures, which is related to server-side workload related task scheduling. You can configure the Optimization Expert by using the Optimization Expert client or a JCL job, named AOCDDL. AOCDDL is included in the Optimization Expert packaging and is used to create Optimization Expert tables and bind packages.

After you install the code, you configure the databases, including:

- Connecting to the database
- Binding the necessary packages
- Installing the tables for the advisors and the tools
- Enabling the user for using the OSC functions
- Managing users

Figure 5-1 shows the configuration steps on the left side of the Subsystem Properties window.



Subsystem Properties

Enable EXPLAIN

Select an action to enable one or more SQL IDs to use the EXPLAIN function. Complete the related fields and click the appropriate button to complete the action. Then click Next or Finish.

Steps

1. Location
2. Connect
3. Bind Packag...
4. Enable EXPL...
5. Create Tables

Action: Create new EXPLAIN tables

SQL ID: KUEBLER

Qualifier: KUEBLER

Database name: DSNDB04

4 KB table space: DSNSUMTS

8 KB table space: FXNTS

32 KB lob table space: LOBTS1

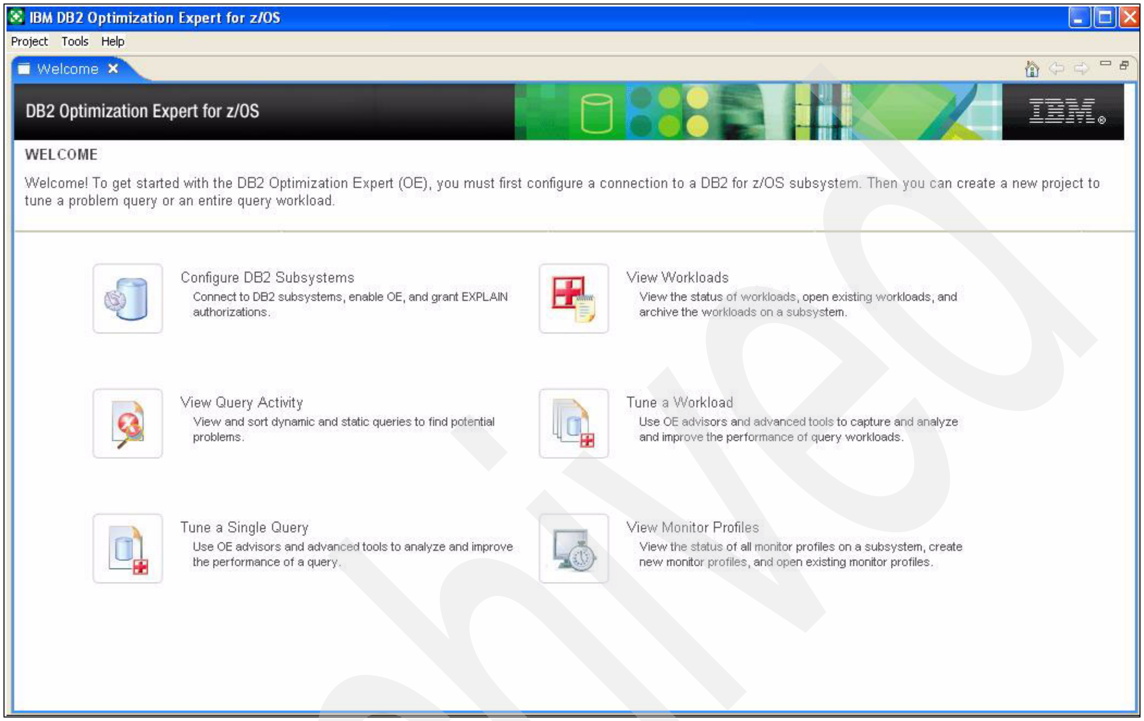
32 KB lob table space: LOBTS2

The following authorization IDs can use the EXPLAIN function on this subsystem, because the necessary EXPLAIN tables exist and they have access to them:

Figure 5-1 The DB2 Optimization Expert and OSC configuration steps

5.3.2 The available functions

Figure 5-2 shows an overview of the available functions.


IBM DB2 Optimization Expert for z/OS
Project Tools Help
Welcome
DB2 Optimization Expert for z/OS
IBM

WELCOME

Welcome! To get started with the DB2 Optimization Expert (OE), you must first configure a connection to a DB2 for z/OS subsystem. Then you can create a new project to tune a problem query or an entire query workload.

Configure DB2 Subsystems
Connect to DB2 subsystems, enable OE, and grant EXPLAIN authorizations.

View Workloads
View the status of workloads, open existing workloads, and archive the workloads on a subsystem.

View Query Activity
View and sort dynamic and static queries to find potential problems.

Tune a Workload
Use OE advisors and advanced tools to capture and analyze and improve the performance of query workloads.

Tune a Single Query
Use OE advisors and advanced tools to analyze and improve the performance of a query.

View Monitor Profiles
View the status of all monitor profiles on a subsystem, create new monitor profiles, and open existing monitor profiles.

Figure 5-2 Optimization Expert function overview

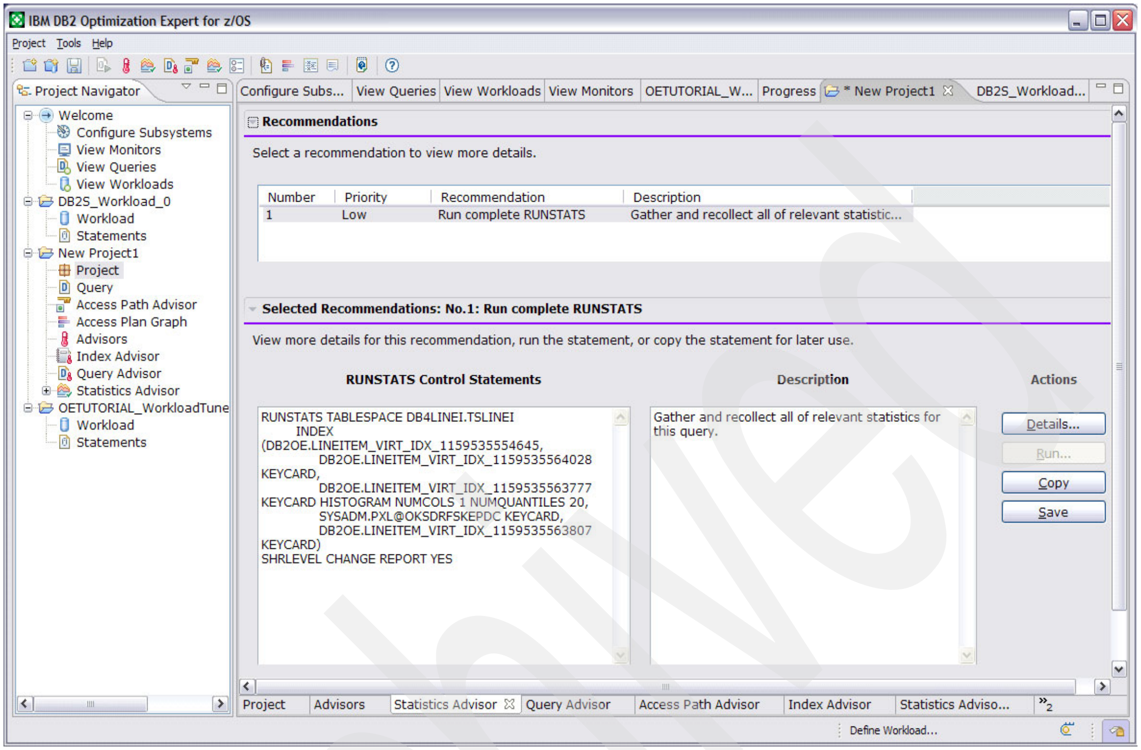
5.3.3 The Advisors

The DB2 Optimization Expert for z/OS includes routines that provide advice for:

- Statistics (RUNSTATS)
- Queries
- Access paths
- Indexes

Additionally, using the Optimization Expert allows you to simulate indexes in order to find the optimized access path to the database content.

Figure 5-3 shows an example for the Statistics Advisor.



Recommendations

Select a recommendation to view more details.

Number	Priority	Recommendation	Description
1	Low	Run complete RUNSTATS	Gather and recollect all of relevant statistic...

Selected Recommendations: No.1: Run complete RUNSTATS

View more details for this recommendation, run the statement, or copy the statement for later use.

RUNSTATS Control Statements

```
RUNSTATS TABLESPACE DB4LINEI.TSLINEI
      INDEX
(DB2OE.LINEITEM_VIRT_IDX_1159535554645,
          DB2OE.LINEITEM_VIRT_IDX_1159535564028
KEYCARD,
          DB2OE.LINEITEM_VIRT_IDX_1159535563777
KEYCARD HISTOGRAM NUMCOLS 1 NUMQUANTILES 20,
          SYSADM.PXL@OKSDRFSKEPDC KEYCARD,
          DB2OE.LINEITEM_VIRT_IDX_1159535563807
KEYCARD)
SHRLEVEL CHANGE REPORT YES
```

Description

Gather and recollect all of relevant statistics for this query.

Actions

Figure 5-3 The Statistics Advisor

In this example, the Statistics Advisor recommends that we run the RUNSTATS utility; it also provided the necessary RUNSTATS control statement.

6

Security

In this chapter, we discuss database roles and trusted security context within DB2 9. We also discuss the support of Secure Sockets Layer (SSL) network encryption and IPv6.

6.1 Database roles and trusted security context

DB2 has added new features in DB2 9 for z/OS in the area of user ID authentication. These features are database role and trusted security context.

Trusted security context is a new concept that enables customers to limit the servers that are allowed to connect to DB2 to the set of SAP application servers. User IDs and password are only valid when used on these application servers. If a system is set up like this, a stolen user ID and password do not allow third parties to connect to DB2, unless they also manage to log on to one of the authorized application servers.

Database roles is a feature that allows customers to strictly control the activities of DBAs. For example, to accomplish a task, a role with the necessary privileges can be assigned to a database administrator (DBA) for a certain amount of time. Afterward the role is revoked again. During this time interval, the activities of the DBA can be audited. This approach allows customers to detect situations in which a DBA explores sensitive data.

Network trusted context is an object that gives users a specific set of privileges. It is available when the user connects to the database through a trusted connection. An example of this is in a Web application environment. The application receives a trusted connection through the trusted context. Then the user can have their identity passed to the database server. This trusted context then allows you to create a unique set of interactions between DB2 and your application.

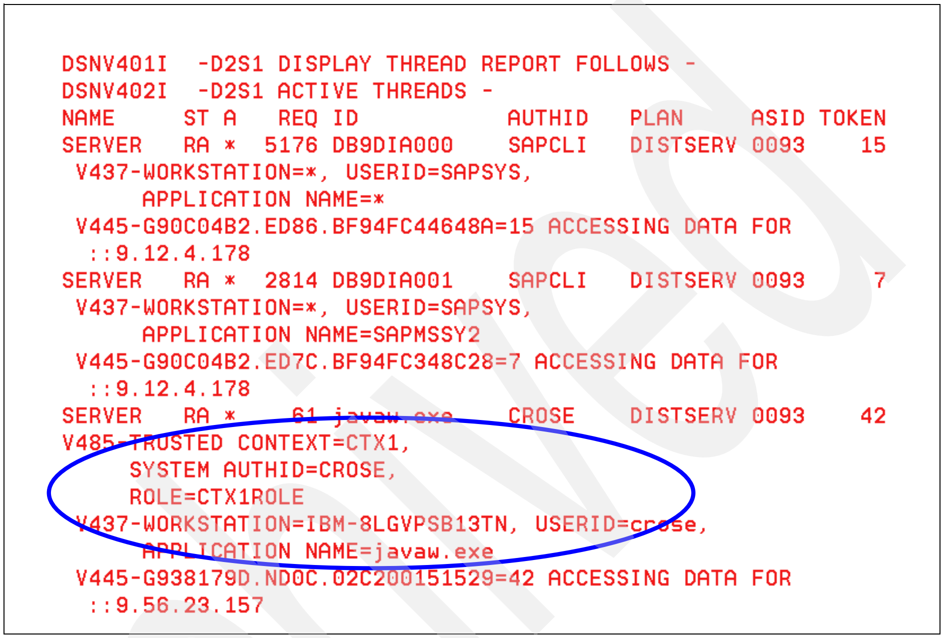
A *role* is a database entity that groups together one or more privileges. It can be assigned to users or objects, if objects are created in a trusted context with the role defined as the owner, by specifying the ROLE AS OBJECT OWNER clause in the trusted context definition. Databases, table spaces, tables, indexes, and views can be implemented in a trusted context with a role as the owner of the created objects. A role is available within a trusted context. You can define a role for a trusted context and assign it to authorization IDs. When associated with a role, and using the trusted connection, an authorization ID inherits all the privileges that the role grants.

In general, the concept of *trusted context* allows multi-tier applications to connect to DB2 and switch the authorization ID without having to re-logon or provide a password. However, the authorization IDs to which the connection switches must exist in RACF®. Trusted context does not apply to SAP because there are usually many SAP user IDs without corresponding RACF user IDs.

To view the trusted context, you can see the threads from DB2 on z/OS by using the following command:

```
-DIS THD(*)
```

Figure 6-1 shows an example.



```
DSNV401I  -D2S1 DISPLAY THREAD REPORT FOLLOWS -
DSNV402I  -D2S1 ACTIVE THREADS -
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
SERVER   RA *  5176 DB9DIA000    SAPCLI   DISTSERV 0093    15
 V437-WORKSTATION=*, USERID=SAPSYS,
      APPLICATION NAME=*
 V445-G90C04B2.ED86.BF94FC44648A=15 ACCESSING DATA FOR
  ::9.12.4.178
SERVER   RA *  2814 DB9DIA001    SAPCLI   DISTSERV 0093     7
 V437-WORKSTATION=*, USERID=SAPSYS,
      APPLICATION NAME=SAPMSSY2
 V445-G90C04B2.ED7C.BF94FC348C28=7 ACCESSING DATA FOR
  ::9.12.4.178
SERVER   RA *    61 javaw.exe    CROSE    DISTSERV 0093    42
V485-TRUSTED CONTEXT=CTX1,
     SYSTEM AUTHID=CROSE,
     ROLE=CTX1ROLE
 V437-WORKSTATION=IBM-8LGVPSB13TN, USERID=crose,
      APPLICATION NAME=javaw.exe
 V445-G938179D.ND0C.02C200151529=42 ACCESSING DATA FOR
  ::9.56.23.157
```

Figure 6-1   Results of the Display Thread command from the server

There are many features of the network trusted context. You can find more information in Chapters 4 and 5 in the *DB2 Version 9.1 for z/OS Administration Guide*, SC18-9840.

6.2 Support of SSL Network Encryption

DB2 has a new Distributed Relational Database Architecture (DRDA) secure port, which allows use of SSL authentication methods. SSL authentication methods support secure communications between a client and DB2.

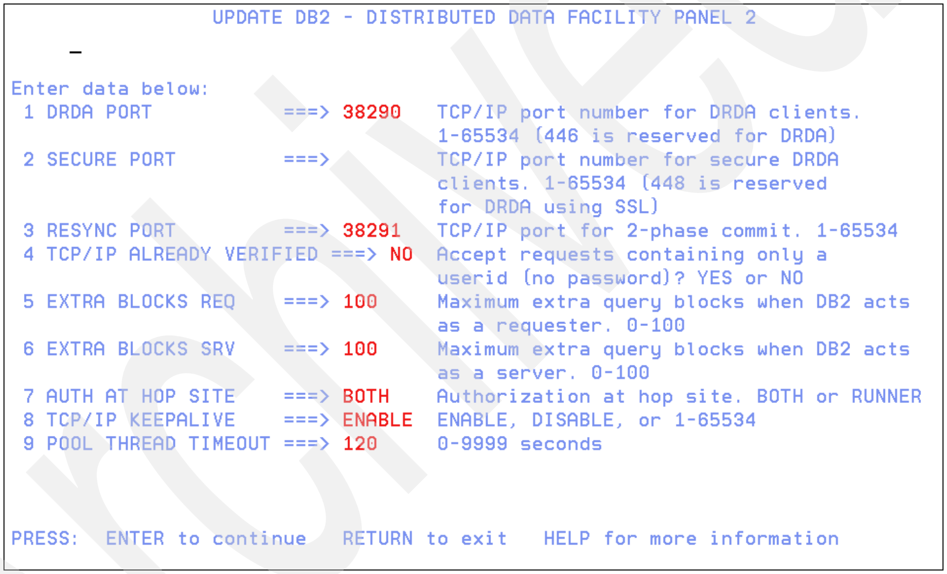
To implement SSL support for a DB2 server, the TCP/IP SQL Listener service task of the distributed data facility (DDF) must be capable of listening to a secondary secure port for inbound SSL connections. The TCP/IP Listener accepts regular (non-SSL) connections on the DRDA port. The secure port accepts only SSL connections to provide secure communications with a partner.

Client connections are assured of getting the SSL protocol connection that they require. The system administrator specifies the secure port number.

The secure port is used only to accept secure connections that use the SSL protocol. In order to define a secure port, you can do so either in the installation panel or via the communication records of the DB2 bootstrap data set (BSDS). This is done through the supplied DSNJU003 utility, which is normally stored in SDSNSAMP PDS.

SSL encryption allows you to encrypt the data stream between the SAP application server and DB2. The main challenge here is to keep the encryption overhead at an acceptable level.

Figure 6-2 shows an example of configuring the DDF on DB2 9 for z/OS.



```
                 UPDATE DB2 - DISTRIBUTED DATA FACILITY PANEL 2

     _

Enter data below:
 1 DRDA PORT            ===> 38290   TCP/IP port number for DRDA clients.
                                     1-65534 (446 is reserved for DRDA)
 2 SECURE PORT          ===>         TCP/IP port number for secure DRDA
                                     clients. 1-65534 (448 is reserved
                                     for DRDA using SSL)
 3 RESYNC PORT          ===> 38291   TCP/IP port for 2-phase commit. 1-65534
 4 TCP/IP ALREADY VERIFIED ===> NO   Accept requests containing only a
                                     userid (no password)? YES or NO
 5 EXTRA BLOCKS REQ     ===> 100     Maximum extra query blocks when DB2 acts
                                     as a requester. 0-100
 6 EXTRA BLOCKS SRV     ===> 100     Maximum extra query blocks when DB2 acts
                                     as a server. 0-100
 7 AUTH AT HOP SITE     ===> BOTH    Authorization at hop site. BOTH or RUNNER
 8 TCP/IP KEEPALIVE     ===> ENABLE  ENABLE, DISABLE, or 1-65534
 9 POOL THREAD TIMEOUT ===> 120      0-9999 seconds



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

Figure 6-2 The DDF definitions

The DB2 JDBC Universal Driver (JCC) supports SSL encryption of the DRDA data stream from the client side. For SSL encryption, it requires the DataSource approach as a means to specify the connection to DB2 and not the DriverManager approach, which is issued by SAP. When SAP supports the DataSource approach, SSL encryption can be employed with SAP Java. DB2 on z/OS uses the ICSF (Integrated Cryptographic co-processors) on the IBM System z9™ platform for data encryption and decryption (done by the hardware). When data is encrypted on the z/OS side, performance is not impacted, but rather the client side might feel a much larger impact.

7

Optimized recovery

Maintaining SAP databases means that you must manage the consistency of the whole database system. With DB2 9 for z/OS, many new features are available. In this chapter, we discuss some of them such as:

- Volume-based utility enhancements
- MODIFY recovery
- Point-in-time recovery

Some of these new features are linked to the use of DFSMShsm 1.8.

7.1 Volume-based utility enhancements

Within SAP, the complete DB2 database is considered as one entity. To recover this entity in case of logical failures (and even in some cases of physical damage), the volume-based utilities BACKUP SYSTEM and RESTORE SYSTEM are well-suited for this purpose. These utilities exploit the fast replication services of DFSMShsm.

The following requirements must be met in advance to exploit the BACKUP SYSTEM utility:

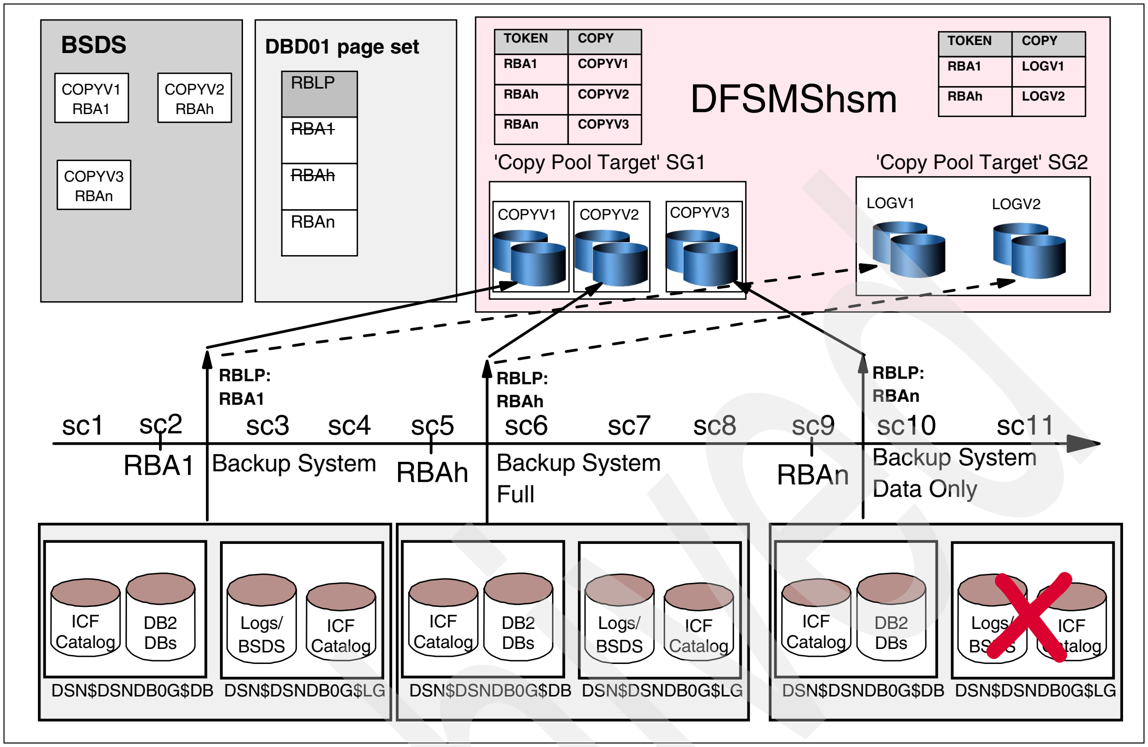
- The data sets that you want to copy are SMS-managed data sets.
- You are running z/OS V1R8 or later.
- You have disk control units that support IBM TotalStorage Enterprise Storage Server (ESS) FlashCopy.
- You have defined a copy pool for your database data that contains all of the table spaces and indexes. If you plan to also copy the logs, define another copy pool for your logs that contains the bootstrap data set (BSDS) and log data sets. Use the DB2 naming convention for both of these copy pools.
- The ICF catalog for the data is on a separate volume than the ICF catalog for the logs.
- You have defined a system managed space (SMS) backup storage group for each storage group in the copy pools.

To take advantage of the enhancements in the area of backup and recovery, you must have z/OS V1.8 or later.

BACKUP SYSTEM can run concurrently with any other utility. However, it must wait for the DB2 events that follow to complete before the copy can begin.

In DB2 V8, dumping to disk is supported only by the database system. Specific information concerning the backup process is held in the BSDS.

BACKUP SYSTEM FULL copies the entire DB2 subsystem. With this option, all data sets that reside on the disk in database copy pool DSN$DSNDB0G$DB and log copy pool DSN$DSNDB0G$LG are copied to backup copy pool SG1 and copy pool SG2. With the DATA ONLY option, only volumes of the database copy pool are copied to the backup pool SG1 (see Figure 7-1).



TOKEN	COPY
RBA1	COPYV1
RBAh	COPYV2
RBAn	COPYV3

TOKEN	COPY
RBA1	LOGV1
RBAh	LOGV2

Figure 7-1 Backup system

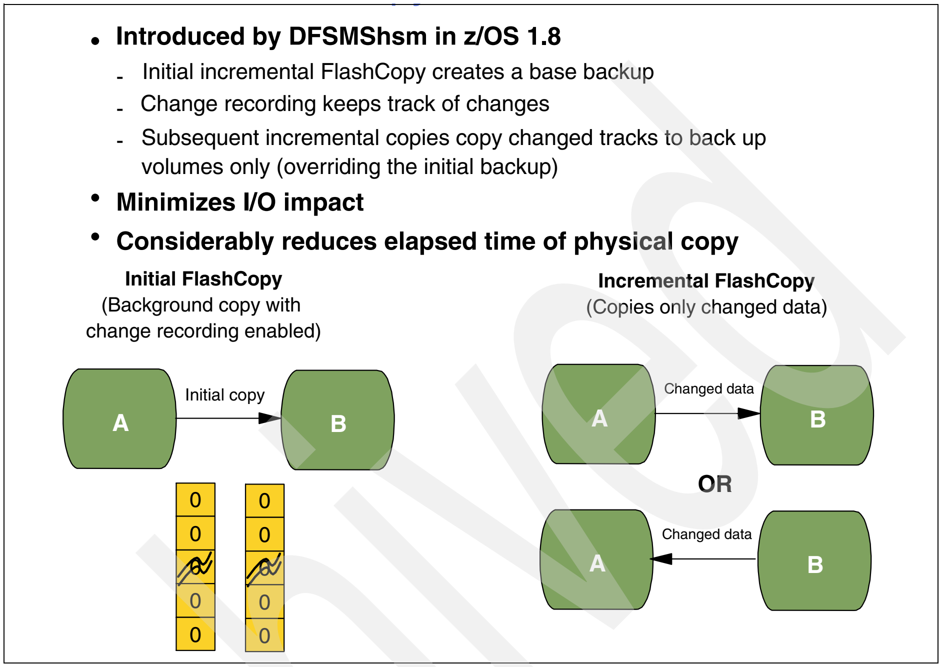
7.1.1 Incremental backup

DFSMShsm creates full FlashCopy volume backups on behalf of the BACKUP SYSTEM utility for the database, log copy pools, or both. It accomplishes this by establishing a relationship between each source volume and a corresponding target volume (backup volume).

The copy is considered successful after the relationship is established. Then a full copy of each source volume is started in the background. Applications can access all of the data while the background copy is being done. However, this can put a strain on performance since the contents of all source volumes are copied to the corresponding target volumes.

Note: The physical background copy of FlashCopy is performed with a priority that is lower than the application priority for I/O.

DFSMShsm introduces incremental FlashCopy support with development APAR OA17314 in z/OS 1.8. Figure 7-2 shows an overview of the highlights of incremental FlashCopy.



- **Introduced by DFSMShsm in z/OS 1.8**
  - Initial incremental FlashCopy creates a base backup
  - Change recording keeps track of changes
  - Subsequent incremental copies copy changed tracks to back up volumes only (overriding the initial backup)
- **Minimizes I/O impact**
- **Considerably reduces elapsed time of physical copy**

Initial FlashCopy
(Background copy with change recording enabled)

Incremental FlashCopy
(Copies only changed data)

Figure 7-2 The incremental FlashCopy

The incremental FlashCopy hardware feature reduces the amount of data that must be processed in the background, because only changed tracks are copied. This results in a complete copy of the source volume, but with potentially less I/O.

Support has been added to the BACKUP SYSTEM utility to exploit the incremental FlashCopy feature for the DB2 database copy pool with development APAR PK41001. The DB2 log copy pool will always have a full FlashCopy taken.

The persistent incremental FlashCopy relationship is established the first time the ESTABLISH FCINCREMENTAL keywords are specified. The relationship exists until it is withdrawn. Use the ESTABLISH FCINCREMENTAL keywords once to establish the incremental FlashCopy relationship. Most customers are expected to keep one DASD FlashCopy version of their database copy pool. The incremental FlashCopy relationship is created when you use the ESTABLISH FCINCREMENTAL keyword.

All of the tracks of each source volume are copied to their corresponding target volumes during the first copy. All subsequent BACKUP SYSTEM requests only have the changed tracks copied to the target volumes. Each source volume can have only one incremental relationship, so multiple incremental FlashCopy versions are not supported. Customers that keep more than one DASD FlashCopy version of the database copy pool must do full copies for versions other than the incremental version.

For example, a user may decide to keep two FlashCopy DASD versions of their database copy pool as a backup.

1. A BACKUP SYSTEM utility is done with the ESTABLISH FCINCREMENTAL keywords. A full copy of each volume is created since this is the first time a FlashCopy is done after the incremental relationship has been established.
2. The next day another BACKUP SYSTEM is done. This creates FlashCopy DASD version 2, which is a full copy since the incremental relationships are established with the target (backup) volumes in FlashCopy DASD version 1.
3. The next day BACKUP SYSTEM is run again, without the ESTABLISH FCINCREMENTAL keywords. The incremental version is the oldest version so it is used for the FlashCopy. This time only the tracks that have changed, since the last FlashCopy of the incremental version, are copied.

The result is a complete copy of the source volume.

The concept of the incremental FlashCopy is different from the DB2 incremental copy. The first incremental FlashCopy is actually a full copy since every track is copied. All subsequent incrementals copy the changed tracks to the target (backup) volume. Since the same target volume is used for each incremental copy, it ends up being a complete copy of the source volume. The previous incremental copy no longer exists, because the changed tracks from the source were copied to the target. In DB2 terms, the incremental FlashCopy is similar to a full image copy that is merged with an incremental image copy to create a new full image copy.

DFSMShsm allows multiple versions of FlashCopy for a copy pool. As previously stated, there can only be one incremental copy. If incremental copies using FlashCopy are taken, every backup is an overriding incremental backup. This is independent of the number of backups that are specified in DFSMShsm for the database copy pool.

In Example 7-1, we specify that a persistent incremental FlashCopy relationship is to be established (only) for source copy volumes in the database copy pool. Subsequent invocations of BACKUP SYSTEM (without these keywords) automatically process the persistent incremental FlashCopy relationship.

Example 7-1   Establish FCINCREMENTAL

```
//BCKICDB2 EXEC PGM=DSNUTILB,PARM='DB2S,BCKINC',REGION=0M
//SYSPRINT DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DSNTRACE DD SYSOUT=*
//SYSIN    DD *
  BACKUP SYSTEM FULL DATAONLY ESTABLISH FCINCREMENTAL
```

Example 7-2 specifies that the last incremental FlashCopy is to be taken and that the persistent incremental FlashCopy relationship is to be withdrawn for all of the volumes in the database copy pool. Use these keywords only if no further incremental FlashCopy backups of the database copy pool are desired.

Example 7-2   End FCINCREMENTAL

```
//BCKICDB2 EXEC PGM=DSNUTILB,PARM='DB2S,BCKINC',REGION=0M
//SYSPRINT DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DSNTRACE DD SYSOUT=*
//SYSIN    DD *
  BACKUP SYSTEM FULL DATAONLY END FCINCREMENTAL
```

7.1.2 Backing up to tape

Up to now, fast replication and tape processing did not correspond with each other. To strike a balance between fast and “cheap”, dumping to tape begins after the fast replication of the database copy pool and (if chosen) the log copy pool. Further, the second-level backup with dumping to tape must be prepared and administrated manually by the user. This can be done by using DFSMShsm commands without knowledge of DB2.

With DB2 9 for z/OS, the BACKUP SYSTEM utility now supports direct dumping to tape. The prerequisite for using these new functions with DB2 9 for z/OS is DFSMShsm 1.8.

Backup to tape is done from the backup copy pools, which consist of:

- Database copy pool
- Log copy pool

The backup is processed as soon as the relationships for the fast replication copy are successfully established.

The new options with DB2 9 for z/OS are:

- DUMP: Creates a fast replication to disk with a subsequent dump to tape
- DUMPONLY: Existing fast replication copy is written to tape
- DUMPCLASS: Using DFSMShsm dump class(< 6) for dump to tape

Use the DUMPONLY option (as shown in Example 7-3) to create dumps on tape volume or volumes of an existing fast replication copy of the database copy pool or the log copy pool on DASD that was created by an earlier invocation of the BACKUP SYSTEM utility. You can use the DUMPONLY option to resume processing for a dump that has failed. The unique token that identifies each system-level backup is reported in the PRINT LOG MAP (DSNJU004) utility output. It can be specified as a 36-digit hexadecimal byte string for the DUMPONLY option. If the TOKEN option is omitted, then the most recent system-level backup is dumped.

Example 7-3 DUMPONLY

```
//BCKDB2  EXEC PGM=DSNUTILB,PARM='DB2S,BSYSTEM',REGION=0M
//SYSPRINT DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DSNTRACE DD SYSOUT=*
//SYSIN    DD *
  BACKUP SYSTEM FULL DUMPONLY
```

The DUMPCLASS option (Example 7-4) can be specified for both DUMP and DUMPONLY to process particular DFSMShsm dump classes. If the DUMPCLASS option is not specified, then all of the default dump classes for the database or log copy pool are processed. If DUMP and DUMPONLY are omitted, then only the fast replication copy is created.

Example 7-4 DUMP with DUMPCLASS

```
//BCKDB2  EXEC PGM=DSNUTILB,PARM='DB2S,BSYSTEM',REGION=0M
//SYSPRINT DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DSNTRACE DD SYSOUT=*
//SYSIN    DD *
  BACKUP SYSTEM DATA ONLY DUMP DUMPCLASS(dumpClassName)
```

The BACKUP SYSTEM utility does not wait for dump processing to complete for DUMP or DUMPONLY processing. Therefore, the user must use the DFSMShsm LIST COPYPOOL command with the DUMPVOLS option (Example 7-5) to verify that the dump to tape was successful.

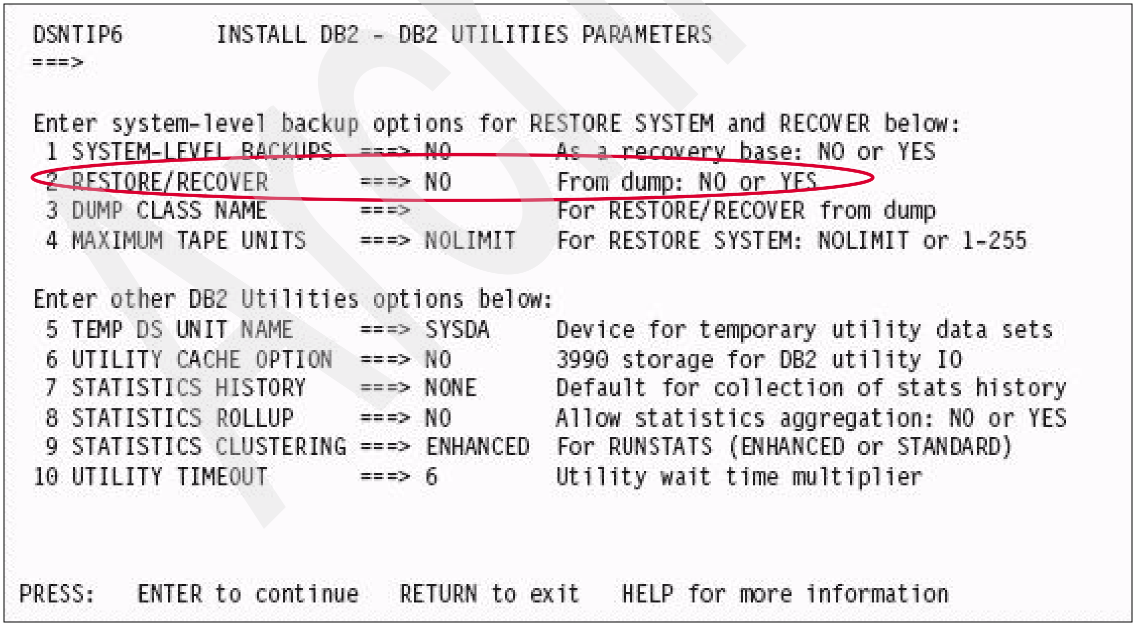
Example 7-5 Using the DUMPVOLS option

```
/F hsmsc04,LIST CP(DSN$DB2S$DB) DUMPVOLS
```

Note that the LIST COPYPOOL output contains the token that corresponds to each system-level backup. If the dump processing fails, then check the DFSMShsm Dump Activity Log for error messages. Invoke the BACKUP SYSTEM utility with the DUMPONLY option to resume dump processing.

7.1.3 Restoring from tape

The RESTORE SYSTEM utility uses the most recent system-level backup of the database copy pool that was taken prior to the SYSPITR log truncation point. If the most recent system-level backup resides on DASD, it is used for the restore. If the most recent system-level backup no longer resides on DASD, and is dumped to tape, then the dumped copy is used for the restore only if the FROMDUMP option is specified on the RESTORE utility statement or on the DSNTIP6 install panel (see Figure 7-3 for specifying “NO” for FROMDUMP).



```
 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 other DB2 Utilities options below:
  5 TEMP DS UNIT NAME     ===> SYSDA      Device for temporary utility data sets
  6 UTILITY CACHE OPTION  ===> NO         3990 storage for DB2 utility IO
  7 STATISTICS HISTORY    ===> NONE       Default for collection of stats history
  8 STATISTICS ROLLUP     ===> NO         Allow statistics aggregation: NO or YES
  9 STATISTICS CLUSTERING ===> ENHANCED   For RUNSTATS (ENHANCED or STANDARD)
 10 UTILITY TIMEOUT       ===> 6          Utility wait time multiplier



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

Figure 7-3 Indicating an option for FROMDUMP

If YES is specified for the RESTORE/RECOVER FROMDUMP install option on the DSNTIP6 installation panel, or the FROMDUMP option (Example 7-6) is specified on the RESTORE utility statement, then only dumps on tape of the database copy pool are used for the restore.

Example 7-6 FROMDUMP with DUMPCLASS

```
//RESTDB2 EXEC PGM=DSNUTILB,PARM='DB2S,RSYSTEM',REGION=0M
//SYSPRINT DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DSNTRACE DD SYSOUT=*
//SYSIN    DD *
  RESTORE SYSTEM FROMDUMP DUMPCLASS(dumpClassName) TAPEUNITS(2)
```

Specify the name of the DFSMShsm dump class that will be used by the RESTORE SYSTEM utility to restore the database copy pool from a system-level backup that has been dumped to tape. The setting is applicable only when you specify YES in DSNZPARM RESTORE_RECOVER_FROMDUMP. You can override the setting for DUMP CLASS NAME by running the RESTORE SYSTEM utility statement or the RECOVER utility statement with the DUMPCLASS keyword. For information about DFSMShsm dump classes, see *MVS/ESA Storage Management Library: Managing Storage Groups*, SC26-3125.

In addition, if a DUMP CLASS NAME is specified on the DSNTIP6 installation panel or the DUMPCLASS option is specified on the RESTORE utility statement, then the database copy pool is restored from that particular DFSMShsm dump class. If a DUMP CLASS NAME is not specified on the DSNTIP6 installation panel or the DUMPCLASS option is not specified on the RESTORE utility statement, then the RESTORE SYSTEM utility issues the DFSMShsm LIST COPYPOOL command and chooses to use the first dump class listed in the output.

When you use the TAPEUNITS keyword with RESTORE SYSTEM, it overrides the setting for RESTORE_TAPEUNITS in DSNZPARM. By using this parameter, the maximum number of tape units or tape drives that the RESTORE SYSTEM utility can allocate when restoring a system-level backup that has been dumped to tape is specified. The default value, NOLIMIT, means that the RESTORE SYSTEM utility allocates as many tape units as necessary to restore the system-level backup.

You specify whether the system-level backup for the RESTORE SYSTEM and the RECOVER utilities is from the disk copy of the system-level backup (NO) or from the dump on tape (YES). You can override the setting of RESTORE/RECOVER by executing the RESTORE SYSTEM utility statement or the RECOVER utility statement with the FROMDUMP keyword. See Example 7-8 for the DSNZPARM settings.

You can ascertain whether the system-level backups of your database copy pool still reside on DASD or have been dumped to tape. You do this by using the output from the DFSMShsm LIST COPYPOOL command with the DUMPVOLS option (see Example 7-7) in conjunction with the DB2 system-level backup information in the PRINT LOG MAP (DSNJU004) utility output. For a data sharing system, run the PRINT LOG MAP utility on each member.

Example 7-7 LIST copy pool

```
/F hsmsc04,LIST CP(DSN$DB2S$DB) DUMPVOLS
```

Restoration of each volume in the database copy pool from a fast replication copy on DASD occurs virtually instantaneously. DB2 does not wait for the background copy to complete before commencing the application of the log records. Restoration of the database copy pool from dumps on tape volume or volumes takes much longer.

7.1.4 Recovering individual objects from system-level backups

In DB2 9 for z/OS, the RECOVER utility has been enhanced to work with system-level backups. To enable the individual object-level recovery, specify whether the RECOVER utility should use system-level backups as a recovery base (in addition to image copies and concurrent copies) in the DSNZPARM DSN6SPRM section.

Specify YES if you take system-level backups with the BACKUP SYSTEM utility.

Example 7-8 DSNZPARM settings

```
SYSTEM_LEVEL_BACKUPS=YES,
RESTORE_RECOVER_FROMDUMP=YES,
RESTORE_TAPEUNITS=NOLIMIT,
UTILS_DUMP_CLASS_NAME=,
```

The dump class is used by the RECOVER utility to restore objects from a system-level backup that has been dumped to tape.

Prerequisite: A prerequisite for using this option is DFSMShsm Version 1.8 or later if you have enabled system-level backups. If you do not have this prerequisite, you receive the error message:

```
DSNU1608I RECOVER SYSTEM UTILITY FAILED.  REQUIRED DFSMSHSM SERVICES
NOT AVAILABLE
```

7.1.5 Extending encryption

Currently, full volume dumps created by DFSMShsm are not encrypted. The data on the unprotected dump tapes is particularly vulnerable for customers who transport the tapes to off-site storage locations; the tapes could be misplaced or fall into the wrong hands. The BACKUP SYSTEM utility, depending upon which keywords are specified, can have DFSMShsm create dump tapes, which can be used by the RESTORE SYSTEM utility.

There are no keyword changes for BACKUP SYSTEM because the encryption parameters are specified in the DFSMShsm DUMPCLASS specification. It is possible for a customer to change the key label that was in effect when the volume was created, in which case, the correct RSA key label would have to be provided to the RESTORE system utility. As a result, a new keyword, RSA, which stands for the Rivest-Shamir-Adelman algorithm, is being added for this purpose. See Example 7-9.

Example 7-9 RESTORE SYSTEM with RSA

```
//RESTDB2 EXEC PGM=DSNUTILB,PARM='DB2S,RSYSTEM',REGION=0M
//SYSPRINT DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DSNTRACE DD SYSOUT=*
//SYSIN    DD *
  RESTORE SYSTEM TAPEUNITS 2 FROMDUMP
       DUMPCLASS(DB2STGD2)
       RSA
(&&#$$$66666666687GY56FHR645KBMNKI8UYT543KFHVBGRT65LKJGUY765D7124)
```

7.2 MODIFY RECOVERY

The MODIFY RECOVERY utility has been enhanced to allow SYSIBM.SYSLGRNX records to be deleted for a table space or index (with the COPY YES attribute) even if no SYSIBM.SYSCOPY records were deleted. This enables customers who use the BACKUP SYSTEM and RESTORE SYSTEM utilities to clean up their SYSIBM.SYSLGRNX records even though they do not take traditional image copies of their objects with the COPY utility. A new SYSIBM.SYSCOPY record with ICTYPE='M' and STYPE='R' is inserted by a MODIFY RECOVERY job so that the RECOVER utility can ensure that it has all of the recovery information that it needs to use system-level backups as the recovery base during object-level recoveries. MODIFY RECOVERY always inserts a SYSIBM.SYSCOPY row with ICTYPE=”M” and STYPE=”R” to record the RBA or LRSN (in case of data sharing) of the most recent SYSCOPY or SYSLGRNX record deleted.

A new keyword “RETAIN(n)” is added to the MODIFY RECOVERY utility control statement, which is used to determine the cleanup date. Only records with ICTYPE=F (full copy), ICBACKUP=BLANK (LOCALSITE primary copy), and DSNUM as stated for the specified table space are checked. Depending on the chosen option (LAST, LOGLIMIT, or GDGLIMIT), the specific records are retained in SYSIBM.SYSCOPY.

With respect to SAP databases, *never* use either GDGLIMIT as an option nor generation data set (GDS) or GDGs for creating image copies. Here are some disadvantages when using GDGs and SAP:

- The date and time of creating the image copy are not visible in the data set, but only in SYSIBM.SYSCOPY.
- There is no selection of specific GDGs when creating them on the same day as the recovery.
- Dynamic alteration of database and tablespace names results in higher administration to define a new GDG base.
- GDG entries in SYSCOPY are not usable or empty because of copy failures, such as due to a B37.

7.2.1 RETAIN LAST(n)

RETAIN LAST(n) specifies the number of recent records to retain in SYSIBM.SYSCOPY. RETAIN works with a date and not with a complete timestamp. This can result in more copies kept as specified by RETAIN. For example, if the most recent five copies were taken on the same day and “RETAIN LAST(2)” is specified, all five copies of this day remain in SYSIBM.SYSCOPY.

7.2.2 RETAIN LOGLIMIT

RETAIN with LOGLIMIT option queries the BSDS to determine the oldest archive log timestamp and deletes records in SYSIBM.SYSCOPY and SYSIBM.SYSLGRNX older than the timestamp. For data sharing, DB2 queries the BSDS of all data sharing members to determine the overall oldest log timestamp.

7.2.3 Reason to use MODIFY RECOVERY on a frequent basis

One of the main reasons to use the MODIFY RECOVERY utility on a weekly or daily basis is the large number of objects. Usually there are about 20,000 to 30,000 table spaces and 40,000 to 60,000 indexes in SAP databases. If only three backups remain in SYSIBM.SYSCOPY, the average count of image copies is about 100,000 data sets for only one SAP system.

Some of the objects only change a few times a year. One of the goals of DB2 administrators is to avoid unnecessary image copies and reduce CPU and space overhead. To face this demand, it is a good practice to obtain new image copies only when changes occur.

7.2.4 Conditional restart

SAP considers the whole DB2 system as one single point of consistency. In the case of recovering a logical error to a prior point in time, DB2 supports the process of getting consistency, especially during restart. This usually results in a conditional restart that is to be prepared by DSNJU003 (see Example 7-10) and is written as a Conditional Restart Control Record (CRCR) in the BSDS.

Example 7-10 Conditional restart with ENDRBA

```
//BSDSCHG EXEC PGM=DSNJU003,REGION=0M
//STEPLIB  DD DISP=SHR,DSN=DB8H8.SDSNLOAD
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD DISP=OLD,DSN=D2S1.BSDS01
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN    DD  *
CRESTART
CREATE,ENDRBA=007029900000
```

Remember that DB2 usually runs with identical bootstrap data sets (BSDS01 and BSDS02, in our case). Therefore, you must either add a SYSUT2 DD card in the CRESTART job or perform an additional IDCAMS REPRO after you complete the CRESTART without errors.

Log truncation by timestamp

In DB2 9 for z/OS, the DSNJU003 (Change Log Map) utility was improved to run even with given time formats to specify the CRCR. Prior to this release, the administrator had to convert the given time to the necessary RBA or LRSN. Now the syntax of CRESTART has been modified to accept the keyword ENDTIME. Also, SYSPITR has been modified to accept a log-truncation-timestamp instead of RBA or LRSN for log or truncation point.

The process of finding an adequate RBA or LRSN to the given time was time consuming and complex. Some locations where RBA or LRSN are assigned to times are:

- The SYSIBM.SYSCOPY timestamp and HEX(START_RBA)
- BSDS in the checkpoint queue GMT
- Logs to scan with DSN1LOGP

Now with the modifications of DSNJU003, this is a thing of the past.

On the CRESTART statement, ENDTIME is a timestamp value that is to be used as the log truncation point. A *valid log truncation point* is any GMT timestamp for which a log record exists with a timestamp that is greater than or equal to the specified timestamp value. Any log information in the BSDS, the active logs, and the archive logs with a timestamp greater than ENDTIME is discarded. DSNJU003 converts the timestamp to a store-clock (STCK) value. It also uses the first 6 bytes (plus any data sharing LRSN delta) to generate an LRSN or STCK value for log truncation in a data sharing environment. For non-data sharing, log records with an LRHTIME value greater than the STCK value generated from the ENDTIME timestamp are truncated.

For the SYSPITRT keyword, the SYSPITRT=log-truncation-timestamp specifies the timestamp value that represents the point-in-time log truncation point for system recovery. Before you run the RESTORE SYSTEM utility to recover system data, you must use the SYSPITR or SYSPITRT option of DSNJU003. This option enables you to create a conditional restart control record to truncate the logs for system point-in-time recovery.

The log-truncation-timestamp specifies a timestamp value that is to be used as the log truncation point. A valid log truncation point is any GMT timestamp for which a log record exists with a timestamp that is greater than or equal to the specified timestamp value. Any log information in the BSDS, the active logs, and the archive logs with a timestamp greater than SYSPITRT is discarded. Use the same timestamp value for all members of the data sharing group that require log truncation.

Finding the previous checkpoint

Since the checkpoint interval can be set to specific times (with DB2V7), many customers have changed the LOGLOAD parameter to CHKTIME. You can do this either with the DB2 command SET LOG or initially at every start of DB2 as entered in the DSNZPARM CHKFREQ command. If you want to use system-level backups as a recovery base, we recommend that you set the interval between checkpoints using the number of minutes instead of number of log records that are written.

LOGLOAD(n) specifies the number of log records that DB2 writes between the start of successive checkpoints. CHKTIME(n) specifies the number of minutes between the start of successive checkpoints.

When starting DB2, the last checkpoint is searched in the logs to scan the logs from this point for any backout or recovery of uncommitted work.

Example 7-11 Restarting a status table during a DB2 restart

```
DSNR003I  -D2S1 RESTART...PRIOR CHECKPOINT RBA=007153F7776C
DSNR004I  -D2S1 RESTART...UR STATUS COUNTS  038
IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0, POSTPONED ABORT=0
DSNR005I  -D2S1 RESTART...COUNTS AFTER FORWARD  039
RECOVERY
IN COMMIT=0, INDOUBT=0
DSNR006I  -D2S1 RESTART...COUNTS AFTER BACKWARD  040
RECOVERY
INFLIGHT=0, IN ABORT=0, POSTPONED ABORT=0
DSNG007I  -D2S1 DB2 CATALOG LEVEL (910) CODE LEVEL (910) MODE (N*)
DSNR002I  -D2S1 RESTART COMPLETED
```

Of course the restart time increases with the number of log records to search for uncommitted work.

Each checkpoint written in the logs is recorded in BSDS and can be monitored with DSNJU004 (Example 7-12). In case of data sharing, each member has its own checkpoint queue.

