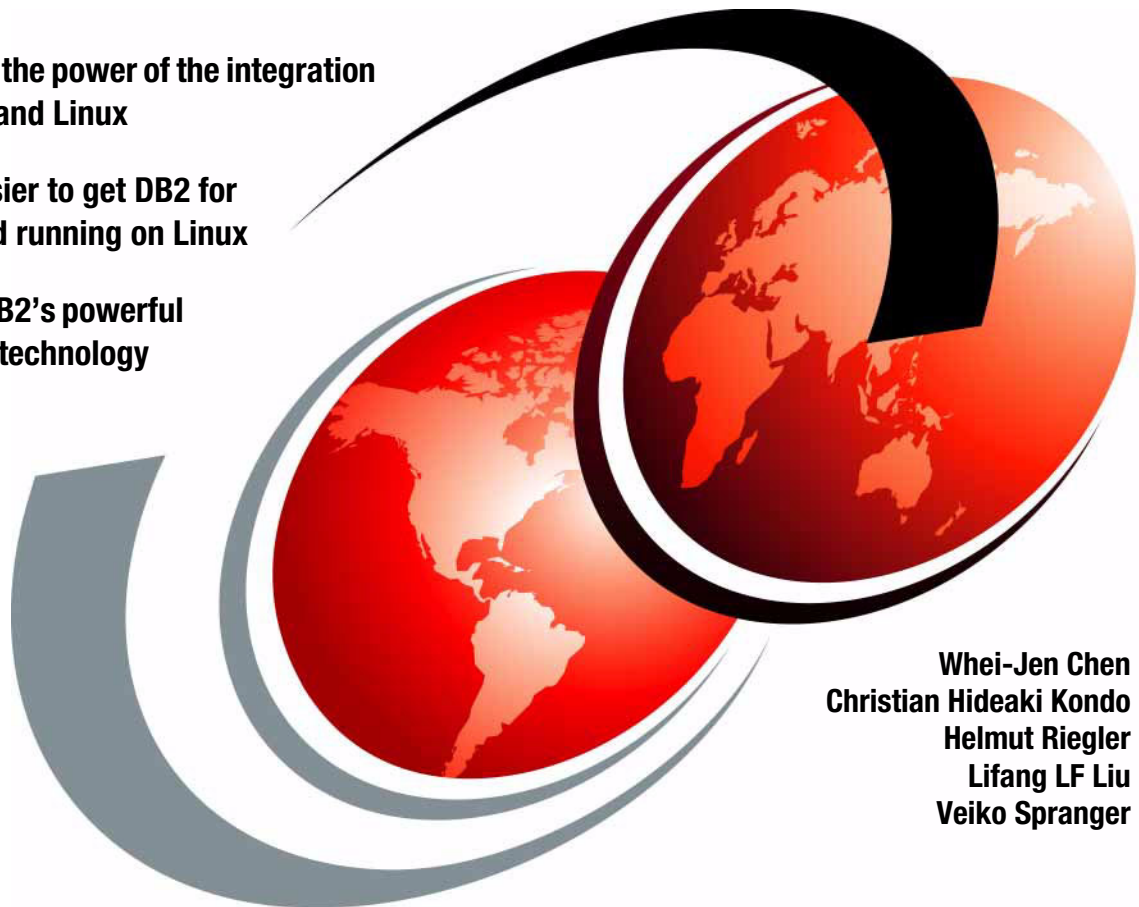


Up and Running with DB2 on Linux

Experience the power of the integration of DB2 9.5 and Linux

Make it easier to get DB2 for LUW up and running on Linux

Leverage DB2's powerful autonomic technology



Whei-Jen Chen
Christian Hideaki Kondo
Helmut Riegler
Lifang LF Liu
Veiko Spranger



International Technical Support Organization

Up and Running with DB2 on Linux

April 2008

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

First Edition (April 2008)

This edition applies to DB2 for Linux, UNIX, and Windows Version 9.5, Red Hat Enterprise Linux (RHEL) 5, SUSE Linux Enterprise Server (SLES) 10, and Ubuntu 7.10.

This document created or updated on April 17, 2008.

© Copyright International Business Machines Corporation 2008. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	iii
Trademarks	iv
Preface	iii
The team that wrote this book	iii
Acknowledgements	v
Become a published author	vi
Comments welcome	vi
Summary of changes	iii
April 2008, First Edition	iii
Chapter 1. Introduction	1
1.1 Overview	2
1.2 DB2 for Linux features and offerings	3
1.2.1 Features	3
1.2.2 Supported platforms	6
1.2.3 DB2 products and packages	7
1.3 DB2 environment	10
1.3.1 Deployment topologies	10
1.3.2 DB2 database objects	14
1.4 Parallelism with DB2	16
1.4.1 SMP environments	16
1.4.2 Database clusters	17
1.4.3 Partitioned database	19
Chapter 2. Installation	25
2.1 Basic requirements	26
2.1.1 Hardware supported by DB2	26
2.1.2 Linux distributions supported by DB2	26
2.1.3 Required disk spaces	27
2.1.4 Memory requirements	28
2.1.5 Communication requirements	29
2.1.6 Kernel parameter values	29
2.1.7 Additional software requirements	32
2.2 Installation considerations and planning	35
2.2.1 Considerations	35
2.2.2 Installation methods	38
2.2.3 Storage planning	39

2.2.4	Lab environment	45
2.2.5	Setting up NFS for a partitioned database environment installation .	46
2.2.6	User and group setup	49
2.2.7	Enabling ssh for a partitioned database environment installation . .	52
2.3	Installing DB2	58
2.3.1	Non-root installation	58
2.3.2	Root installation.	65
2.3.3	Installing a partitioned database environment.	77
2.3.4	Installing DB2 license files	84
2.3.5	Installing DB2 documentation	85
Chapter 3. Post installation tasks		89
3.1	Control Center setup	90
3.2	Preparation for the database creation	93
3.2.1	Code page considerations	93
3.2.2	Table space considerations	94
3.2.3	Linux specific configuration	101
3.2.4	Multiple partitioned specific configuration	101
3.3	Creating a database	104
3.3.1	Creating a single partitioned database	104
3.3.2	Creating a multiple partitioned database.	109
3.4	Further configuration	116
3.4.1	Change online log path	116
3.4.2	Change the archive log path	119
3.4.3	Change the path for db2diag.log.	119
3.5	Add database partition	121
3.6	DB2 configuration	128
3.6.1	STMM - Self tuning memory management	128
3.6.2	Database manager configuration parameters.	132
3.6.3	Database configuration	136
3.6.4	DB2 registry and environment variables	142
3.7	Security	145
3.8	Client configuration	148
3.8.1	Installing IBM Data Server Client	149
3.8.2	Configuring IBM Data Server Client	150
3.9	Configuring licensing	155
3.9.1	DB2 License Center	156
3.9.2	db2licm tool.	159
Chapter 4. Migration and Fix Packs		163
4.1	Migration planning	164
4.1.1	Migration requirements	165
4.1.2	Planning considerations	167

4.1.3 Migration test consideration	168
4.2 Migrating multi-partitioned database	169
4.2.1 Pre-migration tasks	169
4.2.2 Install DB2 Version 9.5	174
4.2.3 Migrating instances and databases.	174
4.2.4 Post-migration tasks	176
4.2.5 Enabling new DB2 9.5 functionality.	178
4.2.6 Migrating the DB2 Administration Server	178
4.2.7 Migrating DB2 clients	180
4.3 Migrating single-partitioned database	182
4.4 32-bit to 64-bit conversion	184
4.5 Migrating to a new database	185
4.6 Fix pack installation	190
Chapter 5. IBM Data Studio	193
5.1 Introduction	194
5.2 Installation	196
5.2.1 Installing IBM Data Studio	196
5.2.2 Installing IBM Data Studio Administration Console	203
5.2.3 Migration from the Developer Workbench	206
5.3 Features and functions	206
5.3.1 Terminology	207
5.3.2 Team function	212
5.3.3 XML editing	213
5.3.4 ER Diagramming	216
5.3.5 Data and object management	217
5.3.6 Data Web Services	219
5.3.7 Tips	220
Chapter 6. Administering databases	223
6.1 DB2 database backup and recovery	224
6.1.1 Enable roll forward recovery and log archiving	226
6.1.2 Recovery history file	231
6.1.3 Enable usage of TSM	233
6.1.4 Backup utility	235
6.1.5 DB2 database recovery	249
6.1.6 Redirected restore sample scenario	260
6.1.7 Recovering a dropped table sample scenario	264
6.2 Table/index reorganization and statistics collection	269
6.2.1 Automatic table maintenance	269
6.2.2 Manual table maintenance	275
6.3 Moving data using EXPORT, IMPORT, LOAD, and db2move	297
6.3.1 Export data to files from database tables and views	298

6.3.2	Import data from files into database tables or views	301
6.3.3	Load data from files into database tables	305
6.3.4	Using db2move utility	330
6.4	Task Center, Scheduler, and DB2 Tools Catalog	334
6.4.1	DB2 Administration Server and Tools Catalog Database	334
6.4.2	Task Center and Scheduler	335
Chapter 7.	Monitoring and troubleshooting DB2	347
7.1	Health monitor and Health Center.	348
7.1.1	Health indicator and health monitor	348
7.1.2	Monitoring with the Health Center.	349
7.2	Memory Visualizer and Memory Tracker.	374
7.3	db2top	379
7.3.1	db2top installation	379
7.3.2	Monitoring with db2top in interactive mode.	380
7.3.3	Running db2top in the background mode	383
7.4	Log files for troubleshooting	389
7.4.1	DB2 administration notification log	389
7.4.2	DB2 diagnostic log (db2diag.log)	391
7.4.3	Operating system log	393
7.5	DB2 Tools for troubleshooting.	393
7.5.1	db2diag	393
7.5.2	db2pd	397
7.5.3	DB2 snapshot command.	401
7.5.4	db2ls	404
7.5.5	db2support	405
7.6	Linux system monitoring tools.	406
7.6.1	top	406
7.6.2	vmstat	409
7.6.3	iostat	410
7.6.4	sar	412
7.6.5	Other system monitoring tools	414
Chapter 8.	Application development	417
8.1	Application configuration	418
8.2	DB2 application objects.	420
8.2.1	Triggers.	420
8.2.2	Use defined functions	421
8.2.3	Stored procedures.	423
8.3	Programming languages	425
8.3.1	Perl	426
8.3.2	PHP.	430
8.3.3	C/C++	432

8.3.4 Java	440
8.4 pureXML	446
8.4.1 Storage model	446
8.4.2 Using XML data	447
8.5 Application development tools	453
8.5.1 Explain facility	454
8.5.2 Design advisor	457
8.5.3 Application development with the Data Studio	460
Appendix A. Issuing commands to multiple database partitions	481
db2_all and rah	482
db2_ps	485
db2_call_stack	485
Appendix B. DB2 Tools Catalog creation	487
DB2 tools catalog creation	488
Related publications	495
IBM Redbooks	495
Other publications	495
Online resources	496
How to get IBM Redbooks	498
Index	499

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:

Redbooks (logo) ®
 eServer™
 iSeries®
 i5/OS®
 pureXML™
 pSeries®
 zSeries®
 z9™
 AS/400®
 ClearCase®

DB2 Connect™
 DB2 Universal Database™
 DB2®
 Informix®
 IBM®
 OpenPower®
 OS/390®
 OS/400®
 PowerPC®
 POWER™

Rational®
 Redbooks®
 REXX™
 System i™
 System z™
 System z9®
 Tivoli®
 WebSphere®

The following terms are trademarks of other companies:

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.

DataStage, are trademarks or registered trademarks of Ascential Software Corporation in the United States, other countries, or both.

AMD, the AMD Arrow logo, and combinations thereof, are trademarks of Advanced Micro Devices, Inc.

eXchange, Java, JDBC, JDK, JVM, J2EE, Sun, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Excel, Microsoft, SQL Server, Visual Studio, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

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

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

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

Linux is one of the fastest growing server operating platforms within the past few years. DB2 has long been known for its technology leadership. This IBM Redbooks publication is an informative guide that describes how to effectively integrate DB2 for Linux, UNIX, and Windows (LUW) with SUSE and Red Hat Linux operating systems. This book provides both introductory and detailed information on installing, configuring, managing, and monitoring DB2 in a Linux environment.

We describe the DB2 product family and features for Linux, and provide step-by-step instructions for a single as well as for multiple partitions DB2 system installation and configuration. We cover how to migrate single and multiple partition DB2 to DB2 Version 9.5, and discuss, in detail, DB2 database administration in a Linux environment, procedures and tools for database backup and recovery, online maintenance, and system monitoring. We cover DB2 integrated tools and their features and use.

We discuss aspects of DB2 application development in the Linux environment, and provide general tips about building and running DB2 applications on Linux, and the use of DB2 application development tools.

The team that wrote this book

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

Whei-Jen Chen is a Project Leader at the International Technical Support Organization, San Jose Center. She has extensive experience in database design and modeling, DB2 system administration, and application development. Whei-Jen is an IBM Certified Solutions Expert in Database Administration and Application Development as well as an IBM Certified IT Specialist.

Christian Hideaki Kondo is a DB2 Systems Support member in Brazil. He has seven years of experience in DB2 and Informix. He holds a degree in computer science from Universidade Estadual Paulista - UNESP. His areas of expertise include DB2 Engine and BI solutions including migrations and performance and tuning services. He also is IBM Certified Advanced Database Administrator for DB2 9.

Helmut Riegler is an IT Specialist working for the World Wide Learning Applications Development in Vienna. He is the technical lead for the ongoing support of the learning applications. He has been with IBM since 1998 and has a broad experience in software development and database administration. Before he joined IBM, he developed CAD software. He holds the degree of an Engineer for Electrical Engineering.

Lifang LF Liu is a Senior IT Specialist of Software Group, IBM China. His major job responsibilities are solution design and proposal delivery, technical support at presale stage, and helping the customers finish the PoC/PoT in their projects. Lifang focuses on supporting DB2 OLTP business recently, especially, focuses on large ISV/OEM customers' support and DB2/SAP selling. Prior to this position, he was a DB2 worldwide L2 support engineer working in Technical Support Center, IBM China. He is IBM Certified Advanced Database Administrator of DB2. His areas of expertise include DB2 troubleshooting, database performance management, and database maintenance.

Veiko Spranger is an Advisory IT Specialist in it' Services and Solutions GmbH, an IBM subsidiary. He has 14 years of experience in DB2 focus on DB2 Distributed and Business Intelligence. Veiko is a member of the Central Region Frontoffice support team and the EMEA BI VFE (virtuell frontend) and an IBM Certified Advanced Database Administrator. He is an SAP Support Backoffice Specialist for DB2 for LUW. His primary focus is on DB2 backup, restore, and recover.

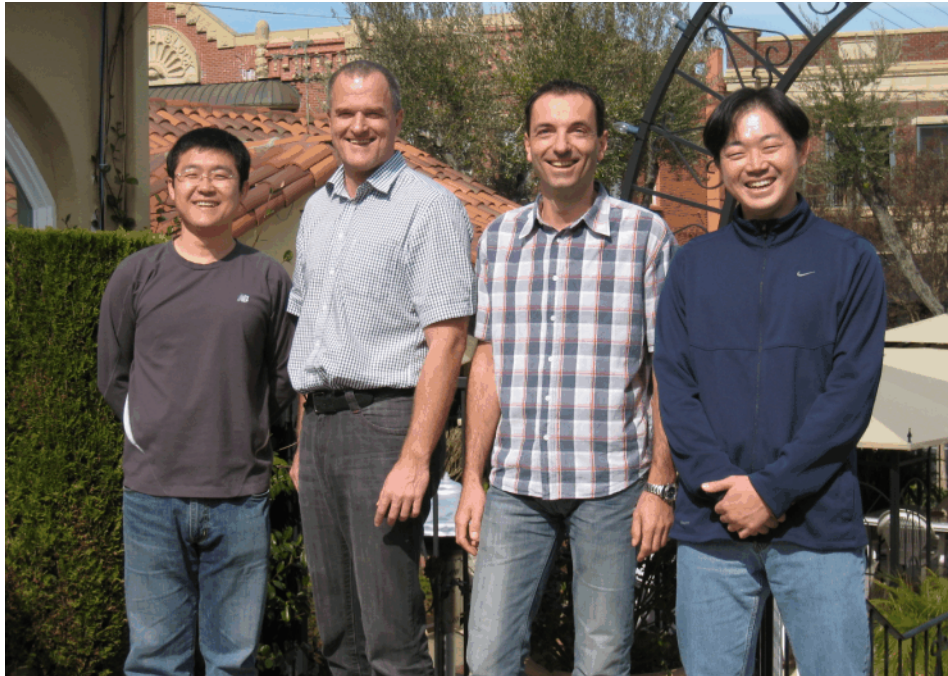


Figure 0-1 Left to right: LiFang, Helmut, Veiko, and Christian

Acknowledgements

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

Boris Bialek
Ian Hakes
Jeff Shantz
Martin Schlegel
Rav Ahuja
Grant Hutchison
Yvonne Chan
Amy Tang
Budi Surjanto
Paul Zikopoulos

IBM Toronto Laboratory, Canada

George Lapis
Gary Lazzotti
Tom Cheung
IBM Silicon Laboratory, USA

Emma Jacobs, Sangam Racherla, Deanna Polm
International Technical Support Organization

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

- Authors of the first edition, *Up and Running with DB2 on Linux*, published in March 2003, were:

Rav Ahuja
Louisa Ford
Paul Jud
Kang Yong Ying

Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will 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 will 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 books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks® in one of the following ways:

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

ibm.com/redbooks

- Send your comments in an e-mail to:

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

Summary of changes

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

Summary of Changes
for SG24-6899-01
for Up and Running with DB2 on Linux
as created or updated on April 17, 2008.

April 2008, First Edition

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

New information

- ▶ DB2 9.5 features and functions

Changed information

- ▶ DB2 installation
- ▶ Post installation tasks
- ▶ Migration
- ▶ Database administration
- ▶ Monitoring
- ▶ Application development



Introduction

With a wide spectrum of offerings and capabilities, DB2 for Linux allows you to access, manage, and analyze all forms of information across the enterprise. DB2 gives you a robust, easy-to-manage database that offers high performance, complementing the stability and reliability of Linux. Industry leading parallel technologies in DB2 make it one of the most scalable and powerful databases in production today, and when combined with Linux clusters, it allows you to manage mission-critical data at a cost lower than other enterprise class databases.

DB2 for Linux test drive can be downloaded from the Web site <http://www.ibm.com/software/data/db2/9/download.html>, <http://www.ibm.com/software/data/db2/9/edition-express-c.html> and visit <http://ibm.com/db2/linux> for further information.

This chapter introduces DB2 Version 9.5 and its offerings for Linux, and provides an overview of the DB2 environment and its parallelism with Linux. Along with an overview, the following topics are covered:

- ▶ DB2 for Linux features and offerings
Covers features, supported platforms, and products and packages.
- ▶ DB2 environment
Covers deployment topologies and db2 database objects.
- ▶ Parallelism with DB2
Covers SMP environments, database clusters, and partitioned databases.

1.1 Overview

It was not too long ago that Linux servers were being used primarily in the academic and scientific domain. In just a few short years, Linux has earned the designation of being one of the fastest growing server operating platforms and is becoming increasingly pervasive in the enterprise. Its openness, flexibility, and ability to lower cost of ownership are just some of the factors that have contributed to this operating system's phenomenal success in the commercial arena of e-business systems. Once relegated to running infrastructure tasks like file and Web serving, Linux has now found its way into the enterprise datacenter. Many companies have moved beyond the initial phases of experimentation and testing and are now reaping the benefits of deploying mission critical applications and databases with Linux and DB2.

IBM DB2 has long been known for its technology leadership. Therefore, it was not surprising when IBM took the lead in bringing DB2's proven performance, scalability, and ease of use features to Linux. Over the years DB2 has kept up its lead by releasing the first database for clustered environments on Linux, showcasing the first commercial database for Intel and AMD powered 64-bit platforms, and continually being first to announce industry leading benchmarks on Linux. IBM's commitment to Linux is further reflected through its ongoing efforts to exploit and enhance the Linux kernel for database workloads.

IBM is the market leader on Linux for relational database systems. The reasons why major companies and governments around the globe choose to deploy DB2 for Linux in the enterprise setting are quite simple. DB2's rich set of features have been running on Linux almost four years and during this time while the Linux kernel matured through the efforts of thousands of programmers and volunteers, the IBM teams were busy further hardening the kernel and DB2 on Linux for enterprise workloads. Today DB2 is the most versatile and powerful database on Linux and capable of effectively handling terabytes of data in both decision support and transactional environments. The combination of DB2 and Linux is a robust database platform for a variety of solutions and vertical applications, including:

- ▶ Back-end for Web and application servers
- ▶ Business Intelligence and Data Warehousing
- ▶ Transactional enterprise systems
- ▶ Enterprise applications like ERP, CRM, SCM
- ▶ Information Integration and Content Management
- ▶ Gateway to mainframe and host data
- ▶ Retail, financial, public sector and manufacturing applications
- ▶ Life sciences and bio-informatics solutions
- ▶ Store for spatial and geographical systems
- ▶ High Performance Computing applications including:

- Financial modeling
- Oil and gas exploration
- Research and scientific

Despite implementing a wide array of solutions in different industries, customers of DB2 for Linux generally talk about a few common themes regarding the benefits they derive. Foremost among them is the exceptional value that DB2 for Linux delivers. DB2 is renowned for critical self-tuning and self-healing features. The Self-tuning Memory Manager (STMM), automatic storage, and others autonomic maintenance features make DB2 easy to use and maintain while spending less time managing it. IBM's alliances with all the major Linux distributors and the ability to get 24x7 support for both DB2 and Linux directly from IBM provides added piece of mind.

DB2 Version 9.5 extends its innovative abilities as a hybrid data server and cost savings of deep compression to enable rapid use of data, staying ahead of threats, and extracting the full value of XML data in operational processes.

The following sections provide an overview of DB2's features and product offerings available on Linux as well as the different architectures for deploying the product in the enterprise.

1.2 DB2 for Linux features and offerings

This section provides a description of DB2 Version 9.5 features, supported platforms, and product offerings.

1.2.1 Features

DB2 is an open-standards, multi-platform, relational database system that is strong enough to meet the demands of large corporations and flexible enough to serve medium-sized and small businesses. Its features include:

- ▶ Exploitation of SMP and cluster based parallelism
- ▶ Federated support for a variety of data sources
- ▶ Advanced built-in OLAP and business intelligence functions
- ▶ Ability to manage multiple data types including Binary Large Objects
- ▶ Built on open industry standards: SQL, DRDA, CLI, ODBC, JDBC, and more

- ▶ Leader in query optimization technology and performance
- ▶ Stored procedures, UDFs, data encryption and globalization
- ▶ Fully Web-enabled and certified for leading enterprise applications
- ▶ High Availability features and Failover support, HADR, TSA, and more
- ▶ J2EE certified and Java application interfaces for JDBC and SQLJ
- ▶ Support for open source interfaces like PHP, Python and Perl
- ▶ Native XML support
- ▶ A graphical, integrated toolset for managing local and remote databases
- ▶ Productivity tools for visual development and migration
- ▶ Support for the InfiniBand I/O high-speed interconnect
- ▶ Exploitation of the latest Linux Kernel capabilities
- ▶ Innovative self-managing technologies to reduce DBA intervention
- ▶ Tools and wizards for monitoring and performance tuning
- ▶ Simplified management of large scale databases on clusters
- ▶ Federated Web Services for Information Integration
- ▶ Connection Concentrator for high user scalability
- ▶ Dynamic configuration (without the need for restarting database)
- ▶ Online tools and utilities like REORG, LOAD, storage and memory management
- ▶ Multidimensional clustering for improved performance of complex queries

DB2 has been commercially available on Linux since Version 6. DB2 for Linux, UNIX, and Windows (LUW) continuously offers industry leading performance, scale, and reliability database features and functions. XML is supported natively in DB2 9.1. DB2 9.5 extends its innovative abilities as a hybrid data server and cost savings of deep compression to enable rapid use of data, staying ahead of threats, and extracting the full value of XML data in operational processes. The new and enhanced DB2 9.5 features include:

- ▶ **pureXML:** Introduced in DB2 9.1, pureXML provides the capability to store XML data natively as a column in a DB2 database. This seamless integration of XML with relational data speeds application development, improves search performance with highly optimized XML indexes, and is flexible because both SQL and XQuery can be used to query XML data. In DB2 9.5, enhancements including non-unicode XML database support are added.
- ▶ **Data compression enhancements:** Data compression dictionaries can be automatically created during data population operations on tables for which you have defined the COMPRESS attribute, that means DB2 can automatically compress the data without DBA intervention.
- ▶ **Database backup enhancements:** DB2 9.5 can automatically delete backup images, load copy images, and old log files that are no longer needed for recovery. There are new options for automated backup such as compression, incremental and delta backups, and log files in backup. In addition, rolling forward to minimum recovery time is possible now with new TO END OF BACKUP parameter.

- ▶ **Workload Manager:** The workload management features help you to identify, manage, and monitor data server workloads. These features provide the first workload management solution truly integrated into the DB2 data server. Support for identity assertion enables you to provide workload management for individual users or groups in a multi-tier application environment. By associating workload definitions to service classes, each unique workload can be prioritized using either a predictive or reactive model. This allows businesses to align their business goals with their IT applications.
- ▶ **Multithreaded architecture extensions:** DB2 uses a multithreaded architecture which improves performance and simplifies configuration and optimization. Also, simplifications to memory management eliminate most agent-level configuration parameters and automate the rest.
- ▶ **Deployment improvements:** Deployment improvements simplify the process of installing and maintaining your DB2 data server.
 - Two previously manual steps required after applying fix packs, the running of the db2iupdt and dasupdt commands, are now automated. In addition, binding occurs automatically at the first connection
 - Non-root users can perform installing, applying or rolling back fix packs, configuring instances, adding new features, and uninstalling.
- ▶ **Easier management of partitioned database systems:**
 - There is a single view of all database configuration elements across multiple partitions. With this new functionality, you can update or reset a database configuration across all database partitions by issuing a single SQL statement or a single administration command from any partition on which the database resides.
 - The BACKUP DATABASE command can now back up all partitions of a multi-partition database at once with the same timestamp.
- ▶ **Automatic storage management enhancements:** Automatic storage automatically grows the size of your database across disk and file systems. It eliminates the need to manage storage containers while taking advantage of the performance and flexibility of database-managed space. Starting DB2 9.1, this feature is enabled by default. In DB2 9.5 further enhancements are provided including free unused space at the end of a table space, set the maximum size for automatic storage database containers, and support for multi-partitioned environment.
- ▶ **Additional automatic configuration parameters:** There are more tuning parameters that the data server automatically handles, without requiring you to stop and restart your instance or database.
- ▶ **Automated maintenance improvements:** You can use new DB2 stored procedures to configure and collect information about automated maintenance for the following areas:

- Maintenance windows
- Automatic backups
- Automatic table and index reorganizations
- Automatic table RUNSTATS operations

The DB2 stored procedures for collect information are

- SYSPROC.AUTOMAINT_GET_POLICY
- SYSPROC.AUTOMAINT_GET_POLICYFILE

To configure automated maintenance use

- SYSPROC.AUTOMAINT_SET_POLICY
- SYSPROC.AUTOMAINT_SET_POLICYFILE

- ▶ **Trusted context:** A Trusted Context is a database object that describes a trust relationship between the database and an external entity such as a middle-tier application server. Trusted context provides greater control while using restricted, sensitive privileges, and allows middle-tier servers or applications to assert the identity of the end-user to the database server.
- ▶ **Self-tuning memory allocation:** Self-tuning memory provides a configuration that is dynamic and responsive to significant changes in workload characteristics. Dynamic fast communication manager (FCM) buffers and new configuration parameters that can be tuned automatically by the DB2 database manager.
- ▶ **Label-based access control (LBAC):** This feature increases the control you have over who can access your data. LBAC lets you decide exactly who has write access and who has read access to individual rows and individual columns.
- ▶ **Automatic statistics:** DB2 automatically runs the RUNSTATS utility in the background to ensure that the correct statistics are collected and maintained by default when you create databases.
- ▶ **Product licenses:** Easier management of product licenses using the License Center and the db2licm command. Licensing changes for the DB2 Runtime Client allows you to freely distribute it.
- ▶ **Multiple DB2 versions:** The ability to install multiple DB2 versions and fix packs on the same computer.
- ▶ **Administrative SQL routines and views:** The administrative routines and views provide a primary, easy-to-use programmatic interface to administer the DB2 database product through SQL. DB2 9.5 provide more views.

1.2.2 Supported platforms

IBM is committed to Linux and supports it on a variety of hardware platforms.

Nowadays DB2 products are supported on the following hardware:

- ▶ x86 (Intel® Pentium®, Intel Xeon®, and AMD™) 32-bit Intel and AMD processors
- ▶ x64 (64-bit AMD64 and Intel EM64T processors)
- ▶ POWER™ (IBM eServer™ OpenPower®, System i™ or pSeries® systems that support Linux)
- ▶ eServer System z™ or System z9™

The supported operating systems for Linux include:

- ▶ Red Hat Enterprise Linux (RHEL) 4 Update 4
- ▶ Red Hat Enterprise Linux (RHEL) 5
- ▶ SUSE Linux Enterprise Server (SLES) 9 Service Pack 3
- ▶ SUSE Linux Enterprise Server (SLES) 10 Service Pack 1
- ▶ Ubuntu 7.10

For the latest information on supported Linux distributions check in:

<http://www-306.ibm.com/software/data/db2/linux/validate/>

To create and run Java applications and java-based tools IBM JDK™ is required. For convenience the IBM SDK 5 SR5 is shipped on DB2 9.5 CDs, and in most cases is automatically installed with DB2 installation. Refer to the validation Web site for the most up to date list of supported products as it is updated frequently.

Note: Non-IBM versions of the SDK for Java are supported only for building and running stand-alone Java applications. For building and running Java stored procedures and user-defined functions, only the IBM SDK for Java that is included with the DB2 Database for Linux, UNIX®, and Windows® product is supported.

1.2.3 DB2 products and packages

DB2 for Linux, UNIX, and Windows offers industry leading performance, scale, and reliability through:

- ▶ DB2 packages for the production environment
- ▶ Additional DB2 Enterprise features
- ▶ Additional DB2 Workgroup and Express features
- ▶ Products for accessing legacy and host data

DB2 packages for the production environment

DB2 for Linux, UNIX, and Windows provides different packages for users to select, based on their business need. This section introduces the various DB2 packages.

DB2 Express-C Edition (DB2 Express-C 9.5) is a fully functional edition of the DB2 data server and is available for download and deployment at no charge. DB2 Express-C helps developers, midmarket partners, and multibranch companies accelerate their time to value and reduce total cost of ownership. DB2 Express-C 9.5 includes pureXML and the autonomic features available in the larger enterprise editions. To upgrade to the other editions of DB2 9.5, you simply install the license certificate, with no changes to your application code. The no-charge edition of DB2 Express-C can utilize up to two processor cores and 2 GB of memory, while a licensed support option can use four cores and 4 GB of memory.

DB2 Express Edition (DB2 Express 9.5) is a full-function DB2 data server, which provides very attractive entry-level pricing for the small and medium business (SMB) market. DB2 Express 9.5 includes the same autonomic manageability features of the more scalable editions. You never have to change your application code to upgrade — simply install the license certificate. DB2 Express 9.5 can be deployed on systems with up to two processors. The DB2 data server cannot use more than 4 GB of memory.

DB2 Workgroup Server Edition (DB2 Workgroup 9.5) is the data server of choice for deployment in a departmental, workgroup, or medium-sized business environment. DB2 Workgroup 9.5 can be deployed on systems with up to four processors and 16 GB of memory.

DB2 Enterprise Server Edition (DB2 Enterprise 9.5) is designed to meet the data server needs of mid-size to large-size businesses. It can be deployed on servers of any size, from one processor to hundreds of processors. DB2 Enterprise 9.5 is an ideal foundation for building on demand enterprise-wide solutions such as high-performing 24 x 7 available high-volume transaction processing business solutions or Web-based solutions. It is the data server backend of choice for industry-leading ISVs building enterprise solutions such as business intelligence, content management, e-commerce, enterprise resource planning, customer relationship management, or supply chain management. Additionally, DB2 Enterprise 9.5 offers connectivity, compatibility, and integration with other enterprise DB2 and Informix® data sources. DB2 Enterprise 9.5 includes table partitioning, HADR, online reorganization, materialized query tables (MQTs), multidimensional clustering (MDC), query parallelism, connection concentrator, the governor, and Tivoli System Automation for Multiplatforms.

DB2 Database Partitioning Feature (DPF): The Database Partitioning Feature is available as part of the DB2 warehouse products. This feature allows DB2 Enterprise 9.5 customers to partition a database within a single system or across a cluster of systems. The DPF capability provides the customer with multiple benefits including scalability to support very large databases or complex workloads and increased parallelism for administration tasks.

Products for accessing legacy and host data

With the following DB2 products, you can extend your enterprise system to access the legacy system.

DB2 Connect™ Personal Edition: DB2 Connect Personal Edition provides the API drivers and connectivity infrastructure to enable direct connectivity from Windows and Linux desktop applications to System z and System i database servers. This product is specifically designed and licensed for enabling two-tier client/server applications running on individual workstations and, as such, is not appropriate for use on servers.

DB2 Connect Enterprise Edition: DB2 Connect Enterprise Edition is a combination of DB2 Connect server and Data Server Client software designed to address the needs of organizations that require robust connectivity from a variety of desktop systems to System z and System i database servers. Data Server Client software is deployed on desktop systems and provides API drivers that connect client/server applications running on these desktop systems to a DB2 Connect server. Designed to provide connectivity for client/server applications in large-scale, demanding environments, DB2 Connect server provides connection pooling and connection concentrator functions to maximize application availability, while minimizing System z and System i resources.

DB2 Connect Application Server Edition: DB2 Connect Application Server Edition is identical to the DB2 Connect Enterprise Server in its technology. Just like the DB2 Connect Enterprise Edition, it is designed for large-scale, demanding environments. However, its licensing terms and conditions are meant to address specific needs of multitier client/server applications as well as applications that utilize Web technologies. DB2 Connect Application Server Edition license charges are based on the size or the number of processors available to the application servers where the application is running.

DB2 Connect Unlimited Edition for zSeries: DB2 Connect Unlimited Edition is ideal for organizations with extensive usage of DB2 Connect, especially where multiple applications are involved. This product provides program code of the DB2 Connect Personal Edition as well as program code identical to the DB2 Connect Application Server Edition for unlimited deployment throughout an organization.

The license fees for DB2 Connect Unlimited Edition for zSeries are based on the size of the DB2 System z server (measured in MSUs) and are not affected by either the number of processors available to the DB2 Connect servers or the number of processors available to application servers where the application is running.

DB2 Connect Unlimited Edition for iSeries: This product has the same specification as *DB2 Connect Unlimited Edition for zSeries* except that the license fees for DB2 Connect Unlimited Edition for iSeries are based on the number of processors allocated to the data source running on i5/OS® or OS/400® operating systems. Unlimited users are permitted.

DB2 for pervasive platforms

The last, but not the least, DB2 offering is:

DB2 Everyplace: DB2 Everyplace is a relational database and enterprise synchronization server that enables enterprise applications and enterprise data to be extended to mobile devices such as personal digital assistants (PDAs) and smart phones. DB2 Everyplace can also be embedded into devices and appliances to increase their functionality and market appeal. The product can be used as a local independent database on a mobile device or query information on remote servers when a connection is available.

1.3 DB2 environment

This section describes the various DB2 application configurations on Linux. It also discusses the parallelism on SMP and clusters and gives a description of DB2 architecture.

1.3.1 Deployment topologies

DB2 for Linux can be used with a wide range of applications, whether they are developed in-house or pre-packaged. The applications can be deployed with DB2 using a number of configurations, which are:

- **Single-tier**

In this configuration the application and the database reside on the same system. In enterprise environments, it may be rare to see such a configuration, because remote access to a database server is typically required. Nonetheless this is quite common for developing applications that can later be deployed transparently in a multi-tier DB2 environment without any changes.

- **Client/Server or 2-tier**

The application and the database reside on separate systems. The machines where the application runs typically have a DB2 client installed, which communicates over the network to a database server. For the application, the physical location of the data is transparent. The application communicates with the DB2 client using a standard interface (for example, ODBC) and the DB2 client takes over the task of accessing the data over the network. In some cases, such as browser or Java based access, it is not necessary to have the DB2 client running on the same machine where the application executes.

DB2 provides exceptional flexibility for mixing and matching client and server platforms in a heterogeneous environment. DB2 client and server code is available for a wide variety of platforms. For example, the application can execute on a Windows based machine with a DB2 client for Windows, which can then access a DB2 database on a Linux server. Likewise, the Linux machine can act as a client and access data from UNIX servers or mainframes.

► **Multi-tier**

In a multi-tier configuration, the application, DB2 client and the data source typically reside on separate systems. Examples of such configurations include, but are not limited to, scenarios illustrated below (Table 1-1).

Table 1-1 Multi-tier configuration examples

Client	Middle-Tier	Server
Web-browser	Web Server DB2 Client	DB2 Database Server
Application Client	Application Server DB2 Client	DB2 Database Server 1 DB2 Database Server 2
Application DB2 Client	DB2 Connect Gateway	OS/390® AS/400® VM/VSE
Application DB2 Client	DB2 Server	Secondary Data Sources (for example, Mainframe DB2, Non-DB2, non-relational)

IBM recognizes that in many cases there may be a need for accessing data from a variety of distributed data sources rather than one centralized database. The data sources can be from IBM, such as DB2 or Informix, or non-IBM databases, such as Oracle®, or even non-relational data, such as files or spreadsheets. As illustrated in the last scenario in the table above, IBM offers the most comprehensive business integration solution by allowing federated access to a variety of distributed data sources.

Figure 1-1 on page 12 shows an example of the scalable 3-tier architecture.

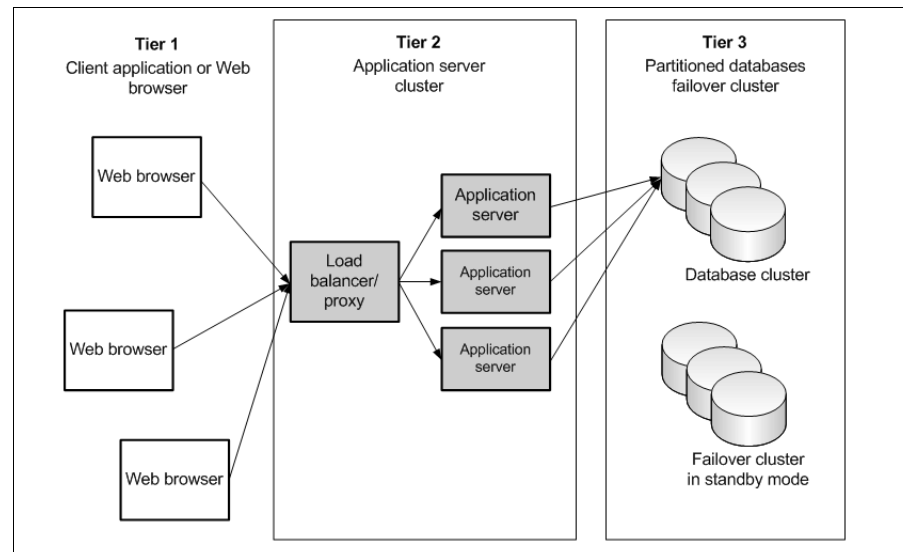


Figure 1-1 Sample scalable 3-tier architecture

Tier 1 represents clients which connect to the application through an Intranet or Internet. Tier 2 is the application hosted in an application server like WebSphere. A load balancer and a proxy component distribute the requests equally to the different application servers. If the traffic increases new application servers can be added easily in order to increase the throughput.

Tier 3 is the database tier. It consists of 2 partitioned database clusters. If the load on one database cluster increases a partition can be added to the database cluster easily to adhere to the new load. The standby cluster is a copy of the running cluster. In case of an outage of the primary database cluster the application server requests will be transparently redirected to the standby cluster.

In the Chapter 8, “Application development” on page 417, you can see a more detailed how applications can be configured to implement the different tiers.

► OLTP versus DSS

You can distinguish between *Online Transaction Processing* (OLTP) and *Decision Support System* (DSS) systems based on the type of their workload. OLTP systems process the business related transactions. DSS based systems typically have large and long running reports. DB2 offers many features to tune both types of workload. Figure 1-2 shows an example architecture.

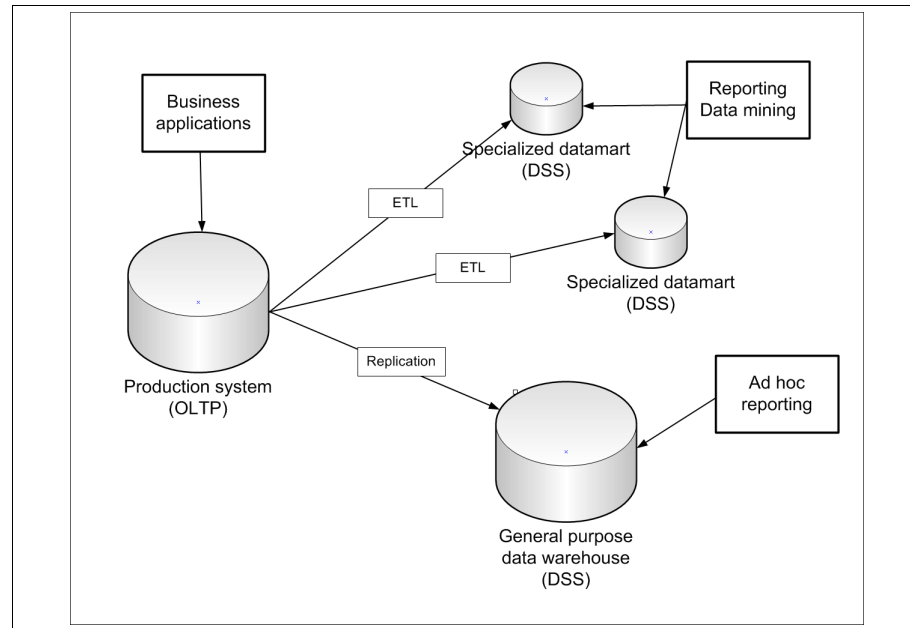


Figure 1-2 Deployment based on workload

Figure 1-2 shows an example architecture based on the workload type. The business transactions take place on the production systems (OLTP). Here you have a lot of transactions which write to the database. The type of queries and transaction are defined very well. The data model is usually highly normalized. These systems usually require a very high availability.

There is a general purpose data warehouse which receives the majority of the data from the production system on a regular basis, by replicating the data with the data propagator.

For instance by replicating the data with the data propagator. Here users can perform ad hoc reports. On such a system the transactions are not known very well because users are theoretically able to execute any query. The availability of this warehouse can be lower than the one from the production system. The data model can be normalized at the same level as the production system or can already be a little bit denormalized. It depends on the needs.

The data marts are very specialized data warehouses. They are designed to run a quite well defined set of reports. The data is mostly denormalized and transformed into a *CUBE* or into *star schemas*. Cubes and star schemas are data models designed to support the queries running on these systems. You have ETL (Export Transform Load) processes to transfer the data from the production system to the data marts on a regular base. That can be done by

tools like WebSphere DataStage™ or WebSphere QualityStage. The availability of a data mart is usually lower than for the production systems.

1.3.2 DB2 database objects

In this section, we introduce some DB2 objects and their relationship to each other.

Instances

An instance (sometimes called a database manager) is DB2 code that manages data. It controls what can be done to the data, and manages system resources assigned to it. Each instance is a complete environment. It contains all the database partitions defined for a given parallel database system. An instance has its own databases (which other instances cannot access directly), and all its database partitions share the same system directories. It also has separate security from other instances on the same machine (system), allowing for example both production and development environments to be run on the same machine under separate DB2 instances without interfering with each other.

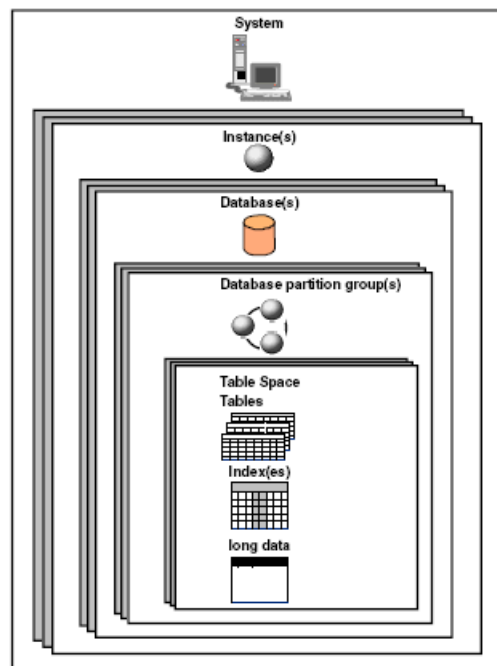


Figure 1-3 DB2 database objects

Databases

A relational database presents data as a collection of tables. A table consists of a defined number of columns and any number of rows. Each database includes a set of system catalog tables that describe the logical and physical structure of the data, a configuration file containing the parameter values allocated for the database, and a recovery log.

Database partition groups

A database partition group is a set of one or more database partitions. Before creating tables for the database, you first need to create the database partition group where the table spaces will be stored, and then create the table space where the tables will be stored. If a partition group is not specified, there is a default group where table spaces are allocated. In earlier versions of DB2, database partition groups were known as node groups. In a non-partitioned environment, all the data resides in a single partition, therefore it is not necessary to worry about partition groups.

Table spaces

A database is organized into parts called table spaces. A table space is a place to store tables. When creating a table, you can decide to have certain objects such as indexes and large object (LOB) data kept separately from the rest of the table data. A table space can also be spread over one or more physical storage devices.

Table spaces reside in database partition groups. Table space definitions and attributes are recorded in the database system catalog. Containers are assigned to table spaces. A container is an allocation of physical storage (such as a file or a device). A table space can be either system managed space (SMS), or database managed space (DMS). For an SMS table space, each container is a directory in the file space of the operating system, and the operating system's file manager controls the storage space. For a DMS table space, each container is either a fixed size pre-allocated file, or a physical device such as a disk, and the database manager controls the storage space.

Tables

A relational database presents data as a collection of tables. A table consists of data logically arranged in columns and rows. All database and table data is assigned to table spaces. The data in the table is logically related, and relationships can be defined between tables. Data can be viewed and manipulated based on mathematical principles and operations called relations. Table data is accessed through Structured Query Language (SQL), a standardized language for defining and manipulating data in a relational database. A query is used in applications or by users to retrieve data from a database. The query uses SQL to create a statement in the form of:

```
SELECT <data_name> FROM <table_name>
```

Buffer pools

A buffer pool is the amount of main memory allocated to cache table and index data pages as they are being read from disk, or being modified. The purpose of the buffer pool is to improve system performance. Data can be accessed much faster from memory than from disk; therefore, the fewer times the database manager needs to read from or write to a disk (I/O), the better the performance. (You can create more than one buffer pool, although for most situations only one is sufficient.) The configuration of the buffer pool is the single most important tuning area, because you can reduce the delay caused by slow I/O.

1.4 Parallelism with DB2

A system with a single processor and disk has limitations for the amount of data, users, and applications that it can handle. Parallel processing, which is key for enterprise workloads, involves spreading the load across several processors and disks, allows for large volumes of data and high transaction rates to be processed. In many environments, the data volumes are growing at a phenomenal rate, therefore a database management system needs to be able to easily scale to increased loads with the addition of more disks and CPUs.

DB2's parallel technology enables highly scalable performance on large databases by breaking the processing into separate execution components that can be run concurrently on multiple processors. Elapsed times for queries can be dramatically reduced by processing the individual queries in parallel. DB2 supports a scalable growth path to easily add more processing power to an existing system by either "scaling up" (SMP) or "scaling out" (MPP) or both.