Example 7-12 DSNJU004

```
CHECKPOINT QUEUE
                      18:59:57 OCTOBER 18, 2006
      TIME OF CHECKPOINT       05:42:03 OCTOBER 18, 2006
      BEGIN CHECKPOINT RBA            007153F7C090
      END CHECKPOINT RBA              007153F7E4E9
      END CHECKPOINT LRSN             BF91F9D64CAB
      TIME OF CHECKPOINT       04:27:08 OCTOBER 18, 2006
      BEGIN CHECKPOINT RBA            007153F7776C
      END CHECKPOINT RBA              007153F79B74
      END CHECKPOINT LRSN             BF91E9175DBB
      TIME OF CHECKPOINT       04:17:08 OCTOBER 18, 2006
      BEGIN CHECKPOINT RBA            007153F73C9E
      END CHECKPOINT RBA              007153F760B8
      END CHECKPOINT LRSN             BF91E6DB2B48
      TIME OF CHECKPOINT       04:07:08 OCTOBER 18, 2006
      BEGIN CHECKPOINT RBA            007153F701A2
      END CHECKPOINT RBA              007153F725EB
      END CHECKPOINT LRSN             BF91E49EF744
```

Unfortunately only the last 100 checkpoints can be stored in the checkpoint queue. When specified as LOGLOAD, this is not a problem, because new checkpoints are created only when log changes occur. If a lot of traffic is on the logs, this results in a log switch and archives of both -logs and BSDS, and the new checkpoint queue corresponds to the new log. As we mentioned before, specific time intervals (for example, every 30 minutes) often force a new checkpoint to be taken.

At restart, DB2 scans the BSDS queue to find the earliest checkpoint to determine a starting point in searching the logs. If the checkpoint queue does not contain any checkpoint prior to ENDRBA=007029900000, no restart control record can be written to the BSDS, and the utility fails with RC 8. Example 7-13 shows the failure of DSNJU003 in this case.

Example 7-13   DSNJU003 failure

```
DSNJCNVB CONVERSION PROGRAM HAS RUN   DDNAME=SYSUT1
CRESTART CREATE,ENDRBA=007029900000
DSNJ407E  DSNRJFCK NO VALID CHECKPOINT RBA FOUND
DSNJ411I  DSNRJRCR CRESTART CREATE FOR CRCRID = 0001, DDNAME = SYSUT1
DSNJ221I  PREVIOUS ERROR CAUSED CRESTART OPERATION TO BE BYPASSED
DSNJ201I  DSNJU003 CHANGE LOG INVENTORY UTILITY PROCESSING WAS
UNSUCCESSFUL
```

If a failure happens, the administrator must manually find a valid checkpoint. The administrator might do this to print the prior archived checkpoint queue with DSNJU004 from the correct BSDS and find the last written checkpoint, which is somewhat complex. Or the administrator might do this to print the log with DSN1LOGP to find the last checkpoint, which is even more complicated.

Finally the administrator must set up DSNJU003 again with the valid checkpoint. Example 7-14 shows a successful run.

Example 7-14 DSNJU003 successful

```
DSNJCNVB CONVERSION PROGRAM HAS RUN   DDNAME=SYSUT1
CRESTART CREATE,ENDRBA=007029900000,CHKPTRBA=007029871214
DSNJ411I  DSNRJRCR CRESTART CREATE FOR CRCRID = 0001, DDNAME = SYSUT1
DSNJ225I  CRESTART OPERATION COMPLETED SUCCESSFULLY
DSNJ200I  DSNJU003 CHANGE LOG INVENTORY UTILITY PROCESSING COMPLETED
SUCCESSFULL
```

Now the CRCR can be written, and DB2 uses this checkpoint during the restart (Example 7-15).

Example 7-15 DSNJU004 CRCR

```
CONDITIONAL RESTART CONTROL RECORD
                      20:36:25 OCTOBER 18, 2006
 **** ACTIVE CRCR RECORD ****
 CRCR IDENTIFIER  0001
      USE COUNT   0
      RECORD STATUS
          CRCR ACTIVE
          CRCR NOT USED
      PROCESSING STATUS
          FORWARD = YES
          BACKOUT = YES
      STARTRBA                        NOT SPECIFIED
      ENDRBA                          007029900000
      ENDLRSN                         NOT SPECIFIED
      ENDTIME                         NOT SPECIFIED
      EARLIEST REQUESTED RBA          000000000000
      FIRST LOG RECORD RBA            000000000000
      ORIGINAL CHECKPOINT RBA         000000000000
      NEW CHECKPOINT RBA (CHKPTRBA)   007029871214
      CRCR CREATED             20:34:01 OCTOBER 18, 2006
      RESTART PROGRESS                 STARTED     ENDED
                                       =======     =====
          CURRENT STATUS REBUILD          NO         NO
          FORWARD RECOVERY PHASE          NO         NO
          BACKOUT RECOVERY PHASE          NO         NO
 ****
```

As shown in Example 7-15, the chosen CHKPTRBA (007029871214) is inserted in the CRCR in the BSDS.

With DB2 9 for z/OS, the DSNJU003 utility has been enhanced. The utility now places a warning in the CRCR indicating that a backward scan of the log to locate the checkpoint is required. See Example 7-16.

Example 7-16 DSNJU003 no valid checkpoint

```
DSNJCNVB CONVERSION PROGRAM HAS RUN   DDNAME=SYSUT1
CRESTART CREATE,ENDRBA=007029900000
DSNJ407I  DSNRJFCK WARNING - NO VALID CHECKPOINT RBA FOUND.  LOG WILL
BE SCANNED AT RESTART
DSNJ411I  DSNRJRCR CRESTART CREATE FOR CRCRID = 0001, DDNAME = SYSUT1
DSNJ200I  DSNJU003 CHANGE LOG INVENTORY UTILITY PROCESSING COMPLETED
SUCCESSFULLY
```

The prior CHKPTRBA is searched automatically by DB2 at restart (SCAN REQUIRED) as shown in Example 7-17.

Example 7-17 DSNJU004 CRCR - SCAN REQUIRED

```
CONDITIONAL RESTART CONTROL RECORD
                    20:42:55 OCTOBER 18, 2006
**** ACTIVE CRCR RECORD ****
CRCR IDENTIFIER  0001
     USE COUNT   0
     RECORD STATUS
         CRCR ACTIVE
         CRCR NOT USED
     PROCESSING STATUS
         FORWARD = YES
         BACKOUT = YES
     STARTRBA                       NOT SPECIFIED
     ENDRBA                         007029900000
     ENDLRSN                        NOT SPECIFIED
     ENDTIME                        NOT SPECIFIED
     EARLIEST REQUESTED RBA          000000000000
     FIRST LOG RECORD RBA            000000000000
     ORIGINAL CHECKPOINT RBA         000000000000
     NEW CHECKPOINT RBA (CHKPTRBA)   SCAN REQUIRED
     CRCR CREATED           20:41:28 OCTOBER 18, 2006
     RESTART PROGRESS                 STARTED     ENDED
                                      =======     =====
          CURRENT STATUS REBUILD         NO        NO
          FORWARD RECOVERY PHASE         NO        NO
          BACKOUT RECOVERY PHASE         NO        NO
****
```

At the beginning of the recovery phase of the subsequent DB2 restart, the new message (DSNR054I) indicates that a scan of the log to locate the restart checkpoint is to be performed. The log is read backwards from the log truncation point, until a complete checkpoint is found.

Example 7-18 JESMSGLG: DB2 scans for checkpoint

```
DSNR001I  -D2S1 RESTART INITIATED
DSNR054I  -D2S1 RESTART...BEGIN SCAN FOR CHECKPOINT
DSNR003I  -D2S1 RESTART...PRIOR CHECKPOINT RBA=0073B456E604
DSNR004I  -D2S1 RESTART...UR STATUS COUNTS  153
IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0, POSTPONED ABORT=0
DSNR005I  -D2S1 RESTART...COUNTS AFTER FORWARD  154
RECOVERY
IN COMMIT=0, INDOUBT=0
DSNR006I  -D2S1 RESTART...COUNTS AFTER BACKWARD  155
RECOVERY
INFLIGHT=0, IN ABORT=0, POSTPONED ABORT=0
DSNX230I  -D2S1 A RESOURCE IS UNAVAILABLE WHEN TRYING  156
TO BUILD THE TRUSTED CONTEXT CACHE REASON = 00C9000A TYPE OF RESOURCE
= 00000100 RESOURCE NAME = DSNDB01.DBD01
DSNR050I  -D2S1 DSNRRPRC DB2 STARTED IN SYSTEM RECOVER PENDING MODE
DSNR002I  -D2S1 RESTART COMPLETED
DSNY014I  -D2S1 DSNYSTRT DB2 WAS STARTED WITH ACCESS(MAINT)
DSN9022I  -D2S1 DSNYASCP 'START DB2' NORMAL COMPLETION
DSN3029I  -D2S1 DSN3RRSR RRS ATTACH PROCESSING IS AVAILABLE
```

7.3 Recovering to a point in time with consistency

Recovering to a point in time is a viable alternative in many situations in which recovery to the current point in time is not possible or not desirable. To make recovery to a point in time work best without losing data consistency, we recommend that you take periodic quiesce points at points of consistency.

In a transaction system that has real high volume, taking quiesce points blocks the application access on the objects that are being quiesced. Taking quiesce points frequently may create a lot of overhead in a production system. Also, in reality, many point-in-time recoveries must be done to unplanned recovery points. If recovery to an inconsistent point in time must be done, then manual repair of the data left in an inconsistent state is required due to the uncommitted change at recovery point.

This process can be time consuming and often error prone, and requires deeper DB2 knowledge. Also, the ability to recover a list of objects to a point in time with consistency completes the backup and recovery picture when using system-level backups as a recovery base.

Since system-level backups are not taken at consistent points, the enhancement added by this line item allows you to use system-level backups as a recovery base for point-in-time recoveries on a list of objects with consistency. If you want to use system-level backups as a recovery base, we recommend that you set the interval between checkpoints, using the number of minutes instead of the number of log records written.

This enhancement of the RECOVER utility automatically detects the uncommitted transactions that are running at the recover point in time and rolls back their changes on the recovered objects. After recovery, objects are left in their transactionally consistent state.

Recover TOLOGPOINT and TORBA have the recover with consistency as its default behavior. Recover TOCOPY, TOLASTCOPY, and TOLASTFULLCOPY using SHRLEVEL CHANGE copy still have today's recovery behavior, so no consistency is ensured. To achieve consistency with a recovery to a SHRLEVEL CHANGE copy, you can use a desired SHRLEVEL CHANGE copy as the recovery base and specify a recovery point that was right after the copy completed via the TORBA or TOLOGPOINT syntax. You can locate the entry in the SYSCOPY table for the SHRLEVEL CHANGE copy that is being used and use the RBA or LRSN value in the PIT_RBA column as the recovery point. Use the REPORT RECOVERY utility to see the report of the SYSCOPY records

As part of the recovery process to recover to a point in time with consistency, the changes made on the recovered objects by units of recovery (URs) that are INFLIGHT, INABORT, and POSTPONED ABORT during the recovery time are rolled back. INDOUBT URs during the recovery point are treated as INABORT, and their changes on the recovered objects are also rolled back. INCOMMIT URs are treated as committed, and no rollback happens. If a UR that was active during the recovery to a point in time changed multiple objects in its life span, only the changes made by it on the objects being recovered in current RECOVER utility are rolled back. The changes made by it on other objects are not rolled back.

Therefore, it is essential to include all of the relevant objects in the same RECOVER utility job to ensure consistency from an application point of view. For point-in-time recovery, the RECOVER utility checks to see if all objects in a referential integrity (RI) set, base and LOB relationship, and base and XML relationship are recovered in the same list. If all objects are not specified in the list to be recovered, RECOVER sets the appropriate prohibitive pending state. For example, if the parent table space in an RI set is recovered to a point in time, but its dependent table space is not recovered in the same list, then the RECOVER utility places the dependent table space in a check pending (CHKP) state.

During recovery to a point in time with consistency, after the RECOVER utility applies the redo log records in the LOGAPPLY phase, if any DB2 member has an outstanding UR during the recovery point and has modified the objects that are being recovered, the RECOVER utility enters the log analysis phase.

LOGCSR is a new recovery phase and occurs after the RECOVER LOGAPPLY phase. LOGCSR stands for “log analysis current status rebuild”. If the RECOVER utility determines that there are no active units of recovery during the recovery point, is skips the LOGCSR phase. During the LOGCSR phase, for each DB2 member that has URs that have changed the objects that are being recovered and was outstanding during the recovery point, the RECOVER utility reads the log forward from the last checkpoint on this DB2 member prior to recovery point. It identifies the URs that were both active (INFLIGHT, INABORT, INDOUBT, or POSTPONED ABORT) during the recovery point and changed the objects that are being recovered. This is done both sequentially and on a member basis.

Before the start of log analysis for each DB2 member, a new message DSNU1550I is issued. It shows the name of the DB2 member whose log will be analyzed, along with the prior checkpoint RBA value from which the log analysis will start. After the log analysis is done for this DB2 member, a new message DSNU1551I is issued. It marks the end of log analysis for this member and indicates the elapsed time spent on this DB2 member’s log analysis. The RECOVER utility then starts the log analysis for the next DB2 member.

After the log analysis is done for all DB2 members, the RECOVER utility issues another new message, DSNU1552I. This message marks the end of the LOGCSR phase and indicates the total elapsed time used by the RECOVER utility for all DB2 members during the LOGCSR phase. The new messages DSNU1550I, DSNU1551I, and DSNU1552I are all part of the RECOVER utility job output.

After log analysis in the LOGCSR phase, the RECOVER utility issues a new DSNU1553I message that shows the UR status in a table format. This table shows a combination of the active URs and the recovered objects they changed. It also shows the name of DB2 member that the URs belong to and the RBA value of the earliest log record written by each active UR when it made the change on the object. Each UR’s status during the recovery point is also displayed, such as INFLIGHT, INABORT, POSTPONED ABORT, and INDOUBT. INCOMMIT URs are not displayed since they do not need to be rolled back. This message is displayed in the RECOVER utility job output. This message also shows the total number of INFLIGHT, INABORT, POSTPONED ABORT, and INDOUBT URs during the recovery point. If there is no active UR during the recovery point, these numbers are all zero.

If a UR's change on recovered objects needs to be backed out, a new phase called LOGUNDO for the RECOVER utility happens next. The LOGUNDO phase is entered only when the RECOVER job is at a point in time with consistency, or a there is UR that is active during both the recovery point and the changing objects that are being recovered. During the LOGUNDO phase, the RECOVER utility backs out the changes made on recovered objects by active URs on a member basis.

In a data sharing environment, if multiple DB2 members must be processed, a backout is done one member at a time. The sequence of the backout being done among the members is chosen randomly. A fast log apply process is not used during the LOGUNDO phase even if it is enabled on the DB2 subsystem.

At the beginning of the backout process of each DB2 member, the RECOVER utility issues a new message, DSNU1554I, into the RECOVER job output. It marks the start of the backout process on this member. The member name is part of the message. During the backout process, the RECOVER utility scans the DB2 log written by the DB2 member that is being processed backward in one pass. The backward scan starts from the recovery point. During the backward process, changes made on the recovered objects by this DB2 member's active URs are undone. The active URs were identified in the LOGCSR phase. The backward log scan continues until all the changes made on the recovered objects by this DB2 member's active URs are backed out.

During the backout phase, periodically, the RECOVER utility issues another new message, DSNU1555I, into its job output after it processes a certain number of log records. This message shows the RBA value of the current log record that is being processed by the RECOVER utility and the RBA value of the last log record that is read in this member's LOGUNDO phase. By monitoring this message, the user knows the progress of the recover LOGUNDO phase on this DB2 member. In extreme cases, for example, on one DB2 member, a long running UR may be changing objects that were recovered a long time ago and were active at the recovery point. The backout process for this member could take a long time since it has to roll back all the changes made by this UR on the recovered objects. This case should not happen often if all the applications commit on a regular basis.

If this case does happen, you can terminate the current RECOVER job and perform another RECOVER job on the same objects by choosing a different recover-to-point-in-time where the UR with the problem is not active at that time. The -TERM UTIL command can terminate the RECOVER utility during LOGAPPLY, LOGCSR, and LOGUNDO phases.

After the backout process on one DB2 member is done, the RECOVER utility issues another new message, DSNU1556I, to indicate that the backout is done on this member. The member name is included in the message. The elapsed time spent in the UNDO processing for this DB2 member is also shown in this message. This message is also saved in the RECOVER utility job output. If more DB2 members have log records that need to be processed, the RECOVER utility starts the backout process for the next DB2 member. When no more DB2 members are to be processed, the RECOVER utility issues a new message, DSNU1557I, in the RECOVER utility job output. This message marks the end of the LOGUNDO phase and indicates the total elapsed time spent in this phase.

The job output in Example 7-19 shows the new recovery process.

Prerequisite: A prerequisite for using this option without image copies is DFSMShsm Version 1.8 or later. If you have enabled system level backups in DSNZPARM and are not at the proper level, you receive the message:

```
DSNU1608I RECOVER SYSTEM UTILITY FAILED.  REQUIRED DFSMSHSM SERVICES
NOT AVAILABLE
```

To use this feature with native image copies, you must first disable the entry in DSNZPARM:

```
SYSTEM_LEVEL_BACKUPS=NO
```

Remember that in a data sharing environment, you must specify TOLOGPOINT instead of TORBA. Otherwise you receive this message:

```
DSNU529I  -D2S1 293 11:30:44.94 DSNUCAIN - INVALID SPECIFICATION OF
TORBA = X'0071F3F8A000'
```

The RBA is within the range of the used logs of the specific group member.

Example 7-19   RECOVER with consistency

```
1DSNU000I    293 12:17:51.10 DSNUGUTC - OUTPUT START FOR UTILITY,
UTILID = HEINZREC
 DSNU1044I   293 12:17:51.15 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
0DSNU050I    293 12:17:51.15 DSNUGUTC -  RECOVER TABLESPACE
DSN8D81A.DSN8S81S TOLOGPOINT X'BF94F73EE1E6'
 DSNU459I    293 12:17:51.28 DSNUCBMD - SYSCOPY P RECORD ENCOUNTERED
FOR TABLESPACE DSN8D81A.DSN8S81S  , PIT
 RBA=BF927F24395F
 DSNU515I    293 12:17:51.28 DSNUCBAL - THE IMAGE COPY DATA SET
BAEK.HEELA.T153805.REORG.SYSCOPY1 WITH DATE=20061018 AND
 TIME=113827
                        IS PARTICIPATING IN RECOVERY OF TABLESPACE
DSN8D81A.DSN8S81S
 DSNU504I    293 12:17:55.07 DSNUCBMD - MERGE STATISTICS FOR TABLESPACE
DSN8D81A.DSN8S81S   -
                        NUMBER OF COPIES=1
                        NUMBER OF PAGES MERGED=21134
                        ELAPSED TIME=00:00:03
 DSNU830I  -D2S1 293 12:17:51.34 DSNUCARS - INDEX DSN8810.XPARTS IS IN
REBUILD PENDING
 DSNU831I  -D2S1 293 12:17:51.35 DSNUCARS - ALL INDEXES OF
DSN8D81A.DSN8S81S ARE IN REBUILD PENDING
 DSNU578I  -D2S1 293 12:17:55.11 DSNUCALA - SYSLGRNX INFORMATION FOR
MEMBER D2S1
 DSNU513I  -D2S1 293 12:17:55.11 DSNUCALA - RECOVER UTILITY LOG APPLY
RANGE IS RBA 00716C90DF2C LRSN BF92A6F681E4 TO
                                                           RBA
007175526260 LRSN BF92A978DC4F

 DSNU579I  -D2S1 293 12:17:55.29 DSNUCACL - RECOVER UTILITY LOG APPLY
AT LOGPOINT BF92A6F70971

 DSNU579I  -D2S1 293 12:17:59.58 DSNUCACL - RECOVER UTILITY LOG APPLY
AT LOGPOINT BF92A703A364

  DSNU579I  -D2S1 293 12:18:34.53 DSNUCACL - RECOVER UTILITY LOG APPLY
AT LOGPOINT BF94F73ED03E
 DSNU1510I   293 12:18:34.77 DSNUCBLA - LOG APPLY PHASE COMPLETE,
ELAPSED TIME = 00:00:39
 DSNU1550I -D2S1 293 12:18:34.78 DSNUCALC - LOGCSR IS STARTED FOR
MEMBER D2S1, PRIOR CHECKPOINT RBA = X'0071F0001AE0'
 DSNU1551I -D2S1 293 12:18:35.01 DSNUCALC - LOGCSR IS FINISHED FOR
MEMBER D2S1, ELAPSED TIME = 00:00:00
 DSNU1552I -D2S1 293 12:18:35.01 DSNUCALC - LOGCSR PHASE COMPLETE,
ELAPSED TIME = 00:00:00
 DSNU1553I -D2S1 293 12:18:35.01 DSNUCALC - RECOVER DETECTS THE
FOLLOWING ACTIVE URS:
              INFLIGHT = 1, INABORT = 0, INDOUBT = 0, POSTPONED ABORT =
0

                MEM    T  CONNID     CORRID     AUTHID    PLAN   S
URID       DATE       TIME
```

```
              D2S1     B TSO      RUFLAIR      RUFLAIR  DSNESPCS F
0071E55E52B4 2006-10-20 14.46.11
                    DBNAME  SPACENAME DBID/PSID PART     RBA
                   DSN8D81A DSN8S81S  0105/000A 0000 BF94F73035FD

 DSNU1554I -D2S1 293 12:18:35.15 DSNUCALU - LOGUNDO IS STARTED FOR
MEMBER D2S1
 DSNU1555I -D2S1 293 12:18:35.32 DSNUCACL - RECOVER LOGUNDO STATUS: LOG
RECORD AT RBA X'BF94F73EC7DA' TO RBA
 X'BF94F73035FD' ON MEMBER D2S1
 DSNU1555I -D2S1 293 12:18:58.91 DSNUCACL - RECOVER LOGUNDO STATUS: LOG
RECORD AT RBA X'BF94F7304160' TO RBA
 X'BF94F73035FD' ON MEMBER D2S1
 DSNU1556I -D2S1 293 12:18:58.99 DSNUCALU - LOGUNDO IS FINISHED FOR
MEMBER D2S1, ELAPSED TIME = 00:00:23
 DSNU1557I -D2S1 293 12:18:58.99 DSNUCALU - LOGUNDO PHASE COMPLETE,
ELAPSED TIME = 00:00:23
 DSNU500I    293 12:18:59.15 DSNUCBDR - RECOVERY COMPLETE, ELAPSED
TIME=00:01:08
 DSNU010I    293 12:18:59.15 DSNUGBAC - UTILITY EXECUTION COMPLETE,
HIGHEST RETURN CODE=4
```

8

New functions for SAP high availability in DB2 9 for z/OS

High availability solutions for SAP NetWeaver on DB2 z/OS are available based on Parallel Sysplex® data sharing and SAP sysplex failover. Additionally, the SAP high availability solution uses System Automation for z/OS to automate the starting, stopping, and monitoring of all SAP components.

DB2 9 for z/OS provides new features to reduce the planned and unplanned outages by:

- Data sharing restart high availability
- Online REORG enhancements
  - Elimination of the BUILD2 phase
  - Unload and reload of data in parallel streams
  - Online REORG for LOBs
  - Clone
  - RETRY_DELAY
- Online CHECK DATA and CHECK LOB function
- PIT Recovery: Scan log for prior checkpoint
- Online REBUILD INDEX
- Add REFRESH command for EARLY code

8.1 Data sharing restart high availability

New functions the work with DB2 data sharing members reduce planned and unplanned outages. Additionally the high availability enhancements for data sharing restart improve the performance and usability of DB2 restart.

The following list gives an overview about the advantages of the new functions that are included in the data sharing restart process:

- Avoiding certain locks for Group Buffer Pool (GBP) dependent objects and opening the objects involved during the DB2 restart as early as possible reduces the downtime for DB2.
- Restarting DB2 by automatically initiating the Group Buffer Pool Recovery (GRECP) of the GRECP objects at the end of restart improves the usability.
- Simplify the special open processing at the end of restart and normal open processing during restart.
- If AUTOREC is set, initiate auto GRECP recovery at the end of restart for disaster recovery (DR) members.
- Initiating to open the page set early in the Forward Log Recovery (FLR) phase applies if you not use the DEFER ALL option.

8.2 Online REORG and elimination of the BUILD2 phase

The REORG utility reorganizes a table space from an index space to improve access performance, to reclaim fragmented space, and to optionally compress a table space. It unloads the data of a table space, or index space, and reorganizes the data in optimum order, as specified by the clustering index. This ensures that optimum performance can be achieved by SQL that is referencing the data via the clustering index. REORG and RUNSTATS allow the DB2 optimizer to choose better access paths. A REORG without the BUILD2 phase eliminates outages.

Online tablespace reorganization for partitions has been improved by removing the BUILD2 phase for all types of secondary indexes (see Figure 8-1). This means that reorganization of individual partitions or a set of partitions is much more transparent to the transactions that operate on that object.

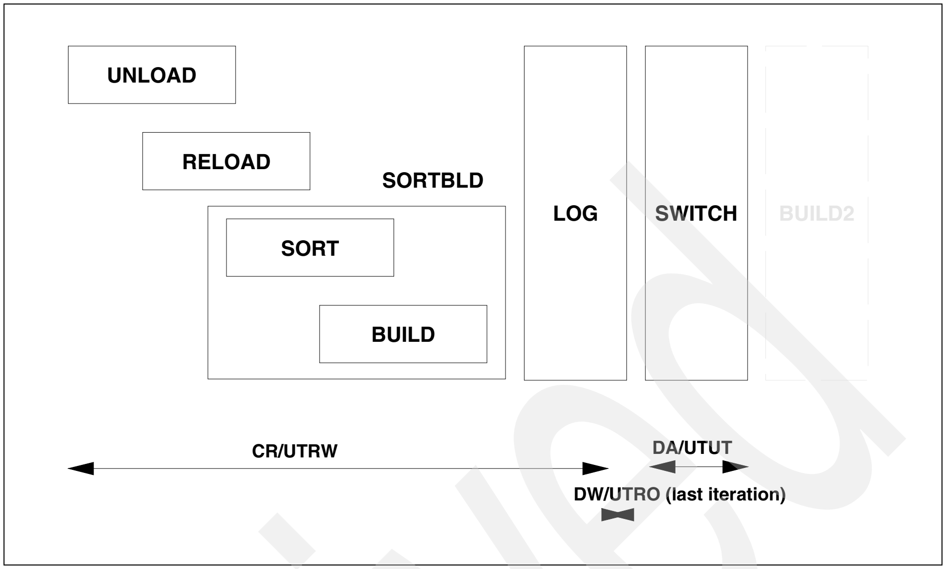


Figure 8-1 Online REORG(SHRLEVEL CHANGE) and online REORG PART phases

Example 8-1 shows the output from a reorganization job that demonstrates the online reorg phases.

Example 8-1 Online REORG PART SHRLEVEL CHANGE

```
1DSNU000I    298 15:51:17.13 DSNUGUTC - OUTPUT START FOR UTILITY,
UTILID = REOGJOB.REORG
 DSNU1044I   298 15:51:17.17 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
0DSNU050I    298 15:51:17.20 DSNUGUTC -  TEMPLATE SYSCOPY UNIT(SYSDA)
DSN(BAEK.REORG.T&TI..SYSCOPY1)
 DSNU1035I   298 15:51:17.20 DSNUJTDR - TEMPLATE STATEMENT PROCESSED
SUCCESSFULLY
0DSNU050I    298 15:51:17.20 DSNUGUTC -  REORG TABLESPACE
DSNDB04.MARATS PART 3 MAPPINGTABLE DSN8910.MAP_TBL SHRLEVEL CHANGE
SORTDEVT SYSDA NOSYSREC
 DSNU1038I   298 15:51:20.05 DSNUGDYN - DATASET ALLOCATED.
TEMPLATE=SYSCOPY
                        DDNAME=SYS00001
                        DSN=BAEK.REORG.T195117.SYSCOPY1
 DSNU1160I   298 15:51:20.09 DSNURPRD - PARTITIONS WILL BE
UNLOADED/RELOADED IN PARALLEL, NUMBER OF TASKS = 1
 DSNU1161I   298 15:51:20.09 DSNURPRD - NON-PARTITIONED INDEXES WILL BE
UNLOADED IN PARALLEL, NUMBER OF TASKS = 1
```

```
 DSNU395I    298 15:51:20.09 DSNURPRD - INDEXES WILL BE BUILT IN
PARALLEL, NUMBER OF TASKS = 4
 DSNU397I    298 15:51:20.09 DSNURPRD - NUMBER OF TASKS CONSTRAINED BY
CPUS
 DSNU251I  -D2S1 298 15:51:20.27 DSNURPUT - UNLOAD PHASE STATISTICS -
NUMBER OF RECORDS UNLOADED=4701 FOR TABLESPACE DSNDB04.MARATS PART 3
 DSNU250I    298 15:51:20.27 DSNURPRD - UNLOAD PHASE COMPLETE, ELAPSED
TIME=00:00:00
 DSNU701I  -D2S1 298 15:51:20.18 DSNURULN - 8503 INDEX ENTRIES UNLOADED
FROM 'BAEK.X1_MARA'
 DSNU701I  -D2S1 298 15:51:21.24 DSNURULN - 8503 INDEX ENTRIES UNLOADED
FROM 'BAEK.X2_MARA'
 DSNU701I  -D2S1 298 15:51:21.30 DSNURULN - 8503 INDEX ENTRIES UNLOADED
FROM 'BAEK.X3_MARA'
 DSNU701I  -D2S1 298 15:51:21.37 DSNURULN - 8503 INDEX ENTRIES UNLOADED
FROM 'BAEK.X4_MARA'
 DSNU701I  -D2S1 298 15:51:21.44 DSNURULN - 8503 INDEX ENTRIES UNLOADED
FROM 'BAEK.X5_MARA'
 DSNU303I  -D2S1 298 15:51:21.46 DSNURWT - (RE)LOAD PHASE STATISTICS -
NUMBER OF RECORDS=4701 FOR TABLE BAEK.MARA_PART PART=3
 DSNU302I    298 15:51:21.47 DSNURILD - (RE)LOAD PHASE STATISTICS -
NUMBER OF INPUT RECORDS PROCESSED=4701
 DSNU300I    298 15:51:21.47 DSNURILD - (RE)LOAD PHASE COMPLETE,
ELAPSED TIME=00:00:01
 DSNU394I  -D2S1 298 15:51:22.56 DSNURBXA - SORTBLD PHASE STATISTICS -
NUMBER OF KEYS=4701 FOR INDEX DSN8910.XMAP_TBL
 DSNU394I  -D2S1 298 15:51:22.50 DSNURBXA - SORTBLD PHASE STATISTICS -
NUMBER OF KEYS=13204 FOR INDEX BAEK.X1_MARA
 DSNU394I  -D2S1 298 15:51:23.52 DSNURBXA - SORTBLD PHASE STATISTICS -
NUMBER OF KEYS=13204 FOR INDEX BAEK.X2_MARA
 DSNU394I  -D2S1 298 15:51:24.55 DSNURBXA - SORTBLD PHASE STATISTICS -
NUMBER OF KEYS=13204 FOR INDEX BAEK.X3_MARA
 DSNU394I  -D2S1 298 15:51:24.57 DSNURBXA - SORTBLD PHASE STATISTICS -
NUMBER OF KEYS=13204 FOR INDEX BAEK.X4_MARA
 DSNU394I  -D2S1 298 15:51:24.60 DSNURBXA - SORTBLD PHASE STATISTICS -
NUMBER OF KEYS=13204 FOR INDEX BAEK.X5_MARA
 DSNU391I    298 15:51:24.62 DSNURPTB - SORTBLD PHASE STATISTICS.
NUMBER OF INDEXES = 6
 DSNU392I    298 15:51:24.62 DSNURPTB - SORTBLD PHASE COMPLETE, ELAPSED
TIME = 00:00:03
 DSNU1162I   298 15:51:24.62 DSNURLGD - LOG APPLIES WILL BE PERFORMED
IN PARALLEL, NUMBER OF TASKS = 4
 DSNU386I    298 15:51:25.71 DSNURLGD - LOG PHASE STATISTICS. NUMBER OF
ITERATIONS = 1, NUMBER OF LOG RECORDS = 0
```

```
 DSNU385I    298 15:51:25.71 DSNURLGD - LOG PHASE COMPLETE, ELAPSED
TIME = 00:00:01
 DSNU400I    298 15:51:25.75 DSNURBID - COPY PROCESSED FOR TABLESPACE
DSNDB04.MARATS PART 3
                      NUMBER OF PAGES=837
                      AVERAGE PERCENT FREE SPACE PER PAGE = 10.35
                      PERCENT OF CHANGED PAGES =100.00
                      ELAPSED TIME=00:00:05
 DSNU387I    298 15:51:26.76 DSNURSWT - SWITCH PHASE COMPLETE, ELAPSED
TIME = 00:00:01
 DSNU428I    298 15:51:26.76 DSNURSWT - DB2 IMAGE COPY SUCCESSFUL FOR
TABLESPACE DSNDB04.MARATS PARTITION 3
 DSNU010I    298 15:51:27.99 DSNUGBAC - UTILITY EXECUTION COMPLETE,
HIGHEST RETURN CODE=0
```

There is no partitioned table space at the beginning. Figure 8-2 shows the window on which you see key values for the partitioned table space from the segment table space (SAP transaction SE14).

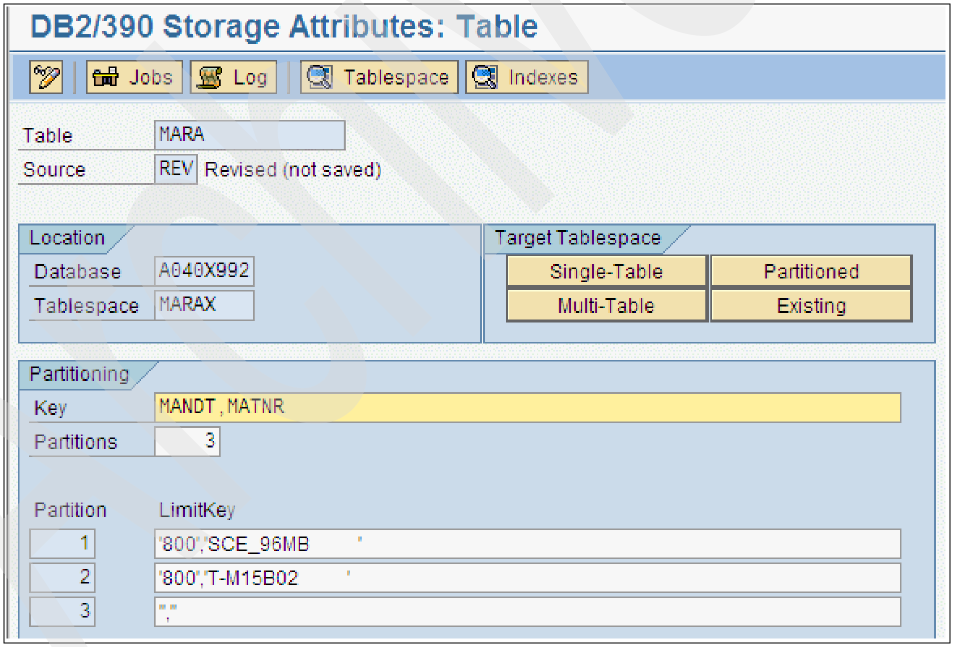


Table: MARA
Source: REV Revised (not saved)

Database: A040X992
Tablespace: MARAX

Key: MANDT,MATNR
Partitions: 3

Partition	LimitKey
1	'800','SCE_96MB '
2	'800','T-M15B02 '
3	'',''

Figure 8-2 Example of producing part key values

8.3 Intra-REORG parallelism

DB2 9 for z/OS can considerably accelerate the processing of the REORG utility by running crucial phases of the REORG utility in parallel. REORG attempts to unload and reload tablespace partitions in parallel. Also REORG with the options of SHRLEVEL REFERENCE or CHANGE attaches one or more subtasks during the LOG phase to accelerate the processing of log records.

REORG aims to unload and reload table space partitions in parallel in any of the following situations:

- The NOSYSREC keyword is specified or defaulted as in SHRLEVELCHANGE.
- The UNLDDN keyword is specified with a template name where the template's dataset name pattern includes a partition number.

However, the following situations prevent unload parallelism:

- DATAWKnn DD statements are specified.
- The SORTDEVT keyword is not specified.
- The UTPRINT data set is not allocated to SYSOUT.
- The REBALANCE keyword is specified.
- The possibility exists that rows may be moved from one partition to another.

Non-partitioning index (NPI) unload subtasks appear in the -DIS UTIL DSNU111I message with a phase name of UNLOADIX. Example 8-2 shows the output for the NPI unload subtasks.

Example 8-2 -DIS UTIL for NPI parallelism

```
DSNU105I  -D2S1 DSNUGDIS - USERID = BAEK
                MEMBER = D2S1
                UTILID = REOGJOB.REORG
                PROCESSING UTILITY STATEMENT 1
                UTILITY = REORG
                PHASE = RELOAD   COUNT = 0
                NUMBER OF OBJECTS IN LIST = 1
                LAST OBJECT STARTED = 1
                STATUS = ACTIVE
DSNU347I  -D2S1 DSNUGDIS -
                DEADLINE = NONE
DSNU384I  -D2S1 DSNUGDIS -
                MAXRO = 1800 SECONDS
                LONGLOG = CONTINUE
                DELAY = 1200 SECONDS
DSNU111I  -D2S1 DSNUGDIS - SUBPHASE = UNLOADIX COUNT = 3970
```

```
DSNU111I  -D2S1 DSNUGDIS - SUBPHASE = UNLOAD COUNT = 4615
DSNU111I  -D2S1 DSNUGDIS - SUBPHASE = BUILD COUNT = 0
DSNU111I  -D2S1 DSNUGDIS - SUBPHASE = BUILD COUNT = 25767
DSNU111I  -D2S1 DSNUGDIS - SUBPHASE = COPY COUNT = 2
DSN9022I  -D2S1 DSNUGCCC '-DIS UTIL' NORMAL COMPLETION
***
```

Example 8-3 shows the output that is produced during the unload of non-partitioned indexes.

Example 8-3   NPI parallel UNLOAD,RELOAD

```
1DSNU000I    299 18:30:05.64 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID =
REOGJOB.REORG
 DSNU1044I   299 18:30:05.68 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
0DSNU050I    299 18:30:05.69 DSNUGUTC -  TEMPLATE SYSCOPY UNIT(SYSDA)
DSN(BAEK.&TS..T&TI..SYSCOPY1)
 DSNU1035I   299 18:30:05.69 DSNUJTDR - TEMPLATE STATEMENT PROCESSED
SUCCESSFULLY
0DSNU050I    299 18:30:05.69 DSNUGUTC -  REORG TABLESPACE
DSNDB04.MARATS PART 2 MAPPINGTABLE DSN8910.MAP_TBL
 SHRLEVEL CHANGE SORTDEVT SYSDA NOSYSREC
 DSNU1038I   299 18:30:08.58 DSNUGDYN - DATASET ALLOCATED.
TEMPLATE=SYSCOPY
                        DDNAME=SYS00001
                        DSN=BAEK.MARATS.T223005.SYSCOPY1
 DSNU1160I   299 18:30:08.60 DSNURPRD - PARTITIONS WILL BE
UNLOADED/RELOADED IN PARALLEL, NUMBER OF TASKS = 1
 DSNU1161I   299 18:30:08.60 DSNURPRD - NON-PARTITIONED INDEXES WILL BE
UNLOADED IN PARALLEL, NUMBER OF TASKS = 1
 DSNU395I    299 18:30:08.60 DSNURPRD - INDEXES WILL BE BUILT IN
PARALLEL, NUMBER OF TASKS = 4
 DSNU397I    299 18:30:08.60 DSNURPRD - NUMBER OF TASKS CONSTRAINED BY
CPUS
 DSNU251I  -D2S1 299 18:30:08.82 DSNURPUT - UNLOAD PHASE STATISTICS -
NUMBER OF RECORDS UNLOADED=4615 FOR TABLESPACE
 DSNDB04.MARATS PART 2
```

Another new subtask appears in the -DIS UTIL command output, with the message DSNU111, with phase names of LOGAPPLY. See the text in bold in Example 8-4.

Example 8-4   JOB LOG for parallel log apply

```
PREVIOUS ITERATION:
   ELAPSED TIME = 00:00:00
   NUMBER OF LOG RECORDS PROCESSED = 0
CURRENT ITERATION:
   ESTIMATED ELAPSED TIME = 00:00:00
   ACTUAL ELAPSED TIME SO FAR = 00:01:20
   ACTUAL NUMBER OF LOG RECORDS BEING PROCESSED = 0
CURRENT ESTIMATE FOR NEXT ITERATION:
   ELAPSED TIME = 00:00:01
   NUMBER OF LOG RECORDS TO BE PROCESSED = 0
DSNU111I  -D2S1 DSNUGDIS - SUBPHASE = COPY COUNT = 836
DSNU111I  -D2S1 DSNUGDIS - SUBPHASE = LOGAPPLY COUNT = 0
DSNU111I  -D2S1 DSNUGDIS - SUBPHASE = LOGAPPLY COUNT = 0
DSNU111I  -D2S1 DSNUGDIS - SUBPHASE = LOGAPPLY COUNT = 0
DSNU111I  -D2S1 DSNUGDIS - SUBPHASE = LOGAPPLY COUNT = 0
DSN9022I  -D2S1 DSNUGCCC '-DIS UTIL' NORMAL COMPLETION
Example of -DIS UTIL for parallel log apply
DSNU392I    298 16:18:08.05 DSNURPTB - SORTBLD PHASE COMPLETE, ELAPSED
TIME = 00:00:00
 DSNU1162I   298 16:18:08.05 DSNURLGD - LOG APPLIES WILL BE PERFORMED
IN PARALLEL, NUMBER OF TASKS = 4
 DSNU386I    298 16:18:08.87 DSNURLGD - LOG PHASE STATISTICS. NUMBER OF
ITERATIONS = 1, NUMBER OF LOG RECORDS = 0
 DSNU385I    298 16:18:08.87 DSNURLGD - LOG PHASE COMPLETE, ELAPSED
TIME = 00:00:00
 DSNU400I    298 16:18:08.89 DSNURBID - COPY PROCESSED FOR TABLESPACE
DSNDB04.MARATS PART 3
```

The size of the shadow data sets that is required for NPIs with a REORG TABLESPACE SHRLEVEL CHANGE (or REFERENCE) PART x is now larger. Instead of allocating it using the percentage of space required for the partition compared to the whole table space, the shadow NPI now needs to be as large as the whole original NPI.

The naming convention for the shadow data set for NPIs has changed as shown in the following example. The old naming convention was *catname*.

```
DSNDBx.dbname.psname.y0mmm.Annn
```

Here *y* is I or J, *mmm* is the part number of the first partition in the range that is being reorganized, and *nnn* is the piece number. The new naming convention for these data sets is the same as the naming convention for the data partitions and parts of partitioned indexes, *catname*.

```
DSNDBx.dbname.psname.y0001.Annn.
```

Example 8-5 shows a sample of DATASET while running a reorganization using the online REORG PART.

Example 8-5 Sample DATASET naming rule while online REORG PART

```
DB2SU.DSNDBD.DSNDB04.MARATS.I0001.A001
DB2SU.DSNDBD.DSNDB04.MARATS.I0001.A002
DB2SU.DSNDBD.DSNDB04.MARATS.I0001.A003
DB2SU.DSNDBD.DSNDB04.MARATS.J0001.A003
DB2SU.DSNDBD.DSNDB04.X1RMARA.I0001.A001
DB2SU.DSNDBD.DSNDB04.X1RMARA.J0001.A001
DB2SU.DSNDBD.DSNDB04.X2RMARA.I0001.A001
DB2SU.DSNDBD.DSNDB04.X2RMARA.J0001.A001
DB2SU.DSNDBD.DSNDB04.X3RMARA.I0001.A001
DB2SU.DSNDBD.DSNDB04.X3RMARA.J0001.A001
DB2SU.DSNDBD.DSNDB04.X4RMARA.I0001.A001
DB2SU.DSNDBD.DSNDB04.X4RMARA.J0001.A001
DB2SU.DSNDBD.DSNDB04.X5RMARA.I0001.A001
DB2SU.DSNDBD.DSNDB04.X5RMARA.J0001.A001
```

8.4 REORG LOB

You can reorganize with SHRLEVEL REFERENCE for large object (LOB) table spaces. If you describe REFERENCE for an LOB table space, you must take an online copy during the reorganization.

SHRLEVEL REFERENCE

You can run the REORG utility with SHRLEVEL REFERENCE for the LOB table, which contributes to enhanced availability in SAP environments because the number of tables with LOBs keeps growing. Often, SAP tables with LOB columns are predominantly accessed for read operations. Running REORG with SHRLEVEL REFERENCE on the LOB table space is compatible with these read operations.

Running the REORG utility of the LOB table space with SHRLEVEL REFERENCE makes LOB pages continuous and removes embedded free space. It is also important that it reclaim physical space. In DB2 V8, the REORG utility with SHRLEVEL NONE does not reclaim physical space. For SHRLEVEL REFERENCE, LOBs are unloaded to a shadow data set and physical space is reclaimed.

If you specify SHRLEVEL REFERENCE, LOG NO is required and no updates are logged during the REORG.

Table 8-1 lists the four phases of REORG table space SHRLEVEL REFERENCE on an LOB table space.

Table 8-1 Phases of REORG table space SHRLEVEL REFERENCE on LOB table space

Sequence	Phase	Phase description
1	UTILINIT	Utility initialization
2	REORGLOB	Unloads LOBs to a shadow data set
3	SWITCH	Switches access to a shadow copy of a table space or partition
4	UTILTERM	Utility cleanup

Example 8-6 shows the REORG sample job for the LOB table space with SHRLEVEL REFERENCE.

Example 8-6 REORG table space SHRLEVEL REFERENCE for LOB table sample JCL

```
//REORGLOB JOB (999,POK),'REORG_SAMPLE',CLASS=A,REGION=0M,
//     MSGCLASS=T,MSGLEVEL=(1,1),NOTIFY=&SYSUID
/*JOBPARM S=SC04
//STEP1 EXEC DSNUPROC,UID='REOGJOB.REORG',UTPROC='',SYSTEM='DB2S'
//SYSREC DD DSN=BAEK.REORG.STEP1.SYSREC,DISP=(NEW,DELETE,CATLG),
// UNIT=SYSDA,SPACE=(CYL,(1000,500),,,ROUND)
//* Sort work data sets for use by SORTDATA
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(1000,500),,,ROUND)
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(1000,500),,,ROUND)
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(1000,500),,,ROUND)
//UTPRINT DD SYSOUT=*
//SYSIN DD *
TEMPLATE SYSCOPY UNIT(SYSDA) DSN(BAEK.HEELA.T&TI..SYSCOPY1)
REORG TABLESPACE LX20X999.LDYNPDVF
SHRLEVEL REFERENCE
/*
```

Example 8-7 shows the output from a REORG running with SHRLEVEL REFERENCE for an LOB table.

Example 8-7 REORG running SHRLEVEL REFERENCE for the LOB table

```
DSNU000I    283 16:39:38.19 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID
= REOGJOB.REORG
DSNU1044I   283 16:39:38.22 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
DSNU050I    283 16:39:38.22 DSNUGUTC -  TEMPLATE SYSCOPY UNIT(SYSDA)
DSN(BAEK.HEELA.T&TI..SYSCOPY1)
DSNU1035I   283 16:39:38.23 DSNUJTDR - TEMPLATE STATEMENT PROCESSED
SUCCESSFULLY
DSNU050I    283 16:39:38.23 DSNUGUTC -  REORG TABLESPACE
LX20X999.LDYNPDVF SHRLEVEL REFERENCE
DSNU1038I   283 16:39:39.55 DSNUGDYN - DATASET ALLOCATED.
TEMPLATE=SYSCOPY
                        DDNAME=SYS00001
                        DSN=BAEK.HEELA.T203938.SYSCOPY1
DSNU1151I -D2S1 283 16:42:39.64 DSNURLOB - REORGLOB PHASE COMPLETE -
NUMBER OF RECORDS PROCESSED=50636
DSNU387I    283 16:46:03.48 DSNURSWT - SWITCH PHASE COMPLETE, ELAPSED
TIME = 00:03:23
DSNU428I    283 16:46:03.49 DSNURSWT - DB2 IMAGE COPY SUCCESSFUL FOR
TABLESPACE LX20X999.LDYNPDVF
DSNU400I    283 16:46:03.52 DSNURBID - COPY PROCESSED FOR TABLESPACE
LX20X999.LDYNPDVF
                       NUMBER OF PAGES=276896
                       AVERAGE PERCENT FREE SPACE PER PAGE =  0.00
                       PERCENT OF CHANGED PAGES =100.00
                       ELAPSED TIME=00:06:23
DSNU010I    283 16:46:03.79 DSNUGBAC - UTILITY EXECUTION COMPLETE,
HIGHEST RETURN CODE=0
REORG table space SHRLEVEL REFERENCE for the LOB table job log

Menu  Options  View  Utilities  Compilers  Help
-----------------------------------------------------------------------
DSLIST - Data Sets Matching DB2SU.DSNDBD.LX20X999.LDYNPDV Row 1 of 2
Command ===> .........................................Scroll ===> CSR

Command - Enter "/" to select action                  Tracks %Used XT
Device
-----------------------------------------------------------------------
         DB2SU.DSNDBD.LX20X999.LDYNPDVF.I0001.A001     19470    ?  10
3390
```

```
          DB2SU.DSNDBD.LX20X999.LDYNPDVF.J0001.A001     19470    ?   1
3390
****************** End of Data Set list ***************************
```

If you specify REFERENCE for an LOB table space, you must take an online image copy during the reorganization. You also cannot specify STATISTICS.

8.5 CLONE

While today there is no known use case for clone tables in SAP environments, this may change in the future. With DB2 9 for z/OS, you can reorganize cloned tables from the specified table spaces, although you cannot use the STATISTICS keyword with the reorganization of clones.

A CLONE table is identical structurally to a base table. The base and clone tables each have separate underlying VSAM data sets. They are identified by their VSAM data set instance LOG.

In DB2 9 for z/OS, you can make a table space NOT LOGGED, which is a change to using the LOG option. Table 8-2 shows the effects of REORG on the table spaces logging options.