1.4.1 SMP environments

In the past, complex enterprise workloads have typically run on high-end SMP (symmetric multi-processor) machines. A large number of processors running within the same machine and connected with a high bandwidth bus can deliver excellent performance. Because DB2 for Linux shares much of its code with DB2 running on high-end UNIX systems, it inherits the ability to exploit SMP architectures. DB2's ability to run on SMP systems is only restricted by today's Linux SMP scalability. DB2 can take advantage of multiple processors easily without setting up a clustered system. This is called *intra-parallelism*.

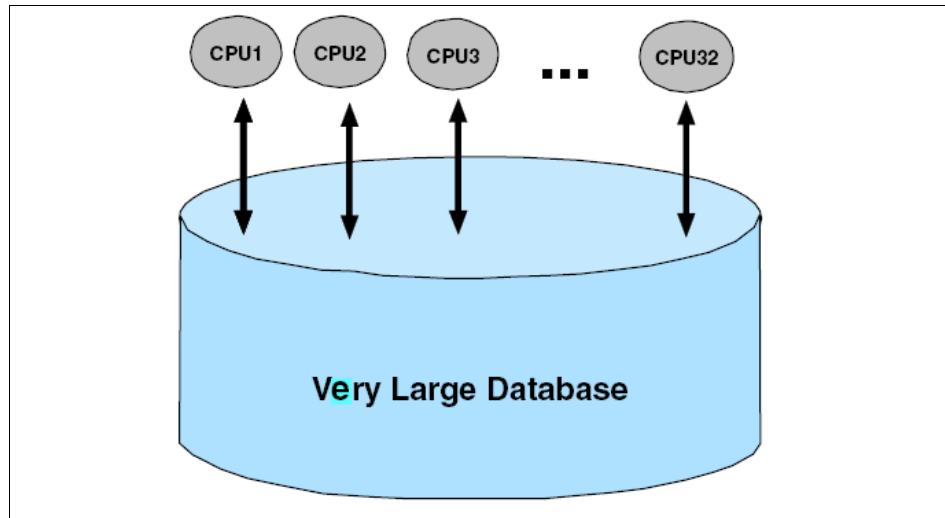


Figure 1-4 Database on a large SMP system

1.4.2 Database clusters

The driving force behind using Linux clusters is that by distributing the load over several low cost servers running an open-source operating system, a larger task can be accomplished faster, more reliably, and much more economically. And if the load increases, the cluster can be extended for managing the additional demand without compromising on performance. DB2 was the first commercial database on Linux to provide built-in capabilities for clusters. That is, DB2 can be deployed across a Linux cluster right out of the box, without the need for additional clustering software. DB2 clusters on Linux are ideal for running demanding transactional applications and warehouses that involve large volumes of data while providing the following benefits:

- ▶ Faster processing time
- ▶ Very large databases
- ▶ Excellent scalability
- ▶ Increased availability
- ▶ Reduced cost

Combining the clustered approach with DB2's support for 64-bit Linux environments provides additional performance and scalability advantages.

For the clustering feature of DB2 you have to setup a partitioned environment. This is called *inter-parallelism*.

Database clustering architectures

There are two primary database clustering architectures: shared-disk and shared-nothing. Shared-disk is used by Oracle RAC. DB2 for Linux employs shared-nothing.

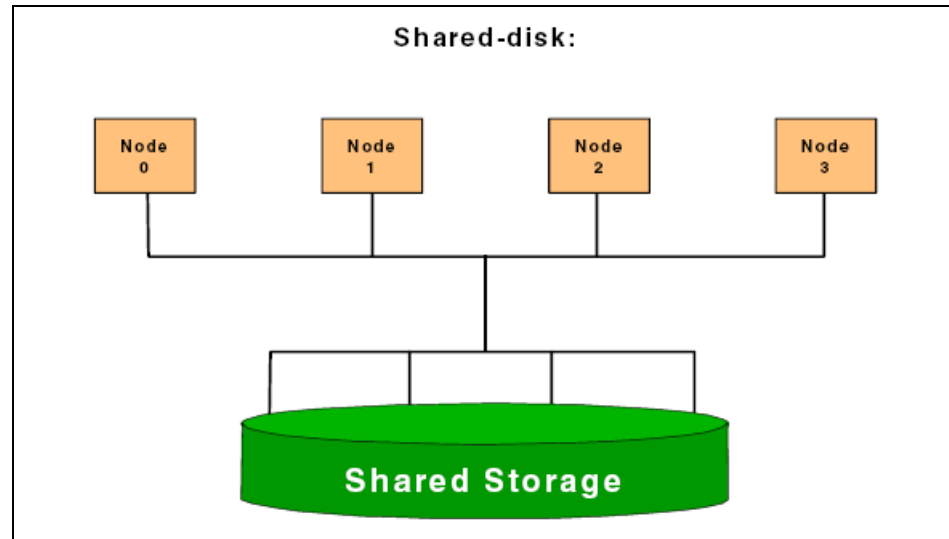


Figure 1-5 Shared disk architecture

Shared-disk

In a shared-disk environment, each database node has its own processors but shares the disks with other nodes. Therefore all the processors can access all the disks in the cluster. This introduces additional overhead for coordinating resources and locks between nodes. For example, if Node 1 wants to access data on a certain disk that has been locked for update by another node in the cluster, Node 1 must wait for other nodes to complete their operations. While this works well when there are few nodes in the cluster (for example, mainframe environments), the overhead for distributed lock management and cache coherency issues can severely limit scalability and introduce performance degradation for four or more nodes, making it impractical to exploit economies of large clusters on Linux. Furthermore, this approach involves specialized hardware and software for shared disk and cache management, making it much more expensive than shared-nothing.

Shared-nothing

As the name implies, partitions (nodes) in a shared-nothing environment do not share processors, memory, or disks with partitions on other machines. Each partition acts on its own subset of data. Because each partition has its own private resources, this approach does not involve any resource contention with

other servers, lending itself to virtually unlimited scalability. This is one reason DB2 for Linux can support up to 1000 partitions. And because there is no coordination overhead for accessing resources, additional machines can easily be added to the cluster with linearly scalable performance, which implies, if doubling of the data volume is matched by the doubling of the cluster resources, the high performance of the database will be maintained at the same level. If one partition in a cluster fails, its resources can dynamically be transferred to another machine, ensuring high availability. Another benefit of the shared nothing approach used by DB2 for Linux is that it does not require specialized hardware, making the solution much simpler, less expensive, and suitable for Linux based “commodity” hardware.

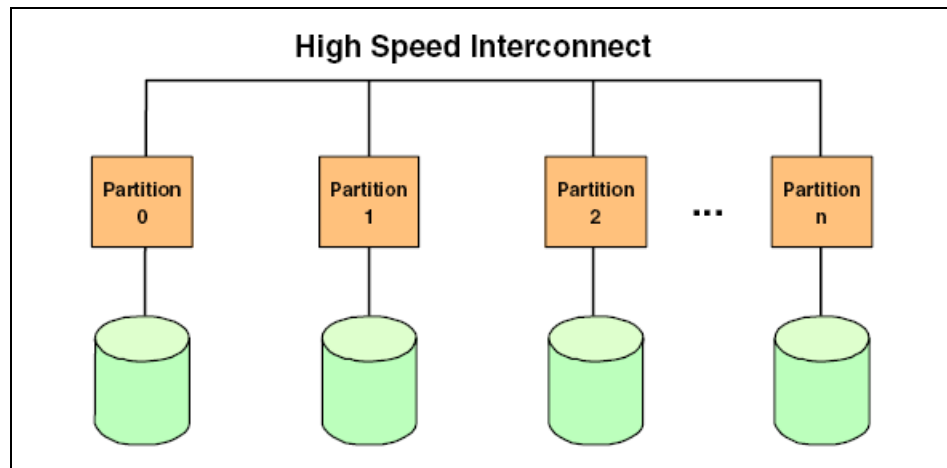


Figure 1-6 Shared-nothing architecture

1.4.3 Partitioned database

DB2 exploits the power of Linux clusters by employing database partitioning. In a partitioned environment, a database is distributed across multiple partitions, usually residing on different machines. Each partition is responsible for a portion of a database's total data. A database partition is sometimes also called a node or a database node. Because data is divided across database partitions, you can use the power of multiple processors on multiple physical nodes to satisfy requests for information. Data retrieval and update requests are decomposed automatically into sub-requests, and executed in parallel among the applicable database partitions.

As an illustration of the power of processing in a partitioned database system, assume that you have 100,000,000 records that you want to scan in a single-partition database. This scan would require that a single database

manager search 100,000,000 records. Now suppose that these records are spread evenly over 20 database partitions; each partition only has to scan 5,000,000 records. If each database partition server scans in parallel with the same speed, the time required to do the scan should be approximately 20 times faster than a single-partition system handling the entire task.

The fact that databases are partitioned across several database partitions is transparent to users and applications. User interaction occurs through one database partition, known as the coordinator node for that user. Any database partition can be used as a coordinator node. The database partition that a client or application connects to becomes the coordinator node. You should consider spreading out users across database partitions servers to distribute the coordinator function.

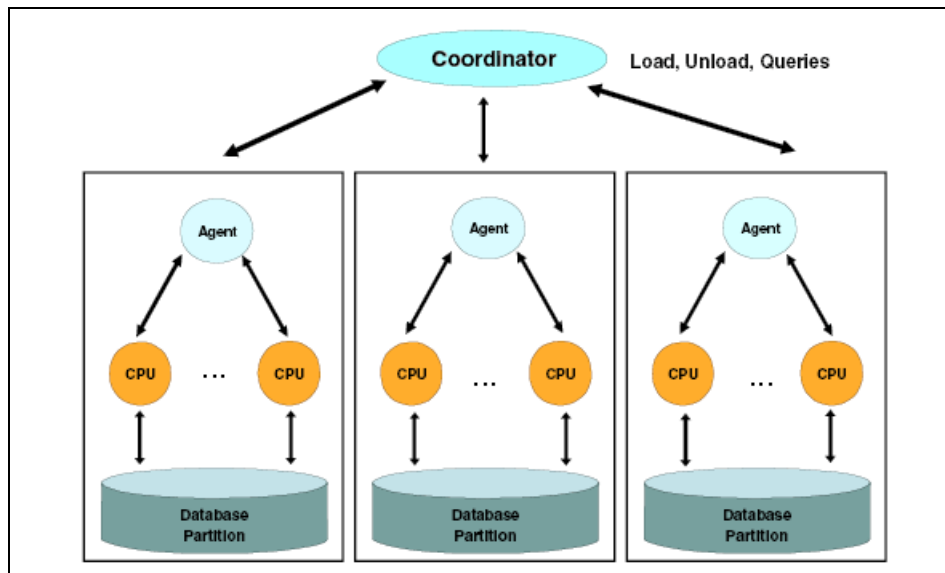


Figure 1-7 DB2 MPP environment

Figure 1-8 on page 21 shows you the relationship of the database objects in a partitioned environment.

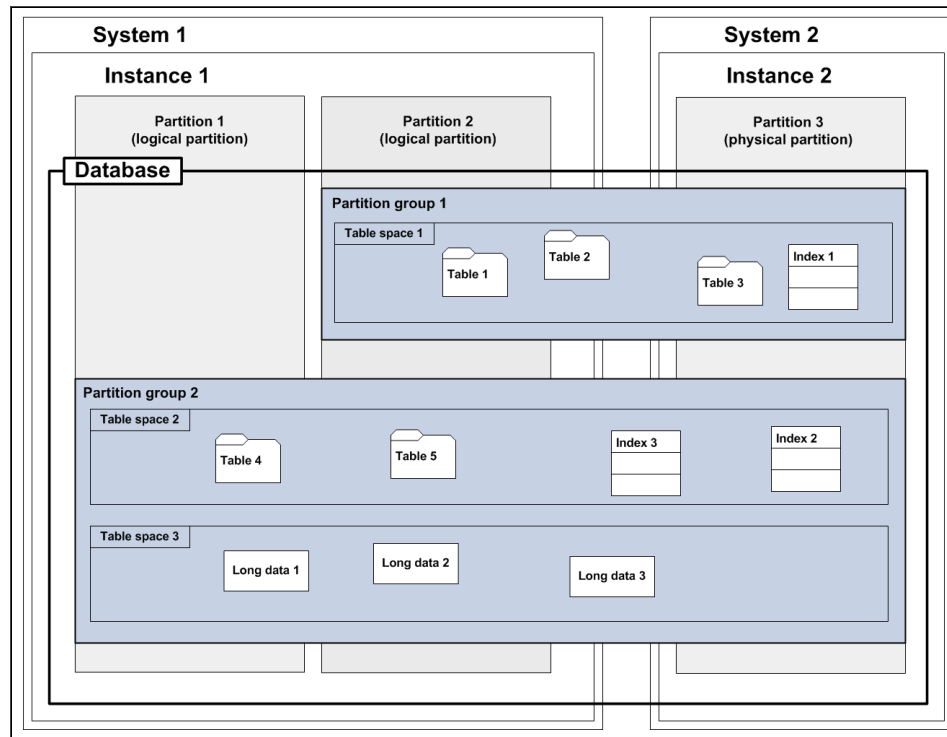


Figure 1-8 DB2 database objects in a partitioned environment

You can see that in a partitioned environment the partition groups are expanded transparently across the partitions. We have got 2 logical partitions on the first system and one physical/logical partition on the second system. Table spaces are assigned to one partition group. After you add a system or a logical/physical partition you just have to extend the partition groups you want to the new partition. In the above example we have one partition group spanning partition 2 and partition 3 and we have one partition group which spans all 3 partitions.

The table data is distributed among the partitions using an updatable partitioning map and a hashing algorithm, which determine the placement and retrieval of each row of data. For example, if a row is being added to a table, the coordinator node checks a partitioning map, which specifies the database partition where the row is to be stored. The row is only sent to that database partition server, with the result that only the interested database partition servers take part in the insert. This keeps communications and coordination overhead between nodes as low as possible. See Figure 1-9.

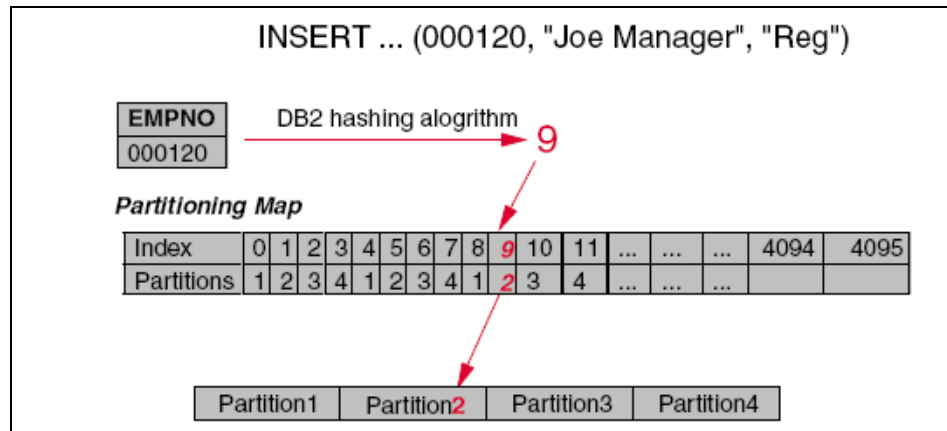


Figure 1-9 Intelligent data distribution across DB2 partitions

The data, while physically split, is used and managed as a logical whole. Users can choose how to partition their data by declaring partitioning keys. Tables can be located in one or more database partitions. As a result, you can spread the workload across a partitioned database for large tables, while allowing smaller tables to be stored on one or more database partitions. Each database partition has local indexes on the data it stores, resulting in increased performance for local data access.

You are not restricted to having all tables divided across all database partitions in the database. DB2 supports partial declustering, which means that you can divide tables and their table spaces across a subset of database partitions in the system. An alternative to consider when you want tables to be positioned on each database partition, is to use materialized query tables and then replicate those tables. You can create a materialized query table containing the information that you need, and then replicate it to each node.

Partitioning on SMP clusters

In the most simplistic scenario, each physical machine in the cluster has a single database partition on it. In this type of a configuration, each partition is a physical database partition and has access to all of the resources in the machine. One partition per machine is typical for systems having one or two processors. When using SMP machines with several processors, it is possible to create more than one partition on the same system. These are called logical database partitions. A logical database partition differs from a physical partition in that it is not given control of an entire machine. Although the machine has shared resources, database partitions do not share all resources. Processors are shared but disks and memory are not. See Figure 1-10.

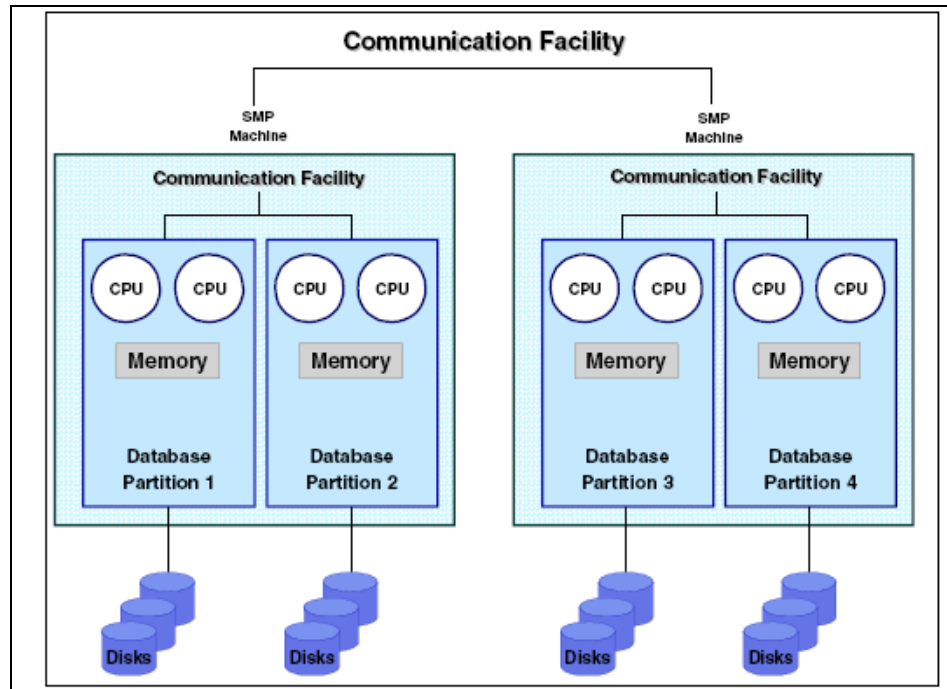


Figure 1-10 A partitioned database across a cluster of SMP machines

There are advantages to using logical partitions on high-SMP systems in both clustered and non-clustered environments. Logical database partitions can provide better scalability. Multiple database managers running on multiple logical partitions may make fuller use of available resources than a single database manager could. The ability to have two or more partitions coexist on the same machine (regardless of the number of processors) allows greater flexibility in designing high availability configurations and failover strategies. Upon machine failure, a database partition can be automatically moved and restarted on a second machine that already contains another partition of the same database.

2



Installation

This chapter guides you through the process of installing DB2 on Linux systems. It covers both single and multiple partition environments and provides instructions for three installation methods: DB2 Setup, db2_install, and response file installation.

The following topics are discussed:

- ▶ Basic requirements
- ▶ Installation considerations and planning
- ▶ Multi-partition installation considerations
- ▶ User and group setup
- ▶ Installing DB2

2.1 Basic requirements

Prior to installing DB2, it is important that you have the required hardware and software. You may be required to configure communications and install additional software packages in order for DB2 to run successfully.

In this section we cover the following installation prerequisites:

- ▶ Hardware supported by DB2
- ▶ Linux distributions supported by DB2
- ▶ Required disk space
- ▶ Memory requirements
- ▶ Communication requirements
- ▶ Kernel parameter values
- ▶ Additional software requirements

2.1.1 Hardware supported by DB2

DB2 products are supported on the following hardware:

- ▶ x86 (Intel Pentium, Intel Xeon, and AMD) 32-bit Intel and AMD processors
- ▶ x64 (64-bit AMD64 and Intel EM64T processors)
- ▶ POWER (IBM eServer OpenPower, System i or pSeries systems that support Linux)
- ▶ eServer System z or System z9

2.1.2 Linux distributions supported by DB2

Make sure that your Linux distribution level is supported by DB2. For the latest information about currently supported Linux distributions, kernels, and libraries, refer to the DB2 for Linux validation Web site:

<http://www.ibm.com/software/data/db2/linux/validate>

The following Linux distributions, show in Table 2-1 and Table 2-2 on page 27, have been successfully validated for use with DB2:

Table 2-1 Recommended environments for DB2 9.5

Kernel architecture	x86	x86_64	PPC64 (POWER)	s390x (zSeries ®)
Red Hat Enterprise Linux (RHEL) 5	✓	✓	✓	✓

Kernel architecture	x86	x86_64	PPC64 (POWER)	s390x (zSeries ®)
SUSE Linux Enterprise Server (SLES) 10	✓	✓	✓	✓
Ubuntu 7.10	✓	✓		

Table 2-2 Validated environments for DB2 9.5

Kernel architecture	x86	x86_64	PPC64 (POWER)	s390x (zSeries)
Asianux Server 3	✓	✓	✓	
Red Hat Enterprise Linux (RHEL) 4	✓	✓		✓
SUSE Linux Enterprise Server (SLES) 9	✓	✓		✓
Ubuntu 7.04	✓	✓		

2.1.3 Required disk spaces

You must take into account the following disk requirements while configuring your system. The DB2 Setup wizard provides dynamic size estimates based on the components selected during a typical, compact, or custom installation, as shown in Table 2-3 and Table 2-4 on page 28:

Table 2-3 Required disk space - non-root installation

Install Type	Description	Required disk space
Typical	The typical setup includes basic database server function, database administration tools, and most product features and functionality.	400 to 490 MB
Compact	Basic DB2 features and functionality will be installed, and minimal configuration will be performed.	390 to 470 MB
Custom	This option allows you to select the features that you want to install and specify your configuration preferences.	390 to 680 MB

Table 2-4 Required disk space - root installation

Install Type	Description	Required disk space
Typical	The typical setup includes basic database server function, database administration tools, and most product features and functionality.	630 to 760 MB
Compact	Basic DB2 features and functionality will be installed, and minimal configuration will be performed.	580 to 700 MB
Custom	This option allows you to select the features that you want to install and specify your configuration preferences.	580 to 1020 MB

You should also allocate disk space for required software, communication products, DB2 documentation, and databases. In DB2 Version 9.5, DB2 Information Center and PDF documentation are now provided on separate DVDs. Installing DB2 Information Center in English requires 131 MB of disk space. Additional space is required for additional languages.

We recommend that you allocate two GB of free space in the /tmp directory.

2.1.4 Memory requirements

We recommend that you allocate a minimum 256 MB of RAM for DB2 and a minimum of 512 MB of RAM for a system running just a DB2 product and the DB2 GUI tools. Additional memory should be allocated for other software and communication products. When determining memory requirements, be aware of the following:

- ▶ We recommend that your SWAP space is at least as twice as much as your RAM.
- ▶ For IBM data server client support, these memory requirements are for a base of five concurrent client connections. We recommend that you allocate an additional 16 MB of RAM per five client connections.
- ▶ Memory requirements are affected by the size and complexity of your database system, as well as by the extent of database activity and the number of clients accessing your system.

For DB2 server products, the self-tuning memory feature simplifies the task of memory configuration by automatically setting values for several memory configuration parameters. When enabled, the memory tuner dynamically distributes available memory resources among several memory consumers including sort, the package cache, the lock list and buffer pools.

2.1.5 Communication requirements

TCP/IP is required to access remote databases. Your Linux distribution provides TCP/IP connectivity if selected during install.

If your Linux machine is installed on an existing network and is required to use a static IP address, information similar to Table 2-5 should be collected from the network administrators.

Table 2-5 Required network information

Name	Example number
Host IP address	192.168.1.3
Subnet mask	255.255.255.0
Gateway	192.168.1.1
Domain name server	192.168.3.1

The above information should be specified on the setup screen during OS installation, or as a post-installation step using your distribution’s setup utility.

Also certain communication software packages are required. These are discussed in 2.1.7, “Additional software requirements” on page 32.