Table 8-2 Effects of REORG on tablespace logging options and tablespace types

REORG keywords	Tablespace keywords	Tablespace types	Logging information	Tablespace status after REORG
LOG YES	LOGGED	Non-LOB	Control records and data	No pending
LOG YES	LOGGED	LOB	Control records and LOB REDO data[a]	No pending
LOG YES	NOT LOGGED	Non-LOB	LOG YES changed to LOG NO	The same as LOG NO
LOG YES	NOT LOGGED	LOB	Control information	No pending
LOG NO	LOGGED	Non-LOB	Nothing	COPY pending
LOG NO	LOGGED	LOB	Nothing	COPY pending
LOG NO	NOT LOGGED	Non-LOB	Nothing	COPY pending
LOG NO	NOT LOGGED	LOB	Nothing	COPY pending

[a] The LOB data UNDO information is never logged for LOB table spaces.

ALTER TABLESPACE with NOT LOGGED is mutually exclusive with the DATA CAPTURE CHANGES parameter of CREATE TABLE and ALTER TABLE. NOT LOGGED is not applied to the table space if any table in the table space specifies DATA CAPTURE CHANGES. Example 8-8 demonstrates the error message that you may receive if you try to run an ALTER TABLESPACE NOT LOGGED with DATA CAPTURE CHANGES.

Example 8-8 ALTER TABLESPACE NOT LOGGED and DATA CAPTURE CHANGES

```
ALTER TABLESPACE DSN00001.TESTRMAR  NOT LOGGED ;
---------+---------+---------+---------+---------+---------+---------+-
DSNT408I SQLCODE = -628, ERROR:  THE CLAUSES ARE MUTUALLY EXCLUSIVE
DSNT418I SQLSTATE   = 42613 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXIATS SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = 380 0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'0000017C'  X'00000000'  X'00000000'
X'FFFFFFFF' X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
---------+---------+---------+---------+---------+---------+---------+-
DSNE618I ROLLBACK PERFORMED, SQLCODE IS 0
```

If the tablespace has a cloned table, an ALTER TABLESPACE fails with an SQLCOLE -650 REASON CODE 19. You must drop the cloned table and rebuild it in order to alter the table space in this case (Example 8-9).

Example 8-9 ALTER TABLESPACE NOT LOGGED with CLONE

```
ALTER TABLESPACE DSN00001.TESTRMAR  NOT LOGGED ;
---------+---------+---------+---------+---------+---------+---------+-
DSNT408I SQLCODE = -650, ERROR: THE ALTER CANNOT BE EXECUTED, REASON 19
DSNT418I SQLSTATE   = 56090 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXIATS SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = 24 0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'00000018'  X'00000000'  X'00000000'
X'FFFFFFFF' X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
---------+---------+---------+---------+---------+---------+---------+-
DSNE618I ROLLBACK PERFORMED, SQLCODE IS 0

ALTER TABLE TEST_MARA DROP CLONE ;
```

You cannot reorganize NOT LOGGED table spaces with SHRLEVEL CHANGE. If you try to do so, you see the following message:

```
DSNU1152I DSNURFIT REORG SHRLEVEL CHANGE OF TABLESPACE
DSN00001.TESTRMAR IS NOT ALLOWED BECAUSE IT HAS A LOGGING ATTRIBUTE OF
NOT LOGGED.
```

As you can see in Example 8-10, the table space, which is defined as NOT LOGGED, can be reorganized with LOG YES. LOG YES is ignored with message DSNU1153I.

Example 8-10 REORG LOG YES with NOT LOGGED table space

```
DSNU1035I   285 13:43:52.52 DSNUJTDR - TEMPLATE STATEMENT PROCESSED
SUCCESSFULLY
0DSNU050I    285 13:43:52.52 DSNUGUTC -  REORG TABLESPACE
DSN00001.TESTRMAR LOG YES SHRLEVEL NONE SORTDEVT SYSDA NOSYSREC
 DSNU1153I -D2S1 285 13:43:53.03 DSNURFIT - LOG YES SPECIFIED FOR THE
NOT LOGGED TABLESPACE DSN00001.TESTRMAR WILL BE IGNORED
 DSNU231I  -D2S1 285 13:43:54.95 DSNURBDC - DICTIONARY WITH 4096
ENTRIES HAS BEEN SUCCESSFULLY BUILT FROM 514 ROWS FOR TABLE SPACE
DSN00001.TESTRMAR
```

You must specify LOG NO for REORG of an LOB table space if you specify SHRLEVEL REFERENCE.

8.6 RETRY_DELAY

The default value was 300 seconds. In DB2 9 for z/OS, if you do not specify this value, the REORG utility uses the smaller of the following two values:

- DRAIN_WAIT value x RETRY value
- DRAIN_WAIT value x 10

Whether the REORG utility uses either of these values depends on whether the RETRY value is bigger than 10.

Example 8-11 shows a sample using the RETRY_DELAY value. If DRAIN_WAIT is 30, RETRY is 4, and you do not specify RETRY_DELAY, the value might be 120 seconds.

Example 8-11 RETRY_DELAY value

```
DSNU391I    285 14:44:27.82 DSNURPTB - SORTBLD PHASE STATISTICS. NUMBER
OF INDEXES = 4
 DSNU392I    285 14:44:27.82 DSNURPTB - SORTBLD PHASE COMPLETE, ELAPSED
TIME = 00:00:03
 DSNU1122I -D2S1 285 14:44:57.84 DSNURLOG - JOB BAEKRRET PERFORMING
REORG
        WITH UTILID REOGJOB.RETRY UNABLE TO DRAIN DSN00001.TESTRMAR.
        RETRY 1 OF 4 WILL BE ATTEMPTED IN 120 SECONDS
 DSNU1122I -D2S1 285 14:47:28.73 DSNURLOG - JOB BAEKRRET PERFORMING
REORG
        WITH UTILID REOGJOB.RETRY UNABLE TO DRAIN DSN00001.TESTRMAR.
        RETRY 2 OF 4 WILL BE ATTEMPTED IN 120 SECONDS
 DSNU1122I -D2S1 285 14:49:59.53 DSNURLOG - JOB BAEKRRET PERFORMING
REORG
        WITH UTILID REOGJOB.RETRY UNABLE TO DRAIN DSN00001.TESTRMAR.
        RETRY 3 OF 4 WILL BE ATTEMPTED IN 120 SECONDS
 DSNU1122I -D2S1 285 14:52:30.39 DSNURLOG - JOB BAEKRRET PERFORMING
REORG
        WITH UTILID REOGJOB.RETRY UNABLE TO DRAIN DSN00001.TESTRMAR.
        RETRY 4 OF 4 WILL BE ATTEMPTED IN 120 SECONDS
 DSNU590I  -D2S1 285 14:55:01.25 DSNURDRN - RESOURCE NOT AVAILABLE,
REASON=X'00C200EA', ON DSN00001.TESTRMAR PROHIBITS PROCESSING
 DSNU012I    285 14:55:01.96 DSNUGBAC - UTILITY EXECUTION TERMINATED,
HIGHEST RETURN CODE=8
```

8.7 Online CHECK DATA and CHECK LOB function

You can run the CHECK DATA or the CHECK LOB utilities in DB2 9 for z/OS using the SHRLEVEL CHANGE option (ONLINE). SHRLEVEL CHANGE allows you to access the SAP objects in DB2 9 for z/OS in read and write mode while the CHECK DATA or CHECK LOB is running. You should run CHECK INDEX before you run CHECK DATA, especially if you specify DELETE YES. Running CHECK INDEX before running CHECK DATA ensures that the indexes that CHECK DATA uses are valid. When checking an auxiliary table index, CHECK INDEX verifies that each LOB is represented by an index entry and that an index entry exists for every LOB.

Important: CHECK DATA is used in conjunction with CHECK INDEX and CHECK LOB to validate the consistency of database objects. CHECK DATA only performs part of this function.

8.7.1 CHECK DATA

The CHECK DATA utility checks table spaces for violations of referential and table check constraints. Additionally CHECK DATA checks for consistency between a base table space and the corresponding LOB or XML table spaces.

We recommend that you run CHECK DATA after a conditional restart or a point-in-time recovery on all table spaces where parent and dependent tables might not be synchronized or where base tables and auxiliary tables might not be synchronized.

CHECK DATA SHRLEVEL REFERENCE optionally copies rows and deletes those rows that violate referential or table check constraints. CHECK DATA SHRLEVEL REFERENCE copies each row that violates one or more constraints to an exception table. If a row violates two or more constraints, CHECK DATA SHRLEVEL REFERENCE copies the row only once.

For SHRLEVEL CHANGE, CHECK DATA generates REPAIR statements that you can run to delete the rows. If the utility finds any violation of constraints, CHECK DATA SHRLEVEL REFERENCE places the table space that it is checking in the CHECK-pending status. CHECK DATA SHRLEVEL REFERENCE resets the CHECK-pending status if it finds no errors or if all rows that contain violations were copied to exception tables and deleted.

> **Important:** To run CHECK DATA, you should read the “Before running CHECK DATA” topic in *DB2 Version 9.1 for z/OS Utility Guide and Reference*, SC18-9855.

For a table with LOB columns, if you plan to run CHECK DATA on a base table space that contains at least one LOB column, complete the following steps prior to running CHECK DATA:

1. Run CHECK LOB on the LOB table space.
2. Run CHECK INDEX on the index on the auxiliary table to ensure the validity of the LOB table space and the index on the auxiliary table.
3. Run CHECK INDEX on the indexes on the base table space.

If the utility finds any violation of constraints, CHECK DATA SHRLEVEL REFERENCE places the table space that it is checking in the CHECK-pending status. You can specify that the CHECK-pending status is to be reset when CHECK DATA SHRLEVEL REFERENCE execution completes.

Because CHECK DATA SHRLEVEL CHANGE operates on copies of the tables, deletion of any rows can cause unpredictable results for applications. If no problems are found, then you must run REPAIR LOCATE DELETE to remove the CHECK PENDING status flag.

Similarly, CHECK DATA SHRLEVEL CHANGE cannot set CHECK PENDING because it may cause disruptions in executing applications. If no problems are found, then REPAIR SET NOCHECKPEND or REPAIR SET NOAUXWARN must be run to remove the CHECK PENDING status flag.

8.7.2 Recommendation

Create the shadows ahead of time for DB2-managed data sets, especially for the shadow data set of the logical partition of nonpartitioned secondary indexes.

Preallocating data sets for indexes

When you preallocate data sets for indexes, create the shadow data sets as follows:

- Create shadow data sets for the partition of the table space and the corresponding partition in each partitioning index and data-partitioned secondary index.
- Create a shadow data set for nonpartitioned secondary indexes.

Use the same naming scheme for these index data sets as you use for other data sets that are associated with the base index, except use J0001 instead of I0001. Also consider the following actions when you preallocate the data sets:

- Allocate the shadow data sets according to the rules for user-managed data sets.
- Define the shadow data sets as LINEAR.
- If the original table space or index space is EA-enabled, define the shadow data sets as EA-enabled.
- Allocate the shadow data sets on the volumes that are defined in the storage group for the original table space or index space.

If you specify a secondary space quantity, DB2 does not use it. Instead, DB2 uses the SECQTY value for the table space or index space. DB2 treats preallocated shadow data sets as DB2-managed data sets.

Estimating the size of pre-allocated data sets

If you have not changed the value of FREEPAGE or PCTFREE, the amount of required space for a shadow data set is comparable to the amount of required space for the original data set.

CHECK LOB

The CHECK LOB online utility on an LOB table space identifies any structural defects in the LOB table space and any invalid LOB values. Run the CHECK LOB utility in the following circumstances:

- Run the utility on an LOB table space that is in CHECK-pending (CHKP) status to identify structural defects. If no defects are found, the CHECK LOB utility turns off the CHKP status.

- Run the utility on an LOB table space that is in auxiliary-warning (AUXW) status to identify invalid LOBs. If no invalid LOBs exist, the CHECK LOB utility turns off AUXW status.
- Run the utility after a conditional restart or a point-in-time recovery on all table spaces where LOB table spaces might not be synchronized.
- Run the utility before you run the CHECK DATA utility on a table space that contains at least one LOB column.

After successful execution, CHECK LOB SHRLEVEL REFERENCE resets the CHKP and AUXW statuses. CHECK LOB SHRLEVEL CHANGE *does not* reset the CHKP and AUXW statuses.

8.7.3 CHECK INDEX online utility and parallelism

The CHECK INDEX online utility tests whether indexes are consistent with the data that they index. It issues warning messages when it finds an inconsistency.

The CHECK INDEX utility with SHRLEVEL CHANGE supports the following functionality:

- Allows you to check integrity of an index while SAP applications are using it
- Invokes data set-level FlashCopy if available to accelerate copy operation
- Runs against a shadow data set
- Provides greater concurrency with the application

Run the CHECK INDEX utility after a conditional restart or a point-in-time recovery on all table spaces whose indexes might not be consistent with the data. Also run CHECK INDEX before you run CHECK DATA, especially if you specify DELETE YES. Running CHECK INDEX before CHECK DATA ensures that the indexes that CHECK DATA uses are valid.

When checking an auxiliary table index, CHECK INDEX verifies that each LOB is represented by an index entry and that an index entry exists for every LOB. For more information about running the CHECK DATA utility on a table space that contains at least one LOB column, refer to the CHECK DATA topic in the IBM Information Management Software for z/OS Solutions information center at:

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp?topic=/com.ibm.db29.doc.ugref/cxutchk.htm

CHECK INDEX has no new syntax. Function is added to perform CHECK processing on the DocID, NodeID, and XML indexes. The PART keyword is supported for both physical and logical partitions. CHECK INDEX extracts keys from data rows and compares those to keys that exist in the indexes, reporting inconsistencies in existing error messages.

In DB2 9 for z/OS, checking indexes *in parallel* reduces the runtime of the utility and improves the availability of the SAP Objects. If you specify more than one index to check, CHECK INDEX checks the indexes in parallel unless they are constrained by available memory or the sorting of work files. Checking indexes in parallel reduces the elapsed time for a CHECK INDEX job by sorting the index keys and checking multiple indexes in parallel, rather than sequentially.

8.7.4 Online REBUILD INDEX

The REBUILD INDEX utility reconstructs indexes or index partitions from the table that they reference. In DB2 9 for z/OS, you can run the REBUILD INDEX utility in online mode (SHRLEVEL CHANGE) and access the SAP objects in change mode.

> **Attention:** Schedule REBUILD with SHRLEVEL CHANGE in the SAP environment when the rate of writing is low and transactions are short. Avoid scheduling REBUILD with SHRLEVEL CHANGE during the high dialog activities.

The REBUILD INDEX has no new syntax. Function is added to rebuild the DocID, NodeID and XML indexes. The PART keyword is supported for both physical and logical partitions.

Specifying access with SHRLEVEL

To rebuild an index or a partition of an index, the SHRLEVEL option lets you choose the data access level that you have during the rebuild.

Log processing with SHRLEVEL CHANGE

When you specify SHRLEVEL CHANGE, DB2 processes the log. This step executes iteratively. The first iteration processes the log records that accumulated during the previous iteration. The iterations continue until one of the following conditions is met:

- DB2 estimates that the time to perform the log processing in the next iteration will be less than or equal to the time that is specified by MAXRO. If this condition is met, the next iteration is the last one.
- The number of log records that the next iteration processes is not sufficiently lower than the number of log records that were processed in the previous iteration. If this condition is met, but the first two conditions are not, DB2 sends message DSNU377I to the console. DB2 continues log processing for the length of time that is specified by DELAY and then performs the action specified by LONGLOG.

Operator actions

LONGLOG specifies the action that DB2 is to perform if log processing is not occurring quickly enough. If the operator does not respond to console message DSNU377I, the LONGLOG option automatically goes into effect. You can execute the TERM UTILITY command to terminate the rebuild process. DB2 does not take the action specified in the LONGLOG phase if any one of these events occurs before the delay expires:

- TERM UTILITY command is issued.
- DB2 estimates that the time to perform the next iteration is likely to be less than or equal to the time specified on the MAXRO keyword.
- REBUILD terminates for any reason, including the deadline.

9

DB2 Connect V9.1

Starting with SAP NetWeaver ‘04, DB2 Connect has replaced ICLI to ensure enhanced connectivity between DB2 for z/OS and SAP Application Servers using Distributed Relational Database Architecture (DRDA). DB2 Connect is also used with WebAS 6.20 and DB2 for z/OS V8 or later.

In this chapter, we discuss the new features that are available with DB2 Connect V9 and the new DB2 Driver for ODBC and CLI that grants remarkable benefits to SAP products for connection with DB2 for z/OS.

In this chapter, we present the following topics:

- The new functions that are delivered in DB2 Connect Version 9
- DB2 Driver for ODBC and CLI (Thin Client)
- Benefits for SAP
- Implementing DB2 Driver for ODBC and CLI in SAP

9.1 DB2 Connect V9 server

DB2 Connect V9.1 for Linux, UNIX, and Microsoft Windows has been available since June 2006. The main new features include:

- Client support for trusted connections

  A client can create trusted connections using ODBC, XA, or new Java methods to database servers (currently only DB2 for z/OS) that support trusted contexts. The user name of the client can then be switched without the database server fully authenticating the new name.

- BINARY, VARBINARY, and DECFLOAT data type support

  DB2 for z/OS now supports data types BINARY, VARBINARY, and DECFLOAT. Support for these data types has been added to DB2 CLI and DB2 .NET Data Provider. Your applications that use DB2 Connect to assess DB2 for z/OS can use DB2 CLI and DB2 .NET Data Provider to take advantage of the new data types.

- IPv6 communication protocol support

  It is now possible to connect to servers using either IPv4 or IPv6 addresses.

- Two phase commit for multi-vendor data sources when using IBM WebSphere® Federation Server

  DB2 Connect applications can use the WebSphere Federation Server to reach data sources that are offered by many IBM and non-IBM vendors.

- Includes the DB2 Driver for ODBC and CLI and DB2 Driver for JDBC and SQLJ

The DB2 Connect 9 server contains all of the available functions. It requires a complete installation of the product. SAP NetWeaver technology recently started to support this configuration to help alleviate high storage and complex setup concerns. Also, many of the DB2 Connect features are not actually used by SAP.

9.2 DB2 Connect V9 application driver

DB2 Connect 9 introduces a new stand-alone DB2 Driver for ODBC and CLI. It contains the following features:

- Can be installed and used to run ODBC and CLI applications without a DB2 client.

  When the driver is installed without a DB2 client, it provides runtime support for ODBC and CLI. It now also provides connectivity for ODBC and CLI applications.

- Supports trusted connections with DB2 on z/OS
- Can be installed on a machine that already has a DB2 client installed
- Coexistence of multiple installations on the same machine
- Allows maintenance without operation interruption
- Requires less storage than the server
- Can reside on an Network File System (NFS) mounted file system

Most of these characteristics make DB2 Driver for ODBC and CLI suitable for the ABAP stack of SAP NetWeaver products. For the Java stack, use the DB2 Driver for JDBC and SQLJ. At the moment, these drivers have not been certified by SAP, but they will be in the near future.

The new Thin Client supports connections to both DB2 for z/OS V8 and V9 and is distributed through SAP Marketplace.

Figure 9-1 shows the compatibilities and the migration possibilities of different connection solutions (ICLI, DB2 Connect V8, and DB2 Connect V9 Thin Client) against the SAP versions.

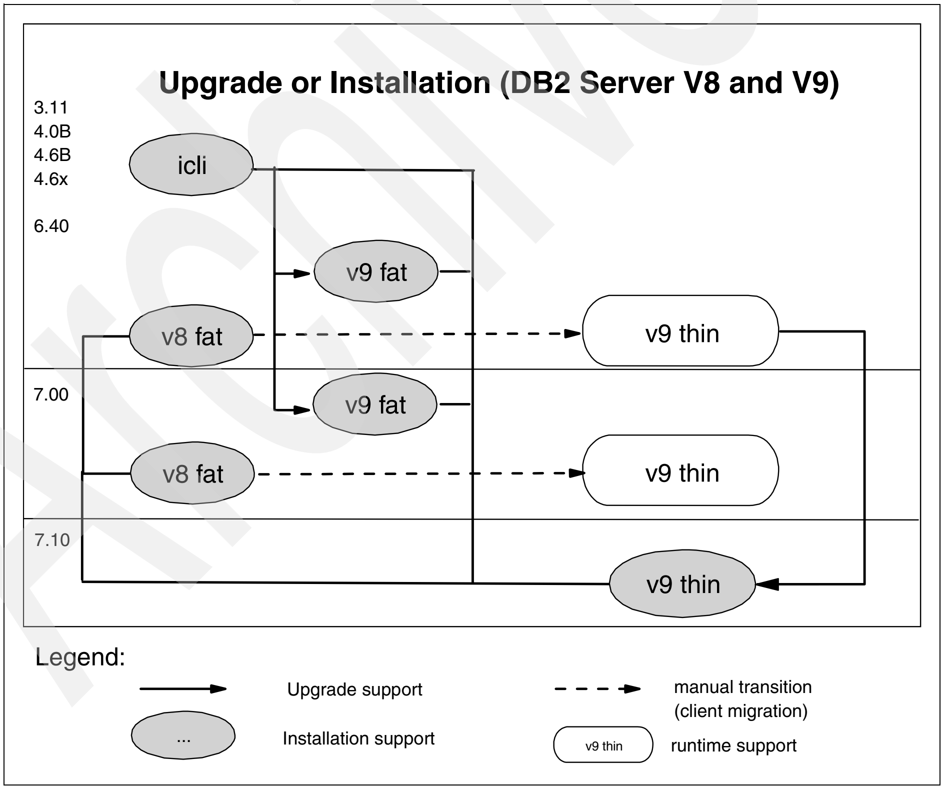


Figure 9-1 Compatibilities and migration possibilities

9.3 Implementing Thin Client for SAP

In this section, we explain how to set up, manage, and exploit DB2 Driver for ODBC and CLI installation in an SAP environment.

9.3.1 Advantages of Thin Client for SAP

There are several advantages in using Thin Client for SAP products instead of the full version of DB2 Connect V9. Installation is easy since it only consists of uncompressing and unpacking a tar file. The Thin Client does not provide an installation program.

When installed, the CLI driver takes about 30 MB of disk space, while the complete DB2 Connect V9 installation takes about 470 MB of space. In UNIX installations, you can access the CLI driver via the NFS. Consequently, in the case of an SAP system with more than one UNIX Application Server, you can install the CLI driver only in one application server and NFS mount it by the others.

Another advantage of the CLI driver is that you can maintain it without interrupting business operations. You can install a new fixpax or a new version of Thin Client without having to stop the Application Server. However, in some circumstances, a bind may be required. The SAP work process can pick up the new CLI driver version by simply disconnecting and reconnecting to DB2.

It is also possible to have different CLI driver versions installed on the same server for rolling maintenance.

9.3.2 Installing Thin Client

We installed and tested the CLI driver on our AIX® 5.3, Maintenance Level 4, SAP Application Server. The SAP SID name was DB9. We used SAP kernel version 640, with patch level 150. Dbsl library dbdb2slib.o was also at level 150.

The DB2 CLI driver version that we had (V9.1.0.1 Fix Pack 1) was in form of a compressed tar file. First we uncompressed and untarred the file. As a result, the product was installed in the /mnt/CLI/clidriver directory.

To make the CLI driver usable by AIX SAP administrator db9adm, we added the lib subdirectory to its LIBPATH environment variable:

```
setenv LIBPATH ${LIBPATH}:/mnt/CLI/clidriver/lib
```

We verified the version of the installed system by running the **db2level** command from the bin directory (see Example 9-1).

Example 9-1 Running db2level to verify the version of the system

```
# cd /mnt/CLI/clidriver/bin
# ./db2level
DB21085I  Instance "DB2" uses "64" bits and DB2 code release "SQL09011"
with level identifier "01020107".
Informational tokens are "DB2 v9.1.0.1", "s061008", "U809676", and Fix
Pack "1".
Product is installed at "..".
```

Then we had to bind the SAP application to DB2. Since db2radm was not able yet to perform the bind using the new CLI driver, we used a DB2 Connect V9 installation and instance. In the future, when the DB2 Driver for ODBC and CLI is certified by SAP, db2tadm will use the binder of the DB2 Driver for JDBC and SQLJ to bind the packages of the CLI driver.

We ran **dbdb2pwd** to create the password file that is contained in the SAP global directory. Finally, we created the connect.ini file in the SAP global directory (see Example 9-2).

Example 9-2 Creating the connect.ini file

```
File connect.ini:
[GROUPS]
  db9group=ewlmaix2
[db9group]
  CON1=db9sc04
  CON2=db9sc47
[db9sc04]
  SSID=D2S1
  HOST=wtsc04.itso.ibm.com
  PORT=38290
  LOCATION=DB2S
[db9sc47]
  SSID=D2S2
  HOST=wtsc47.itso.ibm.com
  PORT=38290
  LOCATION=DB2S
[COMMON]
  RETRY_CNT=3
  SLEEP_TIME=5
```

We then performed the bind, by running the following command from the db9adm user:

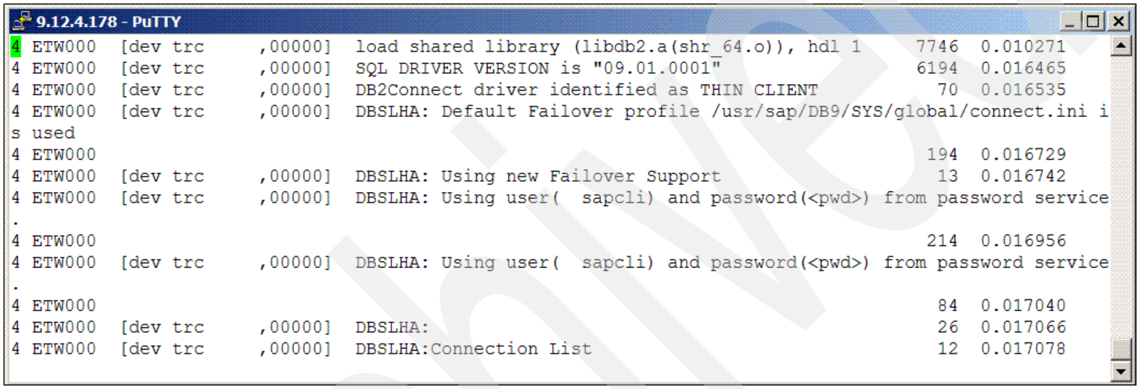
```
db2radm -m db2i -C SAP09.01.0001 -u sapcli -p <pwd>
```

We specified the Collection ID because DB2 Connect V9 uses Collection ID SAP09.01.0000 as default for bind, which is not accepted for connection with the CLI driver.

Then we checked the DB2 connection by running the following command:

```
R3trans -x
```

Figure 9-2 shows an extract of the output file trans.log.



```
4 ETW000  [dev trc     ,00000]  load shared library (libdb2.a(shr_64.o)), hdl 1     7746  0.010271
4 ETW000  [dev trc     ,00000]  SQL DRIVER VERSION is "09.01.0001"                  6194  0.016465
4 ETW000  [dev trc     ,00000]  DB2Connect driver identified as THIN CLIENT           70  0.016535
4 ETW000  [dev trc     ,00000]  DBSLHA: Default Failover profile /usr/sap/DB9/SYS/global/connect.ini i
s used
4 ETW000                                                                             194  0.016729
4 ETW000  [dev trc     ,00000]  DBSLHA: Using new Failover Support                    13  0.016742
4 ETW000  [dev trc     ,00000]  DBSLHA: Using user(  sapcli) and password(<pwd>) from password service
.
4 ETW000                                                                             214  0.016956
4 ETW000  [dev trc     ,00000]  DBSLHA: Using user(  sapcli) and password(<pwd>) from password service
.
4 ETW000                                                                              84  0.017040
4 ETW000  [dev trc     ,00000]  DBSLHA:                                               26  0.017066
4 ETW000  [dev trc     ,00000]  DBSLHA:Connection List                                12  0.017078
```

Figure 9-2 Output from the trans.log file

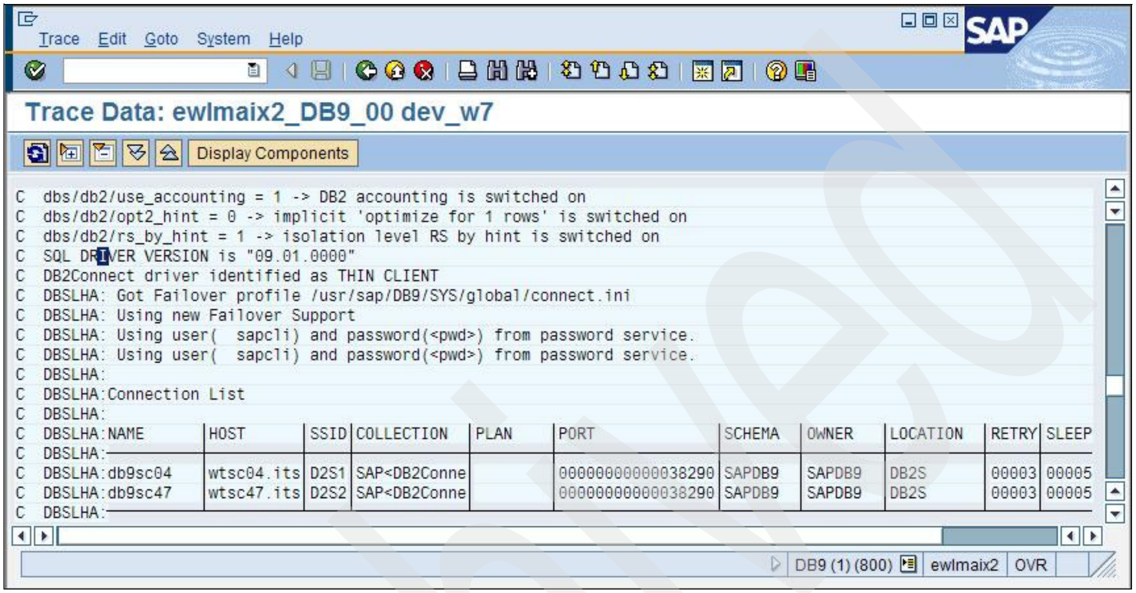
We then started SAP successfully.

> **Important:** If the CLI driver is deployed on an NFS, it uses the locking function of the NFS. If the NFS mount resides on z/OS UNIX System Services (USS), the network lock manager must be started. You can accomplish this by specifying the NLM parameter in the attribute file of the NFS.

9.3.3 Thin Client update

Thin Client update is quite simple too. SAP work processes keep using the same CLI driver that they loaded at SAP startup until they are restarted. At this time, they load the driver again from disk. If we want, we can override the CLI driver directory with another version of the product without any negative impact on the running system.

After the copy, all of the work processes were still running the previous CLI driver version. To verify that the upgrade ran successfully, we manually restarted one work process from transaction SM50 and compared its log file before and after the upgrade. Figure 9-3 shows the trace file before the upgrade.

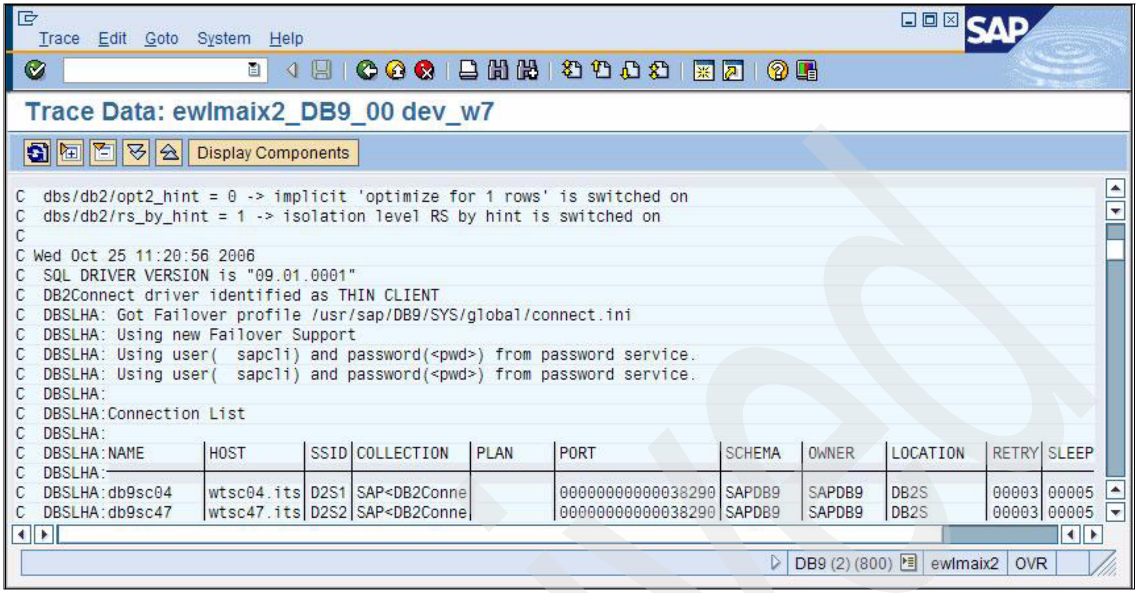


Trace Data: ewlmaix2_DB9_00 dev_w7

```
C  dbs/db2/use_accounting = 1 -> DB2 accounting is switched on
C  dbs/db2/opt2_hint = 0 -> implicit 'optimize for 1 rows' is switched on
C  dbs/db2/rs_by_hint = 1 -> isolation level RS by hint is switched on
C  SQL DRIVER VERSION is "09.01.0000"
C  DB2Connect driver identified as THIN CLIENT
C  DBSLHA: Got Failover profile /usr/sap/DB9/SYS/global/connect.ini
C  DBSLHA: Using new Failover Support
C  DBSLHA: Using user(  sapcli) and password(<pwd>) from password service.
C  DBSLHA: Using user(  sapcli) and password(<pwd>) from password service.
C  DBSLHA:
C  DBSLHA:Connection List
C  DBSLHA:
C  DBSLHA:NAME       |HOST       |SSID|COLLECTION  |PLAN    |PORT              |SCHEMA |OWNER  |LOCATION |RETRY|SLEEP
C  DBSLHA:-----------------------------------------------------------------------------------------------------------
C  DBSLHA:db9sc04    |wtsc04.its |D2S1|SAP<DB2Conne|        |00000000000038290 |SAPDB9 |SAPDB9 |DB2S     |00003|00005
C  DBSLHA:db9sc47    |wtsc47.its |D2S2|SAP<DB2Conne|        |00000000000038290 |SAPDB9 |SAPDB9 |DB2S     |00003|00005
C  DBSLHA:-----------------------------------------------------------------------------------------------------------
```

Figure 9-3 Trace file before the upgrade (version 09.01.0000 is shown)

Figure 9-4 shows the trace file after the upgrade.



```
C  dbs/db2/opt2_hint = 0 -> implicit 'optimize for 1 rows' is switched on
C  dbs/db2/rs_by_hint = 1 -> isolation level RS by hint is switched on
C
C Wed Oct 25 11:20:56 2006
C  SQL DRIVER VERSION is "09.01.0001"
C  DB2Connect driver identified as THIN CLIENT
C  DBSLHA: Got Failover profile /usr/sap/DB9/SYS/global/connect.ini
C  DBSLHA: Using new Failover Support
C  DBSLHA: Using user(  sapcli) and password(<pwd>) from password service.
C  DBSLHA: Using user(  sapcli) and password(<pwd>) from password service.
C  DBSLHA:
C  DBSLHA:Connection List
C  DBSLHA:
C  DBSLHA:NAME        |HOST       |SSID|COLLECTION  |PLAN    |PORT               |SCHEMA  |OWNER   |LOCATION  |RETRY|SLEEP
C  DBSLHA:-----------------------------------------------------------------------------------------------------------------
C  DBSLHA:db9sc04     |wtsc04.its |D2S1|SAP<DB2Conne|        |0000000000038290   |SAPDB9  |SAPDB9  |DB2S      |00003|00005
C  DBSLHA:db9sc47     |wtsc47.its |D2S2|SAP<DB2Conne|        |0000000000038290   |SAPDB9  |SAPDB9  |DB2S      |00003|00005
```

Figure 9-4 Trace file after the upgrade (version 09.01.0001 is shown)

In order to ensure that all work processes have loaded the new CLI driver version and to operate SAP with this new version, the SAP Application Server should be restarted.

Using transaction RZ11, an alternative way to do this is to change the dynamic SAP instance parameter rdisp/wp_auto_restart (Automatic work process restart time) to a value equivalent, for example, to 5 minutes. In this way, all instance workprocesses will restart in 5 minutes and load the new Thin Client version.

9.3.4 Security features

As mentioned before, the CLI driver supports trusted connections with DB2 on z/OS. *Trusted connections* are special connections that allow several users to use the same connection (user switching). A trusted connection relies on a DB2 trusted context, where a valid DB2 user, protocol attributes, allowed IP addresses of the connection clients, and a list of authorized users are specified. These users are then authorized to use the connection without DB2 authentication.

While SAP does not yet employ user switching or trusted connections, the possibility to limit access to SAP Database to specific servers or DBA workstations looks like an important security feature. This adds physical security to the standard user ID and password mechanism. A stolen valid user ID and password combination do not allow access to SAP data from an unauthorized server or workstation, whose physical access should be protected.

9.3.5 LOB performance improvements

There are many cases where an application needs to select a large object value and operate on pieces of it. However, the application does not need or want the entire value to be transferred from the database server into application memory. In these cases, the application can reference an individual LOB value via a large object locator (LOB locator). SAP is such an application. It uses LOB data types for tables that contain source code, such as DYNPSOURCE, DYNPLOAD, REPOSRC, and REPOLOAD.

An *LOB locator* is a token value, defined as type SQLINTEGER, that allows for efficient random access of a large object. When an LOB locator is used, the server performs the query. Instead of placing the value of the LOB column in the result set, it updates the LOB locator with an integer that corresponds to the value of the LOB. When the application later requests the result, the application then passes the locator to the server and the server returns the LOB result.

A LOB locator is not stored in the database. It refers to an LOB value during a transaction and does not persist beyond the transaction in which it was created. A simple token value is created to reference a single large object value, not a column in a row. There is no operation that can be performed on a locator that would have an effect on the original LOB value that is stored in the row.

The processing of large objects (LOBs), particularly from a distributed perspective, has been optimized in DB2 V9 for the retrieval of larger amounts of data. Many applications effectively use locators to retrieve LOB data regardless of the size of the data that is being retrieved. This mechanism incurs a separate network flow to get the length of the data to be returned, so that the requester can determine the proper offset and length for SUBSTR operations on the data to avoid any unnecessary blank padding of the value. For small LOB data, returning the LOB value directly instead of using a locator is more efficient.

SAP makes more and more use of LOBs, especially in the Java stack. SAP LOB objects are contained in specific fields of some tables. These fields can contain very large objects as well as small objects. For this reason, the access strategy has been optimized to take into account the size of the LOB object to retrieve.

The new DRDA LOB flow optimization, available with DB2 Connect V9 and the CLI driver, divide the LOB into three categories, which correspond to different access and data transfer technology.

In small-sized LOBs, data is treated like a normal VARCHAR type and transferred as part of the row, together with any other type of data. In medium-sized LOBs, LOB data is placed in a specific data block (extended query blocks), separated from non-LOB data. Large LOBs use locators that are transmitted in a data query block together with the other data.

You can find more detailed information about LOBs in general and particularly in SAP environments in *LOBs with DB2 for z/OS: Stronger and Faster*, SG24-7270.

10

SAP with System z Integrated Information Processor

IBM has announced the IBM System z9 Advantage for SAP Applications. It unites many recent enhancements to the mainframe environment to deliver the legendary availability, security, and manageability features of the System z9 mainframe. It also offers significant improvements to price/performance options for data serving environments of SAP applications.

Built on years of experience with important SAP solutions and with close collaboration between SAP and IBM, IBM System z9 Advantage for SAP Applications begins with the System z9 Enterprise Class or System z9 Business Class mainframes. These systems are optimized for large scale data and transaction serving and offer reliability and security features for the critical SAP database server. They also leverage the IBM System z9 Integrated Information Processor (zIIP), which may help improve resource optimization and manage software costs for eligible workloads, including those from SAP or other business applications.

In this chapter, we discuss the following topics:

- Description and benefits of zIIP
- How to implement zIIP
- How to monitor the zIIP offloads
- zIIP measurements

10.1 Description and benefits of zIIP

For z/OS V1R8 and later (also available in z/OS V1.6 and z/OS V1.7 with APARs), IBM System z9 Enterprise Class (z9 EC) or IBM System z9 Business Class (z9 BC) servers support integrated information processors (zIIP), which is a new type of processor for a dedicated workload. Conceptually similar to System z Application Assist Processors (zAAP), using zIIPs allows the offload of certain workloads, for example, selected DB2 tasks, from CPUs to zIIPs. This can help free capacity on general purpose processors, which are then available for reallocation to other System z workloads.

With the availability of zIIPs, beginning with DB2 V8, eligible work can be sent to z/OS and zOS.e and be offloaded to zIIPs. Thus, using zIIPs helps to optimize resource usage, contributes to cost-effective System z exploitation, and enhances the role of the mainframe as the data hub of the enterprise.

10.1.1 zIIP history and overview

Table 10-1 shows several specialty engines. In 2006, IBM introduced another speciality engine, the zIIP.

Table 10-1 Technology evolution with mainframe speciality engines

GA year	1997	2001	2004	2006
Engine	Internal Coupling Facility (ICF)	Integrated Facility for Linux (IFL)	System z9 Application Assist Processor (zAAP)	System z9 Integrated Information Processor (zIIP)
Description	Centralized data sharing across mainframes	Support for new workloads and open standards	Incorporation of Java into existing mainframe solutions	Designed to help improve resource optimization for eligible workloads within the enterprise

The zIIP specialty engine helps to improve resource optimization and lower the cost of ownership for eligible data serving workloads. z/OS manages and directs work between standard central processors (CPs) and zIIPs, and no changes are anticipated to DB2 for z/OS applications.

The zIIP is designed so that a program can work with z/OS to have all or a portion of its service request block (SRB) enclave work directed to the zIIP.

10.1.2 SRB enclaves

z/OS dispatches work in either task control block (TCB) mode or supervisor request block (SRB) mode. DB2 parallel tasks use SRB mode and are assigned the same importance as the originating address space.

Preemptable enclaves are used to do the work on behalf of the originating TCB or SRB address space. Enclaves are grouped by common characteristics and service requests. Since they are preemptable, the z/OS dispatcher (and Workload Manager (WLM)) can interrupt these tasks for more important ones, for example to manage a transaction end-to end.

There are two types of preemptable SRBs: client SRBs and enclave SRBs. We focus on an enclave SRB that is scheduled into a target address space but executes work on behalf of an enclave. Dispatching controls are derived from the enclave, including the major dispatch priority. CPU time consumed by each SRB is accumulated back to the enclave. It is reported as an enclave-related CPU service in SMF type 30 records for the address space that created the enclave.

All page faults taken by SRBs in the enclave are interpreted as cross-memory page faults on the target address space. Thus page faults incurred by enclave SRBs manifest themselves as cross-memory paging delay samples even when the SRBs themselves do not run in cross-memory mode. If an enclave SRB makes an I/O request, the associated I/O service is reflected in the home address space that is current at the time of the request.

If the DB2 for z/OS V8 and DB2 9 for z/OS request is coming over a distributed connection (that is Distributed Relational Database Architecture (DRDA) over TCP/IP), then most of the work is executed in enclave SRBs. If the request is coming over a local or native connection, then the work is dispatched between TCBs, client SRBs, and enclave SRBs. Star schema parallel queries and some utility index maintenance now use enclave SRBs.

Regarding the zIIP, only the enclave SRB work, not the client SRB work, non-preemptable SRB work, nor TCB work, is eligible to be redirected to the zIIP. DB2 for z/OS V8 and DB2 9 for z/OS know how its work is dispatched and directs z/OS 1.6 or later to dispatch (redirect) a portion of the eligible work to the zIIP.

10.1.3 Prerequisites for using zIIP

To use zIIPs, the minimum *software requirements* are:

- z/OS V1R6 (5694-A01) or later, or z/OS.e V1R6 (5655-G52) or later
- DB2 V8 (5675-DB2) with zIIP-enabling APARs installed

Starting with z/OS V1R8 and DB2 9 for z/OS, zIIP support is integrated. For systems that run z/OS V1R6, z/OS V1R7, and DB2 V8, the zIIP Web deliverable with the appropriate service APARs must be installed.

To use zIIPs, the *hardware requirements* are:

- IBM System z9 EC with IBM System z9 Integrated Processor Feature Code 7815
- IBM System z9 BC with IBM System z9 Integrated Processor Feature Code 7868

10.1.4 Optimizing SAP applications with zIIP

The following types of DB2 for z/OS workloads can benefit from zIIP:

- SAP OLTP exploiting DRDA using DB2 Connect
- Business Intelligence (BI) batch direct attached
- Complex SAP BW Parallel Queries exploiting DRDA (star schema parallel queries and all other parallel queries)
- Some DB2 utilities

  A portion of the utility functions that are used to maintain index maintenances can be directed to the zIIP, such as LOAD, REORG, and REBUILD INDEX. Figure 10-1 illustrates a simulation of using zIIP offload for utilities.

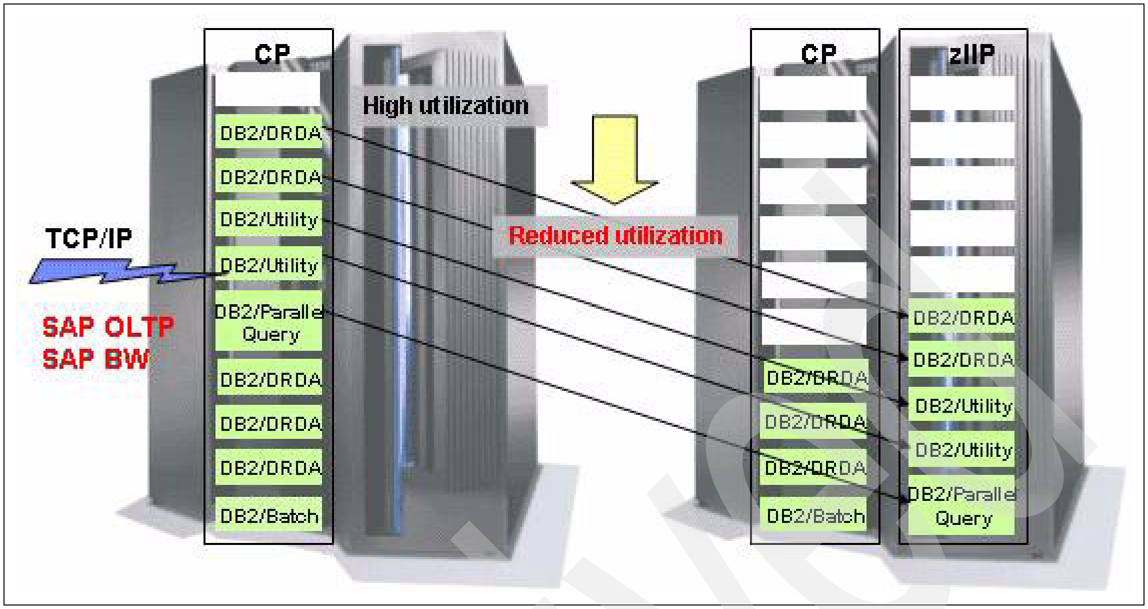


Figure 10-1 zIIP offload simulation (for illustrative purpose only)

DRDA

Database workloads such as CICS®, IMS™, stored procedures, and Batch have become increasingly efficient and cost effective on the mainframe. Today customers are looking to further leverage their data on the mainframe and are turning to the mainframe more and more frequently for new application workloads. Application workloads, such as Enterprise Resource Planning (ERP), Customer Relationship Management (CRM), and business intelligence often use DB2 as a database server.

Added system costs for network access to DB2 results in hardware and software charges that are substantially higher for remote applications, when compared to the same workload running as a local batch application. The zIIP is aimed at helping customers more cost effectively run these database serving applications on the mainframe. Our objective with the zIIP is to help bring the costs of network access to DB2 more closely in line with the costs of running similar workloads under CICS, IMS, or Batch on the mainframe.

Data warehousing

Star schema query processing is also eligible for the zIIP, with a significant percentage of this work eligible to be redirected to zIIP. The possible benefit to a data warehousing application may vary significantly depending on the details of that application. Star schema queries driven by a batch program will likely see a lower benefit than those driven by DRDA, while longer running queries will likely see a higher benefit.

> **Star join and star schema:** A *star join* is a special join technique that DB2 uses to efficiently join tables that form a star schema. A *star schema* is a logical database design that is included in decision support applications. A star schema is comprised of a fact table and a number of dimension tables that are connected to it. A dimension table contains several values that are given an ID, which is used in the fact table instead of all the values.

Utilities

Portions of DB2 Utility (LOAD, REORG, & REBUILD) processing that are related to index maintenance are also eligible for executing on zIIPs. It is expected that the percentage will vary between 10% and 65%, depending on:

- The amount of indexes that are defined on the table
- The number of partitions in the table, if data compression is being used
- Possibly other factors

The CPU time offloaded to the zIIP is expected to be on the lower end of the range for tables with fewer indexes, fewer partitions, or if compression is being used and at the higher end of the range for tables with many indexes or many partitions. Again, the percentages represent work that is eligible to be redirected to zIIPs. The actual percent redirected is less if there is insufficient zIIP capacity to process all the zIIP eligible work.

10.2 Implementing the speciality engine zIIP

zIIP is easy to implement. You acquire it, define it, and activate it, and then begin using it.

10.2.1 Before using zIIP

The SYS1.PARMLIB member IEAOPTxx provides statement PROJECTCPU. Specifying the PROJECTCPU parameter allows you to project zIIP (and zAAP) consumption when a zIIP (or zAAP) processor is not yet defined to the configuration. RMF™ and SMF show the potential calculated zIIP time, so that an accurate zIIP projection can be made. The PROJECTCPU option can be used while running the target workload, after all software is installed that enables hardware sizing data to be produced.

Refer to *OS/390 V2R10.0 MVS Initialization and Tuning Reference*, SC28-1752, for more information about the parmlib member IEAOPTxx. The SMF type 30 record (IFASMFR3) includes zIIP consumption fields.

10.2.2 Defining zIIP

You can define a logical partition (LPAR) to use one or more zIIPs or zAAPs with z/OS 1.6 or later with either of the following combinations.

- One or more dedicated general purpose CPs and one or more dedicated zIIPs/zAAPs
- One or more shared general purpose CPs and one or more shared zIIPs/zAAPs

You can define zIIP on the Hardware Management Console (HMC) and no more action is necessary.

After you define zIIP, you may use the DISPLAY M=CPU command to show if a zIIP processor is defined in the configuration. In the D M=CPU command output, zIIPs are represented by the letter “I”. A zIIP processor is considered to be defined in the offline or reserved state, as well as in the online state. See Example 10-1 for sample output of this command.