2.1.6 Kernel parameter values

You may be required to update some of the default kernel parameter settings for DB2 to run successfully on Linux. The default values for particular kernel parameters on Linux are not sufficient when running a DB2 database system.

Note: You must have root authority to modify kernel parameters.

With Version 8 and later, DB2 checks values of the *kernel.sem*, *kernel.msgmni*, and *kernel.shmmax* parameters automatically at DB2 start time, and changes them for you if the current values are not optimal. The db2start command does the following:

- ▶ Changes SEMMNI kernel parameter to 1024
- ▶ Changes MSGMNI kernel parameter to 1024
- ▶ Changes SHMMAX kernel parameter to 268435456 (32-bit) or 1073741824 (64-bit)

Manually updating the *kernel.shmmax*¹, *kernel.msgmni*, and *kernel.sem* parameters prior to root installations is no longer required. However, for non-root

installations, you should check these kernel parameters before installation and update them if needed.

Manually updating kernel parameters

If any of the default kernel parameter settings do not meet the requirements of your particular system, you can update them manually.

To check your current shared memory segment, semaphore array, and message queue limits, use the **ipcs -l** command. Your output should look something like this:

```

----- Shared Memory Limits -----
max number of segments = 4096           // SHMMNI
max seg size (kbytes) = 32768           // SHMMAX
max total shared memory (kbytes) = 8388608 // SHMALL
min seg size (bytes) = 1

----- Semaphore Limits -----
max number of arrays = 1024             // SEMMNI
max semaphores per array = 250          // SEMMSL
max semaphores system wide = 256000    // SEMMNS
max ops per semop call = 32            // SEMOPM
semaphore max value = 32767

----- Messages: Limits -----
max queues system wide = 1024          // MSGMNI
max size of message (bytes) = 65536    // MSGMAX
default max size of queue (bytes) = 65536 // MSGMNB

```

We recommend that you set the SHMMAX value to be equal to the amount of physical memory on your system. However, the minimum required on x86 systems is 268435456 (256 MB) and for 64-bit systems, it is 1073741824 (1 GB).

In this example, SHMALL is set to 8 GB (8388608 KB = 8 GB). If you have more physical memory than this for DB2, increase the value of this parameter to approximately 90% of your computer's physical memory.

The kernel parameter sem consists of 4 tokens, SEMMSL, SEMMNS, SEMOPM and SEMMNI. SEMMNS is the result of SEMMSL multiplied by SEMMNI. The database manager requires that the number of arrays (SEMMNI) be increased as necessary. Typically, SEMMNI should be twice the maximum number of agents expected on the system multiplied by the number of logical partitions on the database server computer plus the number of local application connections on the database server computer.

¹ For root installations, if kernel.shmmax is set to a low value, you may encounter the memory allocation error while starting the instance, then you have to update kernel.shmmax manually.

MSGMAX should be change to 64 KB (that is, 65536 bytes), and MSGMNB should be increased to 65536.

Modifying kernel parameters on SUSE and Red Hat

In this example, we explain how to update kernel parameters and set them after each reboot.

1. Log in as root.
2. Using a text editor, add the following entries to `/etc/sysctl.conf`:

```
# Example shmmax for a 64-bit system
kernel.shmmax=1073741824
# Example shmall for 90 percent of 16 GB memory. The kernel requires this
value as a number of pages
kernel.shmall=3774873
kernel.sem=250 256000 32 1024
kernel.msgmni=1024
kernel.msgmax=65536
kernel.msgmnb=65536
```

3. Load these entries into sysctl by using the following command:

```
sysctl -p
```

Now if you enter in command `ipcs -l`, you can see that the kernel parameters have been updated in sysctl. To view all sysctl settings, use the command:

```
sysctl -a
```

If you want to update kernel parameters for run-time only, use the `sysctl -w` command. For example, to change `kernel.msgmni` to 1024, enter the following command:

```
sysctl -w kernel.msgmni=1024
```

However, these settings will not remain after the next reboot unless they are saved in the `/etc/sysctl.conf` file.

4. To make the changes effective after every reboot:

- SUSE Linux: Make boot.sysctl active.

```
su - root
# chkconfig boot.sysctl
boot.sysctl off
# chkconfig boot.sysctl on
# chkconfig boot.sysctl
boot.sysctl on
#
```

- Red Hat: The `rc.sysinit` initialization script will read the `/etc/sysctl.conf` file automatically.

2.1.7 Additional software requirements

Depending on your DB2 requirements, you may be required to install additional software packages for DB2 to function properly. Make sure that the following software is installed prior to using DB2.

- ▶ IBM Software Development Kit (SDK) for Java is required to run DB2 graphical tools, and to create and run Java applications, including stored procedures and user-defined functions. If the correct version of JDK is not already installed, IBM SDK 5 Service Release 5 will be installed for you during the installation process if you use either the DB2 Setup wizard, db2_install², or a response file to install the product. IBM SDK 1.4.2 to SDK 5 is also supported.

Note: On Linux, DB2 GUI tools only run on x86 and x86_64 (AMD64/EM64T).

- ▶ One of the following browsers is required to view online help and to run First Steps (db2fs):
 - Mozilla 1.4 and up
 - Firefox 1.0 and up
- ▶ X Window System software, capable of rendering a graphical user interface. You need this if you want to use the DB2 Setup wizard (a graphical installer) to install DB2 or any DB2 graphical tools.
- ▶ Table 2-6 provides additional package requirements for SUSE Linux and Red Hat distributions for DB2 9.5. For single-partition database, only *libaio* and *compat-libstdc++* packages are required.

Table 2-6 Package requirements for Linux

Package name	RPM name	Description
libaio	SLES 10 SP1 x86_64 libaio-0.3.104-14.2.x86_64.rpm SLES 10 SP1 x86 libaio-0.3.104-14.2.i586.rpm RHEL 5 x86_64 libaio-0.3.106-3.2.x86_64.rpm RHEL 5 x86 libaio-0.3.106-3.2.i386.rpm	Contains the asynchronous library required for DB2 servers.

² The db2_install utility does install the JDK for you.

Package name	RPM name	Description
compat-libstdc++	SLES 10 SP1 x86_64 compat-libstdc++-5.0.7-22.2.x86_64.rpm SLES 10 SP1 x86 compat-libstdc++-5.0.7-22.2.i586.rpm RHEL 5 x86_64 compat-libstdc++-33-3.2.3-61.x86_64.rpm compat-libstdc++-33-3.2.3-61.i386.rpm ^a RHEL 5 x86 compat-libstdc++-33-3.2.3-61.i386.rpm	Contains libstdc++.so.5 which is required for DB2 servers and clients.
ksh ^b	SLES 10 SP1 x86_64 ksh-93r-12.28.x86_64.rpm RHEL 5 x86_64 ksh-20060214-1.4.x86_64.rpm	Korn Shell. This package is required for partitioned database environments.
openssh	SLES 10 SP1 x86_64 openssh-4.2p1-18.25.x86_64.rpm RHEL 5 x86_64 openssh-4.3p2-24.el5.x86_64.rpm openssh-clients-4.3p2-24.el5.x86_64.rpm openssh-server-4.3p2-24.el5.x86_64.rpm openssh-askpass-4.3p2-24.el5.x86_64.rpm	This package contains a set of server and client programs which allow users to run commands on (and from) remote computers via a secure shell. This package is not required if you use the default configuration of DB2 with rsh.
rsh-server	SLES 10 SP1 x86_64 rsh-server-0.17-573.2.x86_64.rpm RHEL 5 x86_64 rsh-server-0.17-37.el5.x86_64.rpm	This package contains a set of server programs which allow users to run commands on remote computers, log in to other computers, and copy files between computers (rsh, rexec, rlogin, and rcp). This package is not required if you configure DB2 to use ssh.

Package name	RPM name	Description
nfs-utils	SLES 10 SP1 x86_64 nfs-utils-1.0.7-36.21.x86_64.rpm RHEL 5 x86_64 nfs-utils-lib-1.0.8-7.2.z2.x86_64.rpm	Network File System support package. It allows access for local files to remote computers.

- a. For RHEL 5, both 32-bit and 64-bit compat-libstdc++ packages are required for 32-bit applications to run.
- b. For SLES 10 and RHEL 5, ksh is required instead of pdksh.

To check whether you have these packages installed, use the `rpm -q` command. For instance, to check whether pdksh has been installed, enter the following command:

```
rpm -qa |grep compat-libstdc++
```

To install these packages on Red Hat and SUSE, use `rpm -ivh` command. For example, to install *compat-libstdc++* on SUSE Linux Enterprise Server (SLES) 10 on x86_64, mount the SLES 10 SP 1 x86_64 DVD 1, and enter the following command:

```
rpm -ivh /mnt/dvd/suse/x86_64/compat-libstdc++-5.0.7-22.2.x86_64.rpm
```

- If your Red Hat distribution has X Window System software, you can install these packages using system administration tool *Package Manager*. For example, to install pdksh on Red Hat Enterprise Linux (RHEL) 5, do the following:
 - a. Log on as root.
 - b. Choose **Applications** → **Add/Remove Software**, then choose the appropriate packages.
- If your SUSE distribution has X Window System software, you can install these packages using YaST. For example, to install pdksh on SUSE Linux Enterprise Server (SLES) 10, do the following:
 - a. Log on as root.
 - b. From the YaST2 Control Center, select **Software** → **Install or Remove Software**, then choose the appropriate packages.

Notes:

- ▶ The RPM name may be different depending on the Linux distribution and version you are using. Check the latest Linux information for the RPM name that you require.
- ▶ If required package does not exist in Linux distribution DVD/CD, you may need download it from related Linux support Web site.

2.2 Installation considerations and planning

Now that you have verified that your system meets DB2's basic requirements, it is time to prepare your system for DB2 installation.

In this section we discuss the following topics:

- ▶ Considerations
 - Multiple DB2 copies
 - Non-root installation
 - Support Changes for 32-bit and 64-bit DB2 servers
- ▶ Installation methods: DB2 Setup, db2_install, response file installation
- ▶ Storage planning
 - File systems versus raw devices
 - File system configuration
 - DB2 log space
 - DB2 temporary table space
- ▶ User and group setup
 - User and group requirements
 - Creating users - single-partition
 - Creating users - multi-partition
 - DAS user considerations for a multiple partition environment
- ▶ Network configuration for partitioned database installation
 - Enabling ssh
 - Setting up NFS

2.2.1 Considerations

DB2 9.5 now supports multiple DB2 copies on a single machine, as well as non-root installations. There are also changes in 32-bit and 64-bit server support. These changes should be understood and taken into account before installing DB2 9.5.

Multiple DB2 copies

Beginning with DB2 Version 9, DB2 products can be installed in multiple different locations (as different DB2 copies) on a single machine. A DB2 copy refers to one or more installations of DB2 database products in a particular location on the same computer. Each DB2 Version 9 copy can be at the same or different code levels.

Only one DB2 administration server (DAS) can be created on a machine regardless of the number of DB2 copies that are installed on the machine. We recommend that you migrate your existing DAS to Version 9.5 or create a new DAS in DB2 Version 9.5.

This multiple DB2 copies feature is supported by most DB2 products, except DB2 Information Center.

Non-root installation

Prior to Version 9.5, you could install products, apply and roll back fix packs, configure instances, add features, or uninstall products only if you had root authority. Now, you can install and service a non-root DB2 on Linux platforms.

The DB2 installer automatically creates and configures a non-root instance during a non-root installation. As a non-root user, you can customize the configuration of the non-root instance during the installation. You can also use and maintain the installed DB2 product without root authority. Although non-root installations have most of the functionality of root installations, there are some differences and limitations. Table 2-7 on page 36 shows some of them.

Table 2-7 Differences and limitations between root and non-root installations

Criteria	Root installations	Non-root installations
User can select installation directory	Yes	No. DB2 products are installed under the user's home directory.
Instance creation	Can create instances during installation. This is the recommended method.	A single instance is automatically created and configured. Further instances cannot be created in non-root installations.
Number of DB2 instances allowed	Multiple	One

Criteria	Root installations	Non-root installations
Files deployed during installation	Program files only. Instances are created after installation.	Program files and instance files. The DB2 product is ready for use immediately after installation.
DB2 Administration Server (DAS) and its associated commands	Yes Available	No Not available
Control Center	Yes Available	No Not available
Operating system-based authentication	Yes	No. You can lift this limitation by having a root user run the <i>db2rfe</i> command.
DB2 instance actions can be performed	A DB2 instance created by a user with root authority can be updated or dropped only by a user with root authority.	A non-root instance can be updated or dropped (using the <i>db2_deinstall</i> command) only by the non-root user who owns the non-root instance.
Support partitioned database	Yes	No. Only single-partition databases are supported in non-root installations.

For the details about the limitations of non-root installations, refer to the IBM DB2 9.5 Information Center:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/topic/com.ibm.db2.luw.qb.sserver.doc/doc/c0050568.html>

Support changes for 32-bit and 64-bit DB2 servers

In DB2 9.5, support for 32-bit and 64-bit servers has the following changes:

- ▶ Only 64-bit instances could be created on 64-bit Linux kernel on x86_64, POWER, and zSeries.
- ▶ No 32-bit DB2 Enterprise Server Edition
DB2 9.5 Enterprise Server Edition (ESE) is only supported on 64-bit architectures.

2.2.2 Installation methods

There are four methods in which you can install DB2 on Linux:

- ▶ DB2 Setup wizard
- ▶ db2_install command
- ▶ Response file installation
- ▶ Payload file deployment

Each method has its own advantages and disadvantages. The preferred method often depends on your level of expertise and type of environment, but in general, if a graphical terminal is available, using DB2 Setup wizard is recommended.

DB2 Setup wizard

The DB2 Setup wizard is a Java-based graphical tool that installs DB2. It lays down the DB2 file sets, the Java Developer Kit, and allows you to create a new DB2 instance, create new users and group or configure DB2 to existing users, configure communications, create the tools catalog database, and set up notification.

DB2 Setup also has an option that allows you to create a response file. This is the best method for less experienced users, as most of the configuration is performed for you.

db2_install command

The db2_install script installs all DB2 packages on your Linux system, you cannot select or deselect components. This method is reliable and commonly used by expert users for installing DB2 on larger, more complex partitioned database environments. Tasks such as users and groups set up, instance creation, tools catalog database creation, and notification set up, have to be performed manually after the installation.

Some people prefer this method because it bypasses the configuration performed by DB2 Setup. It allows you to configure DB2 your preferred way in the first place and gives you more control over database management. Particularly for larger, multi-partition systems, we recommend that the databases be stored in a separate file system (for example, /database). Performing manual configuration allows you to allocate your preferred default database directory in the first place.

The db2_install command installs all components for the DB2 product you specify with the English interface support. You can select additional languages to support with the -L parameter.

Considering the higher number of manual configuration tasks, this installation method may take longer. It requires a higher level of skill and cannot create response files.

Response file installation

A response file can be created using the DB2 Setup wizard or by editing a sample response file. It allows you to install DB2 across multiple machines with consistent installation and configuration settings, and can also be used to set up a DB2 cluster. A response file installation is fast, because it bypasses the graphical wizard and does the configuration for you. Another advantage of using a response file is that it creates a Database Administration Server (DAS) on each machine, while with `db2_install` the DAS must be created manually after installation. For more advanced users, we recommend that you create a response file by editing a sample response file, then use this file to install DB2. The sample response file does not only install DB2, but can also configure users, create instance, set up notification, create tools catalog, and configure a large number of DBM parameters. This is the quickest installation method if you already have all the information you need. Unlike the DB2 Setup wizard, the response file installation is not interactive, and it takes a little bit longer to prepare the response file. However, you only have to take that “longer” way once and it can be used to install the same configuration on multiple machines.

Payload file deployment

This method is an advanced installation method that is not recommended for most users. It requires the user to physically install payload files. A payload file is a compressed tarball that contains all of the files and metadata for an installable component.

Like `db2_install`, this installation method requires manual configuration after the product files are deployed.

Note: DB2 product installations are no longer operating system packages on Linux platforms. You can no longer use *rpm* command for installation.

2.2.3 Storage planning

Both DB2 and data to be stored in the database required disk space. DB2 uses both raw device and file systems. It also requires log space. In this section, we discuss the storage planning for DB2 data server.

File systems versus raw devices

The DB2 program, the data to be stored in the databases, and the DB2 logs need disk space. This can be achieved with either raw devices or file systems. In this section, we discuss storage planning for DB2 data servers.

File systems

A popular method for configuring disk space for DB2 on Linux is to use separate file systems to store and run DB2. File systems can be used by DB2 either as system managed storage (SMS) or as database managed storage (DMS). File systems have many benefits. To name a few, they can be distributed across a network and have network-oriented authentication and replication capabilities, which makes them essential for a partitioned database system.

The operating system, by default, caches file data that is read from and written to disk. This behavior of caching data at the file system level is reflected in the **FILE SYSTEM CACHING** clause of the **CREATE TABLESPACE** statement. Since the database manager manages its own data caching using buffer pools, the caching at the file system level is not needed if the size of the buffer pool is tuned appropriately. In some cases, caching at the file system level and in the buffer pools causes performance degradation because of the extra CPU cycles required for the double caching. To avoid this double caching, most file systems have a feature that disables caching at the file system level. This is generically referred to as non-buffered I/O. On Linux, this feature is commonly known as Direct I/O (or DIO). The database manager supports this feature with the **NO FILE SYSTEM CACHING** table space clause. When this is set, the database manager automatically takes advantage of CIO on file systems where this feature exists. This feature might help to reduce the memory requirements of the file system cache, thus making more memory available for other uses.

Prior to Version 9.5, the keyword **FILE SYSTEM CACHING** was implied if neither **NO FILE SYSTEM CACHING** nor **FILE SYSTEM CACHING** was specified. With Version 9.5, if neither keyword is specified, the default, **NO FILE SYSTEM CACHING**, is used. This change affects only newly created table spaces. Existing table spaces created prior to Version 9.5 are not affected. This change applies to Linux with the following exceptions, where the default behavior remains to be **FILE SYSTEM CACHING**:

- ▶ Linux for System z
- ▶ All SMS temporary table space files
- ▶ SMS permanent table space files, except for long field (LF) data and large object (LOB) data files.

Table 2-8 shows the supported configuration for using table spaces without file system caching.

Table 2-8 Supported Linux platform and configuration for using table spaces without file system caching

Platforms	File system type and minimum level required	DIO or CIO requests submitted by the database manager when NO FILE SYSTEM CACHING is specified	Default behavior when neither NO FILE SYSTEM CACHING nor FILE SYSTEM CACHING is specified
Linux distributions SLES 9+ and RHEL 4+ (x86, x86_64, IA64, POWER)	ext2, ext3, ReiserFS	DIO	No file system caching
Linux distributions SLES 9+ and RHEL 4+ (x86, x86_64, IA64, POWER)	VERITAS Storage Foundation 4.1 (VxFS)	CIO	No file system caching
Linux distributions SLES 9+ and RHEL 4+ (on this architecture: zSeries)	ext2, ext3 or ReiserFS on a small computer system interface (SCSI) disks using fibre channel protocol (FCP)	DIO	file system caching

Raw devices

Raw devices can only be used for DMS storage. DMS table spaces require more administration, but provide superior performance. Raw device containers generally provide the best performance, because the database accesses the disk directly, bypassing all of the operating system cache and locking. However, as mentioned in “File systems” on page 40, Direct I/O file systems (DIO) reduce the gap between DMS files and raw devices.

For a detailed procedure of how to set up raw I/O on your Linux machine, refer to Linux documentation.

Note: Since DB2 Version 9.1, the previous raw I/O method that required binding the block device to a character device using the raw utility is deprecated. This raw I/O method is also deprecated in the Linux operating system and will be removed in a future release of Linux. You can use the block device as DB2 DMS table space container now. The block device method uses Direct I/O to achieve an equivalent performance compared to using the character device method.

File system configuration

We recommend that you create different partitions during your Linux OS installation. It has become customary for certain basic directories, such as

/db2home or /software (in our example), to be placed in separate file systems, or partitions, for several reasons:

- ▶ Disk capacity - There is a limited amount of space on each disk, and you might run out of space if you use a single disk for multiple purposes.
- ▶ Performance - The root directory has to be searched linearly every time any path name in Linux is accessed. If the root directory is cluttered, this will impair performance of the entire system.
- ▶ Backup - It is better to separate important and frequently changing data from massive, and seldom changing data. This way you can save system resources by backing up some file systems more frequently than others.
- ▶ User convenience - It's easier to find things if the naming convention is well-organized.

File system configuration, or *partitioning*, is typically performed by using the **fdisk** tool. For SUSE, you can also use the graphical YaST tool. Linux provides several journal file system formats, such as ext3, JFS, VxFS and ReiserFS. In our example, for database related file systems, we use the *ext3* format with *journal*³ mode to create a journal file system.

To show all defined file systems, issue the **df** command, as shown in Example 2-1. This shows the output for a partitioned database environment. For a single partition environment, you do not need NFS exported file system for DB2.

Example 2-1 df command

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/sda2	50572564	21169652	29402912	42%	/
udev	1921324	184	1921140	1%	/dev
/dev/sdb5	20641788	924912	19716876	5%	/export/db2home
/dev/sdb6	4822142	82420	4739722	2%	/database
/dev/sdb7	1010865	41150	969715	5%	/db2temp
/dev/sdb8	544166	41280	502886	8%	/db2log1
/dev/sdb9	544166	41280	502886	8%	/db2log2
/dev/sdb10	1010863	41271	969592	5%	/db2logarc
/dev/sdb11	5099266	2034030	3065236	40%	/software
mensa:/export/db2home	20641792	924896	19716896	5%	/db2home

To show all NFS mounted file systems, issue following command on the host where the file system physically exists (Example 2-2).

showmount -a

³ ordered is the ext3 file system default mode.

Example 2-2 showmount command

```
mensa:~ # showmount -a
All mount points on mensa:
9.43.86.90:mensa.itsosj.sanjose.ibm.com
mensa.itsosj.sanjose.ibm.com:/export/db2home
```

File system configuration recommendations

There are many different ways to configure disk partitions and file systems, depending on your environment. We suggest you follow these recommendations when configuring your system.

- ▶ For single and partitioned database systems, we recommend that you create a separate file system for the DB2 user home directories. In our setup, this file system is called */db2home* for single partition environment. For partitioned database environments, we recommend that you create a separate DB2 home file system on one of the machines in the cluster to be used as the instance home directory. This file system is to be shared between all machines in the cluster via NFS (that is, NFS exported from the NFS server machine, and NFS mounted on the remaining machines). In our setup we created the file system */export/db2home* with a mount point */db2home*.
- ▶ We also recommend that you create a separate file system for storing databases. For partitioned database systems, there should be a separate database file system on each physical system that participates in the partitioned database. In our setup, we created a file system called */database* with mount point */database*.
- ▶ For performance or availability reasons, we recommend that you avoid putting user data on the catalog node. When restoring the catalog node, the restore is faster when there is no data on this node.
- ▶ We also recommend that you create separate partitions for the primary copy of DB2 logs, DB2 mirrored logs, and DB2 temporary table spaces (this is discussed in more detail later in this section). In addition, you may want to create a separate file system for user table spaces in order to separate them from the database files. This configuration is commonly used in many production environments.

Here is an example of one possible configuration (Table 2-9). We will be using this system setup throughout this book.

Table 2-9 Database related file system partition setup sample

Partition name	Description
/db2home ^a	For storing the home directories for DB2 users
/database	For storing the database

Partition name	Description
/db2log1	Used to store primary copy of log files
/db2log2	Used to store DB2 mirroring log files
/db2logarc	Used to store archived log files
/db2temp	For storing DB2 temporary tablespaces
/software	Used for storing software. For example, we downloaded the DB2 install image into this directory.

a. /export/db2home in the partitioned database environment

Log space

By default, DB2 sets the log path to the default database path during database creation. For example, our default database path is /database and our log path is:

```
/database/db2inst1/NODE0000/SQL00001/SQLLOGDIR/
```

We recommend that you store both the primary copy of the logs and the mirror logs each on a physically separate disk, preferably one that is also on a different disk controller.

Mirror logs are created using the MIRRORLOGPATH configuration parameter. Log mirroring allows the database to write an identical second copy of log files to a different path.

Note: Mirroring log files helps protect a database from accidental deletion of an active log and data corruption caused by disk error. While this functionality increases the high availability of a system, log mirroring may impact system performance as all log data will be written to both the log path and the mirror log path.

In Example 2-3, we change the primary log path from the default to /db2log1/ITSODB and set the mirror log path to /db2log2/ITSODB for the database ITSODB.

Example 2-3 db2 logs configuration

```
db2 update db cfg for ITSODB using NEWLOGPATH /db2log1/ITSODB
db2 update db cfg for ITSODB using MIRRORLOGPATH /db2log2/ITSODB
```

The procedure of changing the log path is discussed in 3.4.1, “Change online log path” on page 116.

Note: These changes will only take place after you deactivate and activate your database.

Temp space

DB2 uses system temporary table spaces for many SQL operations, such as JOIN and SORT. DB2's temporary table space, TEMPSPACE1, is one of the three default table spaces (SYSCATSPACE, TEMPSPACE1, and USERSPACE1) that are created during database creation. By default, TEMPSPACE1 is placed in the database path. For larger systems, we recommend that your temporary table spaces are located on a separate file system and disk. For example, in our system, we created the file system /db2temp with mount point /db2temp to store our temporary table spaces.

In a partitioned database environment, the catalog node should contain all three default table spaces, and the other database partitions should each contain only TEMPSPACE1 and USERSPACE1.

Example 2-4 shows how to create a system temporary table space on multiple nodes in the /db2temp file system:

Example 2-4 Create temp table space in a different file system

```
CREATE TEMPORARY TABLESPACE temp space2 IN ibmtempgroup
MANAGED BY SYSTEM
USING ('/db2temp/db2inst1/NODE0000/ITSODB/TEMPSPACE2') ON DBPARTITIONNUM (0)
USING ('/db2temp/db2inst1/NODE0001/ITSODB/TEMPSPACE2') ON DBPARTITIONNUM (1)
EXTENTSIZE      8
PREFETCHSIZE 32
BUFFERPOOL ibmdefaultbp;
DROP TABLESPACE temp space1;
```

2.2.4 Lab environment

In this book, we use the following systems to demonstrate DB2 installation, configuration, and management for the partitioned database environment.

- Mensa
 - Hostname: mensa
 - Domain: itsosj.sanjose.ibm.com
 - Host IP address: 9.43.86.90
 - Linux distribution: SUSE (SLES 10 SP1 x86_64)
 - Partitioned database setup:

- The partitioned instance has one logical partition, node 0, initially on this physical node, that is the coordinate node of the partitioned database environment. We add more logical partitions later.
 - The partitioned instance's home directory /db2home is exported from this machine.
- Gemini
- Hostname: gemini
 - Domain: itsosj.sanjose.ibm.com
 - Host IP Address: 9.43.86.94
 - Linux Distribution: SUSE (SLES 10 SP1 x86_64)
 - Partitioned database setup:
 - The partitioned instance has one logical partition, node 1, on this physical node. We add more logical partitions later.

2.2.5 Setting up NFS for a partitioned database environment installation

If you have a configuration that uses more than one machine for a single database instance, you must configure communications before installing DB2. In this section, we discuss how to set up ssh and Network File System (NFS).

In order to get NFS up and running, you need to configure the */etc/exports* and */etc/fstab* files. The *exports* file is configured on the server side and specifies which directories are to be shared with which clients and the access rights for each client. The *fstab* file is configured on the client side and specifies which servers to contact for each directory, as well as where to place them in the directory tree.

In this section, we lead you through the steps that are required to set up NFS:

- Get the NFS service running on each machine in the cluster.
- On a single machine in the cluster (the NFS server), create a file system to be used as the instance home directory (if not already created).
- Mount this file system locally.
- Set this file system to be mounted at each reboot.
- Export the DB2 home file system using NFS.
- Mount the exported file system on each of the remaining machines in the cluster.

Get NFS service running

To verify that Network File System (NFS) is running on each computer that will participate in the partitioned database system, enter the following command:

```
showmount -e hostname
```

If you enter **showmount** without specifying a hostname, it will check the local system.

If NFS is not running, you will receive a message similar to the following:

```
mount clntudp_create: RPC: Program not registered
```

To start the NFS, perform the following:

- ▶ Make sure you have the *nfs-utils* or *nfs-server* package installed on each machine.
- ▶ Start the server manually by running the following command as root:

```
/etc/init.d/nfs restart
```
- ▶ You may check the status of NFS by running the following command:

```
/etc/init.d/nfs status
```

Once you have verified that NFS is running on each system, check for the *rpc.statd* process using the **ps -ef | grep rpc.statd** command. The *rpc.statd* process is required by DB2.

Create and configure the DB2 home file system to NFS

You need to create a DB2 instance home directory that is shared across all machines that will participate in your partitioned database system. In our example, we allocate a file system */export/db2home* on *mensa*. NFS is used to share this file system.

1. If you haven't already done so, create the DB2 home file system (for example, */export/db2home*) on the NFS server using utilities, such as **fdisk** (to create disk partition) and **mkfs** (to create file system on disk partition), or the YaST tool on SUSE.
2. Locally mount the DB2 home file system using the following command:

```
mount /export/db2home
```
3. Make sure that your new file system has been added to the */etc/fstab* file. This will mount the file system each time the system is rebooted. Most Linux file system creation utilities (for example, Disk Druid, YaST) automatically add an entry to the */etc/fstab* file when you create a new file system. If your DB2 home file system is not found in the */etc/fstab* file, you may add it manually. In */etc/fstab*, you should see an entry for */export/db2home*. In our setup, the entry is:

```
/dev/sdb5          /export/db2home ext3      data=journal,acl,user_xattr
1 2
```

4. Next, add an entry to the `/etc/exports` file to automatically export the NFS file system at boot time. You only need to set up this file on your NFS servers. The `/etc/exports` file follows this format:

```
/directory_to_export machine1_name(permissions) machine2_name(permissions)
machinen_name(permissions)
```

For example, in our `/etc/exports` file, we have:

```
/export/db2home mensa(rw,no_root_squash,sync)
gemini(rw,no_root_squash,sync)
```

Note: In the permissions section, you can specify user ID mappings. By default, permissions are set to `rw` and `root_squash`. The `root_squash` setting means that the root user on a client is not treated as root when accessing files on the NFS server. Although this mode of operation is typically desirable in a production environment, you need to turn it off in order to create users on each of the remaining machines in the cluster. To do this, specify `no_root_squash` in the permissions section.

Before deploying your environment on production, you may consider to change permissions to default setting, that is, `(rw, root_squash)`. For instance, change the `/etc/exports` file to:

```
/export/db2home mensa(rw,sync) gemini(rw,sync)
```

5. Export the NFS directory by running:

```
/usr/sbin/exportfs -a
```

This initializes a file named `/var/lib/nfs/xtab` and exports all directories to this file.

Note: If you make an update to `/etc/exports`, run `exportfs -r` to re-export all directories. It synchronizes `/var/lib/nfs/xtab` with `/etc/exports` and removes entries in `/var/lib/nfs/xtab` that are deleted from `/etc/exports`.

6. On each of the machines in the cluster (including NFS server), first make a directory `/db2home` using the `mkdir` command, then add an entry to the `/etc/fstab` file to NFS mount the file system automatically at boot time. We recommend that you configure the file system to be mounted at boot time, is read-write, is mounted hard, includes the background (`bg`) option, and that `setuid` programs can be run properly. Refer to the following example:

```
mensa:/export/db2home /db2home nfs rw,timeo=7,hard,intr,bg,suid,lock
```

Where, *mensa* is the NFS server machine name in our example.

7. After adding an entry to the `/etc/fstab` file on each machine, NFS mount the exported file system on each of the remaining machines in the cluster by entering the following command:

```
mount /db2home
```

If the mount command fails, make sure the NFS server is started. To re-start the NFS server, run the following command as root on the NFS Server workstation:

```
./etc/init.d/nfs restart
```

You may also use the `showmount` command to check the status of the NFS server. For example:

```
showmount -e mensa
```

Tips:

- ▶ It is considered good convention to place all the directories you want to export in the `/export` hierarchy. If you need the directory to also exist elsewhere in the directory tree, use symbolic links or mount it as a NFS client. For example, if your server is exporting its `/db2home` hierarchy, you should place the directory in `/export`, thereby creating `/export/db2home`. Because the server itself will need access to the `/export/db2home` directory, you should mount it on `/db2home`.
- ▶ If you have an error in your `/etc/exports` file, it is reported in when NFS starts up in *syslog*. You might find this debugging tool very useful.

2.2.6 User and group setup

In this section, we discuss the user IDs and groups that DB2 requires and the process of creating them.

Required users and groups

Three users and groups are required for DB2: the instance-owning user, the DB2 fenced user, and the Database Administration Server (DAS) user⁴. You may use the default names provided by DB2 Setup, or specify your own user and group names. In our setup, we used the DB2 Setup wizard default user ID and group names, which are shown in Table 2-10.

⁴ For non-root installation, since DAS is not available, DAS user is not required.

Table 2-10 Example of required users for DB2

Required user	User name	Group name	Description
Instance owner	db2inst1	db2iadm1	Administers the instance
Fenced user	db2fenc1	db2fadm1	Responsible for executing fenced user defined functions, such as UDFs and stored procedures.
DAS user	dasusr1	dasadm1	Administers the DB2 Administration Server.

Creating users

Root authority is required to create users and groups.

For root installation, there are three ways in which you can create DB2 users and groups:

- ▶ Using the DB2 Setup wizard
- ▶ Using a response file installation
- ▶ Creating users and groups in OS command line or in OS administration tool

For non-root installation, users and groups should exist before installation.

DB2 Setup Wizard

The DB2 Setup Wizard creates all of the required users and groups for you during installation. The default users and groups created are listed in Table 2-10 on page 50. DB2 Setup also gives you an option to specify your own user and group names.

Response file

Users can also be created during a response file installation if you specify user and group information in the response file. Example 2-5 shows the entries in our response file to create the three required users and groups for DB2.

Example 2-5 Using a response file to create users and groups

```

** Instance Creation Settings
db2inst1.NAME           = db2inst1
db2inst1.UID            = 1001
db2inst1.GROUP_NAME     = db2iadm1
db2inst1.GID            = 999
db2inst1.HOME_DIRECTORY = /db2home/db2inst1
db2inst1.PASSWORD       = xxxxxxxxxx
.....
** Fenced User Creation Settings
db2inst1.FENCED_USERNAME = db2fenc1

```

```

db2inst1.FENCED_UID      = 1002
db2inst1.FENCED_GROUP_NAME = db2fadm1
db2inst1.FENCED_HOME_DIRECTORY = /db2home/db2fenc1
db2inst1.FENCED_PASSWORD = xxxxxxxxxx
.....
** Administration Server Creation Settings
DAS_USERNAME      = dasusr1
DAS_UID           = 1003
DAS_GROUP_NAME    = dasadm1
DAS_GID           = 997
DAS_HOME_DIRECTORY = /home/dasusr1
DAS_PASSWORD      = xxxxxxxxxx
.....

```

Creating users and groups in OS command line

To use this method, follow these steps:

1. Log on as root.
2. Create groups for the instance owner, the fenced user, and the DAS user by using the following commands:

```

groupadd -g 999 db2iadm1
groupadd -g 998 db2fadm1
groupadd -g 997 dasadm1

```

In our setup, we used the numbers 997, 998, and 999. Make sure that the numbers you choose do not already exist on your machine by checking */etc/group* file.

3. Create a user that belongs to each group and specify the home directory.

```

useradd -u 1001 -g db2iadm1 -m -d /db2home/db2inst1 -s /bin/ksh db2inst1
useradd -u 1002 -g db2fadm1 -m -d /db2home/db2fenc1 -s /bin/ksh db2fenc1
useradd -u 1003 -g dasadm1 -m -d /home/dasusr1 -s /bin/ksh dasusr1

```

In our setup we used the numbers 1001, 1002, and 1003. Make sure that the numbers you choose do not already exist on your machine by checking */etc/passwd* file.

4. As root user, set a password for each user that you created by entering the following commands:

```

passwd db2inst1
passwd db2fenc1
passwd dasusr1

```

Creating users: Multiple partition considerations

In a partitioned database environment, you only need to create one shared home directory for the instance owner and fenced user (but remember to create users on each machine!). When creating users in a partitioned environment, make sure

that the user and group IDs are the same on each machine. In our setup we have:

- ▶ A shared home directory, `/db2home`, on the instance-owning machine which is NFS-mounted on the remaining machines in the cluster. In this directory we have the home directories for the instance-owning user and fenced user: *db2inst1* and *db2fenc1*.
- ▶ A local home directory for the DAS user on each machine, *dasusr1*, which is stored in the `/home` directory.

DAS user considerations for a partitioned database

We could *not* place the DAS user home directory in the shared folder because we have the same DAS user ID on each machine in the cluster. You should take note of the following DAS user considerations.

- ▶ A DAS must be running on each physical machine in the partitioned database for the graphical administration tools (for example, Control Center) to work.
- ▶ You can only have one DAS on each machine. If you have an existing DAS for old DB2 version, we recommend that you drop it and create a new one in DB2 9.5.
- ▶ Just like an instance, each DAS must be created under a user ID. It does not matter whether a different user ID is used for each DAS in the environment, or whether the same user ID is used and that the user ID's home directory is not shared.
- ▶ If an existing user is used as the DAS user, this user must also exist on all the participating computers before installation.
- ▶ *For response file installs:* If your response file specifies to create new DAS user on each machine in the cluster, and that user already exists on any of the participating computers, then that user must have the same primary group as the new DAS user.

2.2.7 Enabling ssh for a partitioned database environment installation

From DB2 Version 8.2.2 (Version 8, FixPack 9), you can use secure shell (ssh) as an alternative of remote shell (rsh) to executing commands on remote DB2 nodes in the DB2 partitioned database environments. Since ssh is more secure than rhost-based authentication, we recommend that you configure DB2 to use ssh in the partitioned database environments.

To run DB2 ESE successfully, the ssh server and client have to be available on all machines in the partitioned database system. Prior to enabling ssh, ensure the ssh sever and client packages are installed. They are not installed by default in some distributions.

This section shows you how to configure ssh for host-based and public key-based authentication on Linux. You can choose to use either the host-based authentication or the public key-based authentication when setting up ssh. You can also choose to use either the Rivest-Shamir-Adleman algorithm (RSA) or the digital signature algorithm (DSA) encryption as the form of encryption. The decision should be based on your organization security requirements.

Setting up host-based authentication

Host-based authentication allows any user ID from machineA to use ssh to log in as that same user ID on machineB, assuming that the ssh client on machineA is configured to use host-based authentication and the ssh server on machineB is configured to allow host-based authentication. In your partitioned database environment, each partition server needs to be configured to use host-based authentication and each partition server must have the ssh client and ssh server configured correctly.

SSH server configuration

Use the following steps to configure SSH server. These steps should be applied on each physical node.

1. Edit the *sshd_config* file.

The *sshd_config* file can be found in */etc/ssh* on Linux. Log in as root. Change the *HostbasedAuthentication* parameter to *Yes* as shown in Example 2-6. This allows the ssh server to accept host-based authentication requests from the ssh clients. We comment out the default setting and left it in the file.

Example 2-6 Edit the sshd_config file

```
# HostbasedAuthentication no
HostbasedAuthentication yes
```

2. Edit the */etc/ssh/shosts.equiv* file.

The *shosts.equiv* file can be found in */etc/ssh* on Linux. If this file does not yet exist, create the file. Ensure that the file is owned by the root user and only allows user read-write access, group/other read access. See Example 2-7.

Example 2-7 Create shosts.equiv and modify the file properties

```
$ cd /etc/ssh
$ touch shosts.equiv
$ chown root:root shosts.equiv
$ chmod 644 shosts.equiv
```

Each host must be able to communicate with every other host in a partitioned database environment, so you must set up the *shosts.equiv* file such that it

can be reused on all hosts. In our example, we only have two partition servers, mensa and gemini. Edit the file as shown in Example 2-8:

Example 2-8 Edit the shosts.equiv file

```
mensa
mensa.itsosj.sanjose.ibm.com
gemini
gemini.itsosj.sanjose.ibm.com
```

3. Edit the `ssh_known_hosts` file

The ssh server host system needs to have access to the public key of the ssh client host. For host-based authentication, the trust mechanism looks for public keys in the `ssh_known_hosts` file. The `ssh_known_hosts` file is found in the `/etc/ssh` directory on Linux. If this file does not yet exist, create the file. Ensure that it is owned by the root user and only allows user read-write access and group/other read access. See Example 2-9.

Example 2-9 Create ssh_known_hosts and modify the file properties

```
$ cd /etc/ssh
$ touch ssh_known_hosts
$ chown root:root ssh_known_hosts
$ chmod 644 ssh_known_hosts
```

Add unqualified host name, fully qualified host name, and IP address of the client machine to the `ssh_known_hosts` file as shown in Example 2-10 and Example 2-11. You can use the **ssh-keyscan** utility to populate this file.

Example 2-10 Update the ssh_known_hosts file for RSA encryption

```
$ cd /etc/ssh
$ ssh-keyscan -t rsa mensa,mensa.itsosj.sanjose.ibm.com,9.43.86.90
>>ssh_known_hosts
$ ssh-keyscan -t rsa gemini,gemini.itsosj.sanjose.ibm.com,9.43.86.94
>>ssh_known_hosts
```

Example 2-11 Update the ssh_known_hosts file for DSA encryption

```
$ cd /etc/ssh
$ ssh-keyscan -t dsa mensa,mensa.itsosj.sanjose.ibm.com,9.43.86.90
>>ssh_known_hosts
$ ssh-keyscan -t dsa gemini,gemini.itsosj.sanjose.ibm.com,9.43.86.94
>>ssh_known_hosts
```

4. Restart the ssh daemon

`/etc/init.d/sshd restart`

SSH client configuration

Perform the following steps to configure SSH client:

1. Edit the `/etc/ssh/ssh_config` file

Change the line *HostbasedAuthentication* from *no* to *yes* as shown in Example 2-12.

Example 2-12 Edit the ssh_config file

```
# HostbasedAuthentication no
HostbasedAuthentication yes
```

Note that the ssh server configuration file is `sshd_config`.

Add a line to the `ssh_config` file to tell the ssh client to use the `ssh-keysign` utility to read the host's private key as shown in Example 2-13.

Example 2-13 Add a line to the ssh_config file

```
EnableSSHKeySign yes
```

2. Ensure `ssh-keysign` is a `suid-root` executable.

Check `ssh-keysign` setting:

On SUSE:

```
ls -l /usr/lib64/ssh/ssh-keysign
-rwsr-xr-x 1 root root 173952 2007-05-18 03:55 /usr/lib64/ssh/ssh-keysign
```

On Red Hat:

```
ls -l /usr/libexec/openssh/ssh-keysign
-rwsr-xr-x 1 root root 175424 Jul 13 2007 /usr/libexec/openssh/ssh-keysign
```

If the permissions and owner do not resemble the sample output above, then fix the permissions as shown in Example 2-14 and Example 2-15.

Example 2-14 Fix permissions on ssh-keysign on SUSE

```
$ cd /usr/lib64/ssh/
$ chown root:root ssh-keysign
$ chmod 4755 /usr/libexec/openssh/ssh-keysign
```

Example 2-15 Fix permissions on ssh-keysign on Red Hat

```
$ cd /usr/libexec/openssh/
$ chown root:root ssh-keysign
$ chmod 4755 /usr/libexec/openssh/ssh-keysign
```

Setting up public key authentication

Public key-based authentication allows a single user ID (in our case, the DB2 instance owning ID in our partitioned database environment) to log in as the same user ID on each partition server without being prompted for a password. Since the home directory of DB2 partitioned instance owner⁵ is shared across all machines (a pre-requisite for DB2 partitioned database environment), it is possible to complete the entire setup from a single machine.

The steps to set up public key authentication is as follows:

1. Create .ssh directory.

Log in as the DB2 partitioned instance owning user ID. If this ID does not already have a ~/.ssh directory, create one and ensure that the .ssh directory does not allow group or other the write access. At the same time, ensure that your home directory does not allow group or other the write access. ssh views this as a security exposure and will not allow public key-based authentication if the directory permissions are not restrictive enough.

On our testing environment *mensa*, the instance owning ID is db2inst1 and the home directory is /db2home/db2inst1. We create .ssh directory with required authorization as shown in Example 2-16.

Example 2-16 mkdir .ssh directory for user db2inst1 (log in as db2inst1)

```
$ cd /db2home
$ chmod 700 db2inst1
$ mkdir ~/.ssh
$ chmod 700 ~/.ssh
```

2. Generate a key pair.

From ~/.ssh directory, generate a public key/private key pair.

Example 2-17 shows how to generate a public key/private key pair for RSA.

Example 2-17 Generate an RSA-encrypted key pair

```
$ cd ~/.ssh
$ ssh-keygen -t rsa
```

You also can generate a DSA key pair. Example 2-18 on page 56 shows how to generate a public key/private key pair for DSA.

Example 2-18 Generate a DSA-encrypted key pair

```
$ ssh-keygen -t dsa
```

⁵ Refer to “Creating users” on page 50

Whenever prompted for input, press **Enter** to accept the default value. You then are prompted to enter a passphrase. In our environment, we do not want a passphrase so we press **Enter** twice. Two new files are generated in the `~/.ssh` directory, `id_rsa` (the private key) and `id_rsa.pub` (the public key), for RSA encryption. In a similar manner, name files are generated for DSA encryption, `id_dsa` (the private key) and `id_dsa.pub` (the public key).

3. Enable the key pair.

Example 2-19 shows the commands to enable the RSA key pair.

Example 2-19 Enable the RSA key pair

```
$ cd ~/.ssh
$ mv id_rsa identity
$ chmod 600 identity
$ cat id_rsa.pub >> authorized_keys
$ chmod 644 authorized_keys
$ rm id_rsa.pub
```

Example 2-20 shows the commands to enable the DSA key pair.

Example 2-20 Enable the DSA key pair

```
$ cd ~/.ssh
$ mv id_dsa identity
$ chmod 600 identity
$ cat id_dsa.pub >> authorized_keys
$ chmod 644 authorized_keys
$ rm id_dsa.pub
```

4. Set up the host trust relationships.

Because we have not set up any host trust relationships, the first time that `ssh` is issued to a new host, you will be prompted with a message saying that the authenticity of the target host cannot be established. To overcome this situation, you can form a trust relationship with the other hosts in the partitioned database environment. To achieve this trust relationship, use the **ssh-keyscan** utility to gather the public host key for each host in the partitioned database environment and save the keys in the `known_hosts` file.

Example 2-21 shows how to use the `ssh-keyscan` utility to gather RSA keys to set up the host trust relationships.

Example 2-21 Gather the RSA public keys

```
$ ssh-keyscan -t rsa
mensa,mensa.itsosj.sanjose.ibm.com,9.43.86.90>>~/.ssh/known_hosts
$ ssh-keyscan -t rsa
gemini,gemini.itsosj.sanjose.ibm.com,9.43.86.94>>~/.ssh/known_hosts
```

Example 2-22 shows how to use the ssh-keyscan to gather DSA keys to set up the host trust relationships.

Example 2-22 Gather the DSA public keys

```
$ ssh-keyscan -t dsa  
mensa,mensa.itsosj.sanjose.ibm.com,9.43.86.90>>~/.ssh/known_hosts  
$ ssh-keyscan -t dsa  
gemini,gemini.itsosj.sanjose.ibm.com,9.43.86.94>>~/.ssh/known_hosts
```

Verify ssh configuration

To verify ssh configuration, log onto each machine as the DB2 partitioned instance owning user ID, and type following commands as shown in Example 2-23.

Example 2-23 Verify ssh configuration

```
$ ssh mensa echo hi  
$ ssh gemini echo hi
```

These commands should complete successfully without prompting for additional verification.

2.3 Installing DB2

In this section, we explain how to install DB2 using the DB2 Setup, db2_install, and response file installation methods. The following topics are covered:

- ▶ Non-root installation
- ▶ Root installation
- ▶ Installing a DB2 license files
- ▶ Installing DB2 documentation
- ▶ Adding an additional partition

Note: With DB2 Version 9 and later, DB2 license files are shipped separately. You have to install DB2 license files for each DB2 copy no matter which installation method you choose. See 2.3.4, “Installing DB2 license files” on page 84.