Example 10-1 Sample output showing that the ZIIP processor is defined

```
D M=CPU
IEE174I 13.40.53 DISPLAY M 274
PROCESSOR STATUS
ID  CPU                  SERIAL
00  +                     09991E2094
01  +                     09991E2094
02  +A                    09991E2094
03  +A                    09991E2094
04  +I                    09991E2094
05  +I                    09991E2094
06  -
07  -

CPC ND = 002094.S18.IBM.02.00000002991E
CPC SI = 2094.710.IBM.02.000000000002991E
CPC ID = 00
CPC NAME = SCZP101
LP NAME = A09       LP ID =  9
CSS ID  = 0
MIF ID  = 9
```

```
+ ONLINE    - OFFLINE    . DOES NOT EXIST    W WLM-MANAGED
N NOT AVAILABLE

A        APPLICATION ASSIST PROCESSOR (zAAP)
I        INTEGRATED INFORMATION PROCESSOR (zIIP)
CPC ND  CENTRAL PROCESSING COMPLEX NODE DESCRIPTOR
CPC SI  SYSTEM INFORMATION FROM STSI INSTRUCTION
CPC ID  CENTRAL PROCESSING COMPLEX IDENTIFIER
CPC NAME CENTRAL PROCESSING COMPLEX NAME
LP NAME  LOGICAL PARTITION NAME
LP ID    LOGICAL PARTITION IDENTIFIER
CSS ID   CHANNEL SUBSYSTEM IDENTIFIER
MIF ID   MULTIPLE IMAGE FACILITY IMAGE IDENTIFIER
```

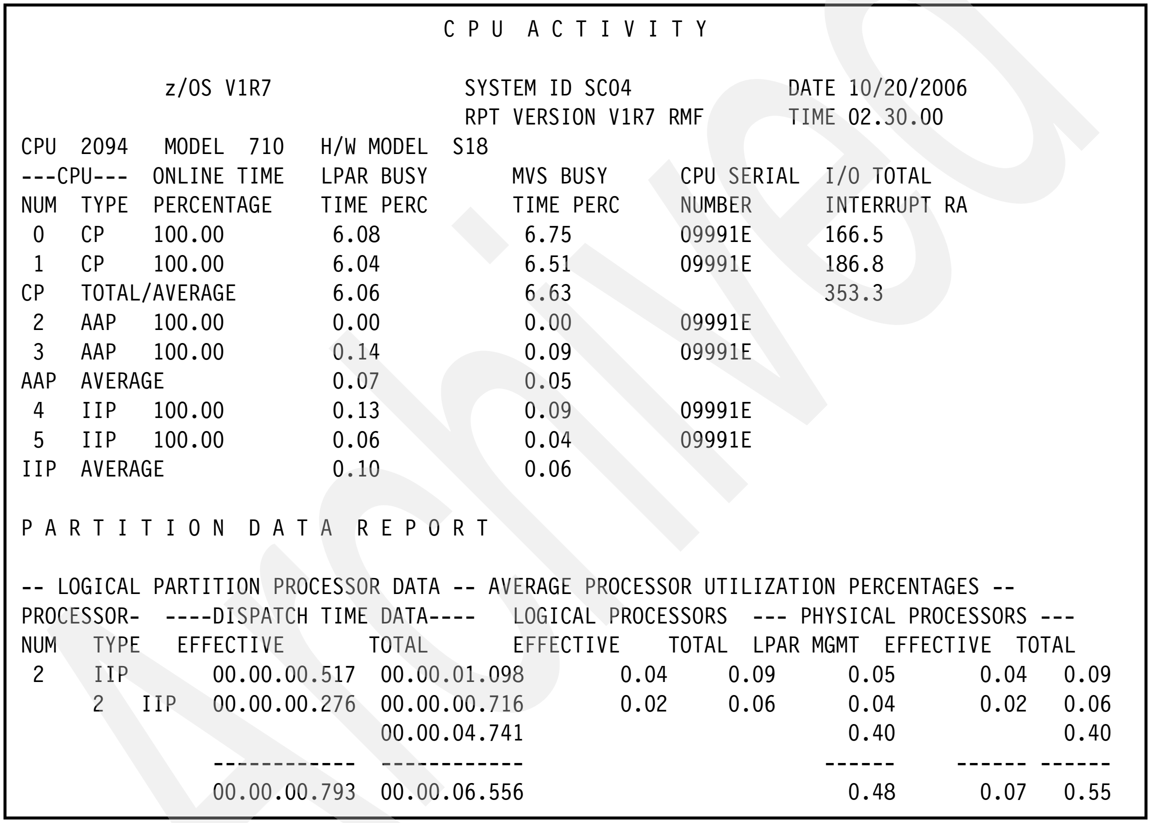
The following requirements are for zIIPs and standard CPs:

- The number of zIIPs must not exceed the number of standard CPs on a server.
- You must have at least one standard CP defined for a partition. The z/OS system needs at least one standard CP online at all times.
- You can set the number of zIIPs for an LPAR, and each processor pool (for example, CPs and zIIPs) can be assigned a unique weight when the processors are being shared.
- There is hard capping for zIIPs, but there is no support of defined capacity, which is the WLM support for a four-hour rolling average). The existing single set of PR/SM™ LPAR processor weights (INITIAL, MIN, MAX) is applied independently to the shared standard CPs; capping applies only to shared standard CPs that are configured to the LPAR. If you use WLM Weight Management for the LPAR with SCRT, then z/OS WLM manages shared standard CPs as it does today, but not the zIIPs.
- zIIPs do not participate in Intelligent Resource Director nor do zIIPs participate in dynamic share management or the number of logical zIIPs online.

10.3 Checking the zIIP offloads

You can use an RMF report for CPU activity. As shown in Figure 10-2, this report gives you the following information:

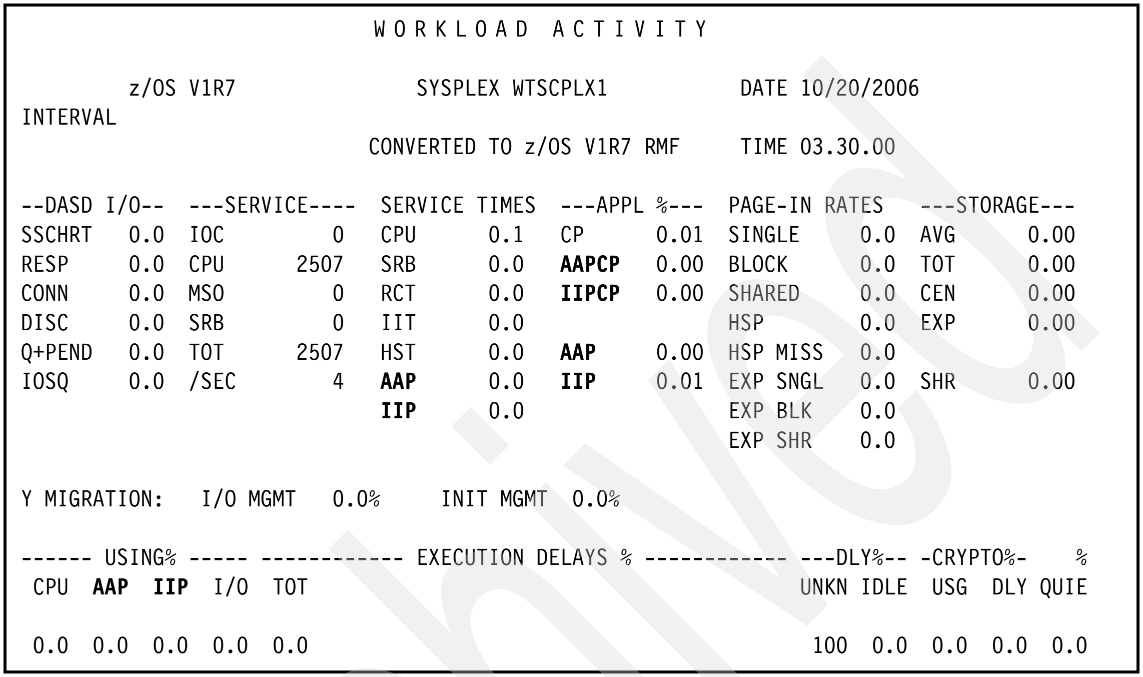
- CPU NUM/TYPE: Specifies the logical CPU number and type (CP, AAP, or IIP)
- IIP AVERAGE: The average value for zIIPs; is only visible if zIIPs are configured; is found in the RMF Postprocessor CPU report



```
                                   C P U  A C T I V I T Y

            z/OS V1R7                SYSTEM ID SC04             DATE 10/20/2006
                                     RPT VERSION V1R7 RMF       TIME 02.30.00
CPU  2094   MODEL  710   H/W MODEL  S18
---CPU---  ONLINE TIME   LPAR BUSY       MVS BUSY      CPU SERIAL  I/O TOTAL
NUM  TYPE  PERCENTAGE    TIME PERC       TIME PERC     NUMBER      INTERRUPT RA
 0   CP    100.00         6.08            6.75         09991E      166.5
 1   CP    100.00         6.04            6.51         09991E      186.8
CP   TOTAL/AVERAGE        6.06            6.63                     353.3
 2   AAP   100.00         0.00            0.00         09991E
 3   AAP   100.00         0.14            0.09         09991E
AAP  AVERAGE              0.07            0.05
 4   IIP   100.00         0.13            0.09         09991E
 5   IIP   100.00         0.06            0.04         09991E
IIP  AVERAGE              0.10            0.06

P A R T I T I O N  D A T A  R E P O R T

-- LOGICAL PARTITION PROCESSOR DATA -- AVERAGE PROCESSOR UTILIZATION PERCENTAGES --
PROCESSOR-  ----DISPATCH TIME DATA----   LOGICAL PROCESSORS  --- PHYSICAL PROCESSORS ---
NUM   TYPE   EFFECTIVE       TOTAL       EFFECTIVE    TOTAL  LPAR MGMT  EFFECTIVE  TOTAL
 2    IIP       00.00.00.517  00.00.01.098       0.04     0.09      0.05       0.04   0.09
      2   IIP   00.00.00.276  00.00.00.716       0.02     0.06      0.04       0.02   0.06
                              00.00.04.741                          0.40              0.40
                ------------  ------------                     ------     ------ ------
                00.00.00.793  00.00.06.556                          0.48       0.07   0.55
```

Figure 10-2 RMF CPU Activity report

You can also create a report using the REPORTS(CPU) option and SMF RECORD 70 - SUBTYPE 1. Figure 10-3 shows a sample RMF Postprocessor Workload Activity report.



```
                             W O R K L O A D   A C T I V I T Y

         z/OS V1R7               SYSPLEX WTSCPLX1           DATE 10/20/2006
INTERVAL
                             CONVERTED TO z/OS V1R7 RMF     TIME 03.30.00

--DASD I/O--  ---SERVICE----  SERVICE TIMES  ---APPL %---  PAGE-IN RATES   ---STORAGE---
SSCHRT   0.0  IOC         0   CPU      0.1   CP      0.01  SINGLE     0.0  AVG      0.00
RESP     0.0  CPU      2507   SRB      0.0   AAPCP   0.00  BLOCK      0.0  TOT      0.00
CONN     0.0  MSO         0   RCT      0.0   IIPCP   0.00  SHARED     0.0  CEN      0.00
DISC     0.0  SRB         0   IIT      0.0                 HSP        0.0  EXP      0.00
Q+PEND   0.0  TOT      2507   HST      0.0   AAP     0.00  HSP MISS   0.0
IOSQ     0.0  /SEC        4   AAP      0.0   IIP     0.01  EXP SNGL   0.0  SHR      0.00
                              IIP      0.0                 EXP BLK    0.0
                                                           EXP SHR    0.0

Y MIGRATION:   I/O MGMT   0.0%     INIT MGMT  0.0%

------ USING% ----- ------------ EXECUTION DELAYS % ------------ ---DLY%-- -CRYPTO%-    %
 CPU  AAP  IIP  I/O  TOT                                         UNKN IDLE  USG  DLY QUIE

 0.0  0.0  0.0  0.0  0.0                                          100  0.0  0.0  0.0  0.0
```

Figure 10-3 RMF WLM report

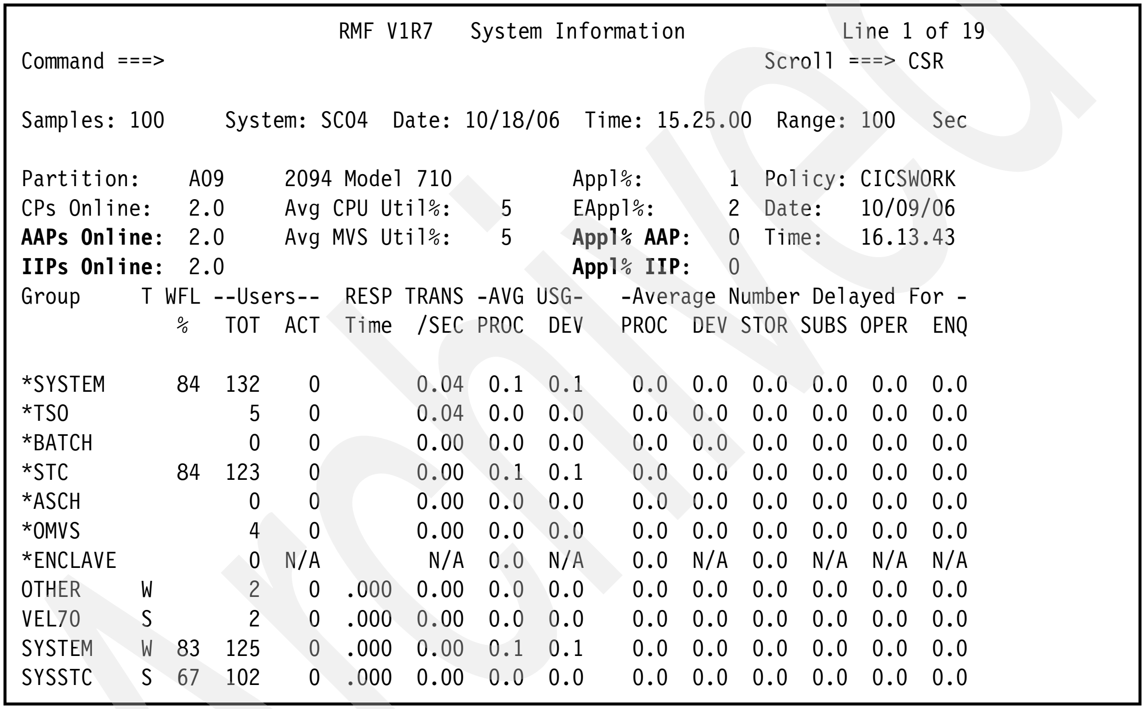
Another report that can be created is for service times. Service times can give you information about:

- **CPU**: Task and preemptible-class SRB (enclave) time in seconds consumed on standard CPs and special purpose processors
- **IIP**: zIIP service time in seconds
- **APPL%**: Percentage of CPU time used by transactions running on the different processor types
- **CP**: Percentage of CPU time used by transactions running on standard CPs in the service or report class period; is calculated using this equation:

```
              CPU + SRB + RCT + IIT + HST -AAP - IIP
APPL% CP = -------------------------------------- x 100
                          Interval length
```

- **IIPCP**: Percentage of CPU time used by zIIP eligible transactions running on standard CPs; a subset of APPL% CP
- **IIP**: Percentage of CPU time used by transactions that are executed on zIIPs in the service or report class period

Figure 10-4 shows a sample report. You can see an overview of the entire system, including zIIP and zAAP information, from this report. *Appl% IIP* is a percentage of CPU time on zIIPs used by all address spaces and enclaves during the report interval. This value is divided by the number of zIIPs that have been active during this interval.



```
                          RMF V1R7   System Information            Line 1 of 19
Command ===>                                                 Scroll ===> CSR

Samples: 100     System: SC04  Date: 10/18/06  Time: 15.25.00  Range: 100   Sec

Partition:    A09     2094 Model 710         Appl%:       1  Policy: CICSWORK
CPs Online:   2.0     Avg CPU Util%:    5     EAppl%:      2  Date:   10/09/06
AAPs Online:  2.0     Avg MVS Util%:    5     Appl% AAP:   0  Time:   16.13.43
IIPs Online:  2.0                            Appl% IIP:   0
Group     T WFL --Users--  RESP TRANS -AVG USG-   -Average Number Delayed For -
            %   TOT  ACT  Time  /SEC PROC  DEV   PROC  DEV STOR SUBS OPER  ENQ

*SYSTEM      84  132    0        0.04  0.1  0.1    0.0  0.0  0.0  0.0  0.0  0.0
*TSO              5    0        0.04  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
*BATCH            0    0        0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
*STC         84  123    0        0.00  0.1  0.1    0.0  0.0  0.0  0.0  0.0  0.0
*ASCH             0    0        0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
*OMVS             4    0        0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
*ENCLAVE          0  N/A         N/A  0.0  N/A    0.0  N/A  0.0  N/A  N/A  N/A
OTHER     W       2    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
VEL70     S       2    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
SYSTEM    W  83  125    0  .000  0.00  0.1  0.1    0.0  0.0  0.0  0.0  0.0  0.0
SYSSTC    S  67  102    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
```

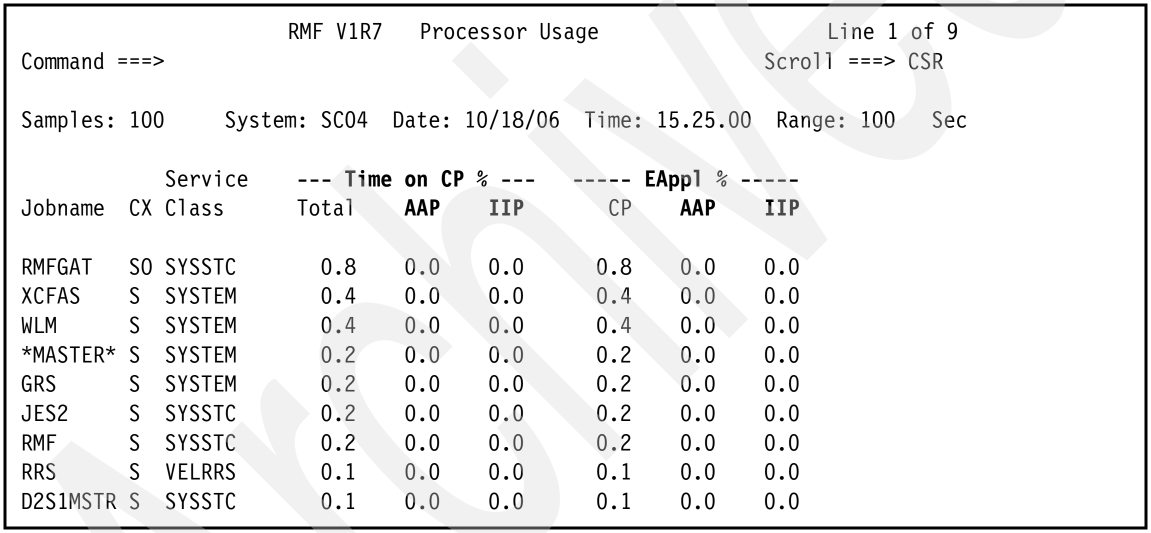
Figure 10-4 RMF MON III SYSINFO report

The monitoring screen in Figure 10-5 shows zIIP or zAAP usage by address space name.

- Time on CP %
  - **Total**: Percentage of CPU time spent on standard CPs as a sum of TCB time, global and local SRB time, and preemptable or client SRB time consumed on behalf of this address space
  - **IIP**: Percentage of CPU time on standard CPs by this address space, which was used by zIIP eligible work; is a subset of the Total percentage

  - **AAP**: Percentage of CPU time on standard CPs by this address space, which was used by zIIP eligible work; is a subset of the Total percentage
- EAppl %
  - **CP**: Percentage of CPU time on standard CPs as sum of TCB time, global and local SRB time, preemptable or client SRB time, and enclave CPU time consumed within this address space
  - **IIP**: Percentage of CPU time consumed on zIIPs within this address space
  - **AAP**: Percentage of CPU time consumed on zAAPs within this address space

Note: For RMF monitoring, you need PTF UA28294.



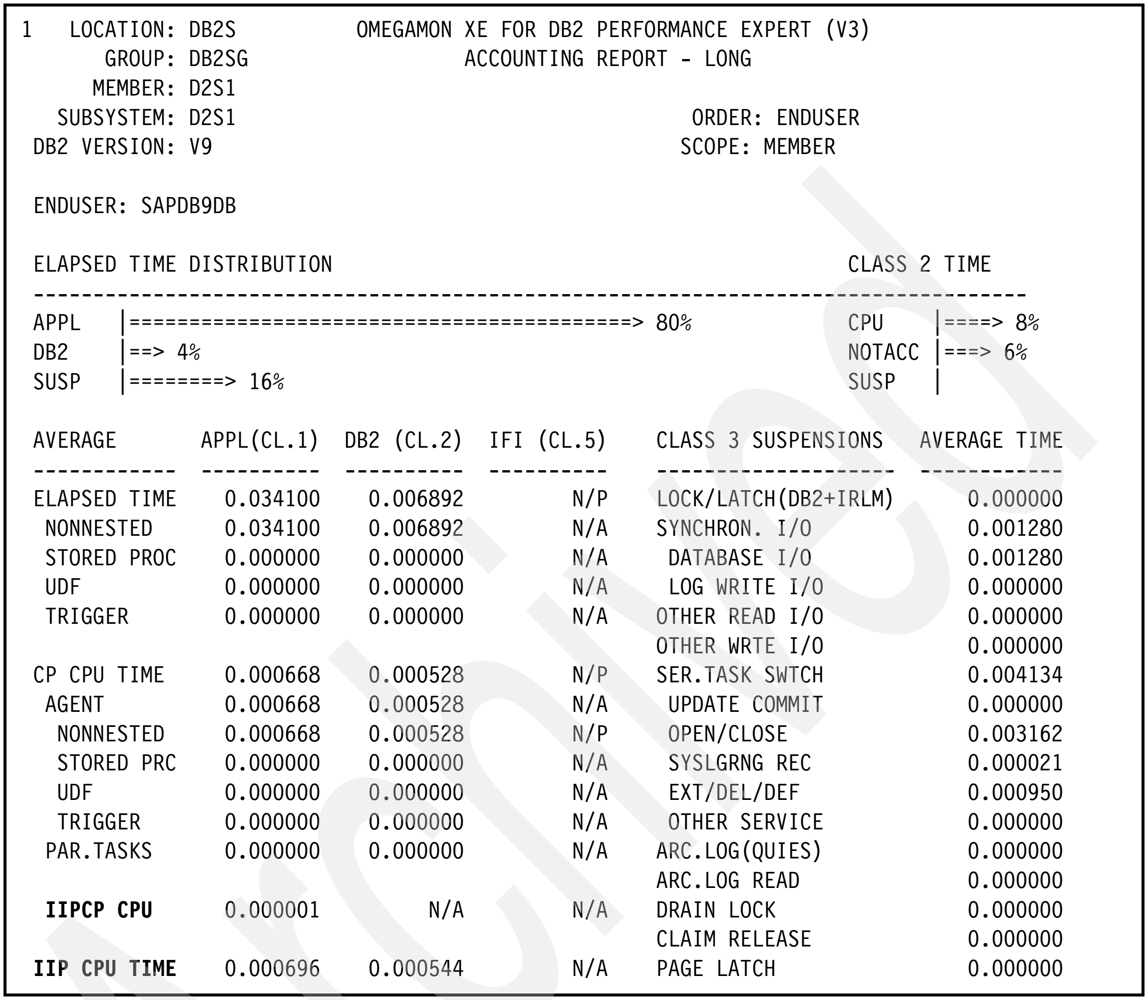
```
                       RMF V1R7   Processor Usage                   Line 1 of 9
Command ===>                                                   Scroll ===> CSR

Samples: 100     System: SC04  Date: 10/18/06  Time: 15.25.00  Range: 100   Sec

            Service    --- Time on CP % ---   ----- EAppl % -----
Jobname  CX Class      Total    AAP    IIP       CP    AAP    IIP

RMFGAT   SO SYSSTC       0.8    0.0    0.0      0.8    0.0    0.0
XCFAS    S  SYSTEM       0.4    0.0    0.0      0.4    0.0    0.0
WLM      S  SYSTEM       0.4    0.0    0.0      0.4    0.0    0.0
*MASTER* S  SYSTEM       0.2    0.0    0.0      0.2    0.0    0.0
GRS      S  SYSTEM       0.2    0.0    0.0      0.2    0.0    0.0
JES2     S  SYSSTC       0.2    0.0    0.0      0.2    0.0    0.0
RMF      S  SYSSTC       0.2    0.0    0.0      0.2    0.0    0.0
RRS      S  VELRRS       0.1    0.0    0.0      0.1    0.0    0.0
D2S1MSTR S  SYSSTC       0.1    0.0    0.0      0.1    0.0    0.0
```

Figure 10-5 RMF MON III Processor Usage

10.3.1 DB2 REPORT

Using OMEGAMON® XE for DB2 PM/PE, you can check the zIIP offload percentage by application. Figure 10-6 shows a sample DB2 report.

Note: APAR PK25395 is required for OMEGAMON XE for DB2 PM/PE V310.



```
1   LOCATION: DB2S          OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (V3)
       GROUP: DB2SG                  ACCOUNTING REPORT - LONG
      MEMBER: D2S1
   SUBSYSTEM: D2S1                                      ORDER: ENDUSER
 DB2 VERSION: V9                                       SCOPE: MEMBER

 ENDUSER: SAPDB9DB

 ELAPSED TIME DISTRIBUTION                                            CLASS 2 TIME
 ------------------------------------------------------------------------------------
 APPL   |==========================================> 80%              CPU    |====> 8%
 DB2    |==> 4%                                                       NOTACC |===> 6%
 SUSP   |========> 16%                                                SUSP   |

 AVERAGE       APPL(CL.1)  DB2 (CL.2)  IFI (CL.5)    CLASS 3 SUSPENSIONS    AVERAGE TIME
 ------------  ----------  ----------  ----------    --------------------   ------------
 ELAPSED TIME    0.034100    0.006892         N/P    LOCK/LATCH(DB2+IRLM)       0.000000
  NONNESTED      0.034100    0.006892         N/A    SYNCHRON. I/O              0.001280
  STORED PROC    0.000000    0.000000         N/A     DATABASE I/O              0.001280
  UDF            0.000000    0.000000         N/A     LOG WRITE I/O             0.000000
  TRIGGER        0.000000    0.000000         N/A    OTHER READ I/O             0.000000
                                                     OTHER WRTE I/O             0.000000
 CP CPU TIME     0.000668    0.000528         N/P    SER.TASK SWTCH             0.004134
  AGENT          0.000668    0.000528         N/A     UPDATE COMMIT             0.000000
   NONNESTED     0.000668    0.000528         N/P     OPEN/CLOSE                0.003162
   STORED PRC    0.000000    0.000000         N/A     SYSLGRNG REC              0.000021
   UDF           0.000000    0.000000         N/A     EXT/DEL/DEF               0.000950
   TRIGGER       0.000000    0.000000         N/A     OTHER SERVICE             0.000000
  PAR.TASKS      0.000000    0.000000         N/A    ARC.LOG(QUIES)             0.000000
                                                     ARC.LOG READ               0.000000
  IIPCP CPU      0.000001         N/A         N/A    DRAIN LOCK                 0.000000
                                                     CLAIM RELEASE              0.000000
 IIP CPU TIME    0.000696    0.000544         N/A    PAGE LATCH                 0.000000
```

Figure 10-6 OMEGAMON XE for DB2 PM/PE

IP CPU time and IIPCPU time: *IP CPU time* is the CPU time using the zIIP. *IIP CPU time* is the CPU time when zIIP eligible work runs on a normal CP, when zIIP is busy, or when zIIP is not configured (providing projection information).

10.4 zIIP measurements (SAP OLTP, BI Batch, SAP BW, utility workloads)

In this section, we describe some measurements done in the IBM Laboratory using zIIP hardware and software. Four types of DB2 workloads were measured:

- SAP OLTP exploiting DRDA (using DB2 Connect)
- BI batch direct attached
- SAP BW query exploiting DRDA
- DB2 Utility

You can find more specific test environments in Appendix C, “zIIP measurements” on page 231.

Analysis of various aspects of these measurements was done. For the analysis used in some cases, see *Large Systems Performance Reference*, SC28-1187. You should be familiar with the terms therein, especially External Throughput Rate (ETR), Internal Throughput Rate (ITR), and the corresponding ratios (ETRR and ITRR). In this section, we define ITR as being based on regular CP utilization. That is, ITR did not take into account zIIP utilizations.

10.4.1 SAP OLTP workload

The SAP OLTP workload exploited zIIPs due to its use of DRDA. There was no use of the ProjectCPU=Yes function. See 10.4.4, “DB2 utilities workload” on page 135, for more information about this. There was enough zIIP CPs so that all zIIP-eligible work was redirected. No stored procedures were used. No DB2 accounting information was collected because of the volume of data generated by the tool for this workload. The results show several interesting items, some of which are shown graphically in Figure 10-7.

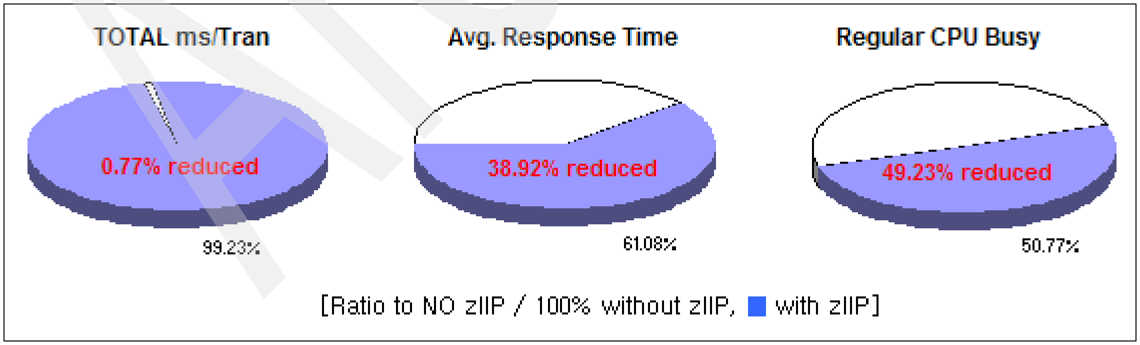


Figure 10-7 zIIP effects on the SAP OLTP workload

First, in the zIIP case, the total milliseconds of CP time per transaction was about 1% less. This indicates that zIIP exploitation is about as efficient in terms of total CP time as non-zIIP.

Second, the response time of the zIIP run was much better. This is due to the fact that the zIIP measurement had a total of eight CPs of relatively low utilization compared to the non-zIIP case, which had only four total CPs at a much higher utilization. However, in both cases, the response time was sub-second. The regular CP busy time dropped to almost half. Similarly, ITR based on regular CPs about doubled. This reflects the fact that the configuration with zIIPs has about twice the throughput potential.

Finally, the zIIP measurement showed that 49.8% of the entire LPAR was redirected to the zIIPs. This includes both captured, uncaptured, and LPAR management times.

However, this was not what we expect of a typical production environment. More normal production environments should expect lower redirect percentages.

10.4.2 BI Batch workload

In this workload, there was no use of DRDA nor was there use of the ProjectCPU=Yes function. See 10.4.4, “DB2 utilities workload” on page 135, for more information about this. There was enough zIIP CPs so that all zIIP eligible work was redirected. In addition, no stored procedures were used, because the batch application was directly attached to DB2 in the same z/OS image. The exploitation of zIIPs in this particular workload came entirely from its use of parallel queries. Because this workload was batch, and therefore had relatively few “transactions”, we ran DB2 accounting. The results show several interesting items, some of which are shown graphically in Figure 10-8.

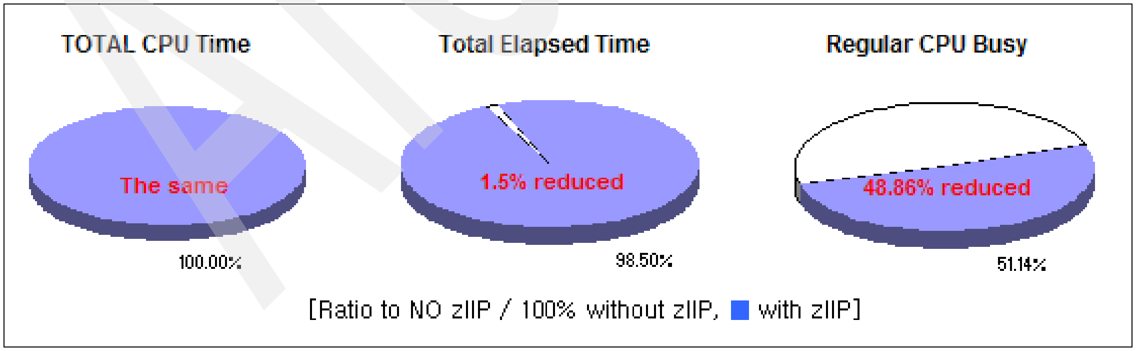


Figure 10-8 zIIP effects on the BI Batch workload

Again, the total CPU time is virtually the same with and without zIIPs. In this case, the total elapsed time was only slightly less with zIIPs. In contrast to the previous workload, which was CPU bound, this workload is dominated by waiting for database I/O. The regular CPU busy fell 49% with zIIPs. From the DB2 accounting data, the No zIIP run's IIPCP CPU estimate for zIIP potential redirect was close to the zIIP run's IIP actual CPU time. Finally, the zIIP measurements show that 49.04% of the entire LPAR was redirected to the zIIPs. Because this data is from WLM, it does not count uncaptured time.

However, this was not what we expect of a typical production environment. More normal production environments should expect lower redirect percentages.

10.4.3 SAP BW Query workload

SAP BW environments have the potential to exploit zIIPs for two reasons. First, more recent SAP releases use DRDA. Further, SAP BW environments also have the potential to have parallel queries. There was no use of the ProjectCPU=Yes function. See 10.4.4, "DB2 utilities workload" on page 135, for more information about this. There were enough zIIP CPs so that all zIIP eligible work was redirected. No stored procedures were used. No DB2 accounting information was collected because of the volume of data generated by the tool for this workload. The results show several interesting items, some of which are shown graphically in Figure 10-9.

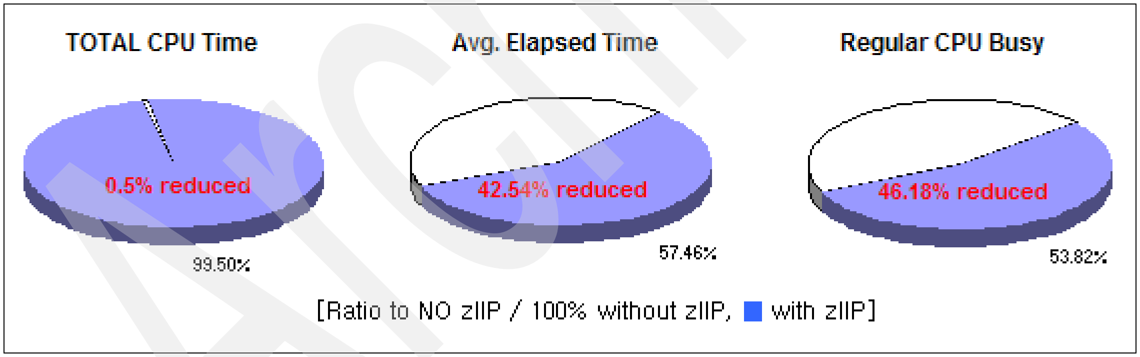


Figure 10-9 zIIP effects of the SAP BW Query workload

Again, in the zIIP case, the milliseconds CPU per transaction was virtually the same, about 0.5% less. This indicates that zIIP exploitation is about as efficient in terms of total CPU usage as non-zIIP. Second, the response time of the zIIP run was 43% better. Like the SAP OLTP measurement, the zIIP measurement had a total of eight CPs of relatively low utilization compared to the non-zIIP case, which had only four total CPs at a much higher utilization. The regular CP busy time dropped 46%. However, because with zIIPs, there was more throughput, the ITR based on regular CPs doubled. This reflects the fact that the configuration with zIIPs has about twice the throughput potential.

Finally, the zIIP measurement showed that 50.1% of the entire LPAR was redirected to the zIIPs. This includes both captured, uncaptured, and LPAR management times. This is close to the same redirect as the SAP OLTP workload.

Normally you might expect BW with DRDA to get more redirect than regular SAP with DRDA. However, the SAP BW Query workload consists of short queries that get little opportunity to exploit zIIPs.

However, these results are not what we expect of a typical production environment. More normal production environments should expect lower redirect percentages. However, parallelism is more likely chosen when queries include partitioned table spaces, and SAP BW provides the capability to partition both the F and E fact tables. Thus, customers have the opportunity to increase the likelihood that their queries will run in parallel.

10.4.4 DB2 utilities workload

The amount of redirect for utilities using this zIIP configuration is:

- 5% to 20% for Rebuild Index
- 10% to 20% for Load or Reorg of a partition with one index only, Load of an entire table, or Reorg of an entire table space
- 40% for Rebuild Index of an LPAR of a non-partitioning index
- 40% to 50% for Reorg Index
- 30% to 60% for Load or Reorg of a partition with more than one index

Variations in percentage redirect for various utilities are primarily determined by the percentage of CPU time consumed by build index processing, which is redirected, to the total CPU time for a given utility. For example, Load or Reorg with more than one partition spends most of the CPU time in build index phase. Consequently it is in a position to gain the biggest redirect percentage, especially with more indexes.

The ProjectCPU function in WLM enables estimation of how much redirect can occur as reported in IIPCP CPU time when zIIP is not available. When zIIP is available, there is no difference between ProjectCPU No and Yes.

Consider a case when there is no zIIP processor, but ProjectCPU is requested in order to estimate the potential redirect, or when zIIP is present. In such a case, a new execution unit switch from utility TCB to an enclave SRB takes place during the index build phase of DB2 z/OS Load, Reorg, and Rebuild Index for each block of index entries. This newly added processing to enable real or simulated CPU time redirect typically results in less than a 10% increase in the job total CPU time and elapsed time. Most of the added CPU time is actually redirected to zIIP when available. This extra CPU and elapsed time tend to go up as the ratio of the number of zIIPs to the number of CPs goes down. Increasing the ratio of the number of zIIPs to the number of CPs closer to one tends to result in less extra CPU and elapsed time and a maximized redirect percentage.

Including the ProjectCPU costs, we saw a net reduction in regular CP time. This ranged from a 4% to 57% reduction. As you might suspect, more indexes result in more utility CPU time. All of the zIIP measurements were with four regular CPs and two zIIPs. In all six index measurements, the IIPCP CPU time indicates that there was some overflow to regular CPs and that these measurements could have exploited even more than two zIIPs.

This set of measurements is not intended to represent customer environments. As mentioned earlier, some customers have some tables that are much larger. In some cases, these customers have had to carefully plan when to do this sort of utility work. zIIPs might help this as well as possibly reduce the costs of this work.

10.4.5 Conclusions

These measurements show exploitation of zIIPs in several environments. In all of these measurements, the amount of redirect to zIIPs is more than any possible additional overheads due to ProjectCPU. In some cases, there were significant redirects. One was almost 60%. The number of zIIPs configured to a system is limited to less than or equal to the number of regular CPs configured on the system. However, this is at a system level. Individual LPARs on that system can exploit more zIIPs than they have regular CPs.

These measurements were run in a laboratory environment. We expect real customer DB2 systems to run more functions that are related to the “care and feeding” of a production DB2 database system. These functions can range from monitoring and tracing options to service and administrative programs such as Data Facility Hierarchical Storage Management (DFHSM). These functions do not exploit zIIPs, and any environment using them will have their potential zIIP redirect percentage of an LPAR diluted. Similarly, running applications that are unrelated to DB2 can dilute observed redirect LPAR percentages.

As such, you should understand that these results are limited in scope and you should not use them for sizing. We expect that your workloads will show different results.

11

Performance topics

In this chapter, we discuss the features in DB2 9 for z/OS that enhance the performance capabilities of SAP. These features include:

- Global query optimization
- FETCH FIRST *n* ROWS ONLY clause in subselects
- Reduction of CPU for utilities
- Histogram statistics
- Automatic buffer pool size management
- Performance improvement of VARCHAR

11.1 Global query optimization

Global query optimization addresses query performance problems that are caused when DB2 breaks a query into multiple parts and optimizes each of those parts independently. While each of the individual parts may be optimized to run efficiently, when these parts are combined, the overall result may be inefficient. The purpose of new DB2 9 for z/OS feature is to improve query performance by enhancing the DB2 optimizer so that more efficient access paths are generated for queries that involve multiple parts. SAP systems with more complex queries mainly benefit from this new feature.

For example, consider the query in Example 11-1.

Example 11-1 Sample query

```
SELECT *
FROM T1
WHERE EXISTS (SELECT 1
FROM T2, T3
WHERE T2.C2 = T3.C2
     AND T2.C1 = T1.C1);
```

DB2 breaks this query into two parts: the correlated subquery and the outer query. Each of these parts will be optimized independently. The access path for the subquery does not take into account the different ways in which the table in the outer query may be accessed and vice versa.

DB2 may choose to do a table scan of T1, resulting in much random I/O when accessing T2, while a non-matching index scan of T1 avoids the random I/O on T2. In addition, DB2 does not consider reordering these two parts. The correlated subquery is always performed after accessing T1 to obtain the correlation value. If T1 is a large table, and T2 is a small table, it may be much more efficient to access T2 first and then T1, especially if there is no index on T2.C1 but there is an index on T1.C1.

In summary, global query optimization allows DB2 to optimize a query as a whole rather than as independent parts. It accomplishes this by allowing DB2 to consider the effect of one query block and to consider reordering query blocks.

With the global query optimization in DB2 9 for z/OS, the new general approach is as follows:

- The subquery is represented as a “virtual table” in the FROM clause that contains the predicate with the subquery.
- This virtual table may be moved around within the referencing query in order to obtain the most efficient sequence of operations.
- Predicates may be derived from the correlation references in the subquery and from the subquery SELECT list.
- These predicates can be applied to either the subquery tables or the tables that contain the correlated columns depending on the position of the “virtual table”.
- When determining the access path for a subquery, the context in which the subquery occurs is taken into consideration.
- When determining the access path for a query that references a subquery, the effect that the access path has on the subquery is taken into consideration.

For optimization purposes, DB2 represents a subquery, which cannot be transformed to a join, as a virtual table. This is similar to the way in which materialized views and table expressions are represented. A *virtual table* is an abstract representation of a subquery. This additional level of abstraction allows the DB2 optimizer to easily rearrange the order and manner in which the subqueries are processed without incurring the cost of fully transforming the query into each form that the optimizer wants to consider.

Subqueries that can be transformed to joins continue to be transformed. Virtual tables only apply to those subqueries that cannot be transformed to joins, or cases where transformation to join reduce the choices that are available to the optimizer, for example with updatable cursors.

11.2 FETCH FIRST n ROWS ONLY clause in subselects

As part of certain SAP BW administrative functions, a large number of selected rows (a few million) of large tables with hundreds of millions of rows are often deleted. These rows are not tied to any partition. Concurrent access to other rows of the table is necessary. This poses the following challenges with respect to the DELETE operation:

- Rollback is fairly slow.
- A large number of locks are accumulated or there is no concurrent access if locks are escalated.

- If the delete operation fails, the work that has already been accomplished becomes lost and the delete operation has to start over.

DB2 9 for z/OS addresses this difficulty by supporting the FETCH FIRST n ROWS ONLY clause in subselects. Before V9, this clause was already available, but only at the level of full SELECT statements. With SAP, this has been used to mimic singleton SELECT statements in the ABAP language. In addition to the FETCH FIRST n ROWS ONLY clause, DB2 9 for z/OS also supports ORDER BY in subselects.

SAP BW can take advantage of these new capabilities by splitting up a large delete operation into multiple smaller units of work. In Example 11-2, assume that the primary key of table TABLE1 consists of the columns Key1, Key2, and Key3. The rows to be deleted are specified in predicate indicated in the example as <DEL_PRED>. To split up this delete into a loop of delete operations with each one deleting 10000 rows only, the statement, FETCH FIRST 10000 rows only, can be used. After each execution of the DELETE statement, a commit can be performed. The loop continues until no more rows need to be deleted.

Example 11-2   Using the FETCH FIRST 10000 rows only statement

```
DELETE FROM TABLE1 X
WHERE  EXISTS
  (SELECT *
   FROM TABLE1 Y
   WHERE X.KEY1 = Y.KEY1
   AND   X.KEY2 = Y.KEY2
   AND   X.KEY3 = Y.KEY3
   AND   <DEL_PRED>
   FETCH FIRST 10000 rows only)
```

11.3 CPU reduction of the utilities

For optimal performance of utilities, DB2 creates specific assembler statements called KPROCs to copy the data into the index keys based on the row and index key layout. The REORG and LOAD utilities mainly exploit these statements to efficiently build index keys from data rows.

However, the generated assembler statements do not support nonpadded indexes, which were introduced with DB2 V8 and which are used by SAP. DB2 9 for z/OS adds support for nonpadded indexes to the KPROCs. Therefore, the performance advantage previously existing for padded indexes also applies to nonpadded indexes.

CPU time is reduced when extracting nonpadded keys from data rows for the following utilities:

- REORG
- REBUILD INDEX
- CHECK INDEX
- LOAD

11.4 Histogram statistics

To enable the DB2 optimizer to determine optimal access paths for queries with predicates on table columns with skewed data distributions, you can use the RUNSTATS utility. You can instruct this utility to collect frequency statistics (also known as *distribution statistics*) by specifying the FREQVAL keyword. RUNSTATS then collects the frequency of certain values (single-column frequencies) or value combinations (multi-column frequencies). The keywords COUNT, MOST, LEAST, and BOTH determine the number of values or value combinations for which frequencies are collected and whether the frequencies are collected for the most frequently or least frequently occurring values.

While frequency statistics provide exact information, it is usually not possible to collect them on each distinct value of a column due to concerns with catalog space and query preparation performance. Therefore, only selected frequency statistics are collected that provide a partial picture of how data is skewed. Consider Table 11-1 which shows the frequency statistics for the YRS_OF_EXPERIENCE column of the EMPLOYEE table with RUNSTATS option COUNT 4 BOTH.

Table 11-1 Frequency statistics

Value	Frequency
3	15%
25	12%
26	11%
27	7%
12	0.02%
13	0.01%
40	0.0001%
41	0.00001%

For queries shown in Example 11-3 through Example 11-5, the single-value based frequency statistics can hardly help DB2 with the predicate selectivity other than uniform interpolation on the rest of (uncollected) value range.

Example 11-3 EQ predicate with unmatched value

```
SELECT EMPID FROM EMPLOYEE T
WHERE T.YRS_OF_EXPERIENCE = 6;
```

Example 11-4 RANGE predicate

```
SELECT T.EMPID FROM EMPLOYEE
WHERE T.YRS_OF_EXPERIENCE BETWEEN 5 AND 10;
```

Example 11-5 Non-local predicate

```
SELECT T1.EMPID FROM EMPLOYEE T1, OPENJOBS T2
WHERE T1.SPECIALTY = T2.AREA AND
T1.YRS_OF_EXPERIENCE > T2.YRS_OF_EXPERIENCE;
```

DB2 9 for z/OS introduces histogram statistics to address cases such as these, particularly range predicates. Histogram statistics describe the data distribution over the entire value range. Histogram statistics is a way of summarizing data distribution on an interval scale (either discrete or continuous). It divides the range of possible values in a table into quantiles, for which a set of statistics parameters are collected. The whole value range is cut into quantiles so that each quantile has about the same number of rows.

Table 11-2 shows an example of the histogram statistics.

Table 11-2 Histogram statistics on column YRS_OF_EXPERIENCE of table EMPLOYEE

SEQ NO.	LOWVALUE	HIGHVALUE	CARD	FREQUENCY
1	0	4	4	20%
2	5	11	7	20%
3	12	22	11	19%
4	25	26	2	23%
5	27	41	10	18%

Even if there is no perfect match between the ranges of the predicate and the low and high values of the quantiles, the predicate selectivity interpolation now is done within the relevant quantiles. With the interpolation done in a much smaller granularity, the predicate selectivity is expected to be evaluated with more accuracy.

Because it is more costly to collect histogram statistics compared to cardinalities of tables, they should not be collected indiscriminately for all SAP tables. Only if queries on a table benefit from histogram statistics should you collect these statistics for the table.

Figure 11-1 shows the correlation-stats-spec portion of the RUNSTATS syntax diagram.

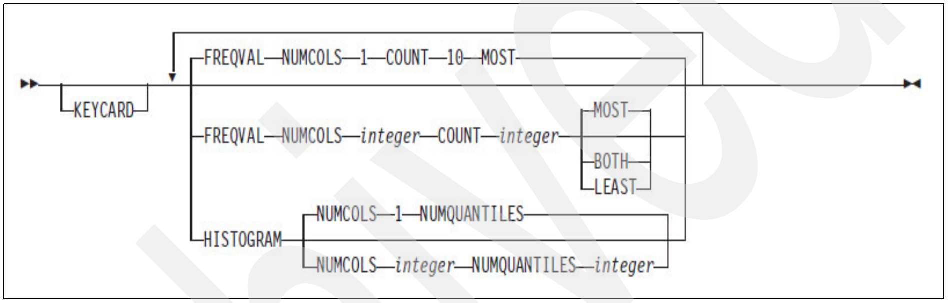

KEYCARD
FREQVAL—NUMCOLS—1—COUNT—10—MOST
FREQVAL—NUMCOLS—integer—COUNT—integer
MOST
BOTH
LEAST
HISTOGRAM
NUMCOLS—1—NUMQUANTILES
NUMCOLS—integer—NUMQUANTILES—integer

Figure 11-1 Correlation-stats-spec portion of the RUNSTATS syntax diagram

Multi-column histogram statistics can also be collected. They are collected similarly as multi-column frequency statistics, that is concatenated columns, and treat the concatenated value as a single value. Figure 11-2 shows the colgroup-stat-spec portion of the RUNSTATS syntax diagram.

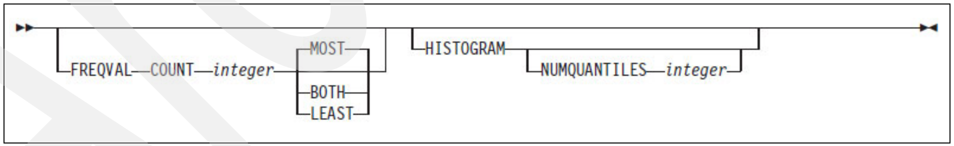


Figure 11-2 Colgroup-stat-spec portion of the RUNSTATS syntax diagram

There may be skipped values between adjacent quantiles. If there are, then it means those skipped values are of zero appearance in the table.