2.3.1 Non-root installation

For non-root installation, you can use one of the following methods to install single-partition database environments:

- ▶ DB2 Setup wizard
- ▶ Response file installation - db2setup command with a response file (silent install)
- ▶ db2_install utility

DB2 Setup wizard

For non-root installation, you can use DB2 Setup wizard as follows:

1. Ask Linux system administrator who has root authority to mount DB2 product CD or DVD using the following command:

```
mount -t iso9660 -o ro /dev/dvd /mnt/dvd
```

Where /mnt/dvd represents the mount point of the CD or DVD.

2. Log in as a non-root user, run **db2setup**⁶, and select **Install a Product** on the first window.

Note:

If you use X Window System software, you may need to set the DISPLAY variable for non-root user to starting a Java-based graphical tool. You can get the DISPLAY variable setting from root user. For example, as root, enter the following command:

```
$ echo $DISPLAY  
127.0.0.1:1.0
```

This shows the DISPLAY variable for root user is 127.0.0.1:1.0. You need run **xhost** command to allow clients access at first. For example, as root enter the following command:

```
$ xhost + 127.0.0.1
```

Then, as the non-root user, enter the following command:

```
$ export DISPLAY=127.0.0.1:1.0
```

3. Select **Install New**. On *Welcome to the DB2 Setup Wizard* window, click **Next**.
4. On *Software License Agreement* window, read the license agreement and then select **Accept** and click **Next**.
5. Select the installation type as show in Figure 2-1 on page 60. You may choose a Typical, Compact, or Custom installation.

⁶ You can use **db2setup -i <language>** to run the DB2 Setup wizard in another language.

- *Typical* installs most DB2 components except for the *Base application development tools*. Click the View Features button to see what components will get installed.
- *Compact* installs only basic DB2 features and functionality.
- *Custom* allows you install whatever components you want. An advantage of the Custom option is that it allows you to install the Base application development tools with DB2. With the Typical option, the Base application development requires a separate installation.

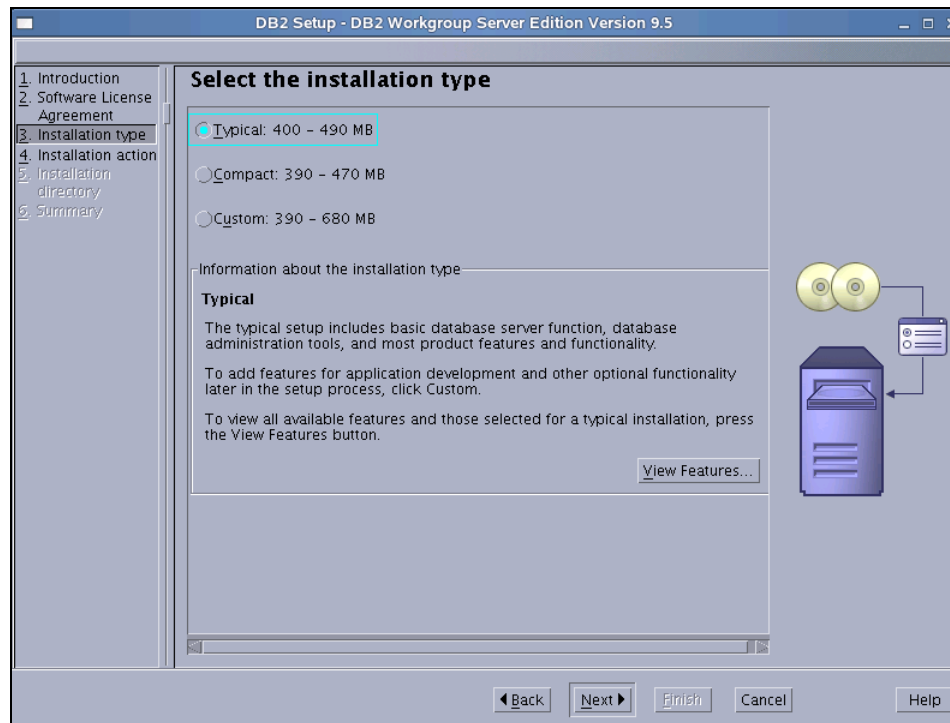


Figure 2-1 Non-root installation -Select the installation type

6. On the *Select the Installation, response file creation, or both* window as shown in Figure 2-2 on page 61, we recommend you that select install DB2 and save your settings in a response file. Then, click **Next**.

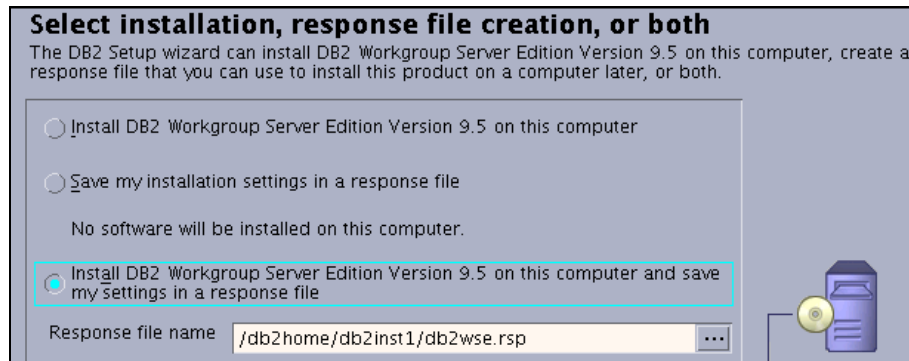


Figure 2-2 Non-root installation - Select Installation, response file creation, or both

7. On the *Installation directory* window, as shown in Figure 2-3, click **Next**.

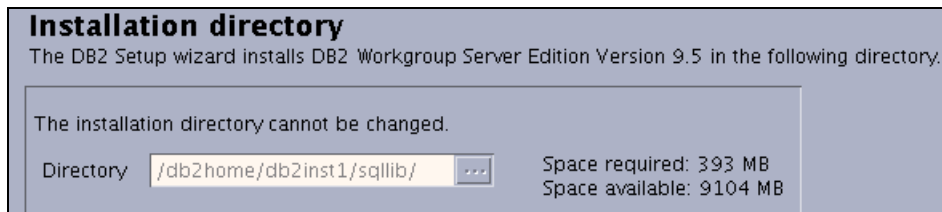


Figure 2-3 Non-root installation - Installation directory

Note: Since it is a non-root installation, you can not change the installation directory.

8. *Start copying files and create response files.* This window provides a summary of your installation and configuration settings as shown in Figure 2-4 on page 62. Scroll through this window to verify that your settings are correct, then click **Finish**.
9. After installation, read the status report tab and check for any errors. If all is well, click **Finish**.

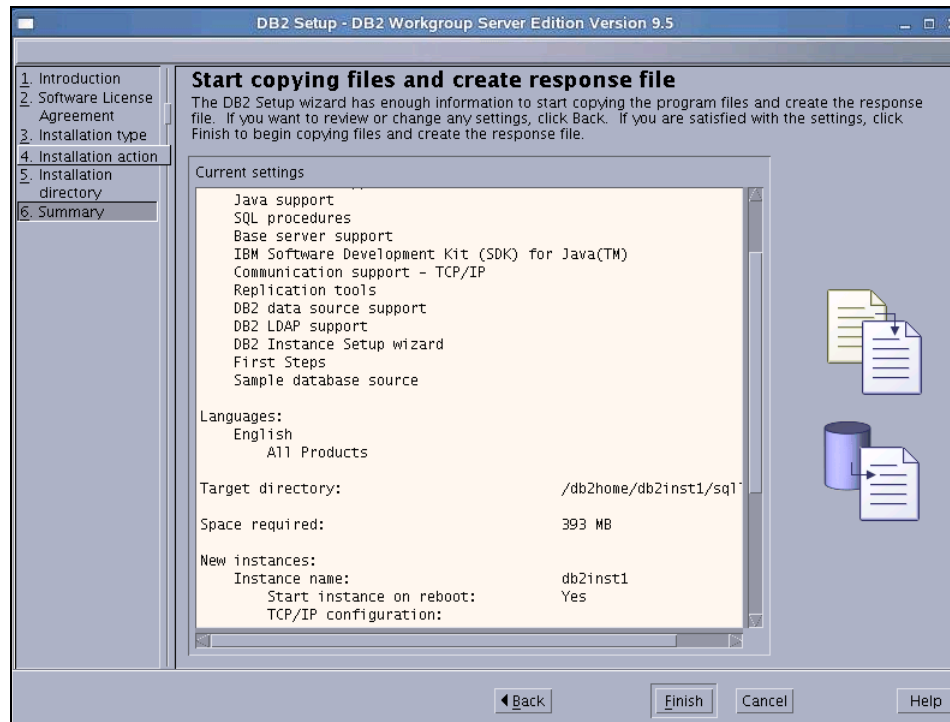


Figure 2-4 Non-root installation - Installation summary window

Note: In this case, the setup logs are located in `/tmp/db2setup_db2inst1.log` and `/tmp/db2setup_db2inst1.err`.

Response file install

Unlike the DB2 Setup wizard, a DB2 response file installation let you install DB2 products without any user interaction. A response file is an English-only text file that contains setup and configuration information. A response file specifies configuration and setup parameters, and the products and components to install.

Response files have a file type of `.rsp`, for example `db2wse.rsp`. See Figure 2-2 on page 61. However, you have the option to change the name and destination directory while using the DB2 Setup wizard.

Creating a response file

You can run the DB2 Setup wizard in graphical mode once and capture the values you enter to a response file.

You also can create the response file manually by using the templates provided. The response files have a .rsp extension and are located in the `/mnt/dvd/db2/linux/samples` or `/mnt/dvd/db2/linuxamd64/samples` directory on the DB2 product DVD, where `/mnt/dvd` represents the mount point of the CD or DVD.

Note: On Linux platforms, a response file created for a root installation might not be usable for a non-root installation. Some response file keywords are valid for root installation only. We do *not* recommend that you edit the sample response file for non-root IDB2 installation.

Creating a response file using the DB2 Setup wizard

Near the beginning of the DB2 Setup wizard, on the *Select the Installation, response file creation, or both* window (Figure 2-2 on page 61), select either **Save my installation setting in a response file** or **Install product on this computer and save my settings in a response file** option. In the *Response file name* field, enter the path where you want the DB2 Setup wizard to place the generated response file. In our sample, we save all installation settings to a file called `db2wse.rsp`.

Example 2-24 shows the `db2wse.rsp` file content.

Example 2-24 Sample response file db2wse.rsp

```
* Product Installation
LIC_AGREEMENT      = ACCEPT
PROD               = WORKGROUP_SERVER_EDITION
FILE               = /db2home/db2inst1/sqllib/
INSTALL_TYPE       = TYPICAL
* -----
* Instance properties
* -----
INSTANCE           = inst1
inst1.TYPE          = wse
* Instance-owning user
inst1.NAME          = db2inst1
inst1.AUTOSTART     = YES
inst1.PORT_NUMBER   = 48786
* -----
* Installed Languages
* -----
LANG               = EN
```

Installing DB2 using the response file

To install DB2 using the response file, use **db2setup** command with **-r** option. For example:

```
./db2setup -r /db2home/db2inst1/db2wse.rsp
```

db2_install utility

The db2_install utility installs the DB2 filesets and creates a non-root instance during a non-root installation. Some people prefer to use db2_install when installing DB2 on a large, complex database system that has special requirements.

Installing DB2

The steps to install DB2 using db2_install are as follows:

1. Ask Linux system administrator who has root authority to mount DB2 product CD or DVD.
2. Log in with the user ID that owns the non-root installation.
3. Change to the directory where the DB2 product DVD is mounted by entering the following command:

```
cd /mnt/dvd
```

where /mnt/dvd represents the mount point of the DVD.

4. Enter the **db2_install** command to start the db2_install script.
5. Enter product keyword when db2_install prompts you for it. For example, to install DB2 Workgroup Server Edition, enter in WSE.

The installation directory for DB2 on Linux is *\$INSTHOME*/sqlib.

Configuring TCP/IP communications

After installing DB2 using db2_install, You must configure your instance with the TCP/IP protocol in order for it to accept requests from remote DB2 clients. Follow these steps:

1. Update the database manager configuration file on the server. To do so:
 - a. Log into system as instance owner.
 - b. Update the *SVCENAME* parameter in the DBM configuration file. You may specify port number. For example, we enter:

```
db2 update dbm cfg using SVCENAME 50001
```

You can check your SVCENAME by entering:

```
db2 get dbm cfg | grep SVC
```

2. Set the *DB2COMM* registry variable to tcp/ip. This will start the DB2 communications manager when the database manager is started. Enter the following command as instance owner:

```
db2set DB2COMM=tcpip
```

3. Stop and re-start the instance for these changes to take effect:

```
db2stop  
db2start
```

2.3.2 Root installation

For root installation, you can use one of the following methods to install both single-partition and multi-partitioned database environments:

- ▶ DB2 Setup wizard
- ▶ Response file installation - db2setup command with a response file (silent install)
- ▶ db2_install utility

DB2 Setup Wizard

For root installation, you can use DB2 Setup wizard in the following way:

1. Log in as root.
2. Mount DB2 product CD or DVD using the following command:

```
mount -t iso9660 -o ro /dev/dvd /mnt/dvd
```

Where /mnt/dvd represents the mount point of the CD or DVD.
3. As root, run **db2setup**⁷, and select **Install a Product** on the first window.
4. Select **Install New**. On *Welcome to the DB2 Setup Wizard* window, click **Next**.
5. On *Software License Agreement* window, read the license agreement and then select **Accept** and click **Next**.
6. Select the installation type as show in Figure 2-5 on page 66. You may choose a Typical, Compact, or Custom installation.
 - *Typical* installs most DB2 components except for the *Base application development tools*. Click the View Features button to see what components will get installed.
 - *Compact* installs only basic DB2 features and functionality.
 - *Custom* allows you install whatever components you want. An advantage of the Custom option is that it allows you to install the Base application development tools with DB2. With the Typical option, the Base application development requires a separate installation.

⁷ You can use **db2setup -i <language>** to run the DB2 Setup wizard in another language.

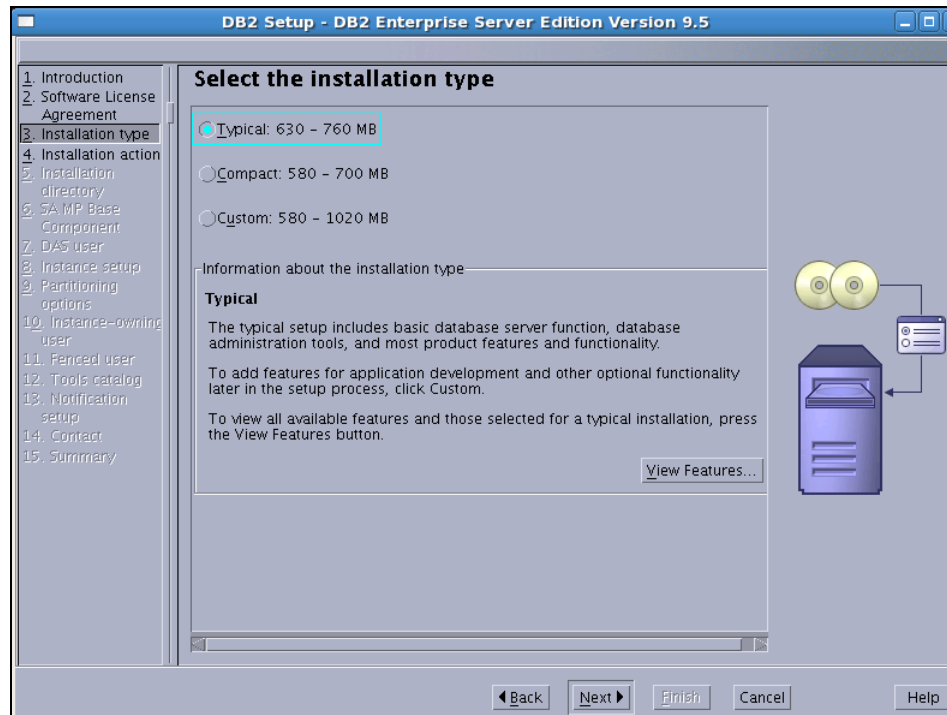


Figure 2-5 Root installation -Select the installation type

7. On the *Select the Installation, response file creation, or both* window (Figure 2-6 on page 66), we recommend you that select install DB2 and save your settings in a response file. Then, click **Next**.

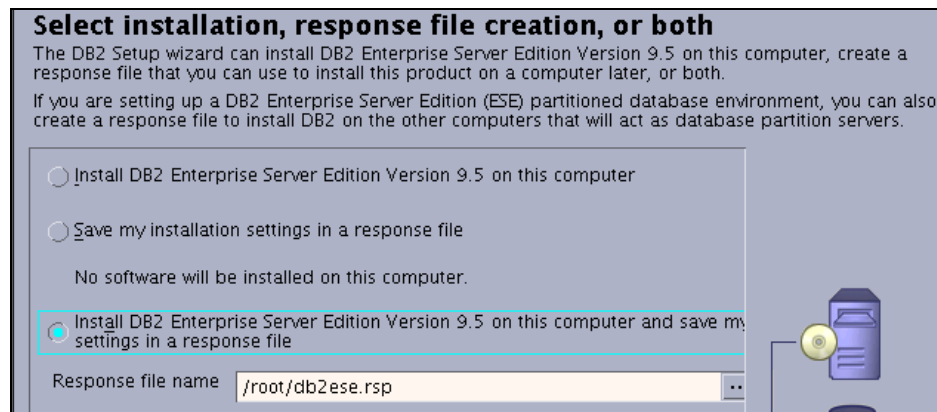


Figure 2-6 Root installation - Select installation, response file creation, or both

8. On the *Select the installation directory* window, as shown in Figure 2-7 on page 67, click **Next**.



Figure 2-7 Root installation - Select the installation directory

9. On the next windows, as shown in Figure 2-8, you can choose to install IBM TSA or not, then click **Next**.

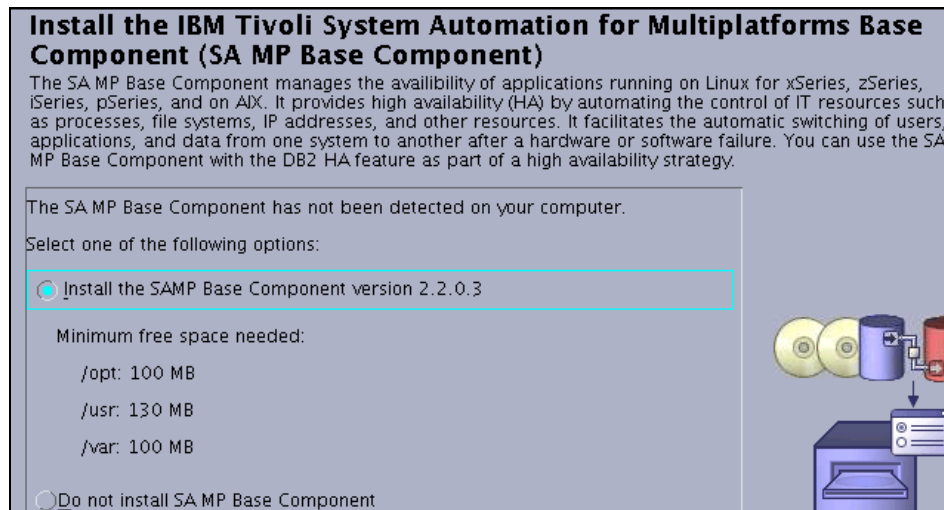


Figure 2-8 Root installation - Install TSA

10. Set user information for the *DB2 Administration Server*, as shown in Figure 2-9 on page 68. This user administers the Database Administration Server.
11. On the *Set up a DB2 instance* windows, select **Create a DB2 instance**. Click **Next**.
12. On the *Set up partitioning options for the DB2 instance* window (Figure 2-10), select single partition instance, click **Next**.

Set user information for the DB2 Administration Server

The DB2 Administration Server (DAS) runs on your computer to provide support required by the DB2 tools. A user with a minimal set of privileges is required to run the DAS. Specify the required user information for the DAS.

☒ **New user**

User name:

UID: ☐ Use default UID

Group name:

GID: ☐ Use default GID

Password:

Confirm password:

Home directory:

☐ **Existing user**

User name:

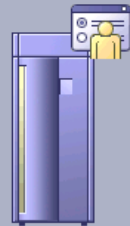


Figure 2-9 Root installation - Set user information for DAS

Set up partitioning options for the DB2 instance

A DB2 instance can have one or more database partitions, which exist on one or more computers. Select the partitioning options for this instance.

☒ **Single partition instance**

The instance will reside only on this computer. Select this option if the instance will not be used in a partitioned database environment.

☐ **Multiple partition instance**

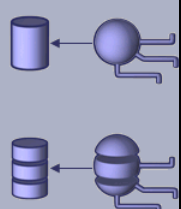
Select this option to prepare to use the partitioning capability of DB2 Enterprise Server Edition Version 9.5 to store data in multiple database partitions. To use this functionality, you must have a Database Partitioning Feature license.

If you select this option, two response files will be saved. See the help for details.

Maximum logical partitions

This computer will be assigned partition number 0. You can specify the maximum number of logical partitions that can exist on each database partition server. This setting applies to all database partition servers in this instance.

Maximum logical partitions: [TCP/IP Settings](#)



1. Introduction
2. Software License Agreement
3. Installation type
4. Installation action
5. Installation directory
6. SA MP Base Component
7. Instance setup
8. Partitioning options
9. Instance-owning user
10. Fenced user
11. Tools catalog
12. Contact
13. Summary

◀ Back Next ▶ Finish Cancel Help

Figure 2-10 Root installation - Set up partitioning options for the DB2 instance

13. On *Set user information for the DB2 instance owner* window (Figure 2-11), set user information for the DB2 instance owner. By default, DB2 Setup creates a new user db2inst1 in group db2adm1. You may change user and group names, or configure an existing user to the new DB2 V9.5 instance by selecting the Existing user option. Note that the default home directory is in /home. We recommend that you change the instance home directory to a DB2-specific directory, such as /db2home.

Figure 2-11 Root installation - Set user information for the DB2 instance owner

14. *Set user info for the fenced user.* This user is responsible for executing fenced user defined functions, such as UDFs and stored procedures. Once again, note the home directory location. We recommend that you change the fenced user home directory to a DB2-specific directory, such as /db2home
15. On the *Prepare the DB2 tools catalog* window, accept default option. Do not prepare the DB2 tools catalog, and click **Next**.
16. *Set up notifications* (Figure 2-12 on page 70). If you choose **Set up your DB2 server to send notifications**, specify a Notification SMTP server here, then click **Next**.

Set up notifications

You can set up your DB2 server to automatically send e-mail or pager notifications to alert administrators when a database needs attention. The contact information is stored in the administration contact list. You need an unauthenticated SMTP server to send these notifications.

☒ Set up your DB2 server to send notifications

Notification SMTP server

Administration contact list location

☒ Local - Create a contact list on this computer

☐ Remote - Use an existing contact list that resides on another DB2 server

Remote DB2 server

☐ Do not set up your DB2 server to send notifications at this time

If you do not set up your DB2 server to send notifications, the health alerts are still recorded in the administration notification log.

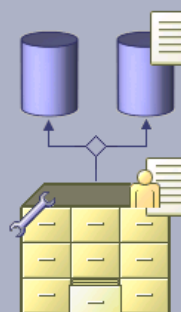


Figure 2-12 Root installation - Set up notifications

17. On *Specify a contact for health monitor notification* window (Figure 2-13 on page 71), you can add Administration contact for this instance, then click **Next**. By default, a health monitor runs on the DB2 instance you are setting up. The DB2 health monitor will send a notification e-mail to this person at the specified mail address when a health indicator threshold is reached. If you check off Address is for a pager, the notification message will be sent to the contact's pager.

Note: If you choose **Defer this task until after installation is complete**, you can specify contacts after installation using the Task Center.