Each quantile has approximately the same number (or percentage) of rows. The highly frequent single value may alone by itself occupy a quantile. The single value is not be broken into more than one quantile. Therefore, the number of quantiles maximum is equivalent to the number of distinct values.

One particular value may be included into a quantile just because it is “closer” to that quantile. The purpose is to avoid the skipped range as much as possible. It is beneficial to have skipped ranges reflected in the histogram statistics. If there are predicates on skipped ranges, the optimizer takes this into consideration.

The maximum number of quantiles collected for the HISTOGRAM option should never exceed the total number of distinct values of the column or column group. However, for storage purposes, DB2 has a maximum limit of 100 quantiles.

11.5 Size management of the automatic buffer pool

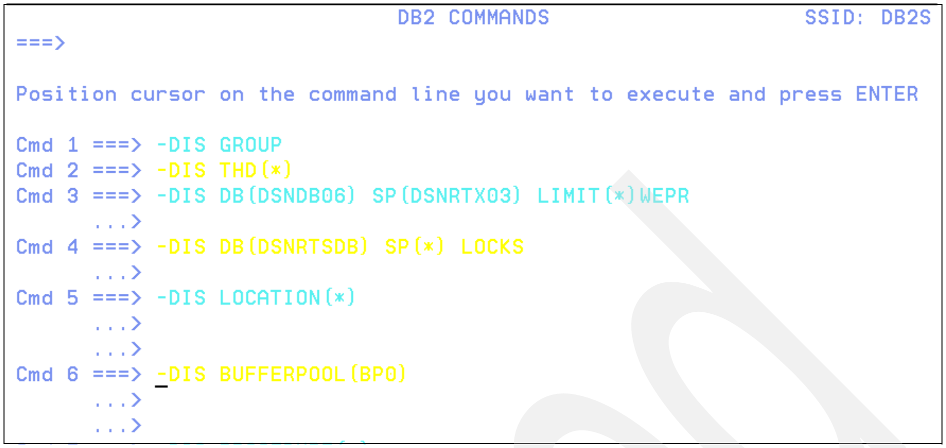
By enabling the size management feature of the DB2 automatic buffer pool, you can reduce the amount of time that you spend monitoring and adjusting DB2 buffer pools. Automatic WLM of DB2 buffer pools is introduced with z/OS 1.8 and DB2 Version 9.

To enable size management feature, you must alter the buffer pool or buffer pools. By default, automatic buffer pool adjustment is turned off. You can activate it by specifying the new AUTOSIZE(YES) option on the ALTER BUFFERPOOL command. After you activate this option, you can deactivate it by specifying ALTER BUFFERPOOL(bpname) AUTOSIZE(NO). The AUTOSIZE attribute is added to the DISPLAY BUFFERPOOL output.

To do this, we decided to turn on AUTOSIZE=YES to demonstrate it for our DB2 9 for z/OS system. We ran a command from the DB2 Commands panel. See Example 11-6 and Figure 11-3.

Example 11-6 Display the buffer pool

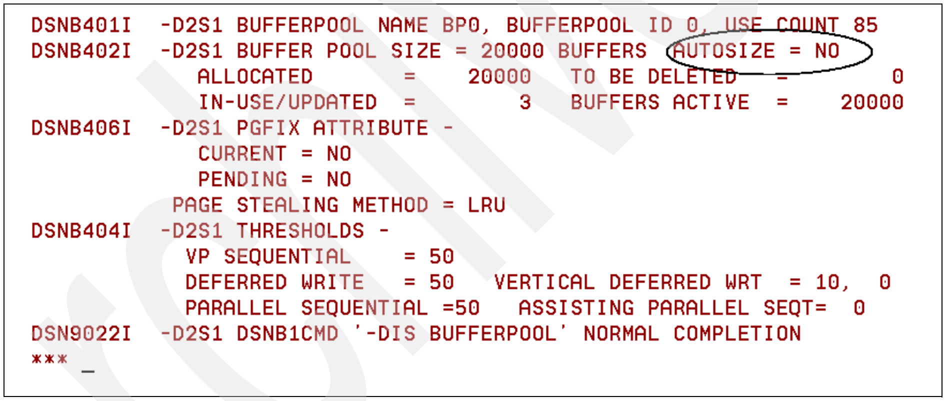
```
-DIS BUFFERPOOL(BP0)
```



```
                                  DB2 COMMANDS                     SSID: DB2S
===>

Position cursor on the command line you want to execute and press ENTER

Cmd 1 ===> -DIS GROUP
Cmd 2 ===> -DIS THD(*)
Cmd 3 ===> -DIS DB(DSNDB06) SP(DSNRTX03) LIMIT(*)WEPR
      ...>
Cmd 4 ===> -DIS DB(DSNRTSDB) SP(*) LOCKS
      ...>
Cmd 5 ===> -DIS LOCATION(*)
      ...>
      ...>
Cmd 6 ===> _-DIS BUFFERPOOL(BP0)
      ...>
      ...>
```

Figure 11-3 Displaying the buffer pool from the DB2 Commands screen

As you can see in Figure 11-4, displaying BP0 shows AUTSIZE = NO.



```
DSNB401I  -D2S1 BUFFERPOOL NAME BP0, BUFFERPOOL ID 0, USE COUNT 85
DSNB402I  -D2S1 BUFFER POOL SIZE = 20000 BUFFERS  AUTOSIZE = NO
             ALLOCATED        =     20000    TO BE DELETED    =        0
             IN-USE/UPDATED   =         3    BUFFERS ACTIVE   =    20000
DSNB406I  -D2S1 PGFIX ATTRIBUTE -
             CURRENT = NO
             PENDING = NO
           PAGE STEALING METHOD = LRU
DSNB404I  -D2S1 THRESHOLDS -
            VP SEQUENTIAL     = 50
            DEFERRED WRITE    = 50   VERTICAL DEFERRED WRT   = 10,  0
            PARALLEL SEQUENTIAL =50    ASSISTING PARALLEL SEQT=  0
DSN9022I  -D2S1 DSNB1CMD '-DIS BUFFERPOOL' NORMAL COMPLETION
*** _
```

Figure 11-4 Display of the buffer pool confirms AUTOSIZE=NO

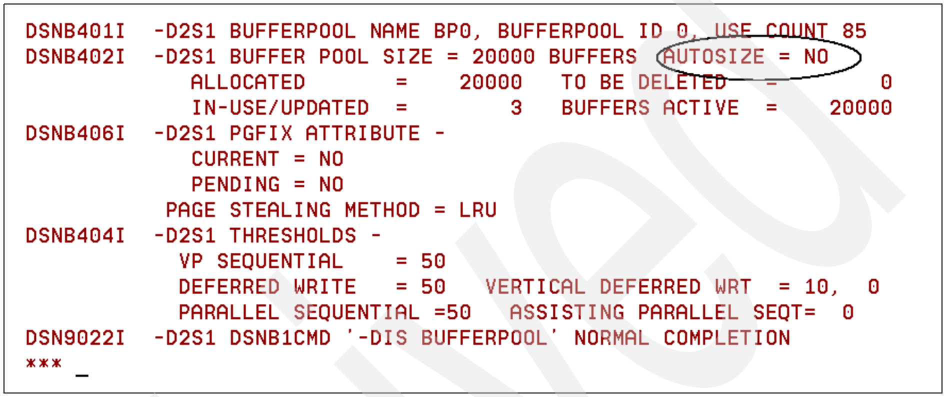
To change AUTOSIZE = YES, we can either use the SQL Processor Using File Input (SPUFI) or run a JCL batch job. We chose the batch JCL, as shown in Example 11-7.

Example 11-7 Batch JCL to change AUTOSIZE=YES

```
//ALTBPTG  JOB ,,MSGLEVEL=(1,1),CLASS=A,MSGCLASS=H,NOTIFY=&SYSUID,
//    REGION=4M
/*JOBPARM SYSAFF=JA0
//JOBLIB    DD  DSN=DB2SU.DB2S1.V9.SDSNLOAD,DISP=SHR
//JSTEP01  EXEC PGM=IKJEFT01
//SYSTSPRT  DD  SYSOUT=*
```

```
//SYSTSIN   DD  *
  DSN SYSTEM(D2S1)
    -ALTER BUFFERPOOL(BP0)      AUTOSIZE(YES)
END
```

We ran the JCL job and received a RC = 0. Then we displayed the buffer pool again. Figure 11-5 shows the output from the -DIS BP(BP0) command.



```
DSNB401I  -D2S1 BUFFERPOOL NAME BP0, BUFFERPOOL ID 0, USE COUNT 85
DSNB402I  -D2S1 BUFFER POOL SIZE = 20000 BUFFERS  AUTOSIZE = NO
             ALLOCATED       =    20000    TO BE DELETED    =       0
             IN-USE/UPDATED  =        3    BUFFERS ACTIVE   =   20000
DSNB406I  -D2S1 PGFIX ATTRIBUTE -
             CURRENT = NO
             PENDING = NO
           PAGE STEALING METHOD = LRU
DSNB404I  -D2S1 THRESHOLDS -
            VP SEQUENTIAL     = 50
            DEFERRED WRITE    = 50   VERTICAL DEFERRED WRT  = 10,  0
            PARALLEL SEQUENTIAL =50   ASSISTING PARALLEL SEQT=  0
DSN9022I  -D2S1 DSNB1CMD '-DIS BUFFERPOOL' NORMAL COMPLETION
*** _
```

Figure 11-5 Redisplay of the buffer pool confirms AUTOSIZE=YES

11.6 Performance improvement of VARCHAR

Prior to DB2 9 for z/OS, the recommendation was that the varying-length column should be in the last position. In DB2 9 for z/OS, the varying length columns are automatically reordered. If a table is from DB2 V8, you should use LOAD or REORG in DB2 9 for z/OS in order for this change to take effect.

11.6.1 Reordered row format from DB2 9 (VARCHAR with not null)

If a table has any varying-length columns, its rows contain varying-length values and are varying-length rows. When a value shorter than the length of the column is stored in a varying-length column, it is not padded to the full length of the column. Therefore, columns that appear after varying-length columns are at variable offsets in the row. Prior to DB2 9, to get to such a column, DB2 scanned the columns sequentially after the first varying length column.

Figure 11-6 shows the basic row format of VARCHAR.

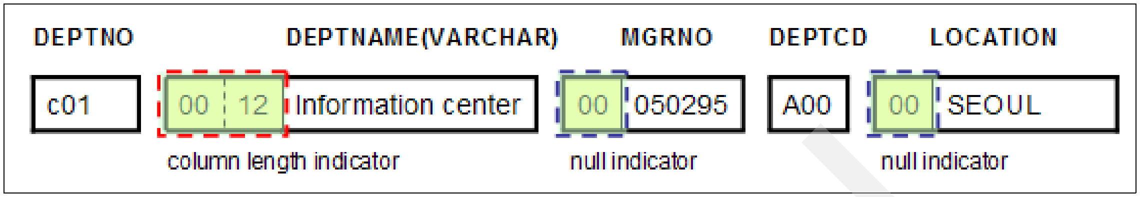


Figure 11-6 Example of the basic row format for VARCHAR

> **Note:** The DEPTNAME column is VARCHAR, and the others are CHAR. MGRNO and LOCATION are nullable columns.
>
> The column length indicator and below offset column are hexadecimal values.

In the reordered row format, if a table has any varying-length columns, all fixed length columns are placed at the beginning of the row. They are followed by the offsets to the varying length columns, which are followed by the values of the varying length columns.

With the reordered row format, it is no longer necessary to scan the prior columns to find the column of interest. The reordered row format of the row enables DB2 to more quickly find columns that follow varying length columns in the row. See Figure 11-7.

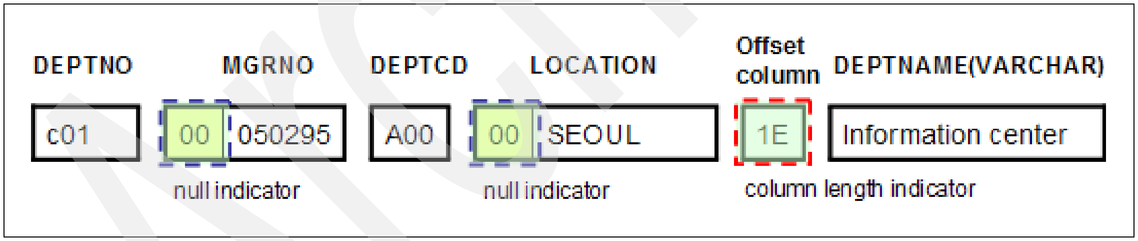


Figure 11-7 Example of the reordered row format for VARCHAR

> **Note:** Assume that the length of the fixed DEPTNO, MGRNO, DEPTCD, and LOCATION columns is 1F.

11.6.2 Reordered row format from DB2 9 for z/OS (VARCHAR with nullable)

A varying-length column can also allow null values. In a basic row format, the value in the length field includes the null indicator byte. See Figure 11-8.

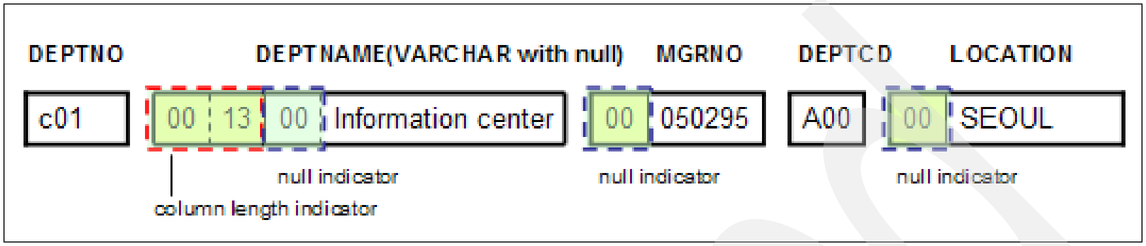


Figure 11-8 Example of VARCHAR with nullable basic row format

In a reordered row format, if a table has any varying-length columns, with or without nulls, all fixed-length columns are placed at the beginning of the row. They are followed by the offsets to the varying-length columns (Figure 11-9), which are followed by the values of the varying-length columns (Figure 11-10).

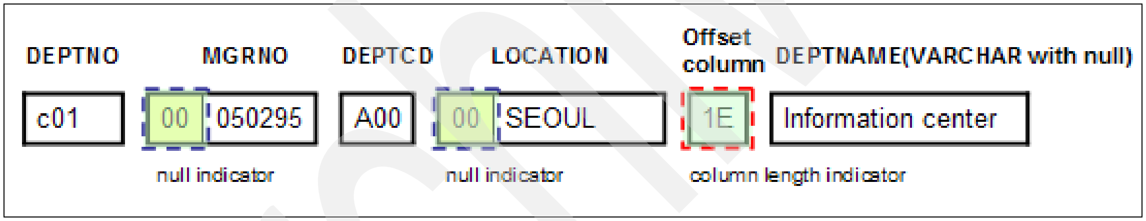


Figure 11-9 Example of VARCHAR with nullable reordered row format

Note: If there were an additional varying-length column after DEPTNAME, it would have an offset of 31 in hexadecimal because of the null indicator byte from the null value in DEPTNAME.

HEX 1E is DEC 30, null indicator 1 byte, and DEPTNAME 18 bytes, so the next offset is DEC 49 Hex31.

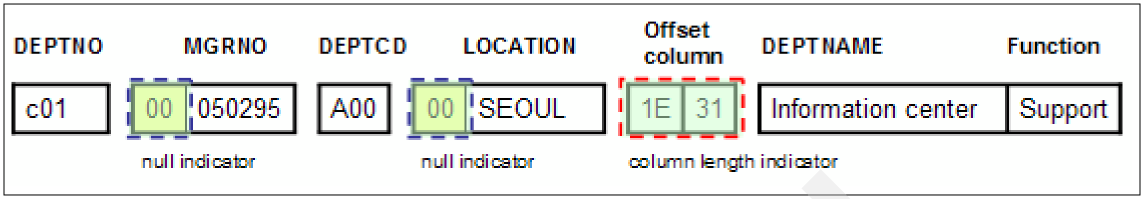


Figure 11-10 Example of adding the VARCHAR column

11.6.3 DSN1PRNT of reordered row

You can see the position of VARCHAR in the reordered row. Example 11-8 shows the sample Data Definition Language (DDL) to create the table.

Example 11-8 Creating a sample table

```
CREATE TABLE EMP_TABLE
(EMP_NO CHAR(5),
EMP_NAME VARCHAR(20),
EMP_DEPT CHAR(5),
ETC VARCHAR(100)) ;
```

We select all of the data for the test by using the following SQL statement:

```
SELECT * FROM EMP_TABLE;
```

Example 11-9 shows the results of this statement.

Example 11-9 Results of SQL SELECT statement

```
---------+---------+---------+---------+---------+---------+----
EMP_NO  EMP_NAME              EMP_DEPT  ETC
---------+---------+---------+---------+---------+---------+----
00010   HYUNKYU JEON          DW        BRONX, NY
00011   JIYOUNG JUNG          ARCHI     MANHATTAN, NY
00013   KIHWAN LIM            CCC       RIO DE JANEIRO, BRAZIL
00014   YOONDONG CHOI         DM        NEW DELHI,INDIA
00016   CHIHOON SONG          NW        BEIJING, CHINA
```

The VARCHAR column EMP_NAME (shown in bold in Example 11-10) is in the middle of the row, but the physical column order is changed.

Example 11-10 VARCHAR column EMP_NAME in middle column

```
//BAEKDSN1 JOB (999,POK),'REORG',CLASS=A,REGION=0M,
//    MSGCLASS=T,MSGLEVEL=(1,1),NOTIFY=&SYSUID
/*JOBPARM S=SC04
//JOBLIB  DD DSN=DB2S9.D2S1.SDSNLOAD,DISP=SHR
//PRINT   EXEC PGM=DSN1PRNT,PARM='PRINT,FORMAT'
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD DSN=DB2SU.DSNDBD.DSN00020.EMPRTABL.I0001.A001,
//        DISP=SHR

GSFLAGS='02'X PGSLTH=45 PGSLTH='002D'X PGSOBD='0000'X PGSBID='01'X
0010001D 00C8E8E4 D5D2E8E4 40D1C5D6 D500C2D9   .00010.DW   .....HYUNKYU
JEON.BRONX, NY

GSFLAGS='02'X PGSLTH=49 PGSLTH='0031'X PGSOBD='0000'X PGSBID='02'X
0010001D 00D1C9E8 D6E4D5C7 40D1E4D5 C700D4C1   .00011.ARCHI.....JIYOUNG
JUNG.MANHATTAN, NY

GSFLAGS='02'X PGSLTH=56 PGSLTH='0038'X PGSOBD='0000'X PGSBID='03'X
0010001B 00D2C9C8 E6C1D540 D3C9D400 D9C9D640   .00013.CCC  .....KIHWAN
LIM.RIO
C2D9C1E9 C9D3                                  DE JANEIRO, BRAZIL
```

11.6.4 Performance of the reordered row format

These tests compare the reordered row format with the basic row format. In our example, the reordered row format table is named DSN8910.PARTS, and the basic row format table is named DSN880.PARTS.

You can determine whether your table is in the basic row format or the reordered row format by querying the SYSIBM.SYSTABLEPART table using a query similar to Example 11-11.

Example 11-11 Query of the SYSIBM.SYSTABLEPART table

```
SELECT DBNAME,TSNAME,FORMAT FROM SYSIBM.SYSTABLEPART
  WHERE TSNAME = 'DSN8S81S'
  OR    TSNAME = 'DSN8S91S' ;
```

Example 11-12 shows the results table. Under the FORMAT column, you see the "R" indicating the reordered row format.

Example 11-12 Reordered table space in the catalog table

```
---------+---------+---------+---------+---------+---------+
DBNAME                    TSNAME                    FORMAT
---------+---------+---------+---------+---------+---------+
DSN8D81A                  DSN8S81S
DSN8D91A                  DSN8S91S                  R
```

To determine the table columns, you run a query like the one shown in Example 11-13.

Example 11-13 Query to determine the table columns

```
SELECT TBCREATOR,TBNAME,NAME,COLNO,COLTYPE,LENGTH
FROM SYSIBM.SYSCOLUMNS
WHERE TBNAME = 'PARTS'
AND TBCREATOR LIKE 'DSN%'
ORDER BY TBCREATOR,COLNO ;
```

Example 11-14 shows the result table from this query.

Example 11-14 Reordered row format test table

```
TBCREATOR          TBNAME       NAME         COLNO     COLTYPE          LENGTH
DSN8810            PARTS        ITEMNUM      1         VARCHAR          10
DSN8810            PARTS        DESCRIPT     2         VARCHAR          30
DSN8810            PARTS        COLOR        3         VARCHAR          8
DSN8810            PARTS        SUPPLIER     4         VARCHAR          15
DSN8810            PARTS        NAME         5         CHAR             10
DSN8810            PARTS        LNAME        6         CHAR             10
DSN8810            PARTS        COMPANY      7         CHAR             10
DSN8810            PARTS        FOOD         8         CHAR             10
DSN8910            PARTS        ITEMNUM      1         VARCHAR          10
DSN8910            PARTS        DESCRIPT     2         VARCHAR          30
DSN8910            PARTS        COLOR        3         VARCHAR          8
DSN8910            PARTS        SUPPLIER     4         VARCHAR          15
DSN8910            PARTS        NAME         5         CHAR             10
DSN8910            PARTS        LNAME        6         CHAR             10
DSN8910            PARTS        COMPANY      7         CHAR             10
DSN8910            PARTS        FOOD         8         CHAR             10
```

Figure 11-11 through Figure 11-15 on page 154 show the performance comparison of the SELECT, INSERT, UPDATE, and DELETE tests that we made between the basic row format table and the reordered row format table. All tests were run five times, and 1,000,000 rows of data were selected, inserted, updated, and deleted each time.

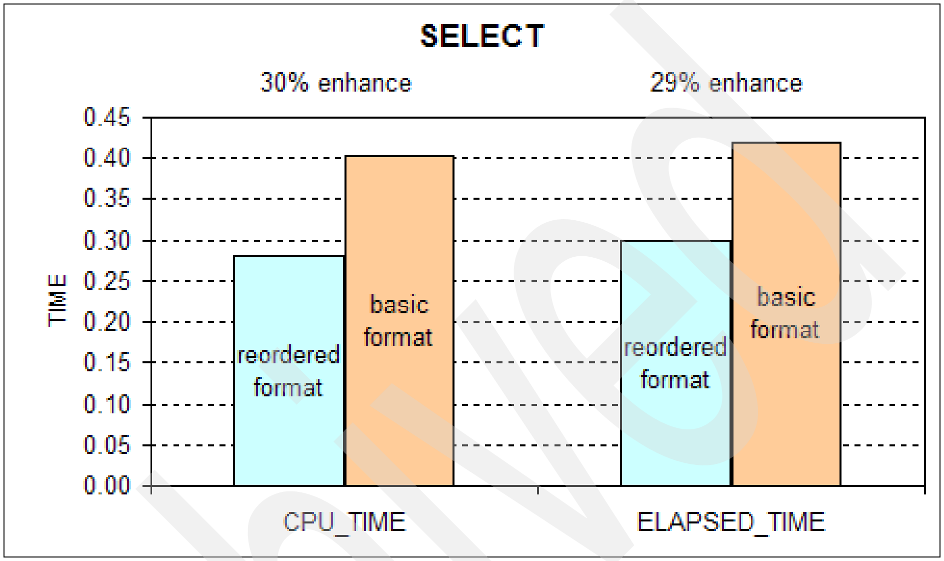


Figure 11-11 Performance test of VARCHAR enhanced SELECT test

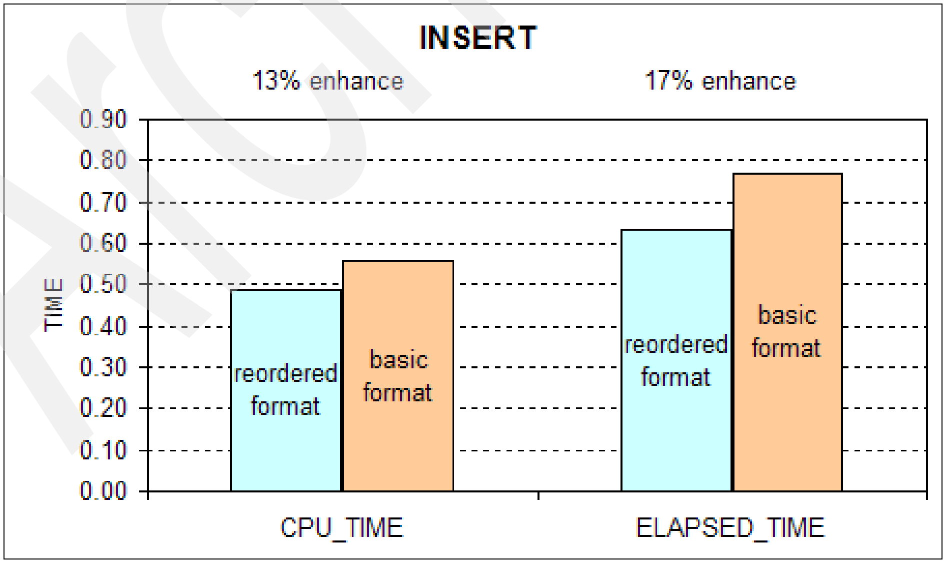


Figure 11-12 Performance test of VARCHAR enhanced INSERT test

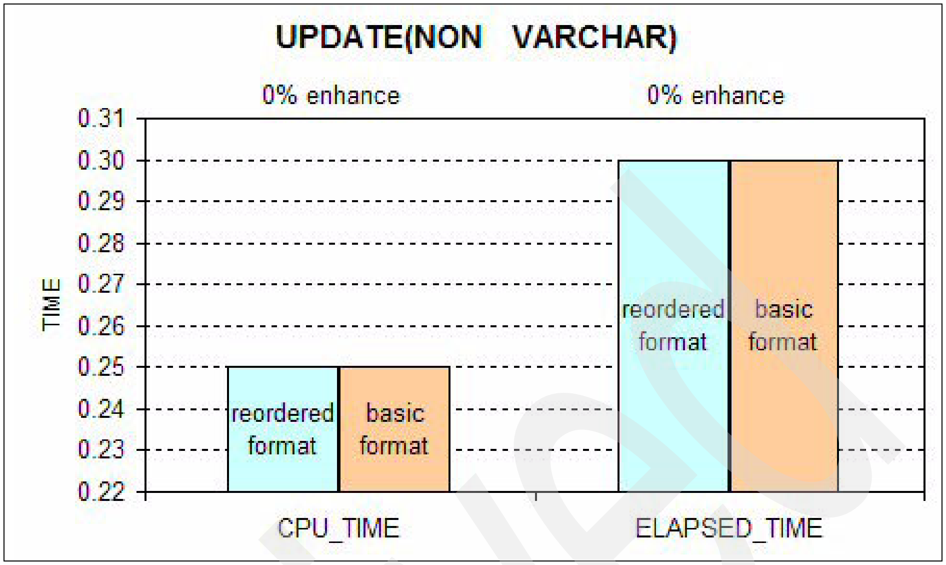


Figure 11-13 Performance test of VARCHAR enhanced UPDATE(NON VARCHAR) test

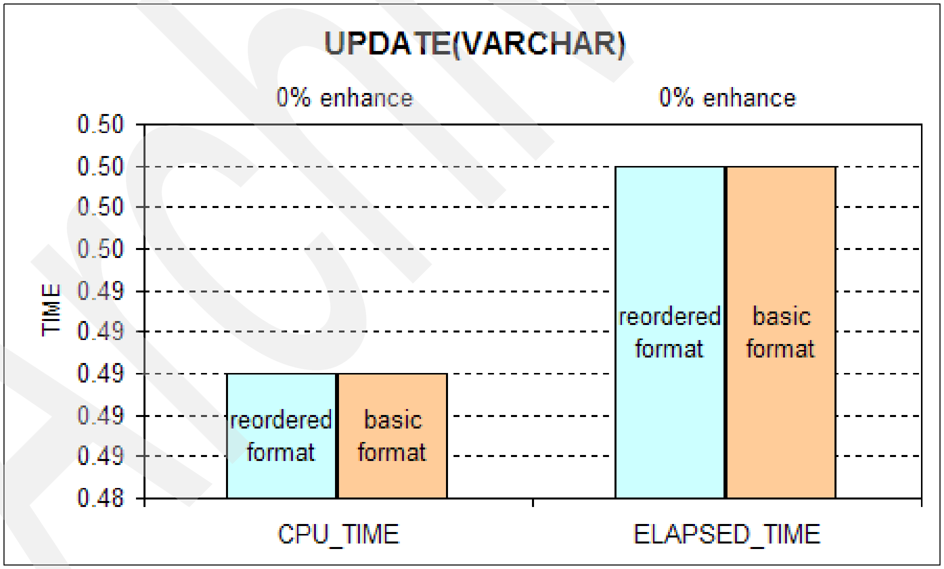


Figure 11-14 Performance test of VARCHAR enhanced UPDATE (VARCHAR) test

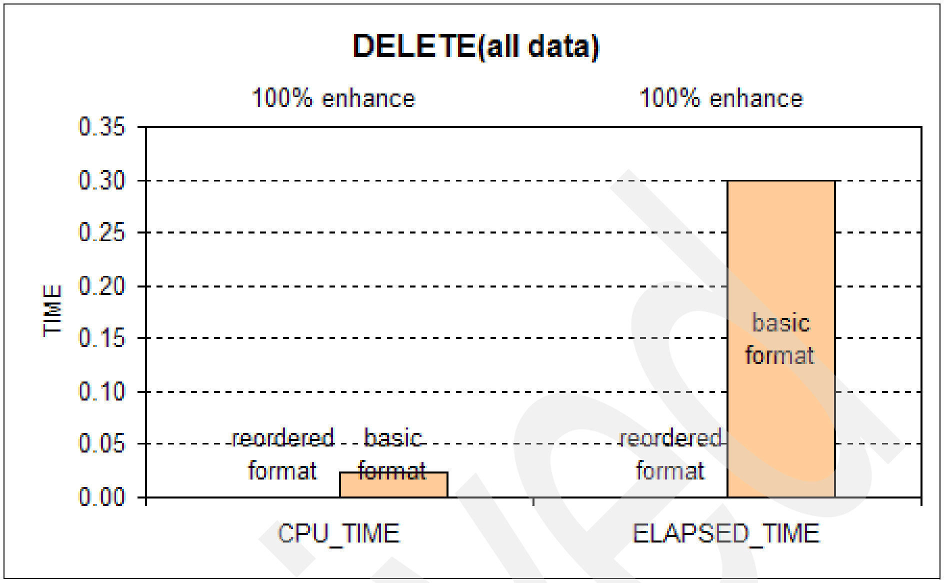


Figure 11-15 Performance test of DELETE all data

The SELECT statements on the reordered row format use approximately 30% less CPU time and elapsed time than the basic row format. INSERT statements also use 13% less CPU time and save 17% less elapsed time. In addition, a significant difference was found when deleting data (see Figure 11-15). However, there is no difference in performance between the reordered row format and the basic format using the UPDATE statement, whether the update is on VARCHAR or NON VARCHAR data.

12

New in SQL

In this chapter, we provide a glimpse of what is new in SQL for DB2 9 for z/OS and how you can use it. We discuss the following topics:

- MERGE
- TRUNCATE
- Partition by growth and APPEND
- NOT LOGGED

12.1 Updates and Inserts MERGE

MERGE is a new SQL statement that combines the conditional UPDATE and INSERT operation on a target using the values from a source. The source data that *matches* the target is used to update the target table rows. The source data that *does not match* the target is used to insert into the target table.

SAP benefits from this MERGE statement. The SAP ABAP language supports the MODIFY statement, which effectively merges new data records into a table. With DB2 9 for z/OS, MERGE is used to implement MODIFY.

Authorization

The privileges held by the authorization ID of the statement must be defined as follows:

- If the insert operation is specified, then the privilege set must include at least one of the following privileges or authorities:
  - The INSERT privilege on the table or view
  - Ownership of the table (tables only)
  - DBADM authority on the database that contains the table (tables only)
  - SYSADM authority
- If the update operation is specified, then the privilege set must include at least one of the following privileges or authorities:
  - The UPDATE privilege on the table or view
  - The UPDATE privilege on each column of the table or view to be updated
  - Ownership of the table (tables only)
  - DBADM authority on the database that contains the table (tables only)
  - SYSADM authority
- If the right side of assignment clause contains a reference to a column of the table or view, or, if the search-condition contains a reference to a column of the table or view, the privilege set that is defined below must include one of the following privileges or authorities:
  - Ownership of the table or view
  - The SELECT privilege on the table or view
  - DBADM authority on the database that contains the table (tables only)
  - SYSADM authority

- □ If the insert-operation or assignment clause includes a subquery, the privilege set must include one of the following privileges or authorities:
  - – The SELECT privilege on every table or view identified in the subquery
  - – Ownership of the tables or views identified in the subquery
  - – DBADM authority on the database that contains the table (tables only)
  - – SYSADM authority

Examples of the MERGE function

Consider Example 12-1, which shows results from selecting all rows from EMP_TABLE. It is now possible to change the existing information or add information for a new employee.

Example 12-1 Results from selecting all rows from EMP_TABLE

```
SELECT * FROM EMP_TABLE      ;
---------+---------+---------+---------+---------+----
EMP_NO  EMP_NAME         EMP_DEPT         ETC
---------+---------+---------+---------+---------+----
20928   HYBAEK           ZIM              NEWYORK
10110   SWYOON           MGR              ARIZONA
10200   KSUM             ARCH             ALASKA
10202   SHRYU            ZIM              CALIFORNIA
10020   EJJANG           ZIM              MICHIGAN
20121   HYHAN            ZIM              NEWJERSEY
```

In order to change the information using the MERGE function, note the SQL statements with their results in Example 12-2.

Example 12-2 SQL statement and results for using the MERGE function

```
MERGE INTO EMP_TABLE A
USING (VALUES('20928','HYBAEK','ZIM','MISSOURI'))
AS B (CHNO,CHNAME,CHDEPT,CHETC)
ON (EMP_NAME = CHNAME)
WHEN MATCHED THEN UPDATE SET EMP_DEPT=CHNAME, ETC=CHETC
WHEN NOT MATCHED THEN INSERT (EMP_NO,EMP_NAME,EMP_DEPT,ETC)
VALUES (CHNO,CHNAME,CHDEPT,CHETC) ;
---------+---------+---------+---------+---------+---------+-----
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0

SELECT * FROM EMP_TABLE;
---------+---------+---------+---------+---------+---------+
EMP_NO  EMP_NAME         EMP_DEPT         ETC
---------+---------+---------+---------+---------+---------+
20928   HYBAEK           HYBAEK           MISSOURI
10110   SWYOON           MGR              ARIZONA
10200   KSUM             ARCH             ALASKA
10202   SHRYU            ZIM              CALIFORNIA
10020   EJJANG           ZIM              MICHIGAN
20121   HYHAN            ZIM              NEWJERSEY
DSNE610I NUMBER OF ROWS DISPLAYED IS 6
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

You can also use the MERGE function to add a new employee as shown in Example 12-3.

Example 12-3 SQL statement using the MERGE function

```
MERGE INTO EMP_TABLE A
USING (VALUES('20041','HSPARK','ZTIVOLI','NEWMEXICO'))
AS B (CHNO,CHNAME,CHDEPT,CHETC)
ON (EMP_NAME = CHNAME)
WHEN MATCHED THEN UPDATE SET EMP_DEPT=CHNAME, ETC=CHETC
WHEN NOT MATCHED THEN INSERT (EMP_NO,EMP_NAME,EMP_DEPT,ETC)
VALUES (CHNO,CHNAME,CHDEPT,CHETC) ;
---------+---------+---------+---------+---------+---------+--
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0

SELECT * FROM EMP_TABLE;
---------+---------+---------+---------+---------+---------+-
EMP_NO  EMP_NAME         EMP_DEPT         ETC
---------+---------+---------+---------+---------+---------+-
20928   HYBAEK           HYBAEK           MISSOURI
10110   SWYOON           MGR              ARIZONA
10200   KSUM             ARCH             ALASKA
10202   SHRYU            ZIM              CALIFORNIA
10020   EJJANG           ZIM              MICHIGAN
20121   HYHAN            ZIM              NEWJERSEY
20041   HSPARK           ZTIVOLI          NEWMEXICO
DSNE610I NUMBER OF ROWS DISPLAYED IS 7
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

To insert or update multiple rows, use the statement shown in Example 12-4.

Example 12-4 SQL statement to insert or update multiple rows

```
MERGE INTO EMP_TABLE A
USING (VALUES(:hv_empno,:hv_empname,:hv_empdept,:hv_etc)
   FOR :hv_nrows ROWS)
AS B (CHNO,CHNAME,CHDEPT,CHETC)
ON (EMP_NAME = CHNAME)
WHEN MATCHED THEN UPDATE SET EMP_DEPT=CHNAME, ETC=CHETC
WHEN NOT MATCHED THEN INSERT (EMP_NO,EMP_NAME,EMP_DEPT,ETC)
VALUES (CHNO,CHNAME,CHDEPT,CHETC)
NOT ATOMIC CONTINUE ON SQLEXCEPTION
```

Note: For a static MERGE statement, if FOR n ROWS is not specified, values-multiple-row is treated as values-single-row. The NOT ATOMIC clause must be specified when values-multiple-row is specified.

12.2 TRUNCATE

The TRUNCATE TABLE statement deletes all data rows for either base tables or declared global temporary tables without activating delete triggers defined on the table (they are ignored). The base table can belong to a simple table space, a segmented table space, a partitioned table space, or a universal table space. If the table contains LOB or XML columns, its corresponding table spaces and indexes are also emptied.

DB2 transforms the new TRUNCATE TABLE operation into a mass delete operation to take advantage of the current, optimized design. It provides greater flexibility for users to deactivate existing Delete Triggers and hardens the results of the truncate operation without issuing a normal commit.

Under specific table attributes and statement options, the TRUNCATE TABLE statement may provide an optimal performance path to an empty the table while in comparison with the current mass delete operation (that is DELETE without a WHERE clause). The performance improvement can be gained only on a table with triggers are defined. Otherwise, it should perform as today. Moreover, it provides a “statement commit” option (IMMEDIATE option) for the statement that allows the truncate operation to become permanent (that is, no ability to undo). It also makes deallocated space immediately available for new, subsequent inserts in the same unit of work.

The truncate operation can run in either a normal way or a fast way. The way that is chosen depends on the following table types and its attributes. DB2 users cannot control which way the truncate statement is processed.

The *normal way* process implies that the truncate operation must process each data page to physically delete data records from the page. The normal way applies to the following situations:

- A table in a simple table space or in a partitioned table space regardless of its table attributes
- A table with either CDC-enabled, MLS-enabled or VALIDPROC-enabled attributes

The *fast way* process implies that the truncate operation deletes data records without physically processing each data page. The fast way applies to a table in a segmented table space or in a universal table space without any table attributes.

The TRUNCATE statement is useful for fast deletions of data in SAP BW processing.

Authorization

For the TRUNCATE statement, in regard to the privilege set, if the statement is embedded in an application program, the privilege set is the privileges that are held by the authorization ID of the owner of the plan or package. If the statement is dynamically prepared, the privilege set is determined by the DYNAMICRULES behavior in effect (run, bind, define, or invoke).

The privilege set must include one of the following privileges or authorities:

- The DELETE privilege for the table
- Ownership of the table
- DBADM authority for the database
- SYSADM authority

In addition, to ignore any delete triggers (that is the IGNORE DELETE TRIGGERS option is specified) that are defined on the table, the privilege set must include one of the following privileges or authorities:

- The ALTER privilege for the table
- Ownership of the table
- DBADM authority for the database
- SYSADM authority

Example for TRUNCATE and DELETE

Each table has 1,000,000 rows. The DSN8910.PARTS table space is segmented. BAEK.PARTS is also a segmented table space but DATA CAPTURE CHANGE is turned on:

```
TRUNCATE TABLE DSN8910.PARTS DROP STORAGE
IGNORE DELETE TRIGGERS ;

TRUNCATE TABLE BAEK.PARTS DROP STORAGE
IGNORE DELETE TRIGGERS ;
```

If you use DELETE without a WHERE clause, it is also fast:

```
DELETE  FROM DSN8910.PARTS ;
DELETE  FROM BAEK.PARTS ;
```

Figure 12-1 shows the difference in time between the fast way of processing and the normal way of processing. The time to process using the fast way was below '0.00'.

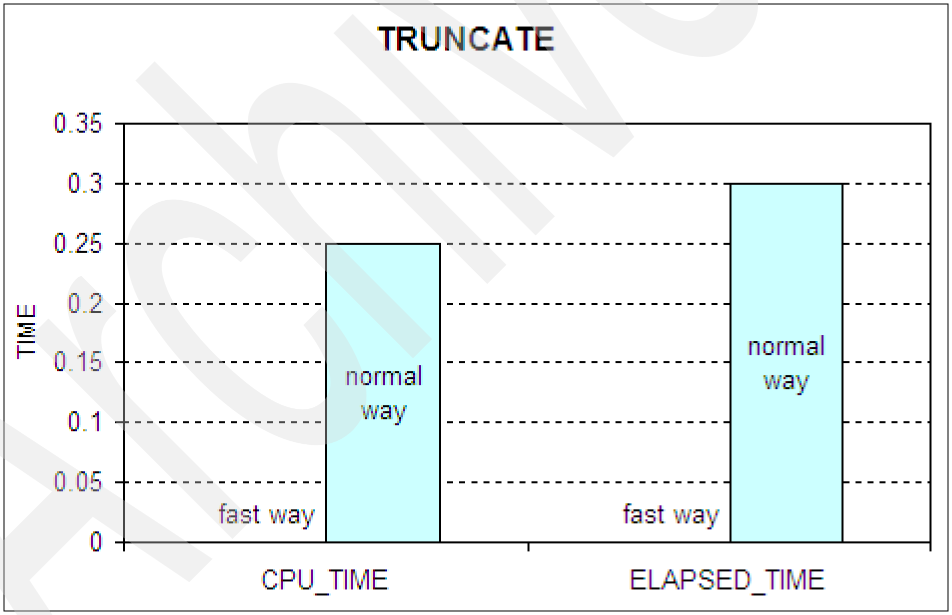


Figure 12-1 Performance test for TRUNCATE

12.3 Partition by growth and APPEND

Use the partition-by-growth table space when a table is expected to exceed 64 GB and does not have a suitable partitioning key for the table. The usage of partition-by-growth table spaces is similar to a single table DB2 managed segmented table space. Partition-by-growth table space are managed by DB2. DB2 automatically adds a new data set when the database needs more space to satisfy an insert. The table space begins as a single-partition table space and automatically grows additional partitions as needed to accommodate data growth. Partition-by-growth table spaces can grow up to 128 TB. The maximum size is determined by the MAXPARTITIONS or DSSIZE value that you specified.

Although the table space is partitioned, it has segmented organization and segmented space management capabilities within each partition. Unlike a non-segmented structure, the segmented structure provides better space management and mass delete capabilities. The partitioning structure allows DB2 utilities to continue partition-level operations and parallelism capabilities.

The APPEND processing option is treated like a TABLE trait. It is specified on the CREATE TABLE statement or may be altered via the ALTER TABLE statement.

The APPEND processing option instructs DB2 to ignore clustering during SQL insert and online LOAD processing. Rather than attempting to insert rows in cluster-preserving order, rows are appended at the end of the table or the appropriate partition. For range-partitioned table spaces, the appropriate partition is the one dictated by the value in the row's partitioning column. For growth-partitioned table spaces, the appropriate partition is any partition with space at the end.

SAP applications can accelerate high volume INSERT processing with APPEND.

Restrictions

Note the following restrictions for using partition by growth:

- If the SEGSIZE keyword is not specified, a default value of 4 is given.
- Table spaces must be DB2 stogroup-managed (not user-managed). This gives DB2 the freedom to create data sets as partitions fill.
- Table spaces cannot be created with the MEMBER CLUSTER option.
- Partitions cannot be explicitly added, rotated, or altered. This means that ALTER TABLE ADD PARTITION, ALTER TABLE ROTATE PARTITION, or ALTER TABLE ALTER PARTITION cannot target a partition of a partition-by-growth table space.

- LOB and XML spaces are always implicitly defined by DB2. They are independent of whether SQLRULES (DB2) or SQLRULES (STD) are in effect.
- A nonpartitioning index (NPI) always uses a 5-byte record identifier (RID).
- Only single tables are supported.
- Partitioned indexes are not supported.

Usage

Partition by growth is useful if a table's growth is unpredictable (it could exceed 64 GB) and there is no convenient key for range partitioning.

You can implement partition by growth in two ways:

- Explicit specification is CREATE TABLESPACE ... MAXPARTITIONS integer.
- Implicit specification is CREATE PARTITIONED BY SIZE EVERY integer G.

The associated SYSTABLESPACES columns are:

- MAXPARTITIONS = max number of partitions
- PARTITIONS = actual number of partitions
- TYPE = G

The benefit of having a clustering index with the APPEND processing option (table trait) is that the clustering order is honored by the REORG and offline LOAD utilities. Therefore, for example, after populating or growing a table via speedy APPEND-obeying INSERTs, the table space can be reorganized to produce a cluster order to the benefit of query processing.

> **Note:** In the future, SAP will create all table spaces as a partition-by-growth table space, which makes it easier to manage DB2.

12.4 Using NOT LOGGED table spaces

You can specify a new logging attribute at a tablespace level. You can specify a LOGGED attribute to indicate that modifications should be recorded on the log. Or you can specify a NOT LOGGED attribute to indicate that modifications should not be recorded on the log. The default logging attribute is LOGGED, except for table spaces in the workfile database and LOB table spaces that contain LOBs with a maximum length greater than 1 GB. For these types of table spaces, the logging attribute is always NOT LOGGED; you cannot specify a logging attribute of LOGGED.

Suppressing the writing of log records, in general, does not improve the performance of your system. The logging facilities in DB2 are specifically tailored to the way that DB2 writes log records and has been finely tuned. Do not sacrifice the recoverability of your data in an attempt to gain performance. In the vast majority of situations, you will not be able to notice a difference in performance after the writing of log records is suppressed. There are several other ways to avoid logging, such as by using LOAD, work files, declared and global temporary tables, LOBs, and REORG LOG NO.

Usage

A *materialized query table* (MQT) is a summarization of data that exists elsewhere. Therefore, MQTs are prime candidates for the NOT LOGGED attribute. Since an MQT can be rebuilt from data that exists elsewhere, there is no need for the changes to the MQT to be logged.

It may also be advantageous to avoid writing log records for changes to tables to which data is propagated. Since the data exists elsewhere, there is not necessarily a reason to log the changes as they are applied to the copied table. If the information becomes damaged, the entire table can be refreshed from its source.

While SAP does not use many MQTs, the most important use case for NOT LOGGED table spaces in SAP is for loading mass amounts of data into DB2 in a faster and more efficient way, for example, when performing a UNICODE migration. It is also useful in an SAP BI/BW query and reporting environment where temporary tables are populated with the result set from a query as an intermediate step by the application and then used by a subsequent query step (join, subset). The results are temporary, and deleted by the application. The data can be recovered by rerunning the application.

Performance

We inserted 1,000,000 rows into each table. One is a LOGGED table space, and the other is a NOT LOGGED table space. Total COLUMN length is about 40 bytes. We found that the larger column length shows the better performance (Figure 12-2).

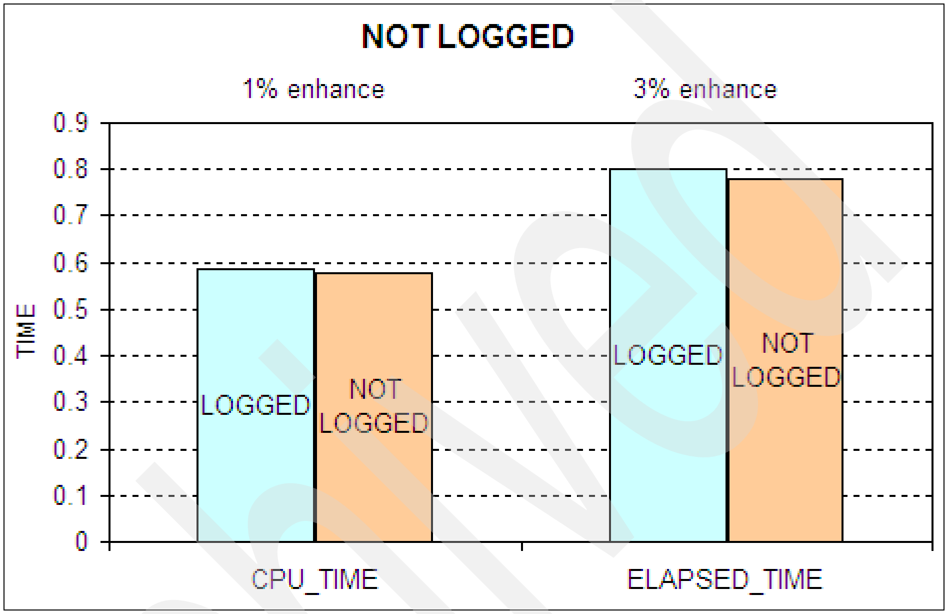


Figure 12-2 Performance test for NOT LOGGED table space

13

Administration

In this chapter, we look at the administrative enhancements that are available in DB2 9 for z/OS. These enhancements include:

- Converge temp space
- Enhanced 64-bit virtual addressing support
- Implicit objects creation using data definition language (DDL)
- Non-partitioned table size limit
- Smart fast mass delete of partitions
- Resource limit facility support for client identifiers

13.1 Converge temp space

Up to version 8, DB2 has supported two databases for temporary files and temporary tables: the WORKFILE database and the TEMP database. The 'AS' clause of the CREATE DATABASE statement with either the 'WORKFILE' or 'TEMP' subclause indicates that the database to be created is either a WORKFILE database or a TEMP database. Each DB2 subsystem or data sharing member has one WORKFILE database and may have one TEMP database.

The WORKFILE database is used by DB2 for storing *created global temporary tables*. It is also used as storage for work files for processing SQL statements that require temporary working space for sorts, materializing views, triggers, nested table expressions, and others.

The TEMP database is used by DB2 for storing the user-defined declared global temporary tables and the DB2 declared global temporary tables for static scrollable cursors.

To facilitate operations, DB2 9 for z/OS converges these two temporary space structures into one: the WORKFILE database. Therefore, there is no longer a need to optimize the setup of two structures and balance them against each other. The WORKFILE database is used as storage for all temporary files and tables, while preserving the external functionality of the temporary tables as it is today. The temporary tables that are currently used by the TEMP database are switched to use the WORKFILE database.