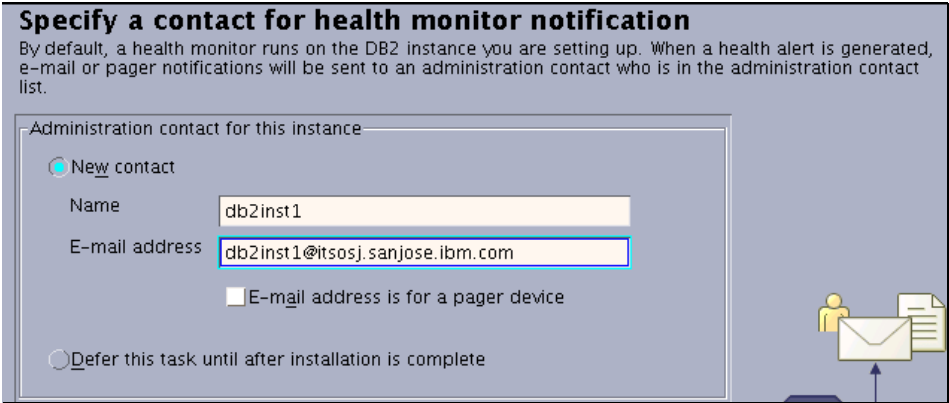


Figure 2-13 Root Installation - Specify a contact for health monitor notification

18. *Start copying files and create response files.* This window provides a summary of your installation and configuration settings as shown in . Scroll through this window to verify that your settings are correct, then click **Finish**.

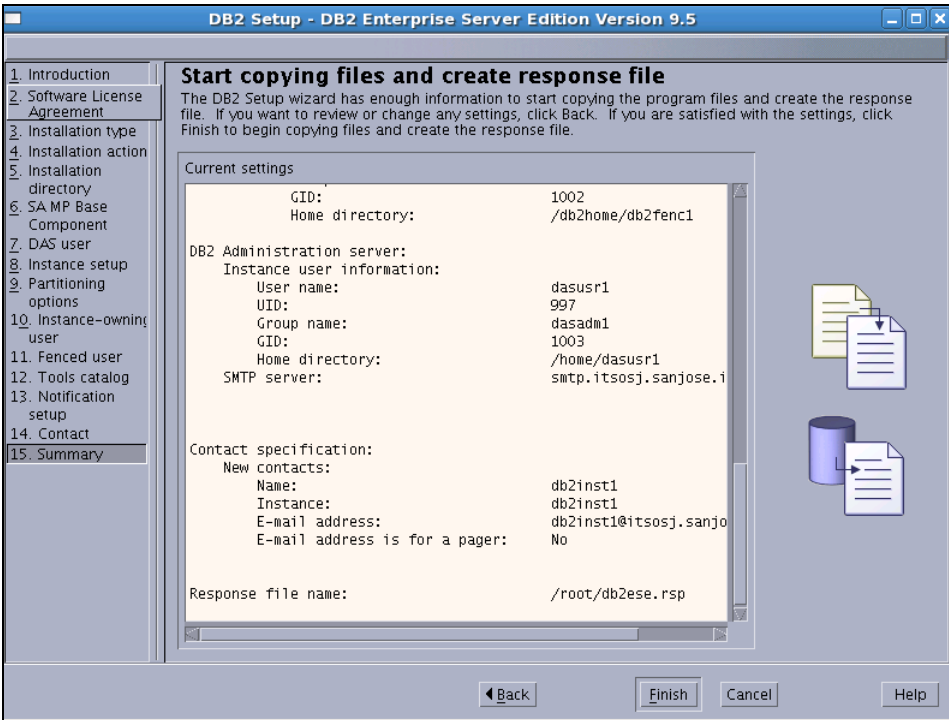


Figure 2-14 Root installation - Installation summary window

19. After installation, read the status report tab and check for any errors. If all is well, click **Finish**.

Note: Install logs are located in /tmp/db2setup.log, and /tmp/db2setup.err.

Response file install

For root installation, you also can use a response file to install DB2 as discussed in “Response file install” on page 62. The response file can be generated by DB2 Setup Wizard or by editing the samples provided by DB2.

Creating a response file using the sample response file

You can create the response file manually by editing the templates provided. The response files have a .rsp extension and are located in the /mnt/dvd/db2/linux/samples or /mnt/dvd/db2/linuxamd64/samples directory on the DB2 product DVD. Where, /mnt/dvd represents the mount point of the CD or DVD. For example, you can find a sample response file for DB2 Enterprise Server Edition in /mnt/dvd/db2/linuxamd64/samples directory, named *db2ese.rsp*.

To create your own response file from the sample, you must copy the sample response file to a local file system and edit it. Activate items in the response file by removing the asterisk (*) to the left of the keyword, then overwrite the default setting with the new setting. The range of possible settings is listed to the right of the equal sign. Refer to the *Quick Beginnings for DB2 Servers*, GC23-5864-00 for guidance on configuring these parameters.

Your custom response file is what you make of it. To give you an idea of the final product, Example 2-25 shows a response file, db2ese.rsp, generated by the DB2 Setup wizard.

Example 2-25 Sample response file db2ese.rsp

```
*-----
* Generated response file used by the DB2 Setup wizard
* generation time: 2/7/08 11:15 AM
*-----
* Product Installation
LIC_AGREEMENT      = ACCEPT
PROD               = ENTERPRISE_SERVER_EDITION
FILE               = /opt/ibm/db2/V9.5
INSTALL_TYPE       = TYPICAL
*-----
* Das properties
*-----
DAS_CONTACT_LIST    = LOCAL
DAS_SMTP_SERVER     = smtp.itsosj.sanjose.ibm.com
```

```

* DAS user
DAS_USERNAME      = dasusr1
DAS_UID           = 997
DAS_GID           = 1003
DAS_GROUP_NAME    = dasadm1
DAS_HOME_DIRECTORY = /home/dasusr1
DAS_PASSWORD      =
3463322748282905151779356453226535001322719040555913873342519614033763535200416
25453423
ENCRYPTED          = DAS_PASSWORD
* -----
* Instance properties
* -----
INSTANCE          = inst1
inst1.TYPE        = ese
* Instance-owning user
inst1.NAME         = db2inst1
inst1.UID          = 999
inst1.GID          = 1001
inst1.GROUP_NAME   = db2iadml
inst1.HOME_DIRECTORY = /db2home/db2inst1
inst1.PASSWORD     =
3463322748282905151779356453226535001322719040555913873342519614033763535200416
25453423
ENCRYPTED          = inst1.PASSWORD
inst1.AUTOSTART    = YES
inst1.SVCENAME     = db2c_db2inst1
inst1.PORT_NUMBER  = 50001
inst1.FCM_PORT_NUMBER = 60001
inst1.MAX_LOGICAL_NODES = 4
* Fenced user
inst1.FENCED_USERNAME = db2fenc1
inst1.FENCED_UID      = 998
inst1.FENCED_GID      = 1002
inst1.FENCED_GROUP_NAME = db2fadm1
inst1.FENCED_HOME_DIRECTORY = /db2home/db2fenc1
inst1.FENCED_PASSWORD =
7234413608936355365120643490665128227774547854466545365132477194452032722963741
33305018
ENCRYPTED          = inst1.FENCED_PASSWORD
* Contact properties
CONTACT           = contact1
contact1.CONTACT_NAME = db2inst1
contact1.EMAIL      = db2inst1@itsosj.sanjose.ibm.com
contact1.PAGER       = false
contact1.NEW_CONTACT = YES
contact1.INSTANCE    = inst1
* -----
* Installed Languages

```

```

*-----
LANG      = EN
*-----
*  SA MP Base Component
*-----
INSTALL_TSAMP = YES

```

Installing DB2 using the response file

To install DB2 using the response file, use **db2setup** command with **-r** option.
For example:

```
./db2setup -r /db2home/db2inst1/db2ese.rsp
```

db2_install utility

For root installation, the db2_install utility installs the DB2 filesets, but does not create an instance, users, or perform any other configuration tasks performed by the DB2 Setup wizard. Some people prefer to use db2_install when installing DB2 on a large, complex database system that has special requirements.

Installing DB2

1. Log in as root.
2. Change to the directory where the DB2 product DVD is mounted by entering the following command:

```
cd /mnt/dvd
```

where /mnt/dvd represents the mount point of the DVD.

3. Enter the **db2_install** command to start the db2_install script.
4. When db2_install prompts you for installation path, enter **no** if you do not want change the default directory.

The default installation directory for DB2 copy on Linux is */opt/ibm/db2/V9.5*.

5. When db2_install prompts you for the product keyword, enter in ESE.

Post-installation tasks

After installing DB2 using db2_install, do the following:

- ▶ Create group and user IDs (if not already done).
- ▶ Create a DB2 Administration Server (DAS).
- ▶ Create a DB2 instance.
- ▶ Create links for DB2 file (optional).
- ▶ Configure TCP/IP communications for the DB2 instance.

Create group and user IDs

Refer to 2.2.6, “User and group setup” on page 49.

Create a DB2 Administration Server (DAS)

The DAS is required if you plan to use the DB2 graphical tools, such as the Control Center and Task Center. You need to have a DAS user created before creating the DAS.

Here is the procedure:

1. Log in as root.
2. Issue the following command to create the DAS:

```
DB2DIR/instance/dascrt -u DASuser
```

Where DB2DIR is the instance directory where the DB2 copy is installed. For the **-u** parameter, enter the DAS user that you created for this machine. For example,

```
/opt/ibm/db2/V9.5/dascrt -u dasusr1
```

Create a DB2 instance

Use the **db2icrt** command to create a new instance:

1. Log in as root.
2. Enter the following command:

```
DB2DIR/instance/db2icrt [-a AuthType] -u FencedID InstNme
```

Where:

- DB2DIR is the location where the current version of the DB2 database product is installed.
- *AuthType* represents the authentication type (SERVER, CLIENT, or SERVER_ENCRYPT) for the instance. If you do not specify this parameter, the default authentication type, SERVER, is assigned.
- *FencedID* represents the name of the DB2 fenced user.
- *InstNme* represents the name of the instance you are creating. The name of the instance must be the same as the name of the instance owning user. This instance will be created in the home directory of the instance owning user.

For example, we enter in:

```
/opt/ibm/db2/V9.5/instance/db2icrt -u db2fenc1 db2inst1
```

Create links for DB2 files (optional)

If you are developing or running applications, you may want to create links for the DB2 files to the /usr/lib directory, and for the include files to the /usr/include directory. This will avoid having to specify the full path to the product libraries and include files. If you create links on one version of DB2, the links on the other version will be overwritten.

To create links:

1. Log in as root.
2. Enter the following command:

```
/opt/ibm/db2/v9.5/cfg/db2ln
```

Assume that DB2 is installed to the default installation directory.

Note: If multiple versions or copies of DB2 co-exist, we recommend that you do not use **db2ln** command to create links for DB2 files. Consider using **db2rm1n** command from a DB2 version or a copy to remove existing links.

Configure TCP/IP for the DB2 instance

You must configure your instance with the TCP/IP protocol in order for it to accept requests from remote DB2 clients.

Follow these steps:

1. Update the `/etc/services` file to specify the service name and port number that the DB2 server will listen on for client requests. To update the `/etc/services` file, use a text editor to add a connection entry. For example, we added:

```
db2c_db2inst1 50001/tcp # DB2 connection service port
```

Where:

- *db2c_db2inst1* represents the connection service name
- *50001* represents the connection port number
- *tcp* represents the TCP/IP communication protocol

The service name is arbitrary but must be unique in the services file. In a partitioned environment, make sure that the port number does not conflict with the port numbers used by the Fast Communications Manager (FCM) or any other applications on the system.

2. Update the database manager configuration file on the server. To do so:
 - a. Log in as instance owner.
 - b. Update the SVCENAME parameter in the DBM configuration file. You may specify either the service name or port number. For example, we enter:

```
db2 update dbm cfg using SVCENAME db2c_db2inst1
```

You can check your SVCENAME by entering:

```
db2 get dbm cfg | grep SVC
```

3. Set the DB2COMM registry variable to `tcp/ip`. This will start the DB2 communications manager when the database manager is started. Enter the following command as instance owner:


```
db2set DB2COMM=tcPIP
```

4. Stop and re-start the instance for these changes to take effect:

```
db2stop
db2start
```

5. Use **netstat** utility to verify if TCP/IP communication port is listening, for example, in our sample, we enter:

```
netstat -an | grep 50001
```

2.3.3 Installing a partitioned database environment

There are several ways in which you can install DB2 on a multi-partition environment:

- ▶ Use the DB2 Setup wizard to install DB2 and create a new instance on the primary machine, then use a response file to install DB2 on the remaining machines participating in the partitioned database.
- ▶ Edit a sample response file and use this file to install DB2 on each machine in the cluster.
- ▶ Use `db2_install` or a combination of DB2 Setup wizard and `db2_install` to set up your environment.

The easiest method is to use the DB2 Setup wizard on the primary machine, and then a response file on the remaining machines participating in the partitioned database system. We discuss this method in this book.

Preparing the environment

The first step of installing a partitioned database environment is preparing the environment, including setting up NFS, creating user and group and enabling ssh. Refer to 2.2, “Installation considerations and planning” on page 35.

Using DB2 Setup wizard to install DB2 on primary machine

When the environment is ready, use DB2 Setup wizard to install DB2 on the primary machine. In our sample, *mensa* is the primary machine of the partitioned database environment.

1. On the primary machine, log in as root.
2. Go to the directory where the DB2 install image is located and run **db2setup**, select **Install a Product** on the first window.
3. Select **Install New** on *Welcome to the DB2 Setup Wizard* window, click **Next**.
4. On *Software License Agreement* window, read the license agreement and then select **Accept** and click **Next**.

5. *Select the installation type.* You may choose a Typical, Compact, or Custom installation.
 - ▶ *Typical* installs most DB2 components except for the *Base application development tools*. Click the View Features button to see what components will get installed.
 - ▶ *Compact* installs only basic DB2 features and functionality.
 - ▶ *Custom* allows you install whatever components you want. An advantage of the Custom option is that it allows you to install the Base application development tools with DB2. With the Typical option, the Base application development requires a separate installation.
6. On the *Select the Installation, response file creation, or both* window (Figure 2-15), select **Install DB2 and save your settings in a response file**. We rename the response file to `db2ese_mpp.rsp`. Then, click **Next**.

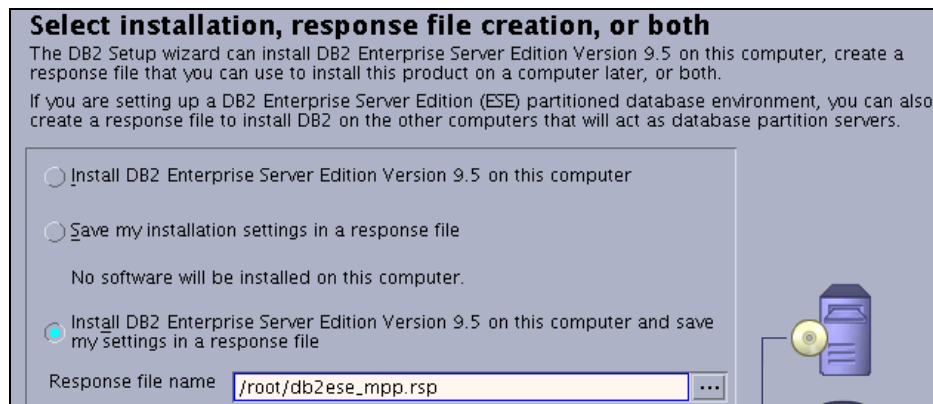


Figure 2-15 MPP Installation - Select installation, response file creation, or both

7. On the next windows, you can choose to install IBM TSA or not, then click **Next**.
8. Set user information for the DB2 Administration Server, as shown in Figure 2-16 on page 79. This user administers the Database Administration Server. Choose existing user *dasusr1*, click **Next**.

Set user information for the DB2 Administration Server

The DB2 Administration Server (DAS) runs on your computer to provide support required by the DB2 tools. A user with a minimal set of privileges is required to run the DAS. Specify the required user information for the DAS.

☐ **New user**

User name:

UID: ☒ Use default UID

Group name:

GID: ☒ Use default GID

Password:

Confirm password:

Home directory: ...

☐ **Existing user**

User name: ...

Figure 2-16 MPP Installation - Set up Information for DAS

9. On the *Set up a DB2 instance* windows, select **Create a DB2 instance**, then click **Next**.
10. On the *Set up partitioning options for the DB2 instance* window, as shown in Figure 2-17 on page 80, select **Multiple partition instance** and specify the number of maximum logical partitions. By default the number of maximum logical partitions is 4. Local services entries will be added in `/etc/services` after this installation and the entries should appear similar to Example 2-26, DB2 reserves four available ports after 60000 for `db2inst1`.

Example 2-26 Local services entries in `/etc/services`

```
.....
DB2_db2inst1    60000/tcp
DB2_db2inst1_1  60001/tcp
DB2_db2inst1_2  60002/tcp
DB2_db2inst1_END    60003/tcp
.....
```

When finished, click **Next**.

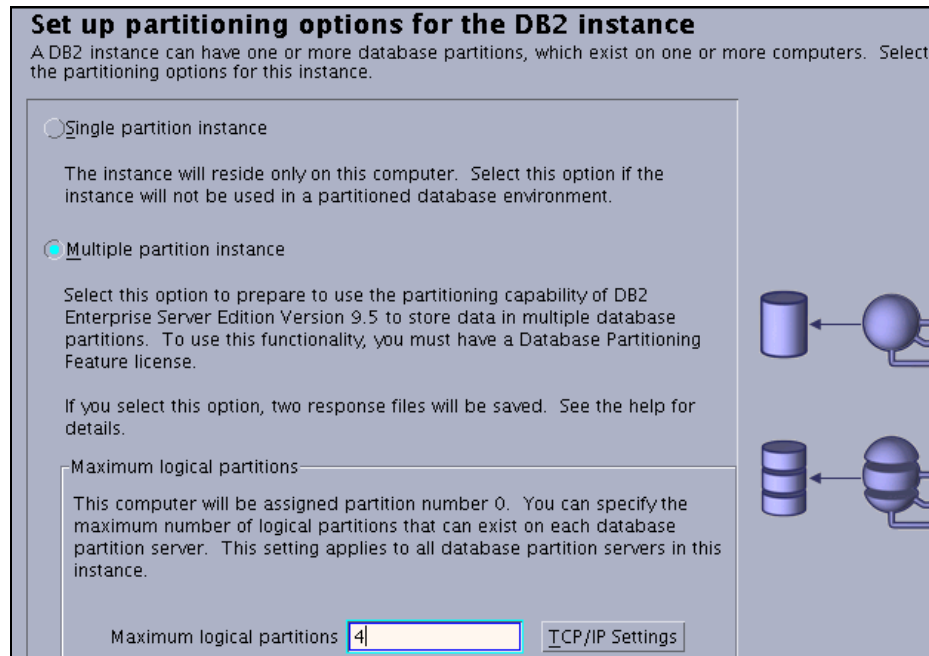


Figure 2-17 MPP Installation - Set up partitioning options for the DB2 Instance

11. On *Set user information for the DB2 instance owner* window, set user information for the DB2 instance owner. Choose existing user *db2inst1*. Note that, in our sample, *db2inst1*'s home directory is created on */db2home* which is NFS shared across the partitioned servers. Click **Next**.
12. Set user information for the fenced user, choose existing user *db2fenc1*. This user is responsible for executing fenced user defined functions, such as UDFs and stored procedures. Once again, please note the home directory location is created on */db2home*.
13. On the *Prepare the DB2 tools catalog* window, accept default option **Do not prepare the DB2 tools catalog**, and click **Next**.
14. Set up notifications. If you choose **Set up your DB2 server to send notifications**, specify a Notification SMTP server here, then click **Next**.
15. On *Specify a contact for health monitor notification* window, you can add Administration contact for this instance, then click **Next**. By default, a health monitor runs on the DB2 instance you are setting up. The DB2 health monitor will send a notification E-mail to this person at the specified mail address when a health indicator threshold is reached. If you check off **Address is for a pager**, the notification message will be sent to the contact's pager.

16. *Start copying files and create response files.* This window provides a summary of your installation and configuration settings. Scroll through this window to verify that your settings are correct, then click **Finish**.
17. After installation, read the status report tab and check for any errors. If all is well, click **Finish**.

Note: Install logs are located in /tmp/db2setup.log, and /tmp/db2setup.err.

18. Since we selected **Create a DB2 instance** in our sample and we created instance owner db2inst1 who uses Korn Shell (ksh) before running DB2 Setup Wizard, check if the Korn Shell .profile file contains the following lines:

```
# The following three lines have been added by IBM DB2 instance utilities.
if [ -f /db2home/db2inst1/sqllib/db2profile ]; then
    . /db2home/db2inst1/sqllib/db2profile
fi
```

Using response file to install DB2 on the remaining machines

After performing a partitioned installation on the primary machine, the DB2 Setup wizard creates two response files: *db2ese_mpp.rsp* and *db2ese_mpp_addpart.rsp*. The *db2ese_mpp.rsp* response file contains instructions to create an instance, instance-owning and fenced users, as well as the tools catalog database (if selected during install). The second response file only installs the appropriate DB2 filesets, specifies DAS properties, and specifies languages to install. Only use the *db2ese_mpp_addpart.rsp* response file when installing database partition servers on the participating computers. Example 2-27 shows the contents of the *db2ese_mpp_addpart.rsp* in our sample.

Example 2-27 The db2ese_mpp_addpart.rsp file

```
*-----
* Generated response file used by the DB2 Setup wizard
* generation time: 2/7/08 3:28 PM
*-----
* Product Installation
LIC_AGREEMENT      = ACCEPT
PROD               = ENTERPRISE_SERVER_EDITION
FILE               = /opt/ibm/db2/V9.5
INSTALL_TYPE       = TYPICAL
*-----
* Das properties
*-----
DAS_CONTACT_LIST    = REMOTE
DAS_CONTACT_LIST_HOSTNAME = mensa.itsosj.sanjose.ibm.com
DAS_SMTP_SERVER     = smtp.itsosj.sanjose.ibm.com
```

```

* DAS user
DAS_USERNAME      = dasusr1
DAS_UID           = 1003
DAS_GROUP_NAME    = dasadm1
DAS_HOME_DIRECTORY = /home/dasusr1
*-----
*  Installed Languages
*-----
LANG              = EN
*-----
*  SA MP Base Component
*-----
INSTALL_TSAMP = YES

```

Use **db2setup** command with **-r** option to install DB2 using the response file on the remaining machines in the cluster. For example:

1. Log in primary machine as instance owner. In our sample, log in *mensa* as *db2inst1*.
2. Issue **db2stop** command to stop DB2 instance.

db2stop force

3. Remove **sqlllib** directory.

rm -rf ~/sqlllib

4. Log in other machine as root. In our sample, log in *gemini* as root.
5. Transfer the **db2ese_mpp_addpart.rsp** file from the primary machine and place it on the **/root** directory.
6. Mount DB2 product DVD.
7. Run **db2setup** command, for example:

/mnt/dvd/db2setup -r /root/db2ese_mpp_addpart.rsp

where **/mnt/dvd** represents the mount point of the DVD.

After installation has finished, check the installation log **/tmp/db2setup.log** to ensure that no errors have occurred.

8. Copy instance related TCPIP service entries in **/etc/services** file on primary machine to **/etc/services** file on current server. In our sample, add following entries in **/etc/services** on *gemini*:

```

db2c_db2inst1  50001/tcp
DB2_db2inst1   60000/tcp
DB2_db2inst1_1 60001/tcp
DB2_db2inst1_2 60002/tcp
DB2_db2inst1_END 60003/tcp

```

9. Issue following command to create physical node on current server.

```
/opt/ibm/db2/V9.5/instance/db2icrt -p db2c_db2inst1 -u db2fenc1 db2inst1
```

10. If there is other participating DB2 server (physical node) in the partitioned database environment, repeat step 3 to step 9 in this section. In our sample, we do only have two DB2 servers, *mensa* and *gemini*.
11. Modify `~/sqllib/db2nodes.cfg` by adding an entry to the `db2nodes.cfg` file for each participating logical database partition server. When you first view the `db2nodes.cfg` file, it should contain an entry similar to the following:

```
0 ServerA 0
```

This entry includes the database partition server number (node number), the TCP/IP host name of the server where the database partition server resides, and a logical port number for the database partition server.