A further advantage of exclusively relying on the WORKFILE database is that this database is created during the installation of a DB2 subsystem. Therefore, it can always be assumed to be available and does not need to be created later by such applications as SAP. Because SAP uses global temporary tables, there have been mechanisms inside the Computing Center Management System to ensure that the TEMP database with its table spaces is created when necessary. This coding is no longer necessary, which streamlines the processing, and is less error prone.

13.2 Enhanced 64-bit virtual addressing support

While most of the DB2 storage structures of the DBM1 address space reside above the 2 GB bar with DB2 V8, some structures jointly use the storage below this bar. To prevent structures from exhausting the virtual storage in this 2 GB area, we recommend that you properly plan for the virtual storage needs of your SAP systems.

DB2 9 for z/OS exploits 64-bit virtual storage for more storage structures that reside in the DBM1 address space. It moves parts of these structures, which used to be restricted to 2 GB of virtual storage in DBM1 in previous releases of DB2, to the storage area above the 2 GB bar of DBM1. More specifically, portions of the EDM pool and of the local dynamic statement cache are moved above the 2 GB bar. Thus, these portions can exploit 64-bit storage.

The SKCT/SKPT pages of the EDM pool move above the bar. With SAP, the predominant consumer of these pages is DB2 Connect with its packages. In this case, these pages typically consume about 25% of the EDM pool. Compared to DB2 V8, around 25% of the size of the EDM pool below the 2 GB bar can be saved when migrating to DB2 9 for z/OS.

Concerning the storage for the local dynamic statement cache below the 2 GB bar, around 70% of this storage moves above the bar. There are significant savings for Insert, Update, and Delete operations. Savings can also be observed for input and output runtime structures of SELECT statements.

Further enhancements in the area of virtual storage below the 2 GB bar in the DBM1 address space have been provided via DB2 V8 APARs. These APARs are:

- PK21237
- PK21861
- PK23523

13.3 Implicit objects creation using DDL

In this section, we discuss implicit objects created using DDL statements.

13.3.1 Automatic creation of objects

DB2 implicitly creates a database if the user does not specify an IN clause or database name on a CREATE TABLE statement. Also, if the containing table space is implicitly created, DB2 creates all the system required objects for the user, such as the enforcing primary key index, enforcing unique key index, and ROWID index on which the column is defined as ROWID GENERATED BY DEFAULT and LOB objects. DB2 9 for z/OS no longer implicitly creates simple table spaces or allows customers to create simple table spaces. However, DB2 still supports simple table spaces that were created prior to V9.

When you use the CREATE TABLESPACE statement and do not specify SEGSIZE, NUMPARTS, or MAXPARTITIONS, a segmented table space with segment size of 4 is created, instead of a simple table space. For an implicitly created segmented table space, the default is always SEGSIZE 4 and LOCKSIZE ROW.

When you use the CREATE TABLE statement with table space or a database, they are created for you by default. The database that is created is created dynamically. This is based on a new subsystem parameter, IMPDB. If it is set to YES, then the database is created. If IMPDB is set to NO, then it defaults to the DSNDB04 database. The naming convention used for the databases is DSN00001 to DSN60000. If the count of 60,000 is reached, the database creation wraps around. Authorization is not checked when you create an implicit table.

13.3.2 New function mode in DB2 9 for z/OS

In DB2 Version 9, the following new system parameters are introduced:

- IMPDSDEF: A new subsystem parameter that corresponds to DEFINE YES and DEFINE NO on a CREATE TABLESPACE statement. It allows the user to specify whether the underlying data set is to be defined when a table space is created in an implicitly created database. The default is DEFINE NO. This field is only valid when IMPDB is YES.
- IMPTSCMP: A new subsystem parameter that corresponds to COMPRESS YES and COMPRESS NO on a CREATE TABLESPACE statement. It permits the user to specify whether the data compression is to be enabled when a table space is created in an implicitly created database. The default is COMPRESS NO. This field only valid when IMPDB is YES.

These features can vastly improve performance for SAP customers.

13.4 Nonpartitioned table size limit

Partition by growth supports large tables without having a partitioning key. A partition-by-growth table space begins as a single-partition table space and automatically grows additional partitions as needed to accommodate data growth. Although such table spaces are partitioned, they possess segmented organization and segmented space management capabilities within each partition. The segmented structure affords better space management and mass delete capabilities over a non-segmented structure.

The underlying partitioning allows utilities to, by and large, continue to provide their partition-level operation and parallelism capabilities. One exception to this is that LOAD PART is not supported for partition-by-growth table spaces.

When does it make sense to partition by growth? Use partition by growth when a table is the only one in its table space, is expected to exceed 64 GB, and does not lend itself to key-range partitioning (that is, it does not have a suitable partitioning key).

When you apply partition by growth at the table-level, the option instructs DB2 to ignore clustering when using SQL to populate a table. This produces a fast insert at the expense of data organization and rapid table space growth.

13.4.1 Partition by growth table space

When you explicitly create a partition-by-growth table space, the new MAXPARTITIONS clause (as opposed to the existing NUMPARTS clause) is specified on the CREATE TABLESPACE statement. The MAXPARTITIONS clause identifies the maximum number of partitions to which the partition-by-growth table space may grow. This cap can protect against run-away applications, for example an insert in an infinite loop, and can be adjusted via ALTER TABLESPACE.

The maximum partition size is based on DSSIZE. The default value is 4 GB. When the CREATE TABLESPACE statement is executed, one partition is then created. Subsequent partitions are created when the partition size is reached and keeps doing so until the value of MAXPARTITIONS is reached.

13.4.2 Partition by growth using CREATE TABLE

Creating an implicit table and employing partition by growth is where you can find value add for your system. You do this by using the new PARTITION BY SIZE clause, which is introduced for the CREATE TABLE statement. This only applies when you create a table without specifying the table space that is to house the table.

DB2 implicitly creates a table space to house the table. If no range partitioning is requested on the CREATE TABLE statement (that is, the PARTITION BY <RANGE> clause is not used), then the implicitly created table space is partition by growth.

The default segsize is 4, the maximum size of each partition is 4 GB, and the maximum number of partitions is 256. This limits the default size of the table to 1 TB. If a different partition size is desired, you can use the PARTITION BY SIZE clause; that is "PARTITION BY SIZE EVERY integer G". The integer gives the maximum size desired for each partition in gigabytes. The maximum number of partitions is still 256.

Note that the 256 partition limit is a default limit chosen by DB2 for implicitly created table spaces. If the limit is inadequate, you can raise it by altering the table space's MAXPARTITIONS via ALTER TABLESPACE. The exception is in the special case of PARTITION BY SIZE EVERY 64G when the page size is 4K, in which case we are already at the architectural limit with 256 partitions. An existing SQLCODE -644 SQLSTATE = 42615 is issued when this architectural limit is reached. The reason for not defaulting to the architectural limit when creating implicit table spaces is to try to find a happy medium between runtime storage requirements and table space growth potential. Unused potential consumes runtime storage.)

Partition by growth using CREATE TABLE can be particularly helpful for LOB and XML tables.

13.4.3 Example of partition by growth

To create a partition-by-growth table space that has a maximum size of 2 GB for each partition, 4 pages per segment, and a maximum of 24 partitions for table space, you use an SQL statement like the one in Example 13-1.

Example 13-1 SQL to create a partition-by-growth table space

```
CREATE TABLESPACE TEST01TS IN TEST01DB USING STOGROUP SG1
DSSIZE 2G
MAXPARTITIONS 24
LOCKSIZE ANY
SEGSIZE 4;
COMMIT;
```

To alter the maximum number of partitions on the partition-by-growth table space, see Example 13-2.

Example 13-2 SQL to alter the maximum number of partitions

```
ALTER TABLESPACE TEST01DB.TEST01TS
MAXPARTITIONS 30 ;
COMMIT;
```

To create a table in a partition-by-growth table space, see Example 13-3.

Example 13-3 SQL to create a table in a partition-by-growth table space

```
CREATE TABLE TEST01TB (
C1 SMALLINT,
C2 DECIMAL(9,2),
C3 CHAR(4))
IN TEST01DB.TEST01TS;
COMMIT;
```

To create a table in a partition-by-growth table space where the table space is implicitly created by a CREATE TABLE statement, see Example 13-4.

Example 13-4 SQL to create a table in a partition-by-growth table space

```
CREATE TABLE TEST02TB (
C1 SMALLINT,
C2 DECIMAL(9,2),
C3 CHAR(4))
PARTITIONING BY SIZE EVERY 4G
IN TEST02DB;
COMMIT;
```

To create or alter a table with APPEND attributes, see Example 13-5.

Example 13-5 SQL to create or alter a table with APPEND attributes

```
CREATE TABLE TEST03TB (
C1 SMALLINT,
C2 DECIMAL(9,2),
C3 CHAR(4))
APPEND YES
IN TEST03DB.TEST03TS;
COMMIT;
ALTER TABLE TEST03TB
APPEND NO
```

13.5 Smart fast mass delete of tables

DB2 9 for z/OS introduces the TRUNCATE TABLE statement, which provides a fast way to delete data from an entire table. This may be a good method to use for emptying data from the entire table space before applying data from DSN1COPY.

The base table can be in a simple table space, a segmented table space, a partitioned table space, or a universal table space. If the table contains LOB or XML columns, the corresponding table spaces and indexes are also truncated.

Truncate can be issued in a program or interactively. To execute TRUNCATE, you should have the following privileges and authorities:

- The DELETE privilege for the table
- Ownership of the table
- DBADM authority for the database
- SYSADM authority

You can find the SYNTAX for TRUNCATE in *DB2 Version 9.1 for z/OS SQL Reference*, SC18-9854. This reference also contains a description of all the features of TRUNCATE.

13.5.1 Examples of TRUNCATE

We found excellent examples of the TRUNCATE statement in *DB2 Version 9.1 for z/OS SQL Reference*, SC18-9854. We share a few of them here:

- **Example 1**: Empty an unused inventory table regardless of any existing triggers and return its allocated space.

  Example 13-6 SQL to empty an unused table using the TRUNCATE statement

  ```
  TRUNCATE TABLE INVENTORY
         IGNORE DELETE TRIGGERS
         DROP STORAGE;
  ```

- **Example 2**: Empty an unused inventory table regardless of any existing delete triggers but preserve its allocated space for later reuse.

  Example 13-7 SQL to empty an unused table but preserve allocated space

  ```
  TRUNCATE TABLE INVENTORY
        REUSE STORAGE
        IGNORE DELETE TRIGGERS;
  ```

- **Example 3**: Empty an unused inventory table permanently (a ROLLBACK statement cannot undo the truncate operation when the IMMEDIATE option is specified) regardless of any existing delete triggers and preserve its allocated space for immediate use.

Example 13-8 SQL to empty an unused table, ignore triggers, and reuse storage

```
TRUNCATE TABLE INVENTORY
       REUSE STORAGE
       IGNORE DELETE TRIGGERS
       IMMEDIATE;
```

13.6 Resource limit facility support for client identifiers

The DB2 resource limit facility (RLF) is a feature that governs the DB2 resources, such as process time. The current RLF and DB2 governor maintains resources by plan, package, or collection name with the authid or the Luname.

In the every day world of applications and workloads, authid, plan, and package names are not unique. This presents problems in maintaining these resources to DB2. A big problem is that middleware servers do not access DB2 with unique individual plans, collections, packages, and authids. If the middleware servers access DB2 through DDF, they use the same plan name, which is DISTSERV. If they use the CLI and JDBC APIs, then a common set of packages is used.

The DB2 RLF function has been enhanced so that you can govern your resources to your middleware server more efficiently.

Enhancements to manage and administer SAP on DB2 9 for z/OS

In this chapter, we discuss the DB2 9 for z/OS enhancements to manage and administer the database and the SAP systems. Manageability is improved in DB2 Version 9.1 with several new enhancements such as:

- Default enablement of autonomic computing features
- CREATE and ALTER TABLE statement enhancements
- Automatic storage support for multipartition databases
- Support for copying schemas between databases
- Enhanced access to DB2 administration commands through SQL
- Automatic table and index reorganization enhancements
- Load support for partitioned tables

These new features and others allow you to spend less time administering your databases and more time managing your business.

14.1 DB2 administration of SAP NetWeaver: An introduction

We recommend that you define and implement an Operation Concept, based on the IT Infrastructure Library® (ITIL®) standard, to administer SAP Data Center.

In DB2 9 for z/OS, new automated database administration features continue to help improve the productivity and effectiveness of your database administrators (DBAs). The DBA is usually responsible for the following tasks:

- To provide availability and performance according to the service-level agreement (SLA) by doing either of the following actions:
  - Monitoring the database
  - Controlling the DB2 using automated tools
- To organize the health of DB2 by using manual or fully automatized cleanup routines, such us:
  - Deleting the old spool data
  - Deleting unnecessary dumps, trace results and logs
  - Controlling the database statistics
  - Reorganizing the tables and indexes
- To ensure database consistency in case of errors, failures, and loss of data by defining backup and recovery policies and implementing them

The Operation Concept should also define the responsibilities. You can assign the role for the DBA to various teams or groups in your enterprise, such as:

- System programmers
- SAP Basis
- Database administrators

ITIL describes the IT processes, such as:

- Operation Management
- Problem Management
- Performance Management
- Security Management

14.1.1 Administration tools

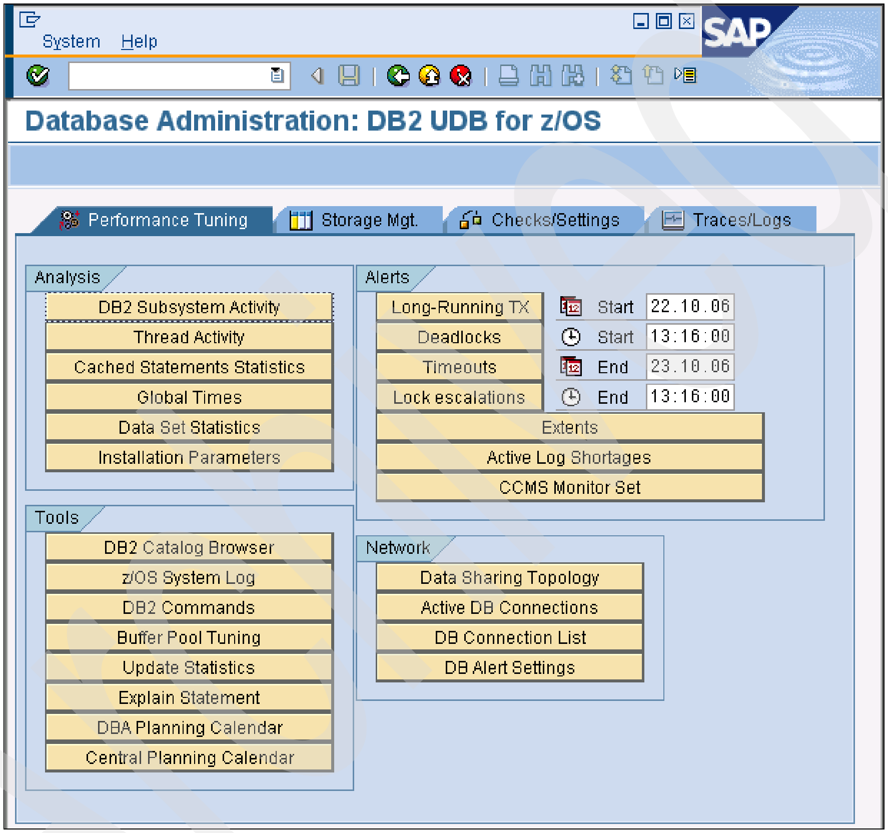
For the implementation of the Operation Management process, you can use a wide variety of tools, such as:

- DB2 Administration Tools in SAP NetWeaver
- DB2 SAP Edition Toolkits provided by IBM

- DB2 Control Center (CC)
- DB2 Tools provided by other vendors

DB2 Administration Tools in SAP NetWeaver

SAP NetWeaver provides transaction ST04 for the Database Administration activities. Figure 14-1 shows the available DB2 Administration Tools included in SAP NetWeaver.



Database Administration: DB2 UDB for z/OS

Start 22.10.06
Start 13:16:00
End 23.10.06
End 13:16:00

Figure 14-1 DB2 Administration using the ST04 transaction

The SAP transaction for the DB2 Administrator ST04 provides tools to manage and control the DB2 subsystem. The tools are available using the register for Storage Management and Performance and Tuning and include:

- DB2 Catalog Browser to review the DB2 catalog
- z/OS System Log to show the z/OS system log
- DB2 Commands to manage the DB2 subsystem
- Buffer Pool Tuning to adapt the DB2 buffer pools
- Update Statistics to updated the statistics for a single object

- Explain Statement to check the access path selected by the optimizer
- DBA Planning Calendar to schedule the DB2 maintenance jobs
- Central Planning Calendar
- Mass Processing to perform mass processing activities
- Empty DB Objects to display the empty tables

Selecting the register for Checks/Settings and Traces/Logs for the ST04 transaction provides the following functions:

- Control and analyze the provided traces for:
  - The SAP system
  - DB2 Connect
  - ABAP
  - SQL statements
  - The Database Service Layer (DBSL)
  - DB2 IFI
- Check the necessary connectivity between the SAP Database Server and the SAP Application Server based on the rfcoscol and saposcol routines

DB2 SAP Edition Toolkits provided by IBM

IBM DB2 Tools are available to support DB2 9 for z/OS. These tools offer many innovative features to assist DBAs and system programmers in managing many aspects of DB2 database environments.

IBM DB2 Tools now deliver the capability to manage your SAP enterprise resource planning (ERP) environment, in addition to the technological leadership that they bring to your data center today. DB2 Tools have made a positive difference in total cost of ownership for hundreds of customers worldwide. This value is now extended to customers with SAP ERP applications running on a dedicated System z machine with the SAP Editions of the following toolkits:

- DB2 Automation Toolkit for z/OS, the SAP Edition

  This toolkit advances the goal of enterprise autonomic computing and offers enhancements to ERP environments. The DB2 Automation Toolkit provides an easy-to-use interface to the DB2 Utilities Suite, which offers a full set of utilities that handle a variety of tasks that help improve productivity, performance, and availability. All the functions of DB2 Automation Toolkit for z/OS are available to your SAP systems.

  The DB2 Automation Toolkit supports SAP NetWeaver environments by automating the execution of copy, reorg, and other utilities to provide better management of system resources.

  You may define exception criteria for the execution of utilities against an object or group of objects. After this definition is done, it is used each time the DB2 Automation Toolkit checks to determine if a utility needs to be executed.

From an autonomic standpoint, after profiles are set up, objects are dynamically added or deleted from processing. This saves time in manual procedures and avoids job control language (JCL) errors.

LOAD balancing is supported in a variety of ways to give your users more control over system resources.

A variety of statistical sources are available to allow you to select the one that best applies to your situation, for example real-time statistics (RTS). You have two choices in how RTS are checked. You can have the DB2 Automation Toolkit do it directly for you, or you can have the DB2 Automation Toolkit use the DSNACCOR stored procedure.

An object counter is displayed on the screen, for example, 50 of 12,000. This feature helps your DBA by giving an automatic and error-free object counter. This is important because SAP NetWeaver environments have to display large numbers of objects.

- DB2 Administration Toolkit for z/OS, the SAP Edition

  The DB2 Administration Toolkit consists of two tools:

  - DB2 Administration Tool for z/OS
  - DB2 Object Comparison Tool for z/OS

  These tools work together to form a function-rich object management solution. All the functions of DB2 Administration Toolkit and DB2 Object Comparison Toolkit are available to SAP NetWeaver.

  DB2 Administration Tool reduces the complexity of managing and executing SAP database changes, which enables your DBA to make faster and more accurate decisions. It also provides important features for the administration of SAP systems. Of particular importance to SAP, DB2 Administration Tool supports the following functions:

  - Easy and fast navigation of a DB2 catalog, allowing you to issue DB2 commands, execute utilities, and examine schema structure
  - Changes of all ZPARMS and explains how to use the “online ZPARM” feature of DB2

  DB2 Object Comparison Tool support for SAP promotes database object integrity when objects are transported from test to production. Of particular importance to SAP, it supports the following functions:

  - Provides quality assurance by comparing objects in SAP NetWeaver databases; helps to identify potential risks or long running changes
  - Alerts your DBA of any user-defined objects, usually indexes, that will be affected by applying the change

- DB2 Performance Toolkits for z/OS, the SAP Edition

  This toolkit consists of two tools:

  - IBM Tivoli® OMEGAMON XE for DB2 Performance Expert on z/OS
  - DB2 Query Monitor for z/OS

  All the functions of DB2 Performance Expert for z/OS V2 and DB2 Query Monitor V2 are available to your SAP systems.

  Tivoli OMEGAMON XE for DB2 Performance Expert builds on the IBM expertise in autonomic computing by providing recommendations for system tuning to gain optimum throughput. Of particular importance to SAP environments, the Tivoli OMEGAMON XE for DB2 Performance Expert supports the following functions:

  - Optimizes the time of your DBA by providing timely and relevant statistics to efficiently make better decisions and recommend actions to tune performance. Specifically provided are the following functions:

    - Rule-of-thumb recommendations on how to tune performance
    - Snapshot™ history on statistics, accounting, and long-term history in Performance Warehouse (PWH)
    - Interval mode and “since reset” support history as opposed to only information from DB2 “start” or from “reset start”
    - A complete data sharing view

      Without DB2 Performance Expert, a data sharing view is available only for Thread Activity and Cached Statements.

    - Network statistics that correlate with DB2 thread, performance check, OS statistics included to detect problems outside DB2, and so on
    - Richer object location simulation through Buffer Pool Analyzer, since the simulation is not restricted to SAP ERP statistics
    - Reporting capabilities for System Management Facilities (SMF) data
    - Capability to define and observe threshold alerts

  - Optimizes system resources by providing query parallelism support and flexibility on how performance data is stored and monitored. Specifically provided are the following features:

    - A complete history of all alerts rather than a subset of alert history
    - CPU and SYSPLEX query parallelism support
    - Flexibility in how collected performance data is stored to help reduce workload on the SAP ERP system

  The DB2 Query Monitor for z/OS enables you to customize and tune your SQL workload to improve performance. It also identifies resource-intensive

SQL that needs to be tuned. In an SAP environment, it is of particular importance that DB2 Query Monitor supports the following functions:

- Provides reports and statistics
- Identifies objects accessed for each SQL statement executed
- Provides statement-level alerts for SQL statements that exceed specified performance thresholds
- Captures comprehensive statistics for SQL statements that trigger alerts
- Allows viewing of alerts and performance data from multiple subsystems on multiple logical partitions (LPARs)

14.1.2 Authorizations in SAP for the DB2 administrator

Finally, you should assign the necessary authorizations in SAP to the database administration roles. To administer DB2 in the SAP NetWeaver environment in an effective and productive way, we recommend that you provide access to the transactions listed in Table 14-1.

Table 14-1 Transaction codes and descriptions

Transaction code	Description
DB02	Tables and Indexes Monitor
DB03	Database Parameters
DB12	Backup Monitor
DB13	DBA Planning Guide
DB2C	DB2 Catalog Browser
DB2J	R/3 JES Interface
RZ20	Alert Monitor
S001	ABAP Workbench
SE11	ABAP Dictionary
SE14	Database Utility
SE16	Data Browser
SE17	General Table Display
SE30	ABAP Runtime Analysis (SQL statement compare)
SE38	ABAP Editor

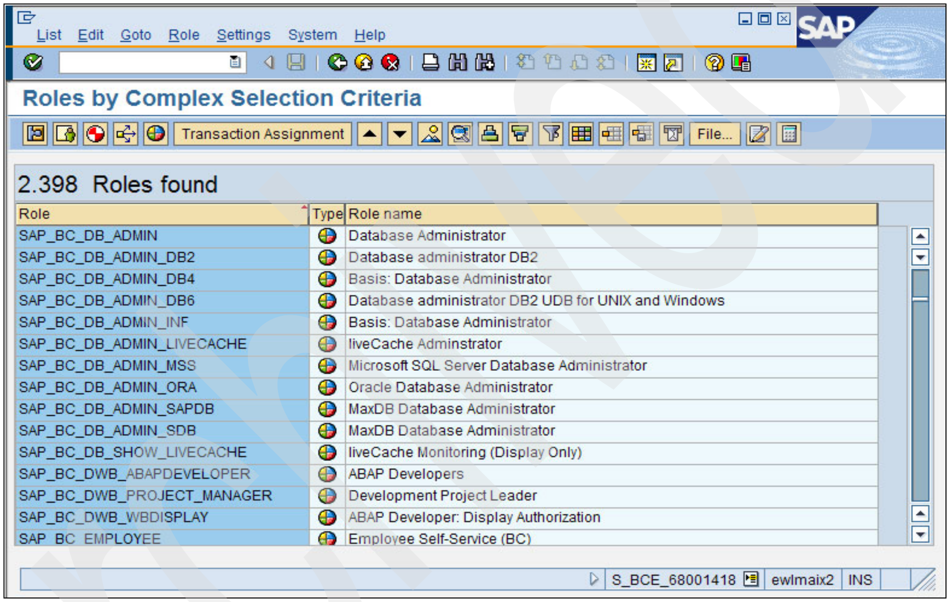
Transaction code	Description
SE80	Repository Browser
SE84	Repository Information System
SM04	Current users on current server
SM13	Update Records
SM21	System Log: Local Analysis
SM31	Table Maintenance
SM37	Background job list
SM39	Background Information
SM50	Work Process Overview
SM51	Server Overview
SPO1	Spooler
ST02	Tuning Summary
ST03	Workload Analysis
ST04	Database Performance Monitor
ST05	Trace Requests
ST06	Operating System Overview
ST07	Application Monitoring
ST08	Network Monitoring
ST10	Table Call Statistics
ST11	Trace Logs
ST22	ABAP Dump Analysis

14.1.3 Role-based administration

SAP NetWeaver provides more than one thousand user and administrator roles. The DBA is responsible for regular day-to-day maintenance and support of DB2.

DB2 9 for z/OS provides the definition and implementation of roles. For details, see Chapter 6, “Security” on page 59. The SAP NetWeaver System provides many predefined roles that can be displayed with the SU01 transaction by using the `/nsu01` command.

Figure 14-2 shows predefined DB ADMIN roles in SAP.



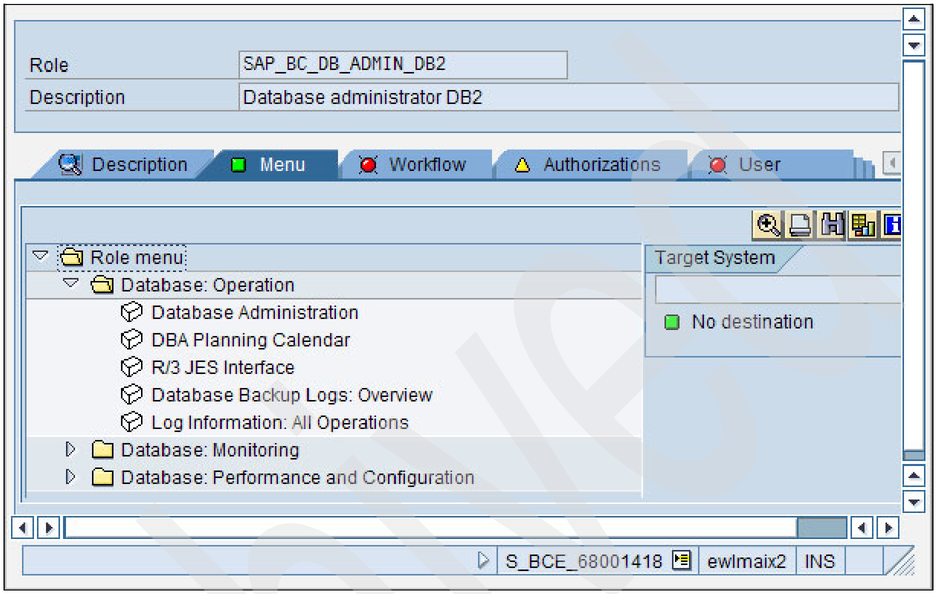
Roles by Complex Selection Criteria

2.398 Roles found

Role	Type	Role name
SAP_BC_DB_ADMIN	●	Database Administrator
SAP_BC_DB_ADMIN_DB2	●	Database administrator DB2
SAP_BC_DB_ADMIN_DB4	●	Basis: Database Administrator
SAP_BC_DB_ADMIN_DB6	●	Database administrator DB2 UDB for UNIX and Windows
SAP_BC_DB_ADMIN_INF	●	Basis: Database Administrator
SAP_BC_DB_ADMIN_LIVECACHE	●	liveCache Adminstrator
SAP_BC_DB_ADMIN_MSS	●	Microsoft SQL Server Database Administrator
SAP_BC_DB_ADMIN_ORA	●	Oracle Database Administrator
SAP_BC_DB_ADMIN_SAPDB	●	MaxDB Database Administrator
SAP_BC_DB_ADMIN_SDB	●	MaxDB Database Administrator
SAP_BC_DB_SHOW_LIVECACHE	●	liveCache Monitoring (Display Only)
SAP_BC_DWB_ABAPDEVELOPER	●	ABAP Developers
SAP_BC_DWB_PROJECT_MANAGER	●	Development Project Leader
SAP_BC_DWB_WBDISPLAY	●	ABAP Developer: Display Authorization
SAP_BC_EMPLOYEE	●	Employee Self-Service (BC)

Figure 14-2 The predefined roles in SAP

Figure 14-3 shows the predefined role for the DB2 administrator in SAP NetWeaver. You can create a new DB2 administrator role and add additional transactions, or change the authorizations. The authorizations define the access mode to the transactions and objects, such as read, write, change, and delete.



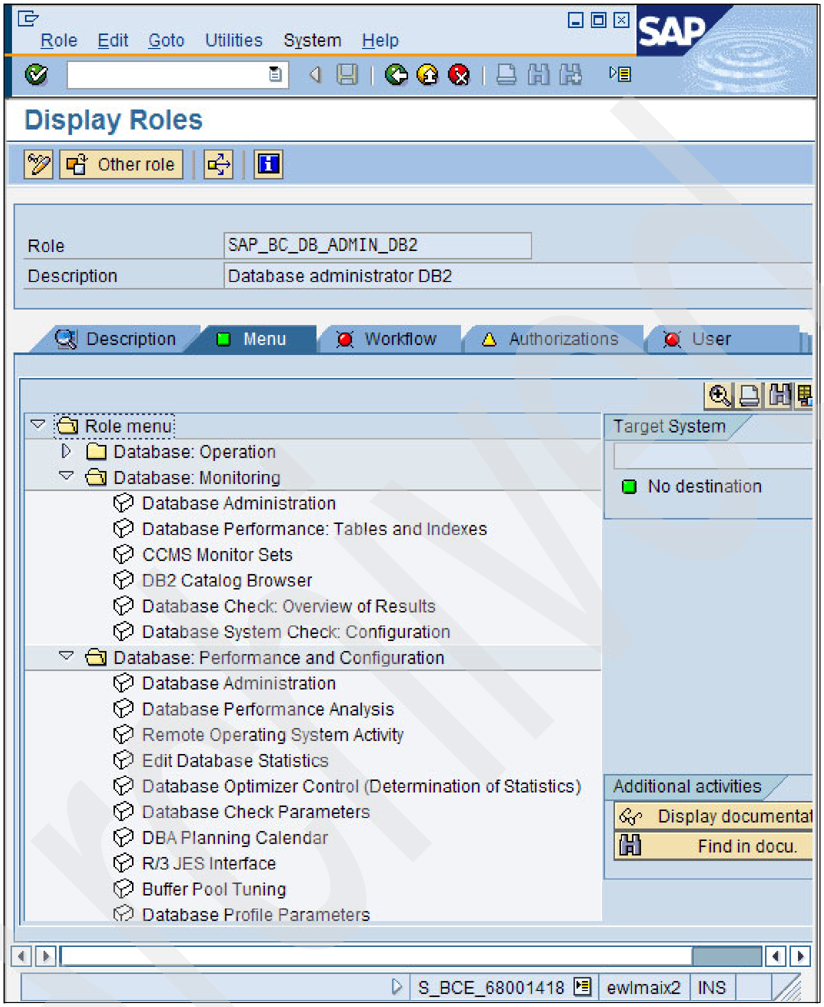
Role: SAP_BC_DB_ADMIN_DB2

Description: Database administrator DB2

- Role menu
  - Database: Operation
    - Database Administration
    - DBA Planning Calendar
    - R/3 JES Interface
    - Database Backup Logs: Overview
    - Log Information: All Operations
  - Database: Monitoring
  - Database: Performance and Configuration

Figure 14-3 DB2 administrator role: Operation

Figure 14-4 shows the assigned transactions for monitoring, performance analysis, and configuration.



Role: SAP_BC_DB_ADMIN_DB2

Description: Database administrator DB2

- Role menu
  - Database: Operation
  - Database: Monitoring
    - Database Administration
    - Database Performance: Tables and Indexes
    - CCMS Monitor Sets
    - DB2 Catalog Browser
    - Database Check: Overview of Results
    - Database System Check: Configuration
  - Database: Performance and Configuration
    - Database Administration
    - Database Performance Analysis
    - Remote Operating System Activity
    - Edit Database Statistics
    - Database Optimizer Control (Determination of Statistics)
    - Database Check Parameters
    - DBA Planning Calendar
    - R/3 JES Interface
    - Buffer Pool Tuning
    - Database Profile Parameters

Figure 14-4 DB2 administrator role: Monitoring and Performance and Configuration

The DB administrator role provides the basic database functions by the menu structure in the launchpad. Depending on the company size and the complexity of the system landscape, some of the predefined database administration roles may be combined.

14.2 Enhancements for administration commands and messages

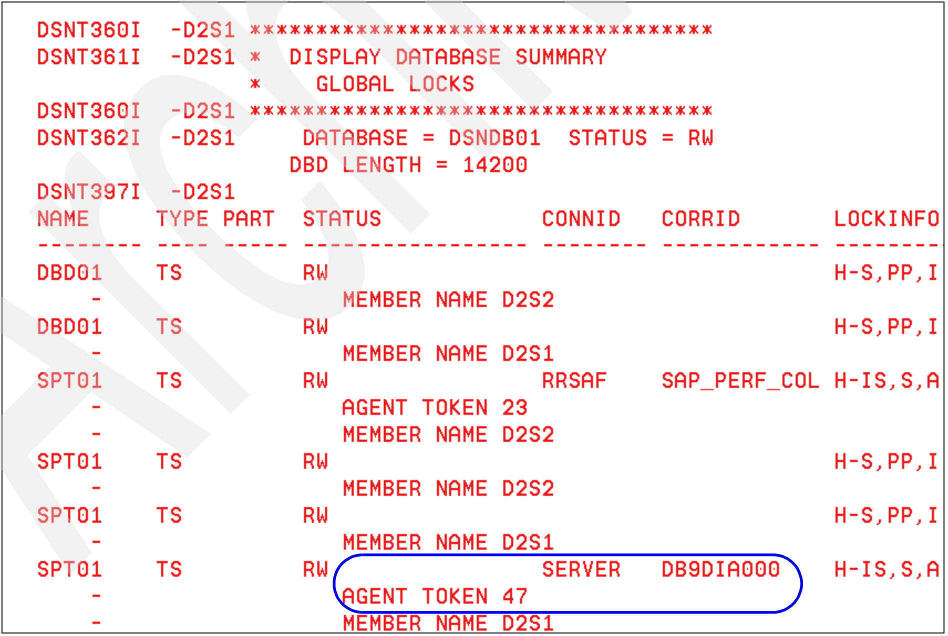
In DB2 Version 9.1, new automated database administration commands and messages continue to help improve the productivity and effectiveness of your DBAs. We discuss the enhancements in the following sections.

14.2.1 Enhanced thread information

A new option keyword for -DISPLAY THREAD (*) TYPE(SYSTEM) shows the DB command and system agents. The DB command and system agents that can be cancelled also have their associated tokens identified. Having identified the appropriate token, you can use the existing -CANCEL THREAD (token) command to cancel DB commands and system agents in progress.

Added thread identifier

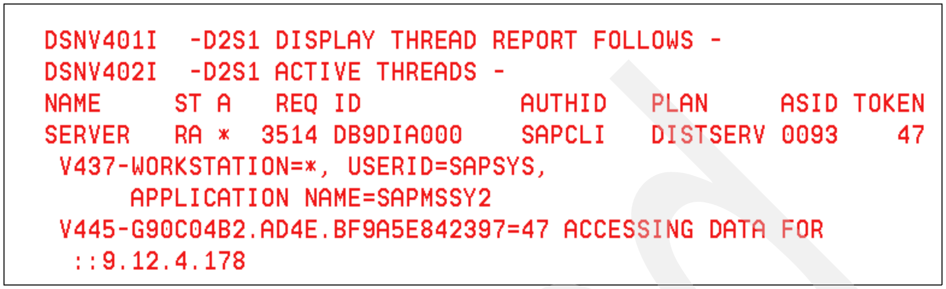
When displaying the DB2 objects, such as databases, DB2 9 for z/OS provides more detailed information. In addition to the necessary information, DB2 shows the Correlation Identifier for SAP-related threads. Figure 14-5 shows an example of Display Database with the Correlation Identifier.



```
DSNT360I  -D2S1 ***********************************
DSNT361I  -D2S1 *  DISPLAY DATABASE SUMMARY
                *    GLOBAL LOCKS
DSNT360I  -D2S1 ***********************************
DSNT362I  -D2S1     DATABASE = DSNDB01  STATUS = RW
                   DBD LENGTH = 14200
DSNT397I  -D2S1
NAME     TYPE PART  STATUS            CONNID   CORRID       LOCKINFO
-------- ---- ----- ----------------- -------- ------------ ---------
DBD01    TS         RW                                      H-S,PP,I
    -                  MEMBER NAME D2S2
DBD01    TS         RW                                      H-S,PP,I
    -                  MEMBER NAME D2S1
SPT01    TS         RW                RRSAF    SAP_PERF_COL H-IS,S,A
    -                  AGENT TOKEN 23
    -                  MEMBER NAME D2S2
SPT01    TS         RW                                      H-S,PP,I
    -                  MEMBER NAME D2S2
SPT01    TS         RW                                      H-S,PP,I
    -                  MEMBER NAME D2S1
SPT01    TS         RW                SERVER   DB9DIA000    H-IS,S,A
    -                  AGENT TOKEN 47
    -                  MEMBER NAME D2S1
```

Figure 14-5 Display Database with the Correlation Identifier

You can see the CORRID DB9DIA000 with agent token 47 and a thread name of SERVER. Now you can run -DIS THD(SERVER) to find token 47 to see what is happening.



```
DSNV401I  -D2S1 DISPLAY THREAD REPORT FOLLOWS -
DSNV402I  -D2S1 ACTIVE THREADS -
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
SERVER   RA *  3514 DB9DIA000    SAPCLI   DISTSERV 0093    47
 V437-WORKSTATION=*, USERID=SAPSYS,
      APPLICATION NAME=SAPMSSY2
 V445-G90C04B2.AD4E.BF9A5E842397=47 ACCESSING DATA FOR
  ::9.12.4.178
```

Figure 14-6   Displaying thread token 47

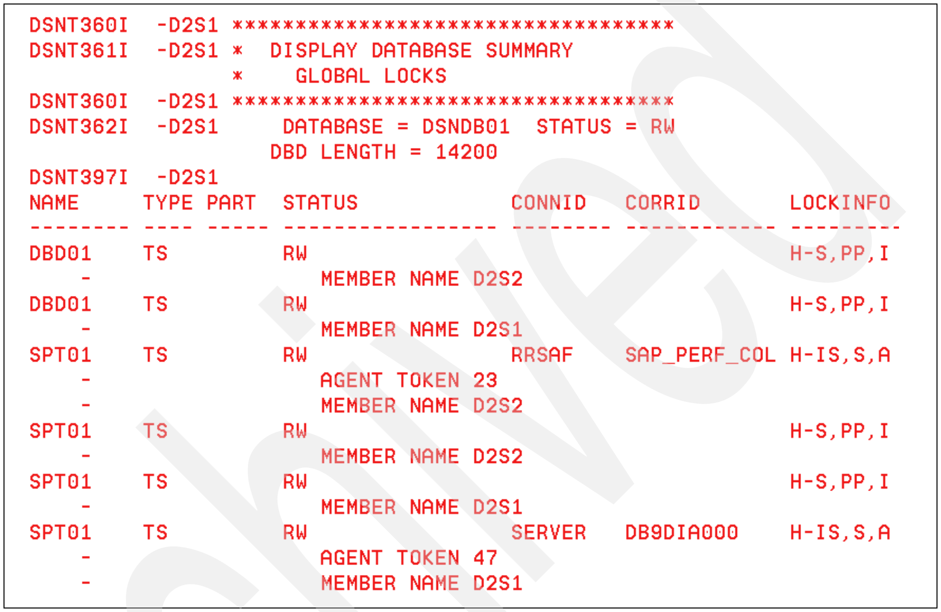
A Correlation Identifier for SAP-related threads Version 9.1 of DB2 for z/OS contains a number of trace enhancements. The trace enhancements in Version 9.1 include more ways to filter trace records.

DB2 Version 9.1 adds these new filter types in the constraint block of a START TRACE, STOP TRACE, or DISPLAY TRACE command:

- PKGPROG: Package
- PKGLOC: Location
- PKGCOL: Collection ID
- USERID: User ID
- APPNAME: Application name
- WKRSTN: Workstation name
- CONNID: Connection ID
- CORRID: Correlation ID

14.2.2 Enhanced LOCK information

Enhanced LOCK information was added in DB2 V8 with APAR PQ97318. If the lock information does not fit into a storage pool that is provided by IFC (24 MB in V8), IRLM returns an error code instead of populating the pool and truncating the lock information. Figure 14-7 shows an example of displaying a database with locks.



```
DSNT360I  -D2S1 ***********************************
DSNT361I  -D2S1 *  DISPLAY DATABASE SUMMARY
                *    GLOBAL LOCKS
DSNT360I  -D2S1 ***********************************
DSNT362I  -D2S1     DATABASE = DSNDB01  STATUS = RW
                   DBD LENGTH = 14200
DSNT397I  -D2S1
NAME     TYPE PART  STATUS            CONNID   CORRID       LOCKINFO
-------- ---- ----- ----------------- -------- ------------ ---------
DBD01    TS         RW                                      H-S,PP,I
    -                  MEMBER NAME D2S2
DBD01    TS         RW                                      H-S,PP,I
    -                  MEMBER NAME D2S1
SPT01    TS         RW                RRSAF    SAP_PERF_COL H-IS,S,A
    -                  AGENT TOKEN 23
    -                  MEMBER NAME D2S2
SPT01    TS         RW                                      H-S,PP,I
    -                  MEMBER NAME D2S2
SPT01    TS         RW                                      H-S,PP,I
    -                  MEMBER NAME D2S1
SPT01    TS         RW                SERVER   DB9DIA000    H-IS,S,A
    -                  AGENT TOKEN 47
    -                  MEMBER NAME D2S1
```

Figure 14-7 Database with locks

14.2.3 Added REFRESH TABLE command

The REFRESH TABLE statement refreshes the data in a materialized query table (MQT). The statement performs the following actions:

- Deletes all rows in the MQT
- Executes the fullselect in the table definition to recalculate the data from the tables specified in the fullselect
- Inserts the calculated result into the MQT
- Updates the catalog for the refresh timestamp and cardinality of the table

The table can exist at the current server or at any DB2 subsystem with which the current server can establish a connection.

You can invoke the REFRESH TABLE command interactively or through an application program. DYNAMICRULES must be implicitly or explicitly specified in order for the statement to run dynamically. The privilege for this is established if you own the MQT, or if you have dbadm or dbctrl where the MQT resides. You also have authority to the REFRESH TABLE if you have SYSADM or SYSCTRL authority.

To use the REFRESH TABLE command, you must have the table as an MQT as shown in Example 14-2. Assume a large transaction table named TRANS contains one row for each transaction processed by a company. The table is defined with many columns. Create an MQT for the TRANS table that contains daily summary data for the date and amount of a transaction (Example 14-1).

Example 14-1 MQT for the TRANS table

```
CREATE TABLE STRANS AS (SELECT YEAR AS SYEAR, MONTH AS SMONTH, DAY AS
SDAY, SUM(AMOUNT) AS SSUM FROM TRANS GROUP BY YEAR, MONTH, DAY) DATA
INITIALLY DEFERRED REFRESH DEFERRED;
```

You now can use the REFRESH TABLE command. Example 14-2 shows the syntax for the command.

Example 14-2 REFRESH TABLE

```
REFRESH TABLE table-name QUERYNO integer
```

Example of REFRESH TABLE

Issue a statement to refresh the content of an MQT that is named SALESCOUNT. The following statement recalculates the data from the fullselect that was used to define SALESCOUNT and refreshes the content of SALESCOUNT with the recalculated results:

```
REFRESH TABLE SALESCOUNT;
```

15

Geographically Dispersed Parallel Sysplex

More and more installations have set up a sysplex over multiple sites for availability, capacity, and workload balancing reasons. However, these configurations provide reduced continuous application availability. If a disaster occurs at the site where the data resides, the surviving portion of the sysplex will be down until lengthy data recovery actions can be completed. Moreover, data recovery can be expected to be incomplete and may lag actual production status by up to 24 hours.

15.1 Minimized application outages

To help switch applications from the primary site to the recovery location, you may want to automate operations, with a program such as Geographically Dispersed Parallel Sysplex™ (GDPS®). Your installation can make this switch with a minimal impact on users. A GDPS is a multi-site availability solution that merges sysplex and remote copy technologies. GDPS provides an integrated disaster survival capability that addresses the system, the network, and the data parts of an application environment. For more information, see the following Web page:

http://www-1.ibm.com/servers/eserver/zseries/gdps.html

The primary objective of GDPS is to minimize application outages that would result from a site failure. It ensures that, regardless of the failure scenario at the failing site, data in the surviving site is consistent and is therefore a valid base for a quick application restart. An installation-defined policy determines whether the switch will occur with limited loss or no loss of data.

In the event of a site failure (including disasters), the surviving site continues to function and absorb the work of the failed site. In the event of a planned site outage, the workload executing in the site undergoing a planned outage is quiesced and restarted at the other site. Current experience indicates that for large operational sites, a planned switch from one site to the other takes less than 60 minutes (including networking). Site unplanned outage recovery takes less than 45 minutes. Only a single keystroke is required to invoke a GDPS action.

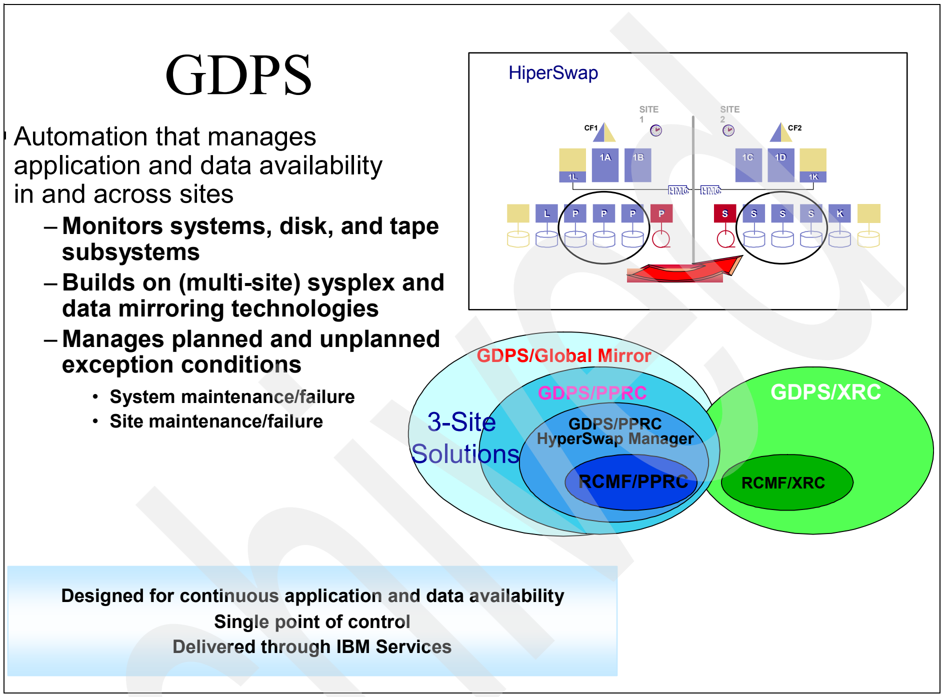
GDPS replaces a manual site switch process that can require more than 20 people to be present to perform their specialized tasks.

15.2 Implementation

GDPS is being implemented as an automation solution, using standard sysplex and IBM Storage Control Unit functions. The base for the implementation of a GDPS is a sysplex spread across two sites, securing diagnostic and control capability in case of a site failure. The two sites may be up to 300 km (synchronous MetroMirror) or several thousands of km (global Mirror with Consistency Groups) apart, depending on the current configuration.

All data that is required for an application restart must be DASD resident. All data that is part of the same group of applications must be in one site. Also PPRC must be used to maintain a synchronous copy of the data in the backup location. Spare processor capacity or an expendable workload must be available in the secondary site so that enough capacity is available to resume the critical workload in the backup location.

Figure 15-1 shows additional features of GDPS.



GDPS

- Automation that manages application and data availability in and across sites
  - Monitors systems, disk, and tape subsystems
  - Builds on (multi-site) sysplex and data mirroring technologies
  - Manages planned and unplanned exception conditions
    - System maintenance/failure
    - Site maintenance/failure

Designed for continuous application and data availability
Single point of control
Delivered through IBM Services

Figure 15-1 GDPS and HiperSwap

The HiperSwap function of GDPS builds on P/DAS technology but provides the total systems solution to allow application programs to continue to run in a seamless fashion following a failure of one or more primary subsystems. HiperSwap takes the P/DAS technology and creates a true business continuance solution for the customer.

16

Benefits of DB2 9 for z/OS to SAP BI

In this chapter, we describe the benefits and new features of DB2 9 for z/OS that are most relevant to SAP BI. We discuss the new dynamic index ANDing, the new histogram statistics, the parallelism enhancements, and some further BI relevant DB2 9 for z/OS features.

16.1 Dynamic index ANDing for star schema queries

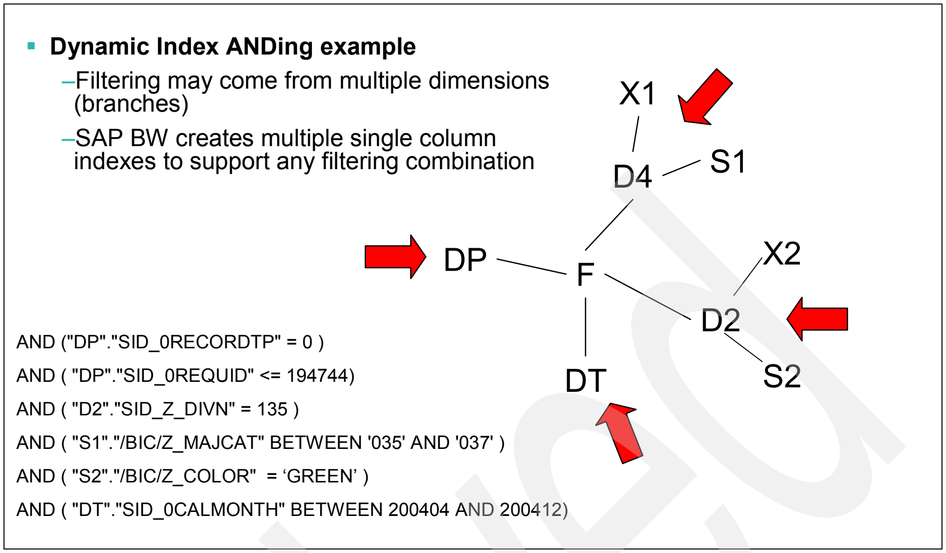
Dynamic index ANDing is a new enhanced star join access method introduced in DB2 9 as a main requirement from SAP BI. The goal of the new access method is to improve and stabilize data warehouse query performance and to lead to more predictable query performance.

How is it done? The method introduces self-automating and self defending access paths and uses parallelism. The access path used at runtime is dynamic and can be changed within the query processing by the DB2 engine. An additional goal of dynamic index ANDing is to simplify the index design on fact tables. The main idea behind dynamic index ANDing access is to apply the most filtering dimensions or snowflake branches before the fact table and to do this in an efficient way.

In short, dynamic index ANDing has the following characteristics:

- Better exploitation of SAP BI single column fact table indexes
  - No need for additional multicolumn indexes for leveraging star join access
- Consistent parallelism
  - Independent filtering dimension access in parallel
  - Fact table access (and post fact table) in parallel
- Adaptive query execution based upon runtime filtering
  - Less filtering dimensions discarded for pre-fact table access
  - Self-automating and self defending access path
  - Runtime fallback to work file for RID processing; avoids RID pool failures
- RID pool overflow to work file
- Less dependent on perfect statistics
  - Optimizer costing still performed
  - Better tolerance of less than perfect access path choice
- Provide more predictable query performance

Figure 16-1 shows a BI snowflake with the fact table F in the middle and the corresponding dimension branches, consisting of dimension tables (Dn), SID-tables (Sn), and X-tables (Xn). The arrows indicate the filtering for any of the four dimension branches.



- **Dynamic Index ANDing example**
  - Filtering may come from multiple dimensions (branches)
  - SAP BW creates multiple single column indexes to support any filtering combination

```
AND ("DP"."SID_0RECORDTP" = 0 )
AND ( "DP"."SID_0REQUID" <= 194744)
AND ( "D2"."SID_Z_DIVN" = 135 )
AND ( "S1"."/BIC/Z_MAJCAT" BETWEEN '035' AND '037' )
AND ( "S2"."/BIC/Z_COLOR"  = ‘GREEN’ )
AND ( "DT"."SID_0CALMONTH" BETWEEN 200404 AND 200412)
```

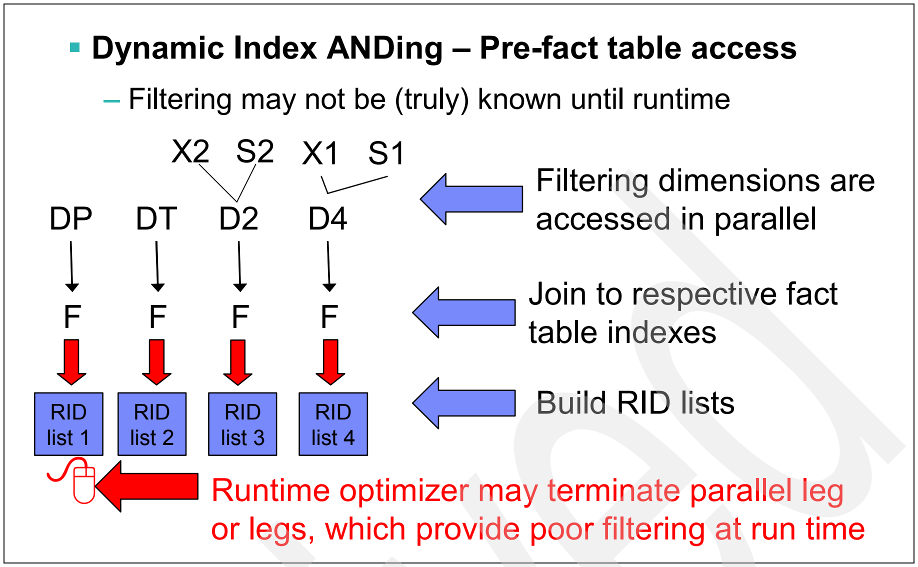
Figure 16-1 Filtering on dimension branches of fact table F

Figure 16-2 and Figure 16-3 on page 200 show the principle of processing dynamic index ANDing for pre-fact table access. Unlike a traditional star join group, the pair-wise join represents each dimension table join with a fact table separately. The result of each pair-wise join is a set of RIDs of the fact table. The RIDs are ANDing together to produce a final RID list for accessing the fact table pages.

The final fact table RIDs are used to retrieve data from the fact table and join back to the dimension table as necessary to obtain data from dimension tables.

To summarize, the pair-wise join exists when the following characteristics are apparent:

1. Joins each dimension table with the fact table through the index independently.
2. Performs a RID Sort and RID Merge (ANDing) of the RID lists that are obtained from step 1 to form the final fact table RID list.
3. Fetches data from the fact table using the final RID list.
4. Joins back the dimension table only if it has selected columns from dimension tables.



- **Dynamic Index ANDing – Pre-fact table access**
  - Filtering may not be (truly) known until runtime

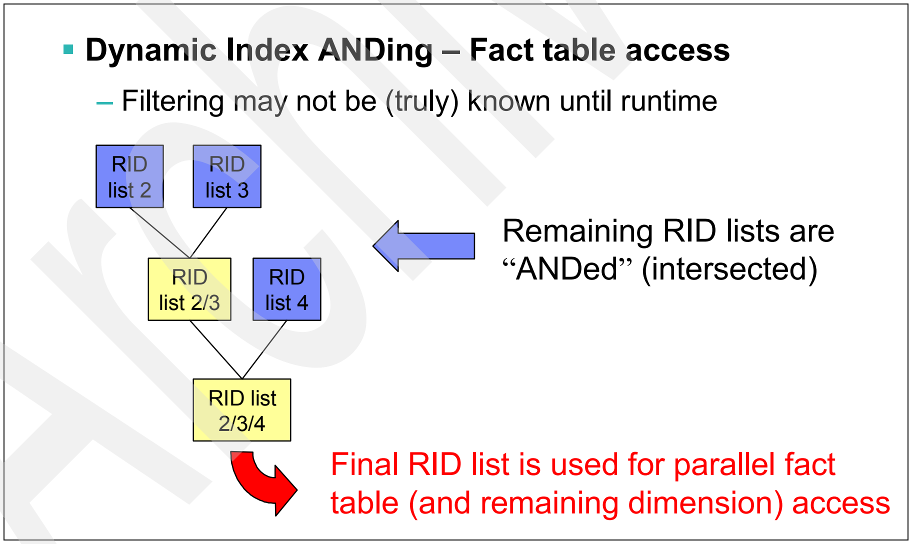
Filtering dimensions are accessed in parallel

Join to respective fact table indexes

Build RID lists

Runtime optimizer may terminate parallel leg or legs, which provide poor filtering at run time

Figure 16-2 Pre-fact table access



- **Dynamic Index ANDing – Fact table access**
  - Filtering may not be (truly) known until runtime

Remaining RID lists are "ANDed" (intersected)

Final RID list is used for parallel fact table (and remaining dimension) access

Figure 16-3 Building the final RID list before fact table access

16.2 Database statistics for BI tables

In this section, we discuss two ways in which DB2 9 for z/OS and SAP work together to process RUNSTATS for BI tables and use histogram statistics to assist in query optimization.

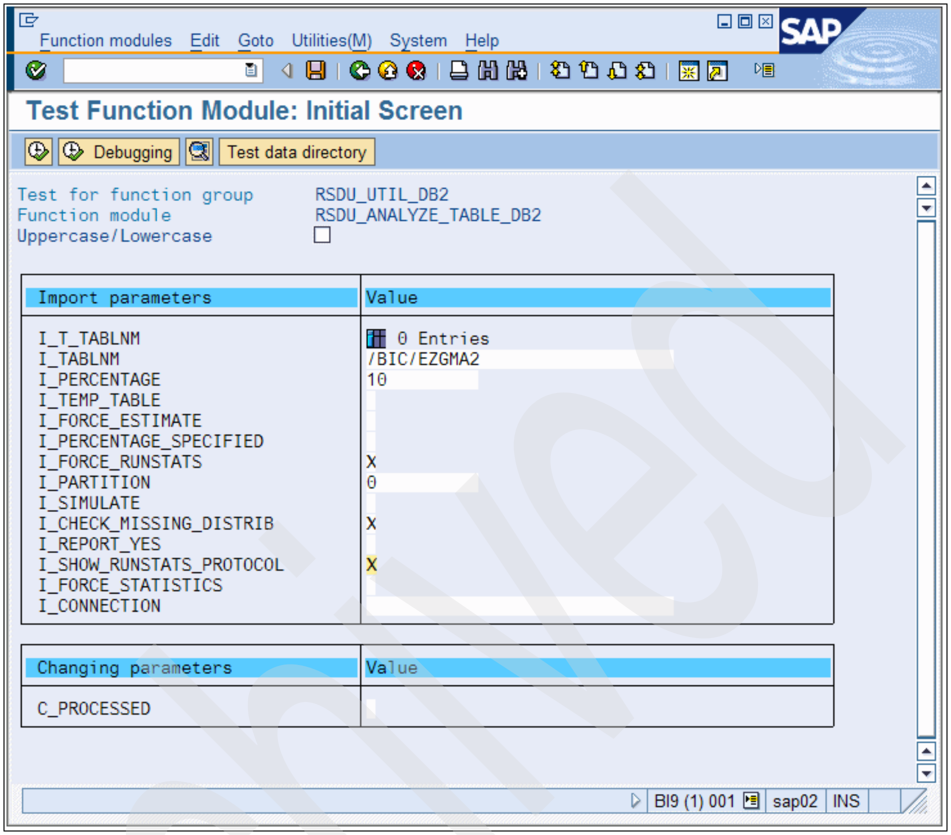
16.2.1 Processing RUNSTATS for BI tables

From BW 3.0 Support Package 30; 3.1C Support Package 24; 3.5 Support Package 16; and BI 7.0 Support Package 6, SAP has optimized the ABAP coding for processing RUNSTATS on BI tables. The two newly introduced main features are:

- RUNSTATS on partition level
- Call RUNSTATS only if necessary

From BW 3.5 Support Package 20 and BI 7.0 Support Package 11, SAP supports the new histogram statistics feature of DB2 9.

On SAP BI systems on DB2 on z/OS, all database statistics for BI tables are refreshed by the function module RSDU_ANALYZE_TABLE_DB2. You can learn how SAP BI calculates DB2 statistics, which include knowing which RUNSTATS commands are fired and which statistics are possibly faked by directly changing statistical data in DB2 system tables. Run the function module RSDU_ANALYZE_TABLE_DB2 over SAP transaction SE37 for a given table. Then mark the parameters I_SHOW_RUNSTATS_PROTOCOL and I_FORCE_RUNSTATS as shown in Figure 16-4.



Test Function Module: Initial Screen

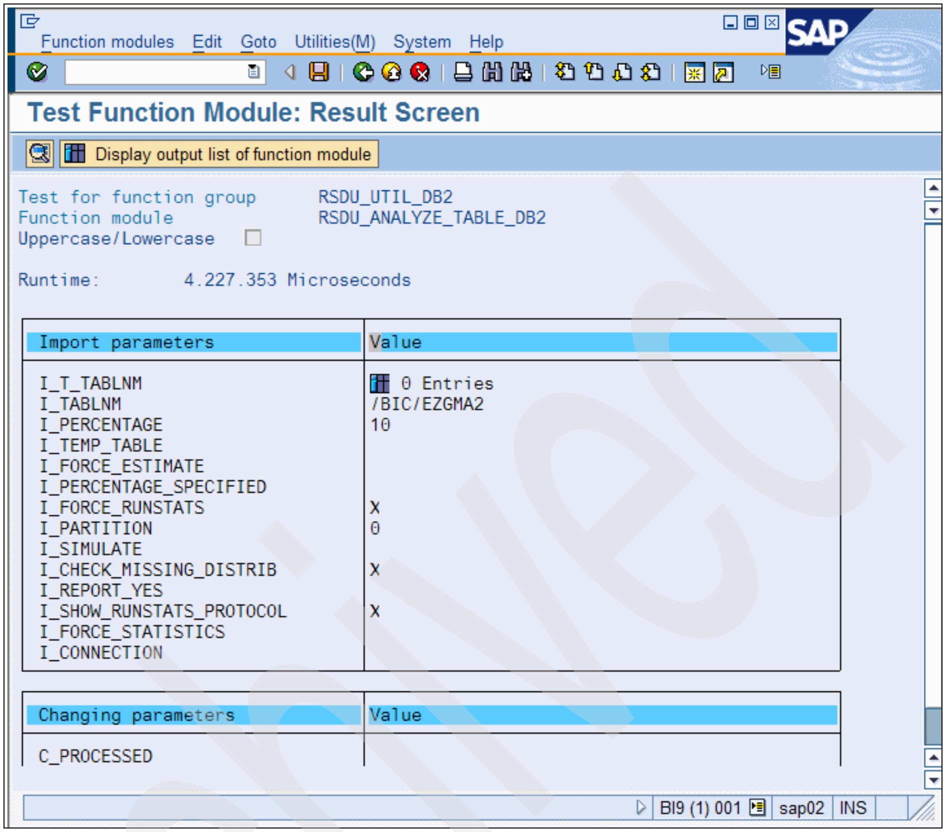
Test for function group RSDU_UTIL_DB2
Function module RSDU_ANALYZE_TABLE_DB2
Uppercase/Lowercase □

Import parameters	Value
I_T_TABLNM	0 Entries
I_TABLNM	/BIC/EZGMA2
I_PERCENTAGE	10
I_TEMP_TABLE	
I_FORCE_ESTIMATE	
I_PERCENTAGE_SPECIFIED	
I_FORCE_RUNSTATS	X
I_PARTITION	0
I_SIMULATE	
I_CHECK_MISSING_DISTRIB	X
I_REPORT_YES	
I_SHOW_RUNSTATS_PROTOCOL	X
I_FORCE_STATISTICS	
I_CONNECTION	

Changing parameters	Value
C_PROCESSED	

Figure 16-4 Executing RUNSTATS by function RSDU_ANALYZE_TABLE_DB2

RUNSTATS is then called over the stored procedure DSNUTILS. You must wait a couple of minutes for large tables. When RUNSTATS is processed, the function comes back with the Result Screen (see Figure 16-5).



Test Function Module: Result Screen

```
Test for function group        RSDU_UTIL_DB2
Function module                RSDU_ANALYZE_TABLE_DB2
Uppercase/Lowercase     □

Runtime:        4.227.353 Microseconds
```

Import parameters	Value
I_T_TABLNM	0 Entries
I_TABLNM	/BIC/EZGMA2
I_PERCENTAGE	10
I_TEMP_TABLE	
I_FORCE_ESTIMATE	
I_PERCENTAGE_SPECIFIED	
I_FORCE_RUNSTATS	X
I_PARTITION	0
I_SIMULATE	
I_CHECK_MISSING_DISTRIB	X
I_REPORT_YES	
I_SHOW_RUNSTATS_PROTOCOL	X
I_FORCE_STATISTICS	
I_CONNECTION	

Changing parameters	Value
C_PROCESSED	

Figure 16-5 Test mode for function RSDU_ANALYZE_TABLE_DB2

Click the **Display output list of function module** button to see the output of the function (Example 16-1).

Example 16-1 Output of function module RSDU_ANALYZE_TABLE

```
22.03.2007                                         SV Parameter                                    1
----------------------------------------------------------------------------------------------------------------------
RUNSTATS TABLESPACE "FA40XK1C"."EZGMA2X"
   TABLE(BI9DB."/BIC/EZGMA2") SAMPLE 010
      COLUMN( KEY_ZGMA21,KEY_ZGMA22,KEY_ZGMA23,KEY_ZGMA24,KEY_ZGMA25,KEY_ZGMA2P,KEY_ZGMA2T,KEY_ZGMA2U )
      COLGROUP ("KEY_ZGMA2P") FREQVAL COUNT 0 HISTOGRAM NUMQUANTILES 50
      COLGROUP ("KEY_ZGMA2U") FREQVAL COUNT 0 HISTOGRAM NUMQUANTILES 50
   INDEX(
      BI9DB."/BIC/EZGMA2~0" KEYCARD FREQVAL NUMCOLS 1 COUNT 0 HISTOGRAM NUMCOLS 4 NUMQUANTILES 50
      ,BI9DB."/BIC/EZGMA2~040" KEYCARD FREQVAL NUMCOLS 1 COUNT 0 HISTOGRAM NUMCOLS 1 NUMQUANTILES 50
      ,BI9DB."/BIC/EZGMA2~050" KEYCARD FREQVAL NUMCOLS 1 COUNT 0 HISTOGRAM NUMCOLS 1 NUMQUANTILES 50
      ,BI9DB."/BIC/EZGMA2~060" KEYCARD FREQVAL NUMCOLS 1 COUNT 0 HISTOGRAM NUMCOLS 1 NUMQUANTILES 50
      ,BI9DB."/BIC/EZGMA2~070" KEYCARD FREQVAL NUMCOLS 1 COUNT 0 HISTOGRAM NUMCOLS 1 NUMQUANTILES 50
      ,BI9DB."/BIC/EZGMA2~080" KEYCARD FREQVAL NUMCOLS 1 COUNT 0 HISTOGRAM NUMCOLS 1 NUMQUANTILES 50
        )
   SHRLEVEL CHANGE REPORT NO
cause for calling RUNSTATS
```

```
parameter force_runstats set
old_card / rows_changed / percent_changed =        0  /          0  /  9.99900E+03
Runtime für RUNSTATS:                   3,817003   secs
20.070.322.203.600,9062580   -  20.070.322.203.604,7232610
1DSNU000I    081 16:36:00.33 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = BI924116
 DSNU1044I   081 16:36:00.39 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
0DSNU050I    081 16:36:00.40 DSNUGUTC -  RUNSTATS TABLESPACE "FA40XK1C"."EZGMA2X" TABLE(BI9DB."/BIC/EZGMA2") SAMPLE
 10 COLUMN(KEY_ZGMA21, KEY_ZGMA22, KEY_ZGMA23, KEY_ZGMA24, KEY_ZGMA25, KEY_ZGMA2P, KEY_ZGMA2T, KEY_ZGMA2U) COLGROUP(
 "KEY_ZGMA2P") FREQVAL COUNT 0 HISTOGRAM NUMQUANTILES 50 COLGROUP("KEY_ZGMA2U") FREQVAL COUNT 0 HISTOGRAM
 NUMQUANTILES 50 INDEX(BI9DB."/BIC/EZGMA2~0" KEYCARD FREQVAL NUMCOLS 1 COUNT 0 HISTOGRAM NUMCOLS 4 NUMQUANTILES 50,
 BI9DB."/BIC/EZGMA2~040" KEYCARD FREQVAL NUMCOLS 1 COUNT 0 HISTOGRAM NUMCOLS 1 NUMQUANTILES 50,
 BI9DB."/BIC/EZGMA2~050" KEYCARD FREQVAL NUMCOLS 1 COUNT 0 HISTOGRAM NUMCOLS 1 NUMQUANTILES 50,
 BI9DB."/BIC/EZGMA2~060" KEYCARD FREQVAL NUMCOLS 1 COUNT 0 HISTOGRAM NUMCOLS 1 NUMQUANTILES 50,
 BI9DB."/BIC/EZGMA2~070" KEYCARD FREQVAL NUMCOLS 1 COUNT 0 HISTOGRAM NUMCOLS 1 NUMQUANTILES 50,
 BI9DB."/BIC/EZGMA2~080" KEYCARD FREQVAL NUMCOLS 1 COUNT 0 HISTOGRAM NUMCOLS 1 NUMQUANTILES 50) SHRLEVEL CHANGE
 REPORT NO
 DSNU042I    081 16:36:01.01 DSNUGLSR - SORT PHASE STATISTICS -
                       NUMBER OF RECORDS=0
                       ELAPSED TIME=00:00:00
 DSNU610I  -DB2T 081 16:36:01.67 DSNUSUTP - SYSTABLEPART CATALOG UPDATE FOR FA40XK1C.EZGMA2X SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:01.67 DSNUSUPT - SYSTABSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:01.68 DSNUSUPC - SYSCOLSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:01.68 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:01.68 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:01.68 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:01.68 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU604I  -DB2T 081 16:36:01.69 DSNUSEF2 - TABLESPACE IS EMPTY
 DSNU610I  -DB2T 081 16:36:01.69 DSNUSUTB - SYSTABLES CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:01.70 DSNUSUCO - SYSCOLUMNS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:01.70 DSNUSUTS - SYSTABLESPACE CATALOG UPDATE FOR FA40XK1C.EZGMA2X SUCCESSFUL
 DSNU623I  -DB2T 081 16:36:01.70 DSNUSEF2 - SYSIBM.SYSCOLDIST CATALOG NOT UPDATED WITH AGGREGATE STATISTICS FOR
 BI9DB./BIC/EZGMA2.KEY_ZGMA2P BECAUSE SOME PARTITIONS HAVE NO VALID STATISTICS
 DSNU623I  -DB2T 081 16:36:01.70 DSNUSEF2 - SYSIBM.SYSCOLDIST CATALOG NOT UPDATED WITH AGGREGATE STATISTICS FOR
 BI9DB./BIC/EZGMA2.KEY_ZGMA2U BECAUSE SOME PARTITIONS HAVE NO VALID STATISTICS
 DSNU610I  -DB2T 081 16:36:02.55 DSNUSUIP - SYSINDEXPART CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~0 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:02.55 DSNUSUPI - SYSINDEXSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~0 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:02.55 DSNUSUPC - SYSCOLSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:02.56 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:02.56 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:02.81 DSNUSUIP - SYSINDEXPART CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~040 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:02.81 DSNUSUPI - SYSINDEXSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~040 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:02.81 DSNUSUPC - SYSCOLSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:02.81 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:02.82 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.10 DSNUSUIP - SYSINDEXPART CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~050 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.10 DSNUSUPI - SYSINDEXSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~050 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.10 DSNUSUPC - SYSCOLSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.10 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.10 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.34 DSNUSUIP - SYSINDEXPART CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~060 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.35 DSNUSUPI - SYSINDEXSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~060 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.35 DSNUSUPC - SYSCOLSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.35 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.35 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.60 DSNUSUIP - SYSINDEXPART CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~070 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.60 DSNUSUPI - SYSINDEXSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~070 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.60 DSNUSUPC - SYSCOLSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.60 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.60 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.85 DSNUSUIP - SYSINDEXPART CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~080 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.85 DSNUSUPI - SYSINDEXSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~080 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.85 DSNUSUPC - SYSCOLSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.85 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.85 DSNUSUPD - SYSCOLDISTSTATS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2 SUCCESSFUL
 DSNU604I  -DB2T 081 16:36:03.85 DSNUSEOF - INDEXSPACE IS EMPTY
 DSNU610I  -DB2T 081 16:36:03.85 DSNUSUCO - SYSCOLUMNS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~0 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.85 DSNUSUIX - SYSINDEXES CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~0 SUCCESSFUL
 DSNU604I  -DB2T 081 16:36:03.87 DSNUSEOF - INDEXSPACE IS EMPTY
 DSNU610I  -DB2T 081 16:36:03.87 DSNUSUCO - SYSCOLUMNS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~040 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.87 DSNUSUIX - SYSINDEXES CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~040 SUCCESSFUL
 DSNU604I  -DB2T 081 16:36:03.87 DSNUSEOF - INDEXSPACE IS EMPTY
 DSNU610I  -DB2T 081 16:36:03.87 DSNUSUCO - SYSCOLUMNS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~050 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.87 DSNUSUIX - SYSINDEXES CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~050 SUCCESSFUL
 DSNU604I  -DB2T 081 16:36:03.87 DSNUSEOF - INDEXSPACE IS EMPTY
 DSNU610I  -DB2T 081 16:36:03.87 DSNUSUCO - SYSCOLUMNS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~060 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.88 DSNUSUIX - SYSINDEXES CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~060 SUCCESSFUL
 DSNU604I  -DB2T 081 16:36:03.88 DSNUSEOF - INDEXSPACE IS EMPTY
 DSNU610I  -DB2T 081 16:36:03.88 DSNUSUCO - SYSCOLUMNS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~070 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.88 DSNUSUIX - SYSINDEXES CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~070 SUCCESSFUL
 DSNU604I  -DB2T 081 16:36:03.88 DSNUSEOF - INDEXSPACE IS EMPTY
 DSNU610I  -DB2T 081 16:36:03.88 DSNUSUCO - SYSCOLUMNS CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~080 SUCCESSFUL
 DSNU610I  -DB2T 081 16:36:03.88 DSNUSUIX - SYSINDEXES CATALOG UPDATE FOR BI9DB./BIC/EZGMA2~080 SUCCESSFUL
 DSNU620I  -DB2T 081 16:36:03.89 DSNUSEOF - RUNSTATS CATALOG TIMESTAMP = 2007-03-22-16.36.00.422387
 DSNU010I    081 16:36:04.01 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
> connection: DEFAULT   DB2 User: SAPR3
```

Since some specific RUNSTATS options are used for BI tables, statistics should not be created for SAP BI systems using external tools. Use the “RUNSTATS on objects needing new statistics” option in the planning calendar (DB13) to update the DB2 statistics. We recommend that you schedule this run daily or at least once a week at night. Since the DB2 statistics for BW programs are already updated automatically for most BI tables, the DB13 statistics update should be quick.

Consider the following remarks:

- Use RUNSTATS on the partition level.

  SAP BI uses only the PART option of RUNSTATS, if the partitioned table has more than 1 million rows. This is done for performance reasons.

- Call RUNSTATS only if necessary.

  This means that RUNSTATS is only called if the old database statistics must be refreshed. For this, the real-time statistics (RTS) data is compared against the old statistics. If more than 10 percent of the data has been changed or if the table is small (lower than 1000 rows), RUNSTATS is called again.

You can find more details in SAP Note 915398. This note also describes the input parameter of the previously used function module RSDU_ANALYZE_TABLE_DB2.

16.2.2 New to DB2 9: Histogram statistics

Data distribution statistics are important for query optimization. DB2 chooses the best access path based on costing. The basic foundation of costing is predicate selectivity estimation, which heavily relies on data distribution statistics.

Up to V8, the only distribution statistics supported in DB2 for z/OS are frequency statistics that are collected on single values, which can be either for single-column or multi-column. For multi-column, it is a concatenated multi-column value. Such frequency statistics are most commonly collected on the most biased values, that is the most frequent, the least frequent, or both.

Often the single-value based frequency statistics can hardly help DB2 with predicate selectivity other than uniform interpolation on the rest of (uncollected) the value range. This is a wild guess and may lead to an undesirable access path.

The histogram describes the data distribution over the entire value range. Predicate selectivity has a more accurate calculation if the searching range matches the boundary of any one quantile or any group of consecutive quantiles. Even if there is no perfect match, predicate selectivity interpolation is now done within one or two particular quantiles. With the interpolation done in a much smaller granularity, predicate selectivity is expected to be evaluated with more accuracy.

Histograms in detail

Histogram statistics are introduced to DB2 9 to enhance predicate selectivity estimation and to enhance the DB2 access path selection in general.

> **Histogram statistics:** Histogram statistics are a way to summarize data distribution on an interval scale, either discrete or continuous. They divide the range of possible values in a data set into quantiles, for which a set of statistics parameters is collected.

Several types of histogram statistics are being researched. DB2 on z/OS uses only *equal-depth histogram statistics*.

Histogram statistics enable DB2 to improve access path selection by estimating predicate selectivity from value-distribution statistics that are collected over the entire range of values in a data set. DB2 chooses the best access path for a query based on predicate selectivity estimation, which in turn relies heavily on data distribution statistics. Histogram statistics summarize data distribution on an interval scale by dividing the entire range of possible values within a data set into a number of intervals.

DB2 creates equal-depth histogram statistics. It divides the whole range of values into intervals that each contain about the same percentage of the total number rows.

The following columns in a histogram statistics table define an interval:

QUANTILENO An ordinary sequence number that identifies the interval

LOWVALUE A value that serves as the lower bound for the interval

HIGHVALUE The value that serves as the upper bound for the interval

Note the following characteristics of the histogram statistics intervals:

- Each interval includes approximately the same number, or percentage, of the rows. A highly frequent single value might occupy an interval by itself.
- A single value is never broken into more than one interval, meaning that the maximum number of intervals is equal to the number of distinct values on the column. The maximum number of intervals cannot exceed 100, which is the maximum number that DB2 supports.
- Adjacent intervals sometimes skip values that do not appear in the table, especially when doing so avoids a large range of skipped values within an interval. For example, if the value 30 has a 1% frequency, placing it in the seventh interval would balance the percentage of rows in the sixth and seventh intervals. However, doing so would introduce a large skipped range to the seventh interval.
- HIGHVALUE and LOWVALUE can be inclusive or exclusive, but an interval generally represents a non-overlapped value range.
- NULL values, if any exist, occupy a single interval.
- Because DB2 cannot break any single value into two different intervals, the maximum number of intervals is limited to the number of distinct values in the column. It cannot exceed the DB2 maximum of 100 intervals.

Example for equal-depth histogram statistics

Equal-depth histogram statistics cut the whole value range so that each quantile has about the same number of rows. The important parameters to describe equal-depth histogram statistics include the total number of quantiles (N) and for each quantile, the LOWVALUE/HIGHVALUE pair, the number of distinctive values (CARD), and the frequency (or the number of rows).

For example, on table IBM_EMPLOYEE column YRS_OF_EXPERIENCE, it may have (N=7) quantiles as shown in Table 16-1.

Table 16-1 Histogram statistics example

SEQUENCE NUMBER	LOWVALUE	HIGHVALUE	CARD	FREQUENCY
1	0	3	4	14%
2	4	15	8	14%
3	18	24	7	12%
4	25	25	7	12%
5	26	26	1	15%
6	27	30	4	16%
7	35	40	6	14%

Note the following points:

- There may be skipped values between adjacent quantiles. If there are, then those skipped values are of zero appearance in the table.
- Each quantile has approximately the same number (or percentage) of rows. The highly frequent single value may alone occupy a quantile. The single-value is not broken into more than one quantiles. Therefore, the number of quantiles maximum is equivalent to the number of distinct values on that column. However, DB2 has a maximum limit of 100 quantiles for storage purposes.
- One particular value may be included into a quantile because it is “closer” to that quantile. The purpose is to avoid the skipped range as much as possible. For example, the value 30 has a frequency equal to 1%. Including it in the sixth quantile would make the fifth and sixth quantile with frequency=15% even. Doing that would introduce a large skipped range [28, 34] to the sixth quantile and we try to avoid that.
- A special case of sequence number 3 in Table 16-1 is the HIGHVALUE or LOWVALUE. When it is too far away from the rest of the value range, the single value of HIGHVALUE or LOWVALUE occupies a quantile by itself.
- A NULL value occupies a quantile by itself.
- Due to aggregation, quantiles may not always have a closed range. That is, the high value or low value can be inclusive or exclusive. Overall the quantiles represent the non-overlapped value range.

You can find more details about histogram statistics in the description of the corresponding system catalog tables (SYSCOLDIST, SYSCOLDISTSTATS, SYSKEYTGTDIST, SYSKEYTGTDIST, ...) in the *DB2 Version 9.1 for z/OS Utility Guide and Reference*, SC18-9855.

16.3 Parallelism enhancements

In this section, we discusses global query optimization and cross-query block optimization

16.3.1 Global query optimization

With DB2 V8, the DB2 optimizer chooses the lowest cost “sequential” plan and then determines how to “parallelize” the access path. With DB2 9 for z/OS, the way of using parallelism has been extended. The V9 optimizer considers multiple sequential plans for parallelism and then executes the one with the lowest costs.

Global query optimization addresses query performance problems caused when DB2 V8 breaks a query into multiple parts and optimizes each of those parts independently. While each of the individual parts may be optimized to run efficiently, when these parts are combined, the overall result may be inefficient.

For example, consider the following query:

```
SELECT * FROM T1
   WHERE EXISTS (SELECT 1 FROM T2, T3 WHERE T2.C2 = T3.C2
                                AND T2.C1 = T1.C1);
```

DB2 V8 breaks this query into two parts: the correlated subquery and the outer query. Each of these parts is optimized independently. The access path for the subquery does not take into account the different ways in which the table in the outer query may be accessed and vice versa.

DB2 V8 may choose to do a table scan of T1, resulting in much random I/O when accessing T2. A nonmatching index scan of T1 avoids the random I/O on T2. In addition, DB2 does not consider reordering these two parts. The correlated subquery is always performed after accessing T1 to get the correlation value. If T1 is a large table and T2 is a small table, it may be much more efficient to access T2 first and then T1, especially if there is no index on T2.C1 but there is an index on T1.C1.

In summary, global query optimization allows DB2 to optimize a query as a whole rather than as independent parts. This is accomplished by allowing DB2 to:

- Consider the effect of one query block on another
- Consider reordering query blocks

16.3.2 Cross-query block optimization

Cross-query block optimization is a new feature that addresses query performance problems caused when DB2 breaks a query into multiple parts and optimizes each of those parts independently. The V9 optimizer treats subqueries as virtual tables, which allows the optimizer to do a cross-query block optimization.

16.4 Extended sparse index and in-memory work file

DB2 9 introduces the new installation parameter MXDTCACH to specify the maximum memory allocated for data caching. The recommended minimal setting of MXDTCACH for SAP BI systems is 128 MB. The recommended settings of new V9 parameters after migration from a V8 SAP system is documented in SAP note 1032273.

The unit of a value specified to MXDTCACH is in MB (1 MB is 1048576 bytes). For example, if the value 300 is specified, the total amount up to 300 MB memory can be allocated for data caching per thread.

- The DB2 default value of 20 MB is far to low.
- When the value of zero is specified, data caching method is not used during query execution. Only sparse index (key+rid) can be applied.
- The possible range of the values is between 0 and 512.
- When data records are cached, the required memory amount is:

  ```
  (number of rows) x ((maximum length of the keys) +
  (total maximum length of all the relevant columns))
  ```

  If the required memory exceeds the limit that MXDTCACH specifies or the memory allocation fails, the query continues without using data caching. Sparse index is used instead.
- When the thread terminates, the allocated memory for data caching is freed from the local pool above the 2 GB bar.

The new installation parameter MXDTCACH can be set on the DB2 installation panel.

Note: The existing installation parameter SJMXPOOL is replaced by MXDTCACH. Accordingly, the data caching space for star join is moved to the local pool above the 2 GB bar.

16.5 TRUNCATE TABLE statement

The TRUNCATE TABLE statement deletes all data rows for either base tables or declared global temporary tables without activating delete triggers defined on the table (they are ignored). The base table can belong to a simple table space, a segmented table space, a partitioned table space, or a universal table space. If the table contains LOB or XML columns, its corresponding table spaces and indexes are also emptied.

DB2 9 transforms the new TRUNCATE TABLE operation into a mass delete operation to take advantage of the current, optimized design. It also provides greater flexibilities for users to deactivate existing Delete Triggers and harden the results of truncate operation without issuing a normal commit.

Under specific table attributes and statement options, the TRUNCATE TABLE statement may provide an optimal performance path to empty the table while in comparison with the current mass delete operation (that is DELETE without a WHERE clause). The performance improvement can be gained only on a table with triggers defined. Otherwise, it performs as it already does. Moreover, it provides a “statement commit” option (IMMEDIATE option) for the statement that allows the truncate operation to become permanent (that is, no undo). It immediately makes deallocated space available for new, subsequent inserts in the same unit of work.

SAP BI uses the TRUNCATE TABLE statement of DB2 9, but because BI tables usually have no triggers, the performance is the same as the mass delete operation.

16.6 FETCH FIRST n ROWS ONLY in subselects

DB2 8 for z/OS disallows the FETCH FIRST n ROWS ONLY clause in a subselect. That is, you could write the following statement, specifying the clauses as part of a SELECT statement:

```
SELECT * FROM T ORDER BY c1 FETCH FIRST n ROWS ONLY
```

However, you could not write the following statement, specifying the clauses within the subselect:

```
INSERT INTO any_table
   (SELECT * FROM T ORDER BY c1 FETCH FIRST n ROWS ONLY)
```

SAP BI needs this feature to implement a more sophisticated algorithm of the *BI Selective Deletion* for Infocube and DSO data. Here *Selective* means that the user can choose an arbitrary selection criteria for the data to be deleted. In this case, the data to be deleted can be distributed over all table partitions of a partitioned table, so that the deletion of whole partitions cannot be used.

When deleting a huge amount of data rows from a fact or DSO table, it is better to separate this deletion in many small portions of delete transactions. Otherwise you might have one long running delete transaction, with the risk of a day-long or even longer rollback in case the delete transaction must be canceled for any reason. This is a nightmare for every database administrator.

With the new DB2 9 feature of using FETCH FIRST n ROWS ONLY within a subselect, the portionwise deletion is implemented in the following way. The large delete operation is split up into multiple smaller units of work.

In Example 16-2, assume that the primary key of table *<deltable>* consists of the columns kf1, kf2, ..., kfn. <delete_condition> is a placeholder for the WHERE condition, generated by SAP BI upon the deletion criteria specified by the BI user. To split up this delete into a loop of delete operations, with each one deleting <paketsize> rows only, the FETCH FIRST n ROWS ONLY statement can be used. After each execution of the DELETE statement, a commit is performed. The loop continues until all rows that are specified are deleted.

Example 16-2   New implementation of huge data deletion

```
repeat
   DELETE FROM <deltable>
      WHERE (kf1, kf2, ..., kfn) IN
         ( SELECT kf1, kf2, ..., kfn FROM <deltable>
            WHERE <delete_condition>
            FETCH FIRST <paketsize> ROWS ONLY );

   COMMIT WORK;
until  nrows_deleted < paketsize.
```

The deletion of the data is performed in a loop with a COMMIT after every deletion of one small paket of rows (paketsize can be configured). This new SAP BI implementation for data deletion will be available for the next SAP BI release coming after BI 7.0. SAP plans to make this new implementation for selective deletion available as well for the BI 7.00 release, running with DB2 9 by setting a special RSADMIN parameter. This will be documented in a future SAP note.

Example 16-3 shows the delete statement that is generated to delete a maximum of 100,000 rows from the E-fact table of cube IUSALES.

Example 16-3   Using FETCH FIRST n ROWS ONLY for E-fact table

```
DELETE FROM  "/BIC/EIUSALES"
WHERE ("KEY_IUSALESP", "KEY_IUSALEST", "KEY_IUSALESU", "KEY_IUSALES1"
      , "KEY_IUSALES2", "KEY_IUSALES3")
   IN ( SELECT "KEY_IUSALESP", "KEY_IUSALEST","KEY_IUSALESU",
              "KEY_IUSALES1", "KEY_IUSALES2","KEY_IUSALES3"
          FROM  "/BIC/EIUSALES"
         WHERE <delete_condition>
         FETCH FIRST 100000 ROWS ONLY )
```

A

Migration technique

In this appendix, we outline the procedures that we used to migrate from a DB2 V8 SAP environment to a DB2 9 for z/OS SAP environment.

Migration best practices

For a guide to best practices, see *SAP on IBM DB2 UDB for OS/390 and z/OS: Best Practice for Installing or Migrating to DB2 V8*, which is available at the following Web address:

https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/5798c93d-0e01-0010-81b4-d641a79c1ab0

Test environment

We used the following environment to test the migration procedures:

- A two-member DB2 V8 data sharing group
  - LPAR SC04 with member D2S1
  - LPAR SC47 with member D2S2
  - z/OS 1.7
- An SAP application
  - SAP ERP 2004 SR1 IDES, 6.40 Web Application Server kernel and ECC 5.0
  - SAP SID DB9
- DB2 Connect
  - DB2 V8.1.1.104, s060120, U805924, FP 11

Configuration of the existing environment

To test the migration process, we used the following configuration of the current environment:

- Database Server
  - SMP/E environment: DB2 V8 Fallback SPE
  - Storage management: ACS routines, FlashCopy enablement
  - SAP buffer pool configuration
  - SAP DB2 V8 ZPARM SAP recommendations
  - Stored procedure Workload Management environments
  - Data sharing configuration components
    - Coupling Facility structures; resize LOCK1, add GBPools
    - Group attach name, locations, TCP/IP port numbers
    - Data set names

- SAP application
  - SAP connect.ini file
  - SAP Client Copy processes: Create, Run, Stop, and Delete
- Tested SAP failover

DB2 9 for z/OS migration preparation

Before we could migrate to DB2 9 for z/OS, we prepared our system in the following way:

1. Used initial program load (IPL) to move DB2 9 for z/OS ERLY code to validate with DB2 V8
2. Reconfigured the system library data sets; each member has its own DSNLOAD
3. Reconciled SAP ZPARM recommendations with DB2 9 for z/OS installation panels
4. Performed a JCL review of install jobs
5. Did a bootstrap dataset (BSDS) conversion
6. Verified the DB2 V8 catalog and directory integrity with DSN1CHKR
7. Performed other optional checks and verifications
   a. STOGROUP not using mix of specific and non-specific volume
   b. Catalog table consistency
8. Obtained FlashCopy of the DB2 V8 baseline

Overview of offline migration to compatibility mode

We used the following steps to perform an offline migration to compatibility mode:

1. Stopped the SAP application
2. Stopped one data sharing group (in our case, D2S1)
3. Migrated D2S1 to compatibility mode
   a. Ran DSNTIJIN: Created new VSAM data sets
   b. Ran DSNTIJIC: Image copied DB2 catalog and directory
   c. Ran Validate for the status of utilities and pagesets
   d. Moved in DB2 9 for z/OS libraries

    e. Ran DSNTIJUZ: ZPARM and DSNHDECP
    f. Edited PROCs
    g. Defined user authorization exits if needed
    h. Started D2S1 in DB2 9 for z/OS MAINT mode
    i. Ran DSNTIJTC: Tailored the catalog
    j. Performed checks on the DB2 catalog and directory
    k. Ran DSNTIJRI: Rebuilt indexes
    l. Ran various other jobs to perform link-edits and binds

4. Migrated D2S2 to compatibility mode
    a. Moved in DB2 9 for z/OS libraries
    b. Ran DSNTIJUZ: ZPARM and DSNHDECP
    c. Edited PROCs
5. Started D2S1 and D2S2 in regular mode
6. Started SAP
    a. Created Client 200
    b. Started Client Copy
7. Stopped Client Copy in preparation for migration to new function mode

Overview of offline migration to new function mode: Group-wide event

We used the following steps to migrate to new function mode:

1. Stopped SAP
2. Stopped D2S2
3. Migrated to ENFM
    a. Started D2S1 in MAINT mode
    b. Verified that no indexes on DB2 catalog were created with user managed data sets
    c. Ran an image copy of the DB2 catalog and directory
    d. Ran DSNTIJEN: Enabled new function mode
        i. Saved ENFM LRSN in BSDS
        ii. Created a new buffer pool and page size for SYSOBJ

4. Migrated to new function mode
   a. Ran DSNTIJNF: New function mode
   b. Ran DSNTIJNG: Reassembled and linked DSNHDECP indicating NEWFUN=YES for both D2S1 and D2S2
5. Started D2S2 and D2S1 in normal mode
6. Started SAP
   a. Started Client Copy
   b. Performed the planned failover successfully

Overview of offline fallback to compatibility mode

We performed the following steps:

1. Validated client copy completion
2. Stopped SAP
3. Performed fallback to compatibility mode from new function mode
   a. Ran DSNTIJCS (CATENFM CMON)
   b. Ran DSNTIJNG (DSNHDECP NEWFUN=NO)
4. Stopped and started D2S1 and D2S2
5. Started SAP
   a. Started a test client copy successfully; ran for 30 minutes
   b. Performed a planned failover successfully; no other client work was running at the time

Pre-online migration activities

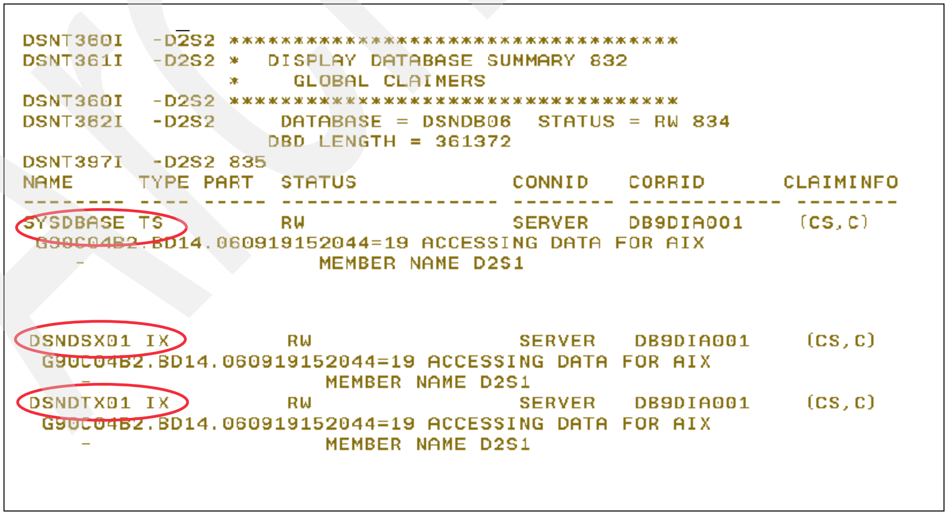
We performed the following activities for pre-online migration:

1. Ran flashback to the DB2 V8 baseline
2. Restored DB2 V8 PROCs
3. Ran DSN1CHKR during the time that the SAP application was down

Overview of online migration test

The online migration test involved the following steps:

1. Started the DB2 V8 data sharing group
2. Started SAP and connected to D2S1
   a. Created client 200
   b. Started client copy
3. Started the online migration process to migrate D2S2 to compatibility mode while client copy was running on D2S1
4. Used the same migration process as the offline migration process
5. DSNTIJTC job could not run while running SAP client copy
   a. DSNTIJTC job hung
   b. SAP work processes hung
   c. DB2 commands were not processing
   d. Brought down D2S1 where SAP was connect by modifying IRLM down
6. Conflict condition stopped DSNTIJTC from running
   a. SAP dialog process was holding a cursor stability until commit claim on SYSDBASE and two of its indexes (see Figure A-1).
   b. After bringing D2S1 down, DSNTIJTC running on D2S2 was able to complete.



```
DSNT360I   -D2S2 ***********************************
DSNT361I   -D2S2 *   DISPLAY DATABASE SUMMARY 832
                 *     GLOBAL CLAIMERS
DSNT360I   -D2S2 ***********************************
DSNT362I   -D2S2       DATABASE = DSNDB06   STATUS = RW 834
                      DBD LENGTH = 361372
DSNT397I   -D2S2 835
NAME      TYPE PART  STATUS             CONNID   CORRID       CLAIMINFO
-------- ---- ----- ----------------- -------- ------------ --------
SYSDBASE TS          RW                 SERVER   DB9DIA001     (CS,C)
  G90C04B2.BD14.060919152044=19 ACCESSING DATA FOR AIX
     -                    MEMBER NAME D2S1


 DSNDSX01 IX          RW                 SERVER   DB9DIA001     (CS,C)
  G90C04B2.BD14.060919152044=19 ACCESSING DATA FOR AIX
     -                    MEMBER NAME D2S1
 DSNDTX01 IX          RW                 SERVER   DB9DIA001     (CS,C)
  G90C04B2.BD14.060919152044=19 ACCESSING DATA FOR AIX
     -                    MEMBER NAME D2S1
```

Figure A-1   Claim information on the SAP dialog process

7. Completed the migration of both DB2 members to compatibility mode offline
   a. DSNTIJTC completed normally when D2S1 came down
   b. Restarted D2S1 to resolve retained locks and clean up after bringing the system down hard
   c. Stopped D2S1 and migrated to compatibility mode
      i. Moved in DB2 9 for z/OS libraries
      ii. Changed PROCs
      iii. Generated new ZPARM and DSNHDECP

Overview of online migration to new function mode

The online migration to new function mode involved the following steps:

1. Started both DB2 members in compatibility mode
   a. Started SAP
   b. Restarted client copy 200
2. Ran DSNTIJRI: Index rebuild
3. Ran DSNTIJTM: Assemble and link-edit DSNTIAD, enables REXX™
4. Ran DSNTIJSG: Package rebinds, stored procedure redefines
5. Checked for view regeneration errors
6. Checked for indexes on catalog using user-defined data sets
7. Ran DSNTIJEN: Enabled new function mode
   a. Saved ENFM LRSN in BSDS
   b. New buffer pool and page size for SYSOBJ
8. Ran DSNTIJNF: New function mode
9. Ran DSNTIJNG: Re-assembled and linked DSNHDECP indicating NEWFUN=YES for both members
10. Ran various other jobs to perform link-edits, binds, and new schema support

Overview of online fallback to compatibility mode

We performed the following steps for online fallback to compatibility mode:

1. Client Copy 200 still running
2. Performed fallback to compatibility mode from new function mode
   a. Ran DSNTIJCS (CATENFM CMON)
   b. Ran DSNTIJNG (DSNHDECP NEWFUN=NO)
   c. Completed fallback to compatibility mode at 17:32:16
3. Started receiving abend RC00C90101 at 17:37:19 on the following items:
   a. DSNDB06.SYSRTSTS (once)
   b. A140X991.SWWLOGHI (repeated)
   c. A140X991.SWWL11VA (repeated)
4. Scheduled client 800 batch jobs running
   a. Started receiving short dumps for runtime errors at 17:38:00 on September 19
   b. Continued receiving errors until 09:05:58 on September 20, when new function mode was re-enabled
5. Completed client copy 200 at about 23:00

Data inconsistency error and short dump messages after fallback to compatibility mode

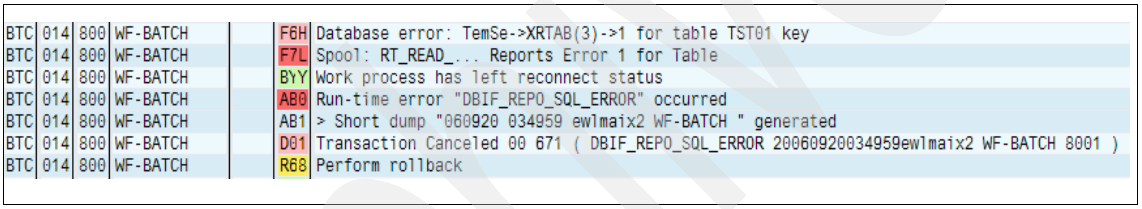
Example A-1 shows the data inconsistency error message that we encountered in the SYSLOG.

Example: A-1 Data inconsistency error message in SYSLOG

```
17.37.19 STC28636  DSNI014I  -D2S1 DSNKISPL DATA IN USE DURING ABEND
335
   335                        REASON 00C90101
   335                        ERQUAL 53C9
   335                        TYPE 00000302
   335                       NAME DSNDB06 .SYSRTSTS.X'00000069'
   335                        CONNECTION-ID=D2S1
   335                       CORRELATION-ID=014.RTSTST00
   335                       LUW-ID=DB2SG.SCPD2S1.BF6E239C38A7=0
17.37.59 STC28636  DSNI014I  -D2S1 DSNKISPL DATA IN USE DURING ABEND
349
```

```
   349                        REASON 00C90101
   349                        ERQUAL 53C9
   349                        TYPE 00000302
   349                       NAME A140X991.SWWLOGHI.X'0000EF69'
   349                        CONNECTION-ID=D2S1
   349                        CORRELATION-ID=DB9BTC013
   349                      LUW-ID=G90C04B2.C3D8.060919174136=133124
17.37.59 STC28636  DSNI014I  -D2S1 DSNKISPL DATA IN USE DURING ABEND
350
   350                        REASON 00C90101
   350                        ERQUAL 53C9
   350                        TYPE 00000303
   350                       NAME A140X991.SWWL11VA.X'0000434F'
   350                        CONNECTION-ID=D2S1
   350                        CORRELATION-ID=DB9BTC013
   350                          LUW-ID=G90C04B2.C3D8.060919174136=133124
```

Figure A-2 shows the SAP log with the short dump message.



```
BTC 014 800 WF-BATCH      F6H Database error: TemSe->XRTAB(3)->1 for table TST01 key
BTC 014 800 WF-BATCH      F7L Spool: RT_READ_... Reports Error 1 for Table
BTC 014 800 WF-BATCH      BYY Work process has left reconnect status
BTC 014 800 WF-BATCH      AB0 Run-time error "DBIF_REPO_SQL_ERROR" occurred
BTC 014 800 WF-BATCH      AB1 > Short dump "060920 034959 ewlmaix2 WF-BATCH " generated
BTC 014 800 WF-BATCH      D01 Transaction Canceled 00 671 ( DBIF_REPO_SQL_ERROR 20060920034959ewlmaix2 WF-BATCH 8001 )
BTC 014 800 WF-BATCH      R68 Perform rollback
```

Figure A-2 SAP database error

Overview of online migration back to new function mode

The online migration back to new function mode entailed the following actions:

1. No recycle of SAP or DB2 data sharing group
2. Started deletion of client 200
3. Ran DSNTIJNF in new function mode using CATENFM=COMPLETE
4. Ran DSNTIJNG to re-assemble and link DSNHDECP, indicating NEWFUN=YES for both D2S1 and D2S2
5. Remigration to new function mode completed at 09:08
   a. RC00C90101 for A140X991.SWWLOGHI and A140X991.SWWL11VA stopped
   b. Short dumps for client 800 stopped

Additional DB2 9 for z/OS feature and function verification

In addition, we verified the following features and functions:

- Object creation, deletion, referential integrity
- Copy, Load and REORG utilities
- DSNTIAUL, DSNTEP2, DSNTIAD
- User defined functions
- Stored procedures
- Materialized query table (MQT)
- LOB; loading and unloading

Results

The results of this migration showed us that preparation of the test environment took more time than the migration itself. Offline migration of DB2 V8 to DB2 9 for z/OS was straightforward and went smoothly.

In addition, we found that the fallback to compatibility mode from new function mode was simple and nondisruptive. The fallback process was tested both offline and online.

An SAP process was abending until the system was migrated back to new function mode.

We found that online migration from DB2 9 for z/OS compatibility mode to new function mode is nondisruptive and can be accomplished while the SAP application is running.

We found that an online migration from DB2 V8 to DB2 9 for z/OS compatibility mode may be disruptive due to the DSNTIJTC job having a high potential for claim conflicts with a running an SAP application. However, if DSNTIJTC fails, it can simply be repeated.

B

Project test environment

In this appendix, we explain the hardware and software environment used for this project. We describe the migration and installation scenario.

Hardware configuration

Figure B-1 shows an overview of the hardware that we used for this project. Two logical partitions (LPARs), SC04 and SC47, provide the needed CPU and memory resources. Additionally the System z Integrated Information Processor (zIIP) and System z Application Assist Processors (zAAP) are installed and used.

The SAP Web Application Server is running on IBM eServer™ pSeries® hardware.

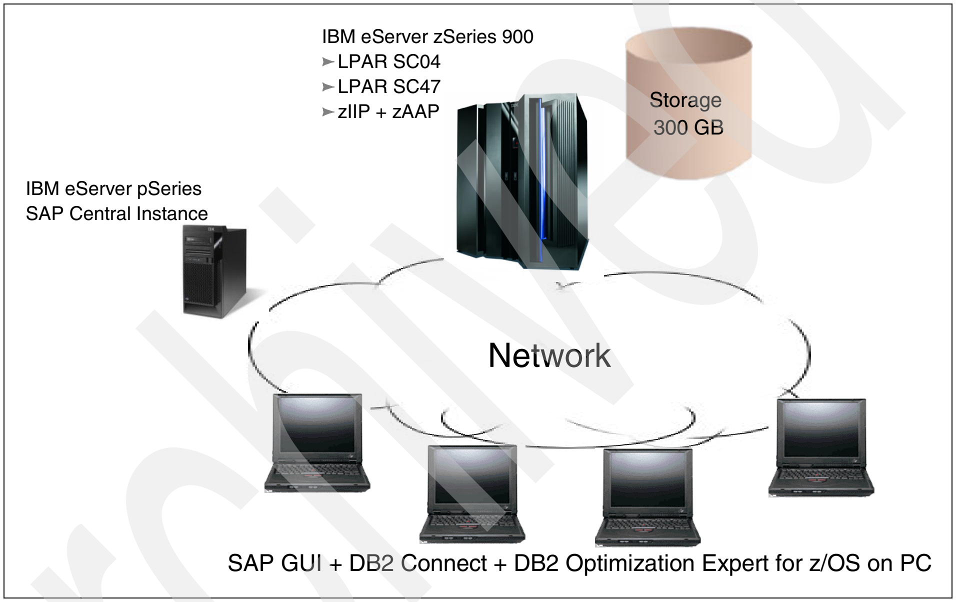


Figure B-1 Project hardware environment

Example B-1 shows the Display Command for the CPU configuration. It displays the implemented zIIPs and zAAPs.

Example: B-1 Displaying the CPU configuration

```
D M=CPU
 IEE174I 09.50.27 DISPLAY M 512
 PROCESSOR STATUS
 ID  CPU                  SERIAL
 00  +                     09991E2094
 01  +                     09991E2094
 02  +A                    09991E2094
 03  +A                    09991E2094
 04  +I                    09991E2094
 05  +I                    09991E2094
 06  -
 07  -

 CPC ND = 002094.S18.IBM.02.00000002991E
 CPC SI = 2094.710.IBM.02.000000000002991E
 CPC ID = 00
 CPC NAME = SCZP101
 LP NAME = A09        LP ID =  9
 CSS ID  = 0
 MIF ID  = 9

 + ONLINE    - OFFLINE    . DOES NOT EXIST    W WLM-MANAGED
 N NOT AVAILABLE

 A        APPLICATION ASSIST PROCESSOR (zAAP)
 I        INTEGRATED INFORMATION PROCESSOR (zIIP)
 CPC ND  CENTRAL PROCESSING COMPLEX NODE DESCRIPTOR
```

Software configuration

In this section, we explain the DB2-related software environment tests, which are based on z/OS 1.7 and the necessary subsystems, such as DFSMS™, JES, the Security Server (RACF), TCPIP, UNIX System Services, and DB2 9 for z/OS. For some of the incremental Backup System tests, we used another system with z/OS 1.8, because the prerequisite for these test is DFSMS 1.8.

Using an AIX SAP Application Server as the SAP Central Instance, we implemented and tested the DB2 Connect V9 and the new call level interface (CLI) as the DRDA clients.

Additionally we implemented and tested the DB2 Connect Server, installed on Microsoft XP systems, to access the SAP database using the Control Center and the Optimization Center for administration and monitoring purposes.

Figure B-2 shows the DB2 environment.

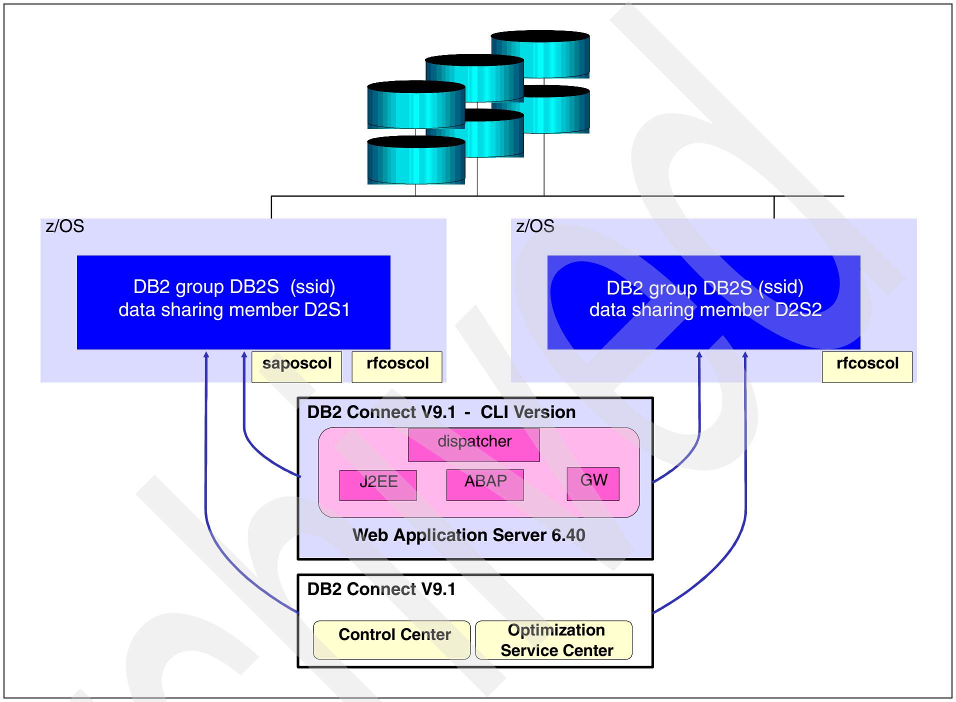
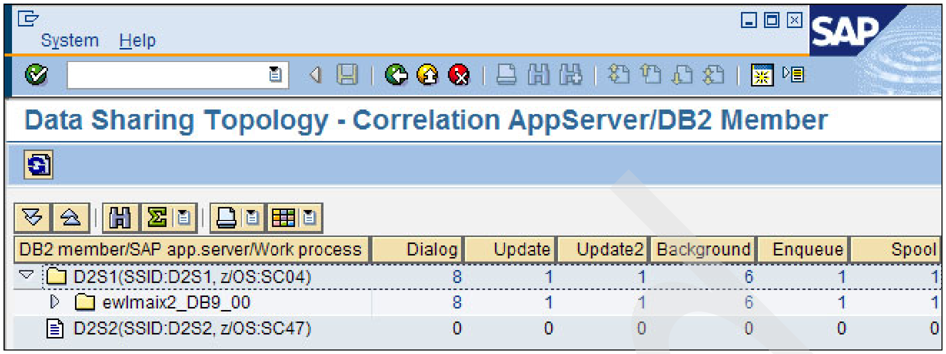


Figure B-2 The DB2 environment

The environment is based on the following items:

- 150 GB storage
- DB2 9 for z/OS with two data sharing members to provide the migration environment

  The SAP transaction ST04 with the display function for the Data Sharing Topology is displayed as shown in Figure B-3.



Data Sharing Topology - Correlation AppServer/DB2 Member

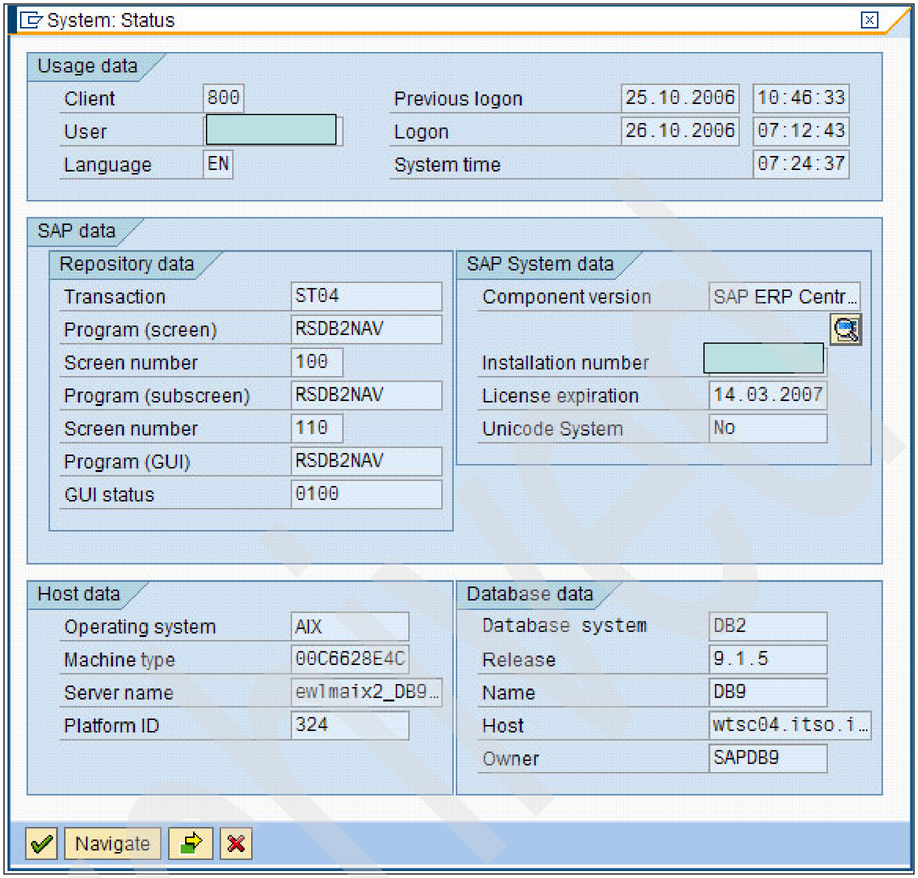
DB2 member/SAP app.server/Work process	Dialog	Update	Update2	Background	Enqueue	Spool
D2S1(SSID:D2S1, z/OS:SC04)	8	1	1	6	1	1
ewlmaix2_DB9_00	8	1	1	6	1	1
D2S2(SSID:D2S2, z/OS:SC47)	0	0	0	0	0	0

Figure B-3 Data Sharing Topology

- DB2 Connect Server V9 on AIX
- DB2 Connect CLI V9 on AIX
- DB2 Connect Server V9 on Microsoft XP
- DB2 Control Center
- DB2 Optimization Service Center
- SAP NetWeaver 2004 with SAP Web Application Server (SAP WAS) 6.40 and SAP Enterprise Resource Planning (SAP ERP) 5.0 with EDIS

The DB2 and SAP releases

The SAP Status screen shown in Figure B-4 summarizes the installed DB2- and SAP-related software. DB2 9.1.5 is installed, along with the SAP ERP Central Component 5.0 under NetWeaver 6.40 with the DB2 CCMS transport SAPK640OCM and the prerequisite SAP Support package 14.



System: Status

Usage data

Client	800	Previous logon	25.10.2006	10:46:33
User		Logon	26.10.2006	07:12:43
Language	EN	System time		07:24:37

SAP data

Repository data

Transaction	ST04
Program (screen)	RSDB2NAV
Screen number	100
Program (subscreen)	RSDB2NAV
Screen number	110
Program (GUI)	RSDB2NAV
GUI status	0100

SAP System data

Component version	SAP ERP Centr...
Installation number	
License expiration	14.03.2007
Unicode System	No

Host data

Operating system	AIX
Machine type	00C6628E4C
Server name	ewlmaix2_DB9...
Platform ID	324

Database data

Database system	DB2
Release	9.1.5
Name	DB9
Host	wtsc04.itso.i...
Owner	SAPDB9

Figure B-4 DB2 and SAP

In addition, to provide an effective test environment, we used the EDIS system, which provides master and transactional data. The list in Example B-2 is an extract of the transaction DB02. It shows that the size of the SAP NetWeaver database is about 140 GB.

Example: B-2 The SAP database size

```
Date         Values          Size/KB               Free/KB        % used
25.10.2006     Total           142.452.720           35.464.884        75
24.10.2006     Total           142.449.744           35.464.884        75
23.10.2006     Total           142.233.648           35.464.884        75
```

C

zIIP measurements

IBM introduced the System z9 Integrated Information Processor (zIIP) as a specialty engine to help improve resource utilization and lower the cost of portions of eligible workloads. zIIPs require at least DB2 for z/OS V8, a System z9 machine, z/OS 1.6, and, for maximum exploitation in an SAP environment, DB2 Connect. zIIPs can also imply specific application software levels such as for SAP.

zIIPs are especially intended for data serving workloads. This test describes some measurements done in the IBM laboratories using zIIP hardware and software. Four types of DB2 V8 workloads were measured:

- SAP OLTP exploiting DRDA using DB2 Connect
- Business Intelligence (BI) batch direct attached
- SAP BW query exploiting DRDA
- DB2 Utility

The purposes of these measurements were to:

- Test these functions
- Quantify the degree of redirect

The same great features that exploit the zIIPs in DB2 V8 are available in DB2 9 for z/OS.

Workload measurement background

As part of the continuous testing and measuring that IBM does with its products, we wanted to investigate the effects of running different workloads with and without zIIPs. We did not tune these measurements as you would for a benchmark. Initially we started this effort by using prototype hardware and software. The results shown here represent the final designs and implementation.

SAP OLTP workload

One of the workloads we measured was the SAP AG mySAP™ Database Server doing OLTP type work. We used DB2 Connect to use DRDA connected to external AIX SAP application servers that submit work to a DB2 database server from AIX SAP application servers.

For this set of measurements, we used a standard Sales and Distribution (SD) workload. However, we did not run the SAP SD Standard Application Benchmark. While there may be some similarity in business functions performed, the measurements we did were stress tests, not SAP-certified benchmarks.

Business information batch workload

We also looked at a unique BI Batch workload that was run direct attached. That is, there was no use of DRDA. The native SQL came from a script that was run in the same z/OS as DB2.

We used a specially developed query set. While this set is based on field experience and observation of customer environments, it was not intended to represent typical customer queries. Rather it was intended to be complex, challenging, and stressful to DB2. It exhibits higher query parallelism than most customers will experience.

While the SAP application is not used, SAP BW data, structure, tables and database are. The query load was run against an SAP BW database with F fact table and dimension tables. The queries were run as a locally attached job (via DSNTEP4, not DRDA). Each query was run sequentially in a single stream. There were 79 queries, with three additional environmental settings:

- set SQLID=SAPR3
- set current degree=any
- set current timestamp

There was a mixture of CPU bound queries and I/O bound queries.

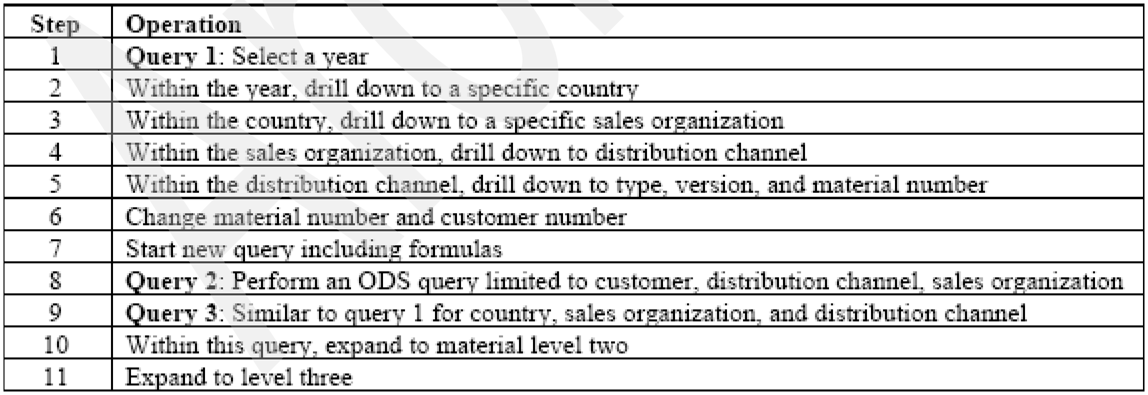
The F fact table was manually partitioned via DB2 to get more parallelism, increasing the zIIP redirect. It is important to note that the partitioning scheme mirrors what was provided by SAP for the F fact table. After partitioning the F fact table, the DB2 Explain tool indicates 57 of the 79 queries are parallel.

SAP BW Query workload

The third workload employed multiple streams (or multiple concurrent users) that submitted queries to the DB2 database server via DRDA from AIX SAP application servers. Each user submitted three queries per loop (see Figure C-1). These queries resulted in ten SQL select statements. Each loop had nine of the selects going against the F table and one going against the Operational Data Store (ODS) table. About 500+ SAP aggregates were used to provide better average response time.

The ODS was a non-star schema type table. In terms of CPU time, the ODS query dominated. Since SAP aggregates are not supported for ODS, the DB2 materialized query table (MQT) provided effective aggregate functions. Further, partitioning of the MQTs reduced the average response time to subsecond. The result was that this workload had short-running queries that were less complex than the BI Batch workload's 79 queries.

We did not run the SAP BW Standard Application Benchmark. While there may be some similarity in business functions performed, we did stress tests, not SAP certified benchmarks.



Step	Operation
1	**Query 1**: Select a year
2	Within the year, drill down to a specific country
3	Within the country, drill down to a specific sales organization
4	Within the sales organization, drill down to distribution channel
5	Within the distribution channel, drill down to type, version, and material number
6	Change material number and customer number
7	Start new query including formulas
8	**Query 2**: Perform an ODS query limited to customer, distribution channel, sales organization
9	**Query 3**: Similar to query 1 for country, sales organization, and distribution channel
10	Within this query, expand to material level two
11	Expand to level three

Figure C-1 Queries submitted by users to test the workload

DB2 utilities workload

The fourth set of workload runs were related to DB2 V8 utilities. DB2 build index processing in the Load, Reorg, and Rebuild Index utility can be redirected to zIIP in DB2 for z/OS V8. These measurements were conducted to understand the performance characteristics of Build Index Redirect, such as the percentage of redirect CPU time of Load Table, Load Partition, Reorg Table Space, Reorg Partition, Rebuild Index, and Reorg Index with a different number of indexes, and the additional cost of redirect. We also had the opportunity with these measurements to test the ProjectCPU function to see what costs it might have.

All of these measurements involved a noncompressed table with 20 million rows with 10 partitions, and 1, 2, or 6 indexes. Note that some customers, such as SAP customers, have tables that are much larger.

Hardware configurations

In this section, we describe the hardware configurations that we used in these measurements.

System z database server

These measurements were done on two different IBM System z9 Enterprise Class systems.

- The SAP OLTP, BI Batch, and SAP BW Query measurements were done on a System z9 S08 704 with four zIIPs configured and a total of 128 GB installed. The regular central processors (CPs) and zIIP CPs were in the same book. This is the maximum number of zIIPs that can be configured on this model. The DB server resided on a z/OS LPAR with four dedicated CPs plus four zIIPs and 16 GB of central memory. The base measurements without zIIP were done with the zIIP processors varied offline.
- The DB2 V8 Utility measurements were done on a System z9 S28 724 partition with four CPs and a total of 8 GB of storage. Initial runs were made without zIIPs configured. Later two zIIPs were configured and varied online. More zIIPs could have been configured on this model. The zIIP CPs were not necessarily in the same book as the regular CPs.

Each of these systems had the requisite service levels installed to support zIIPs. Customers should consult their local IBM representative or make sure they have the proper service levels on their system.

Database DASD

The *SAP OLTP and BI Batch workload measurements* used one Enterprise Storage Server (ESS) Model 800 (2105-800) with a total of eight FICON® attachments to the System z machine. The 2105-800 was a Turbo I model with 16 8-packs of 15K RPM 36.4 GB drives. The actual capacity was 384 3390-9 volumes. It had 16 GB of regular cache, 2 GB of non-volatile storage (NVS), and 8 short-wave FICON ports. Additionally, there were eight ESCON® ports. However, ESCON was not used for these runs. The eight FICON ports were connected via two 2032-064 MCDATA FICON directors to the System z9-704. The host used FICON Express4. However, this Model 800 used only 2 Gbps per port. We exploited the new Modified Indirect Data Address Word (MIDAW) CCW function for improved I/O responsiveness.

The *SAP BW Query measurements* used one dual frame IBM TotalStorage server Model DS8300 (2107-922/92E) with a total of eight FICON attachments to the System z machine. The 2107- 922/92E coupled unit has 48 8-packs of 15K RPM 73 GB drives. The actual capacity was 2496 3390-9 volumes. It had 128 GB regular cache, 4 GB NVS and 32 long-wave FICON ports. However, only eight FICON ports were used for the test, and they were direct connected to the System z9-704. The host used FICON Express4. However, this DS8300 used only 2 Gbps per port. We exploited the new MIDAW CCW function for improved I/O responsiveness.

The *DB2 V8 Utility measurements* used one ESS Model 800 (2105-800) with four FICON Express2 attachment to the System z machine. The 2105-800 was a Turbo I model. The database and workload were spread on 34 volumes with additional 14 volumes for work files. There were 16 GB regular cache, 2 GB NVS, and eight short-wave FICON ports.

pSeries application servers

The pSeries application servers had the following configuration:

- The SAP OLTP workload used three pSeries 690 (p690) servers each with 32 1.3 GHz processors and 128 GB of memory.
- No pSeries servers were used with the BI Batch workloads.
- The SAP BW Query workload used one pSeries 570 with 16 1.9 GHz processors and one p690 with 32 1.3 GHz processors; each has 128 GB of memory.
- No pSeries servers were used with the DB2 V8 Utility workloads.

pSeries presentation servers

The pSeries presentation servers had the following configuration:

- The SAP OLTP workload used one pSeries 620 with four 600 MHz processors and 4 GB of memory.
- No pSeries servers were used with the BI Batch workloads.
- The SAP BW Query workload used one pSeries 630 with four 1.0 GHz processors and 8 GB of memory as well as one pSeries 55A with four 1.9 GHz processors and 32 GB of memory.
- No pSeries servers were used with the DB2 V8 Utility workloads.

Network

Gigabit Ethernet networks were used for all network connections. Each of the p690 and p570 application servers was connected to the System z9-704 via the OSA-Express2 adapter. The following figures show conceptual views of the configurations that were used:

- SAP OLTP is shown in Figure C-2.
- BI is shown in Figure C-3 on page 238.
- SAP BW Query is shown in Figure C-4 on page 239.
- DB2 Utility is similar to Figure C-3 on page 238.

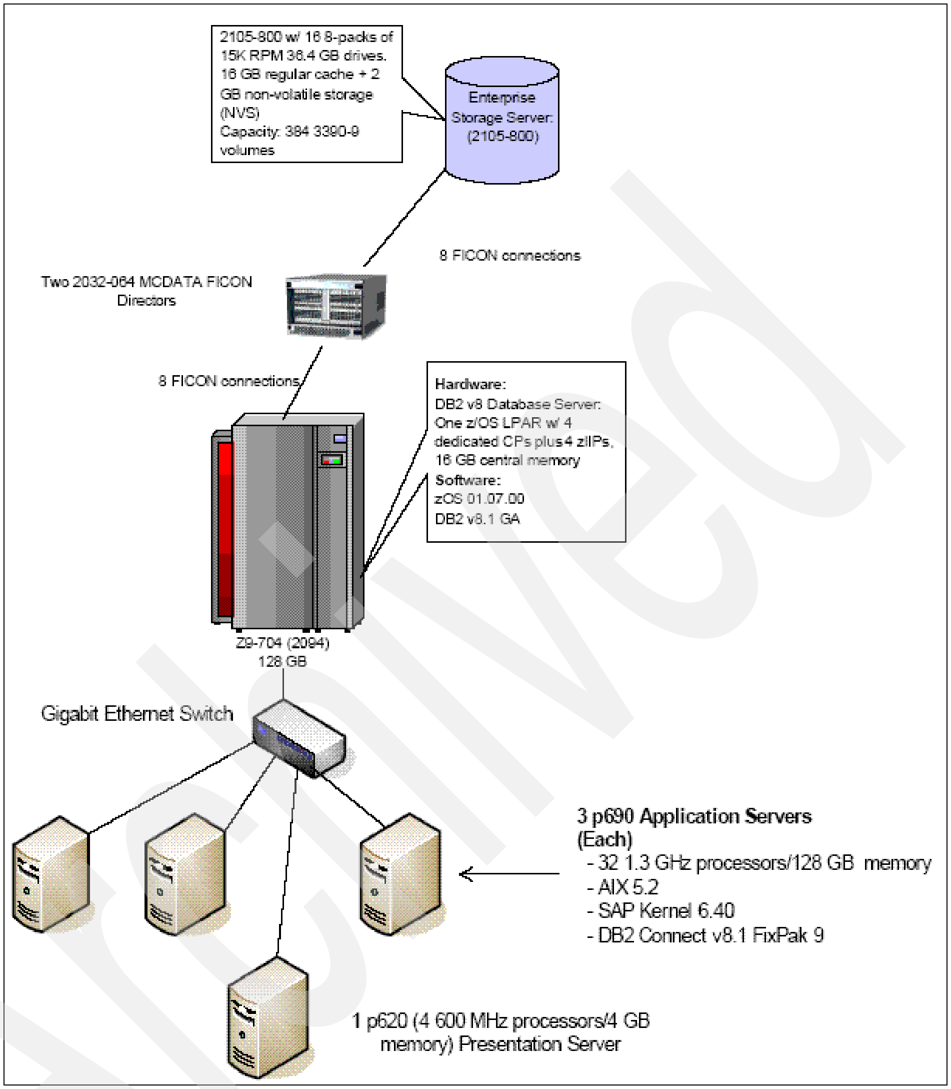

2105-800 w/ 16 8-packs of 15K RPM 36.4 GB drives. 16 GB regular cache + 2 GB non-volatile storage (NVS)
Capacity: 384 3390-9 volumes
Enterprise Storage Server: (2105-800)
8 FICON connections
Two 2032-064 MCDATA FICON Directors
8 FICON connections
Hardware:
DB2 v8 Database Server: One z/OS LPAR w/ 4 dedicated CPs plus 4 zIIPs, 16 GB central memory
Software:
zOS 01.07.00
DB2 v8.1 GA
Z9-704 (2094)
128 GB
Gigabit Ethernet Switch
3 p690 Application Servers
(Each)
- 32 1.3 GHz processors/128 GB memory
- AIX 5.2
- SAP Kernel 6.40
- DB2 Connect v8.1 FixPak 9
1 p620 (4 600 MHz processors/4 GB memory) Presentation Server

Figure C-2 SAP OLTP configuration

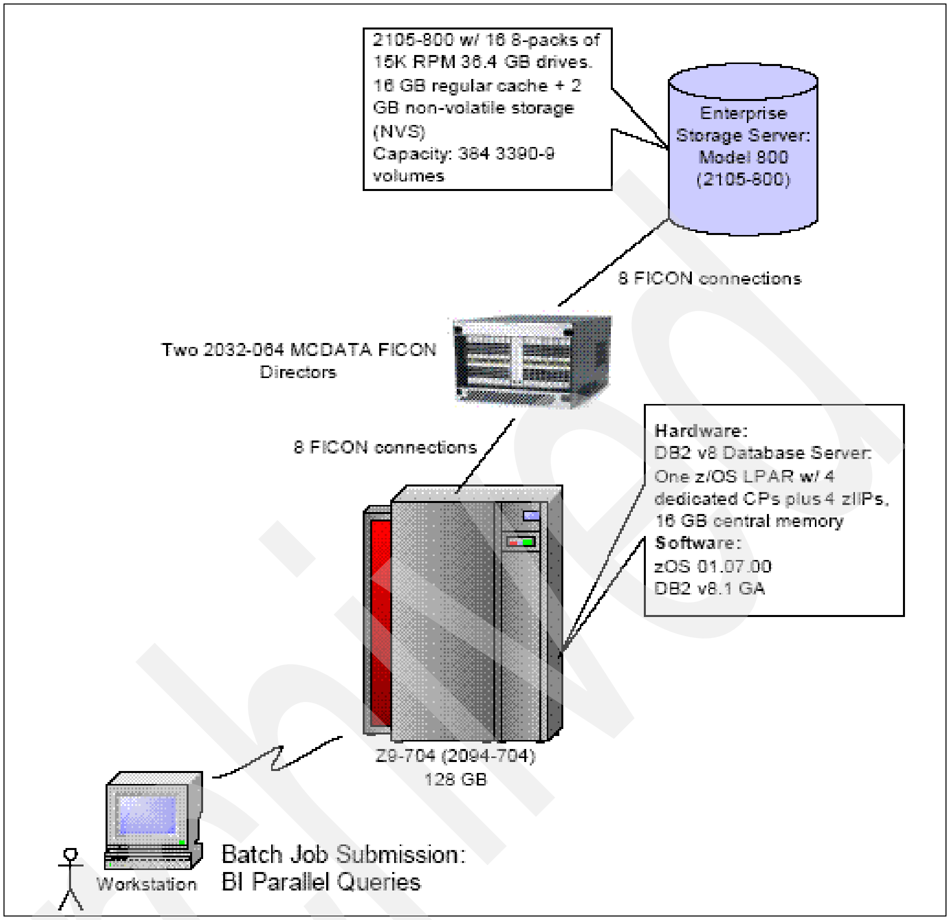


Figure C-3   BI Batch configuration

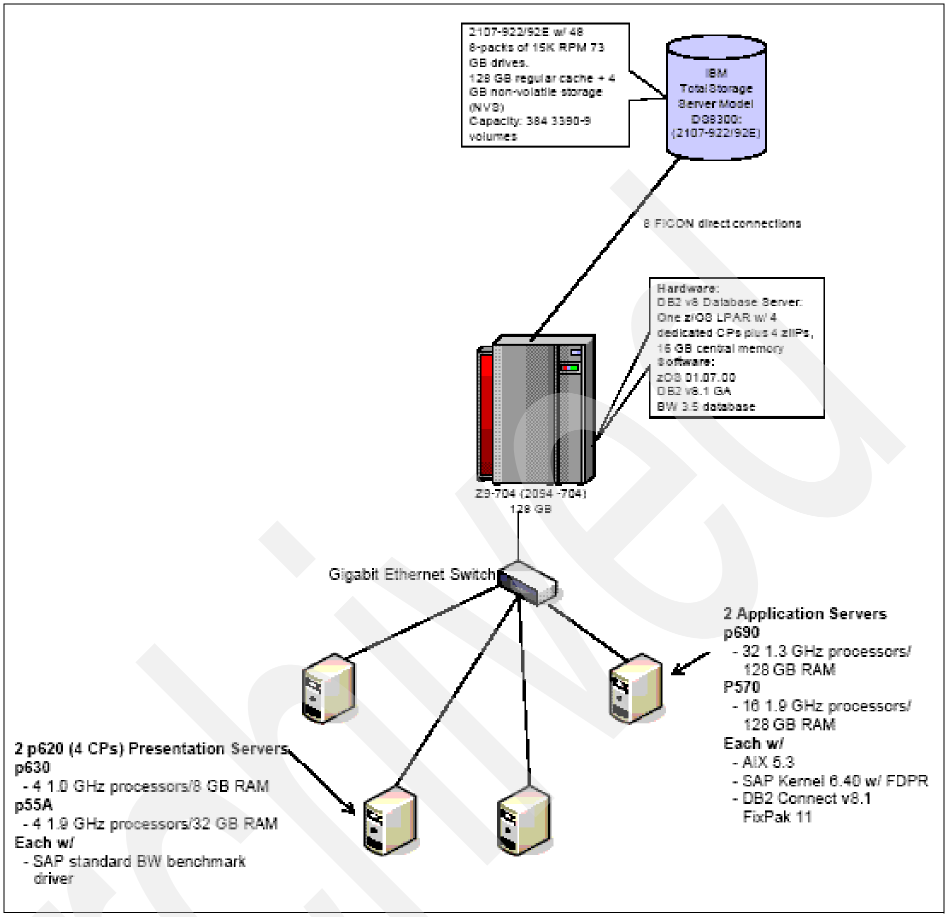


Figure C-4 SAP BW query configuration

System and software environment

In this section, we describe the system and software environment that we used for the measurement tests:

- MVS z/OS release 01.07.00
- DB2 V8.1 GA with PTFs UK16616 and UK17089

  At the time this book was written, this was the most up-to-date DB2 implementation for Parallel Query and DB2 Utility zIIP exploitation.
- DB2 Connect: V8.1 fix pack 9

  Because the zIIP environment continues to evolve, customers should consult their local IBM representative and DB2 Information APAR II14219 to make sure they have the proper service levels on their system.
- AIX: 5.2

SAP application levels

The following SAP application levels were used:

- SAP NetWeaver 04 SR1
- SAP BW 3.5 SP15

SAP disp+work output

The following information is received after running the SAP **disp+work** command:

```
SAP kernel release 6.40, kernel make variant 6.40, compiled on AIX 1 5
00538A4A4C00, compiled for 64 BIT, compile time February 2005 at
21:40:10, update level 0, patch number 58, source id 0.058
```

Note: The authors of this test material included Seewah Chan, Veng K. Ly, Mai N. Nguyen, Howard E. Poole, Michael R. Sheets, and Akira Shibamiya. They performed the test on 02 October 2006.

D

Work file and temporary table space changes

In this appendix, we discuss some of the changes that we made to the work file and temporary table spaces (TEMP spaces). We also discuss some of the changes that we made within DB2 with regard to the TEMP space of the workfile database.

Enhancements for TEMP spaces

DB2 9 contains several enhancements for the TEMP spaces of the workfile database. Among them is the converging of two temporary databases supported by DB2 into one database. You can now create a single database, the workfile database DB2 member of the database group. The workfile database is used as storage for all temporary files and tables, while preserving the external functionality of the temporary tables as it is today.

Other enhancements include the following:

- DB2 9 has discontinued use of the TEMP database and switched the temporary tables that currently use them.
- The TEMP database uses the workfile database.
- Enhancements have been made to the workfile database to compensate for the relevant TEMP database features that were not currently in the workfile database.

For DB2 Version 9, the TEMP database is eliminated. Table spaces used by declared global temporary tables and static scrollable cursors now reside in the workfile database. As in previous versions of DB2, the workfile database supports 4 KB and 32 KB page size table spaces. It continues to be named DSNDB07 on non-data sharing systems. On data sharing systems, the workfile database name for each member is user-specified, as discussed previously.

Prior to DB2 9 for z/OS, the workfile database user managed data sets were defined by the installation process for the workfile table spaces. Now in DB2 9 for z/OS, the installation process generates the DDL to create table spaces in the workfile database. It uses the DB2 managed storage in the SYSDEFLT storage group. If you want your workfile table spaces on user managed data sets, you need to provide your own AMS statements to define them. You must also modify the installation generated DDL to specify the VCAT clause in place of the STOGROUP clause. We did not do this because we let DB2 and storage management subsystem (SMS) manage this for us.

In DB2 Version 9, the installation process is enhanced as part of the Installation and Samples to permit you to specify that installation created objects, including SYSDEFLT, to be defined on storage controlled by the SMS. You can still specify that these objects are to be defined on a specific disk volume serial number, as traditionally.

Creation of SYSDEFLT is moved from installation job DSNTIJSG to installation job DSNTIJTM so that it is available for use by the workfile database.

Size requirements

The workfile database is used as storage for processing SQL statements that require working storage. Table D-1 shows the disk requirement estimates for the temporary work files in the workfile database in cylinders. Other workfile database storage requirements relate to migration.

Table D-1 Workfile database storage requirements

Site size	3380	3390
Small	35	29
Medium	35	29
Large	547	456
Extra large	854	712

You might need more storage for the workfile database if you have a large amount of data to sort and a large amount of concurrent sort activity. If you are sorting compressed data, allow for the same amount of storage that you would need if the data was not compressed. The maximum amount of storage that you need is enough to satisfy the requirements of the largest combination of concurrent activities that use the workfile database. It just so happened that we needed more storage space.

When we attempted to use the SAP application and its functions, we ran into space problems with the workload database. The workload database was built for a smaller system. We had some work on our hands to rebuild this database.

Migration of the workload database

The migration mode needs to account for existing table spaces to avoid name collisions. The configuration CLIST, which derives tablespace names sequentially when generating the DDL used for installation, has no way to discover names of existing table spaces. With the new functionality for the WORKLOAD DATABASE, a DB2 REXX exec called DSNTWFG is provided. The configuration CLIST calls DNTWFG to generate DDL for creating workfile table spaces with unused names. The CLIST then places the DDL in the customized DSNTIJTM job as instream data for processing by the DSNTIAD dynamic SQL processor.

The required inputs for the workload database table spaces are:

- The name of the workfile database
- The number of 4 KB and 32 KB table spaces to be created in the workfile database
- The total amount of storage for each table space type

For each type of workfile table space specified, perform the following tasks:

- Generate the next name in sequence, for example, DSN4K01, DSN4K02, and so on.
- Check SYSIBM.SYSTABLESPACE to see if a table space by that name already exists in the workfile database
  - If it does, generate the next name in sequence and try again.
  - Otherwise, generate and output the CREATE TABLESPACE statement for the current name and the DDL for creating the specified number of table spaces of each type.

Actual work database creation

We came across a problem when working with the SAP application. We started some work that uses the work database and work table spaces. However, the work files were too small for the work to be accomplished. Then we decided to make the work database table spaces larger. We used the JCL in Figure D-1 to make the table spaces larger.

We performed the following actions:

1. Stopped the work database:

   ```
   /-D2S1 STOP DB(WRKD2S1) SP(*)
   ```

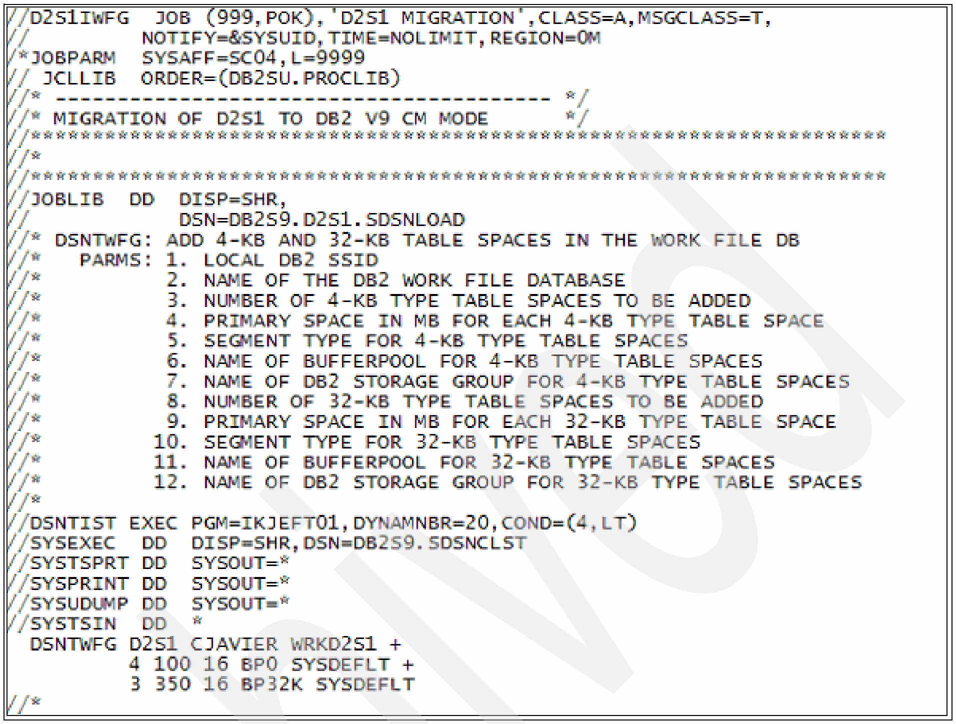
2. Deleted the database and table spaces:

   ```
   Drop Database WRKD2S1
   ```

3. Created the WORKFILE database:

   ```
   CREATE DATABASE WRKD2S1 AS WORKFILE FOR D2S1 ;
   ```

4. Ran the JCL created for the WORKFILE (see Figure D-1).



```
//D2S1IWFG  JOB (999,POK),'D2S1 MIGRATION',CLASS=A,MSGCLASS=T,
//          NOTIFY=&SYSUID,TIME=NOLIMIT,REGION=0M
/*JOBPARM   SYSAFF=SC04,L=9999
// JCLLIB   ORDER=(DB2SU.PROCLIB)
//* ---------------------------------------- */
//* MIGRATION OF D2S1 TO DB2 V9 CM MODE      */
//********************************************************************
//*
//********************************************************************
//JOBLIB   DD  DISP=SHR,
//             DSN=DB2S9.D2S1.SDSNLOAD
//* DSNTWFG: ADD 4-KB AND 32-KB TABLE SPACES IN THE WORK FILE DB
//*   PARMS: 1. LOCAL DB2 SSID
//*          2. NAME OF THE DB2 WORK FILE DATABASE
//*          3. NUMBER OF 4-KB TYPE TABLE SPACES TO BE ADDED
//*          4. PRIMARY SPACE IN MB FOR EACH 4-KB TYPE TABLE SPACE
//*          5. SEGMENT TYPE FOR 4-KB TYPE TABLE SPACES
//*          6. NAME OF BUFFERPOOL FOR 4-KB TYPE TABLE SPACES
//*          7. NAME OF DB2 STORAGE GROUP FOR 4-KB TYPE TABLE SPACES
//*          8. NUMBER OF 32-KB TYPE TABLE SPACES TO BE ADDED
//*          9. PRIMARY SPACE IN MB FOR EACH 32-KB TYPE TABLE SPACE
//*         10. SEGMENT TYPE FOR 32-KB TYPE TABLE SPACES
//*         11. NAME OF BUFFERPOOL FOR 32-KB TYPE TABLE SPACES
//*         12. NAME OF DB2 STORAGE GROUP FOR 32-KB TYPE TABLE SPACES
//*
//DSNTIST EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
//SYSEXEC  DD  DISP=SHR,DSN=DB2S9.SDSNCLST
//SYSTSPRT DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//SYSTSIN  DD  *
 DSNTWFG D2S1 CJAVIER WRKD2S1 +
         4 100 16 BP0 SYSDEFLT +
         3 350 16 BP32K SYSDEFLT
//*
```

Figure D-1 JCL to rebuild workfile database table spaces

In looking at the SYSTSIN DD card of this JCL, we can see the utility named DSNTWFG, using member D2S1 and the SYSADM user ID of CJAVIER. The name of the workfile database is WRKD2S1. For the number of 4k table spaces to be created, with the primary space in MBs, we chose 100. The type of segment for 4 KB table space was 16. For the buffer pool (BP0), we used the SYSDEFLT. The number of 32 KB type table spaces we used was 3. The primary space size of each table space in MBs was 350, which is about 712 cylinders because we have a large database system with SAP. For the next parameter, SEGMENT TYPE for 32 KB table space, we chose 16. BP32K was the name of our buffer pool for the 32 KB table spaces. The name of the storage group to be used for the 32 KB table spaces was SYSDEFLT.

5. Started the database from the DB2 Primary Options Menu (Option 7, DB2 Commands):

```
-STA DB(WRKD2S1) SP(*)
```

6. The workload then used the larger table spaces for its work and did not create an error condition.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information about ordering the following publication, see "How to get IBM Redbooks" on page 248.

- *LOBs with DB2 for z/OS: Stronger and Faster*, SG24-7270

Other publications

These publications are also relevant as further information sources:

- *DB2 Version 9.1 for z/OS Administration Guide*, SC18-9840
- *DB2 Version 9.1 for z/OS Performance Monitoring and Tuning Guide*, SC18-9851
- *DB2 Version 9.1 for z/OS SQL Reference*, SC18-9854
- *DB2 Version 9.1 for z/OS Utility Guide and Reference*, SC18-9855
- *IBM System z9 zIIP Measurements: SAP OLTP, BI Batch, SAP BW Query, and DB2 Utility Workloads*, WP100836

  `http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP100836`
- *Large Systems Performance Reference*, SC28-1187
- *MVS/ESA Storage Management Library: Managing Storage Groups*, SC26-3125
- *OS/390 V2R10.0 MVS Initialization and Tuning Reference*, SC28-1752

Online resources

These Web sites are also relevant as further information sources:

- CHECK DATA topic in IBM Information Management Software for z/OS Solutions information center

  `http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp?topic=/com.ibm.db29.doc.ugref/cxutchk.htm`

- DB2 Optimization Service Center

  `https://www14.software.ibm.com/webapp/iwm/web/reg/download.do?source=swg-db2zosc&S_PKG=dl&lang=en_US&cp=UTF-8#`

- Geographically Dispersed Parallel Sysplex (GDPS)

  `http://www-03.ibm.com/systems/z/gdps/`

- *SAP on IBM DB2 UDB for OS/390 and z/OS: Best Practice for Installing or Migrating to DB2 V8*

  `https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/5798c93d-0e01-0010-81b4-d641a79c1ab0`

- *System Copy for SAP Systems Based on SAP NetWeaver 2004s SR1 Java*

  `https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/20bd5cee-dabe-2910-fb97-c082a7c9c682`

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

Enhancing SAP by Using DB2 9 for z/OS

(0.5” spine)
0.475”<->0.875”
250 <-> 459 pages

Enhancing SAP by Using DB2 9 for z/OS

Understand the new indexing capabilities of DB2 9

Streamline efforts to meet compliance obligations

Examine new utility features and functions

This IBM Redbooks publication presents many of the new and improved features and functions of DB2 V9.1 for z/OS and DB2 Connect V9.1. It explains how they complement and benefit your SAP NetWeaver environment. This book also shares some of our experiences in migrating our DB2 V8 SAP data sharing environment to DB2 9 for z/OS with a minimal amount of outage.

This book is written for SAP and DB2 administrators. Knowledge of these products and of the z/OS environment is assumed.

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

SG24-7239-00 ISBN 0738489026