If you are installing a partitioned database environment with three machines and a logical database partition server on each machine, the updated `db2nodes.cfg` should appear similar to the following:

```
0 ServerA 0
1 ServerB 0
2 ServerC 0
```

If you are installing a partitioned database environment with two machines and two logical database partition servers on each machine, the updated `db2nodes.cfg` should appear similar to the following:

```
0 ServerA 0
1 ServerA 1
2 ServerB 0
3 ServerB 1
```

or

```
0 ServerA 0
1 ServerB 0
2 ServerA 1
3 ServerB 1
```

Example 2-28 shows the content of the `db2nodes.cfg` in our sample.

Example 2-28 db2nodes.cfg

```
db2inst1@mensa:/db2home/db2inst1> more ~/sqllib/db2nodes.cfg
0 mensa 0
1 gemini 0
```

Enabling ssh

Configuring DB2 to use ssh as the remote shell utility

1. Configuring DB2 to use ssh:

To set up DB2 to start with ssh support, you must enable the DB2 registry variable DB2RSHCMD and point it to the path of the ssh command shell. Example 2-29 shows the db2set command. The default is for DB2 to use rsh; therefore, you only need to set this variable if you plan to use ssh.

Example 2-29 Enable DB2 to use ssh

```
$ db2set DB2RSHCMD=/usr/bin/ssh
```

2. To verify ssh configuration in DB2 environment, log on to each machine as the DB2 partitioned instance owning user ID, and enter the commands as shown in Example 2-30.

Example 2-30 Verify ssh configuration in DB2 environment

```
$ ssh mensa echo hi
$ ssh gemini echo hi
$ db2_all echo hi
$ db2_all db2_all echo hi
```

These commands should complete successfully without prompting for additional verification.

Starting/stopping instance manager to verify installation

You can issue db2start command to verify installation now. If all is well, you can use the **db2start/db2stop** commands to start or stop instance manager on primary machine or any participating machine. Example 2-31 shows the output of the **db2start/db2stop** commands on mensa.

Example 2-31 Using db2start/db2stop to verify installation

```
db2inst1@mensa:/db2home/db2inst1> db2start
02/07/2008 16:59:05    0    0    SQL1063N  DB2START processing was successful.
02/07/2008 16:59:06    1    0    SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.
db2inst1@mensa:/db2home/db2inst1> db2stop
02/07/2008 17:05:02    0    0    SQL1064N  DB2STOP processing was successful.
02/07/2008 17:05:02    1    0    SQL1064N  DB2STOP processing was successful.
SQL1064N  DB2STOP processing was successful.
```

2.3.4 Installing DB2 license files

You must install a license key for each DB2 copy. In a partitioned environment, this means that you must install a license key on each machine participating in

the partitioned database system. This is done by adding a DB2 license file using the **db2licm** command.

1. Log in as root user or instance owner. For non-root installation, log in as instance owner.
2. Enter the following command:

```
db2licm -a <filename>
```

Where, *filename* is the filename containing valid license information.

For example, if you want to install the license key with “DB2 Workgroup Server Edition Authorized User Option”, enter the following command:

```
db2licm -a db2wse_u.lic
```

On Linux, the DB2 license keys are located in the *DB2DIR*/license in a file called *node.lock*, where DB2DIR is the instance directory where the DB2 copy is installed.

2.3.5 Installing DB2 documentation

DB2 Version 9.5 provides DB2 Information Center and PDF documentation on separate DVDs. These DVDs are shipped with your client and server DVDs.

- ▶ **DB2 Information Center DVD**
The DB2 Information Center gives you access to all of the information you need to take full advantage of DB2 products. It is installed separately from other DB2 products from its own DVD, which means that you can install DB2 Information Center immediately after installing DB2, or at a later time. You can install the DB2 Information Center on a machine that does not have DB2 installed, such as your company’s internal Web server. This will save space on individual machines.

You can also access the DB2 Information Center from the IBM Web site at:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r5>

- ▶ **PDF documentation DVD**
Unlike the DB2 Information Center, you do not need to ‘install’ the PDF documentation. This CD provides PDF files that you can read directly from the CD, or copy onto your machine. For the latest information about DB2 database product documentation, refer to the Web site:

<http://www-1.ibm.com/support/docview.wss?rs=71&uid=swg27009474>

Installing DB2 Information Center using the DB2 Setup wizard

Following these steps:

1. Log onto system as user with root authority.
2. Insert and mount the DB2 Information Center product DVD.

3. Change to the directory where the DB2 product DVD is mounted by entering the following command:

```
cd /mnt/dvd
```

where /mnt/dvd represents the mount point of the DVD.

4. Enter **./db2setup** to launch the DB2 Setup wizard.
5. Once the launchpad opens, you can view installation prerequisites and the release notes, or you can proceed directly to the installation.
6. Click **Install a Product** and the Install a Product window displays.
7. On the Install a Product page, if you do not have an existing DB2 Information Center installed on your computer, launch the installation by clicking **Install New**.
8. On the Welcome to the DB2 Setup wizard page, click **Next**. The DB2 Setup wizard will guide you through the program setup process.
9. To proceed with the installation, you must accept the license agreement. On the Software License Agreement page, select **Accept** and click **Next**.
10. On the *Select installation, response file creation, or both* window, select **Install DB2 Information Center on this computer and save my settings in a response file** if you want to use a response file to install the DB2 Information Center on this or other computers at a later time. You can specify where the response file will be saved. Click **Next**.
11. On the *Select the languages to install* panel (Figure 2-18) select the languages the DB2 Information Center will install. By default, the DB2 Information Center is installed in the /opt/ibm/db2ic/V9.5 directory. However, you can specify your own installation path. Click **Next**.

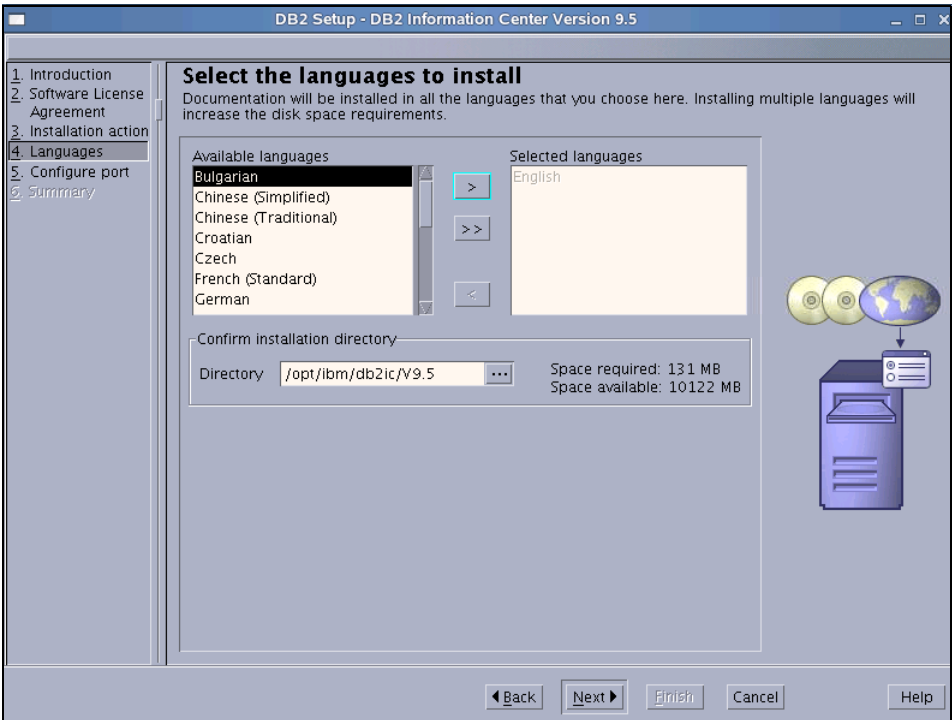


Figure 2-18 Select the languages to install

12.As shown in Figure 2-19, configure the DB2 Information Center for incoming communication on the *Specify the DB2 Information Center port* panel. Click **Next** to continue the installation.

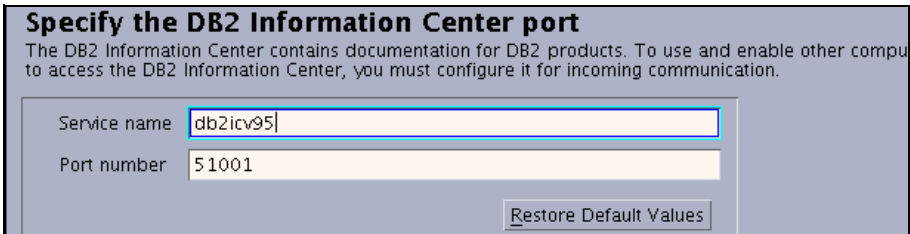


Figure 2-19 Specify the DB2 Information Center port

13.Start copying files. Check **settings**, then click **Finish**.

Installing DB2 Information Center using doce_install command

To install DB2 Information Center manually, use the doce_install command.

1. Log in as root.
2. Decide where you want to install the documentation. In our environment, we created a directory called /documentation which is located in the /db2home directory (for example, /db2home/documentation).
3. Insert and mount the DB2 Information Center DVD.
4. Use **doce_install** command to install DB2 Information Center. The syntax of the doce_install command is as follows:

```
doce_install -b install_path -p doce -n -L language
```

-b *install-path* Specifies the path where the DB2 Information Center is to be installed. The default installation path is /opt/ibm/db2ic/V9.5. This parameter is mandatory when the -n parameter is specified.

-n Specifies non-interactive mode.

-L *language* Specifies national language support. The default is English. To install multiple languages at the same time, this parameter can be specified multiple times. For example, to install both English and Simplified Chinese, specify -L EN -L CN.

For example, enter the **./doce_install** command using the following format:

```
./doce_install -b /db2home/documentation -p doce -n -L EN -L CN
```

Note: The **doce_uninstall** command could be used to uninstall DB2 Information Center.

3



Post installation tasks

In the previous chapter, we provided a detailed process of installing DB2 Version 9.5 on Linux. In this chapter we discuss post-installation tasks that should be performed to have a functional DB2 environment. The following topics are covered:

- ▶ Control Center setup
- ▶ Preparation for database creation
- ▶ Creating a database
- ▶ Further configuration
- ▶ Add a partition
- ▶ DB2 configuration
- ▶ Security
- ▶ Client configuration
- ▶ Configure licensing

3.1 Control Center setup

DB2 has a rich set of graphical tools that are used for database administrators to manage the database system and for application developers to develop stored procedures and user defined functions. The *Control Center* is one of the general administration tools that is used to help you explore and administer your databases. Other such tools include the *Command Center*, which can be used to generate SQL queries, and the *Configuration Assistant*, which is an excellent tool for configuring your database environment.

If you want to administer your database server with the Control Center, you must start the DB2 Administrator Server on each host of your database server environment. The DB2 Administrator service setup procedure is provided in Chapter 2, “Installation” on page 25. To start the DB2 Administrator Server, log on to every host as DAS user and issue **db2admin start**.

If you are logged in as instance owner or privileged DB2 user and have an X-Window started, you can use the **db2cc &** command to start Control Center. If you started your X-Window as the root user, you need to switch to the DB2 instance owner ID and set up the display before you can start the Control Center. You have two ways to set up the environment for running Control Center. Example 3-1 shows the simpler way.

Example 3-1 Start Control Center (method 1)

```
puget:~ # xhost +
puget:~ # su - db2inst1
db2inst1@puget:~> export DISPLAY=:0.0
db2inst1@puget:~> db2cc&
[1] 6051
db2inst1@puget:~>
```

The & sign after the command returns the shell prompt, which allows you to use this shell for other commands. Method 1 is not suggested for security critical production systems because it allows everybody to access your X server. The more secure method would be to use *Xauthority* as demonstrated in Example 3-2.

Example 3-2 Start Control Center (method 2)

```
puget:~ # su - db2inst1
db2inst1@puget:~> scp root@localhost:.Xauthority /tmp/xauth.tmp
Password:
.Xauthority                                100% 161      0.2KB/s   00:00
db2inst1@puget:~> export DISPLAY=:0.0
db2inst1@puget:~> xauth merge /tmp/xauth.tmp
```

```
db2inst1@puget:~> db2cc&
[1] 6052
db2inst1@puget:~>
```

In method 2, we copy the file `.Xauthority` from the server (root user) to the client (db2inst1 user) and merge it with the existing one. We chose the `scp` command for convenience. That allows the user db2inst1 to access the root owned X server. We also set the `DISPLAY` to the same host and use the ampersand sign `&` to run the command in the background.

By using method 2 only the user who uses the `.Xauthority` file is allowed to access the X server.

If you connect from a remote client the steps for method 1 or method 2 could be adjusted accordingly. However, in that case it would be easier to use the `ssh` command as demonstrated in Example 3-3.

Example 3-3 Use SSH for db2cc

```
db2inst1@vmsuse:~> ssh -X db2inst1@puget
Last login: Thu Jan 31 16:55:09 2008 from 9.125.7.139
db2inst1@puget:~> db2cc&
[1] 28832
db2inst1@puget:~>
```

The option `-X` instructs `ssh` to tunnel the X11 port to the local system, set the `DISPLAY` correctly, and to perform the Xauthority authentication. If you have set up the SSH authentication properly, you even do not need to provide a password.

If you come from a Windows system and you use Exceed or Cygwin as a local X server, you can use *putty*. The **putty** program has an option which allows **X11 forwarding**.

After you start the Control Center, you are asked first to specify the view options. See Figure 3-1 on page 92. Select **Advanced**. In this chapter, we provide our demonstrations in that view.

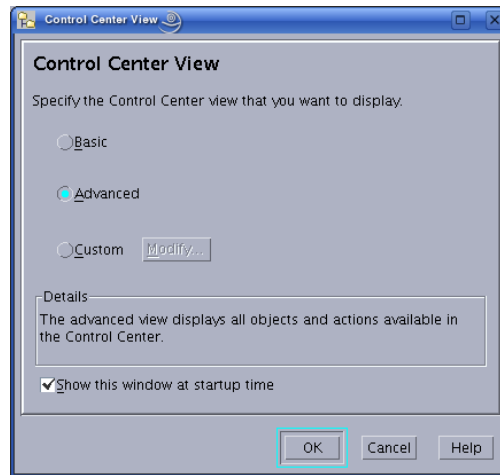


Figure 3-1 Choose the view in db2cc

Figure 3-2 shows the main window of Control Center. From there you can view all the cataloged systems and databases.

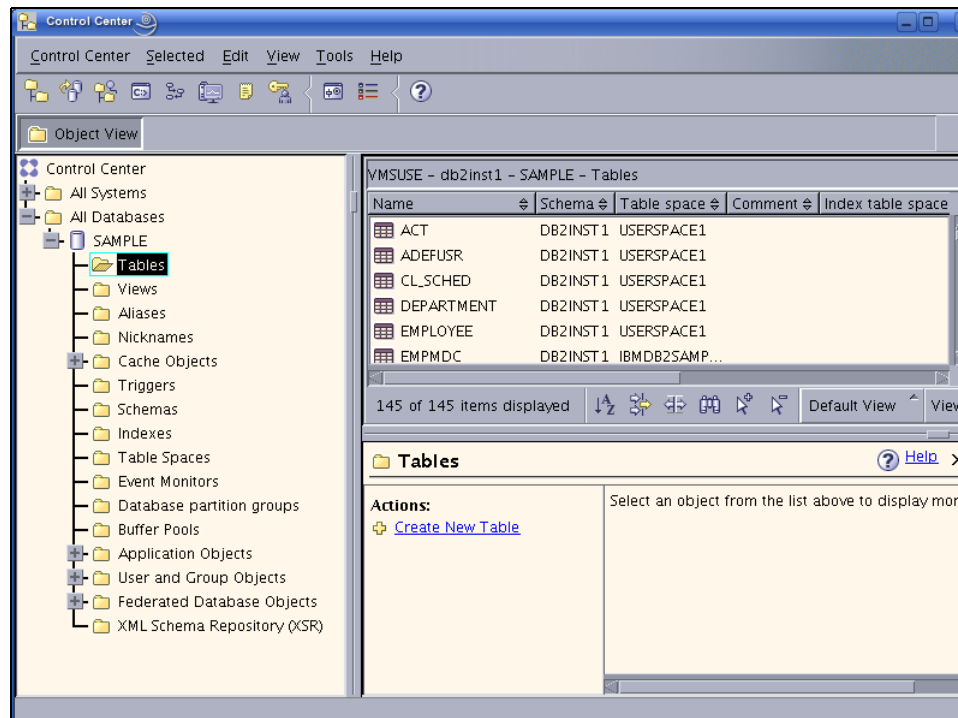


Figure 3-2 Main screen of the Control Center

3.2 Preparation for the database creation

There are a few configuration tasks that need to be done on Linux and DB2 before creating the database. Here we discuss:

- ▶ Code page considerations
- ▶ Table space consideration
- ▶ Linux specific configuration
- ▶ Multi-partition specific configuration

3.2.1 Code page considerations

In DB2 9.5, the default database code page is UTF-8. The code page is specified during database creation time. If you want to change the code page afterwards, the database would have to be recreated.

Figure 3-3 shows an example architecture. We use this architecture to discuss the code page conversion taken place between database and application.

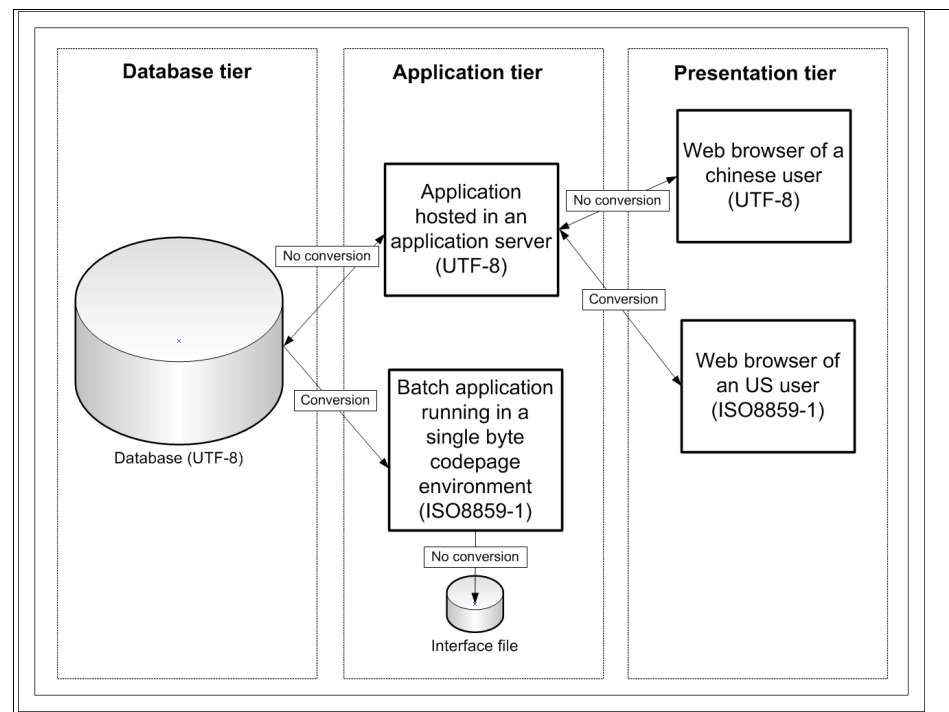


Figure 3-3 Code page conversion scenario

This is a typical application architecture with one database which is accessed by a J2EE™ application from an application server and a batch application which extracts data. The J2EE application is accessed by the users using Web browsers.

The database is created in UTF-8 format. The J2EE application also uses the data in UTF-8 format. The batch application runs on a single byte environment and converts the data into ISO8859-1 format.

Since this is a worldwide application, the user community uses different code pages setup in their Web browsers.

DB2 automatically converts the code page between the client and the server. For applications, this conversion probably needs to be implemented into the code. In Figure 3-3 on page 93, we note where a code page conversion of the transferred data is necessary. Dependent on your architecture you can have many places where a code page conversion may be necessary.

You have to decide before the database creation what code page you want to use for your database because it can not be changed easily afterwards. A later change would require recreating database and a data migration using **db2look** and **db2move**.

UTF-8 offers support for all languages and would be the best choice if you have to provide worldwide access to your application. If you support only your local country and your country uses a single byte code page, you probably want to choose a single byte code page for your database. A database created for a single byte code page requires less storage.

If you have to support several countries with single byte code pages which can be converted between each other, you may want to choose a single code page for your database. It depends on the number of conversions you have to perform in your architecture.

If you use pureXML or if you have UTF-8 clients, you should setup your database as UTF-8. The simplest approach however would be if your whole architecture was setup for UTF-8.

3.2.2 Table space considerations

In this section we describe the different types of table space containers and when it is appropriate to use them.

General table spaces considerations

By default, DB2 creates three table spaces. The *catalog table space* which stores the catalog tables, the *user table space* which stores all table and index data, and the *temporary table space* which is used for storing temporary data while SQL statements are being processed.

However, it makes sense to split the data into more table spaces and to create your own layout. Here are the main reasons for doing that:

- ▶ The caching of the data in the table spaces can be tuned by the related *buffer pools*. Buffer pools are caching areas and are assigned to one or more table spaces. Having different table spaces allows you to tune the caching areas separately for them. You can separate rarely used tables from heavily used ones.

You can also tune the I/O performance of a table space by putting the containers of one table space on different physical disks.

- ▶ You can perform backup on table space level. Separate table spaces allow you to implement different backup strategies.
- ▶ You can grant the permission to create objects to users in table spaces. That allows you to implement more granular security control.
- ▶ At the table creation you can specify that table data should go into one table space, index data into another, and also long data into another one. It is especially recommended to put long data into a different table space because they are not cached in the buffer pool and require file system caching. Once you have defined the table spaces for a table you can change that only by recreating the table.
- ▶ Storage usage of a table is limited within a table spaces. If one table in a table space taken up the entire disk space of a table space, tables in other table spaces will not be affected.

Defining more table spaces requires more design, implementation, and maintenance work. How many table spaces should be defined and how the table spaces should be placed on the disks are depends on the complexity and requirements of the application.

SMS table spaces

System Managed table spaces are handled by the underlying operating system which is Linux in our case. At the creation of the table space you specify directory names as containers. Here some considerations for choosing SMS table space:

- ▶ The disk space is not allocated until it is used.
- ▶ It is easy to increase the SMS table spaces. You only have to increase the underlying file system space. DB2 will use it automatically.

- ▶ Because of the above 2 reasons it is a good practice to use a SMS table space for the temporary table space.
- ▶ You cannot add or delete containers. This can be an issue if you want to enhance the parallel I/O performance by adding another container on a separate physical disk.
- ▶ You may reach file system related limits with the file size of the files created by the database in the SMS managed file systems.
- ▶ The table space is full if *one* of the containers is full.
- ▶ Creating an SMS table space requires less initial work, because you do not have to predefine the container sizes.
- ▶ You cannot split table, index, and long data into separate table spaces for tables which reside within an SMS table space.

DMS table spaces

Database Managed Table spaces are managed by DB2. At the creation of the table space you specify the container files or raw devices and their sizes. Although the size is defined at the table space creation, it can be extended or reduced later on. DB2 can also extend table spaces automatically, as described in “Automatic resizing of table spaces” on page 96.

- ▶ You can split index, data, and long data into separate table spaces for tables which reside in DMS table spaces.
- ▶ The size of a table space can be increased by adding or extending containers, using the **ALTER TABLE SPACE** statement. Existing data can be automatically rebalanced across the new set of containers to retain optimal I/O efficiency.
- ▶ DMS table spaces required more administrative overhead. Work has to be performed on both, Linux and DB2 levels, when creating or extending DMS table spaces.
- ▶ In general, a well-tuned set of DMS table spaces will outperform SMS table spaces.

Automatic resizing of table spaces

DMS table spaces can be created with the option **AUTORESIZE YES** in order to enable the automatic resize feature. By default this option is set to **NO**. You can turn this feature on and off after you have created the table space.

Table spaces with this feature enabled will increase their size automatically up to the file system limit as the space is required. You can specify the initial size, the maximum size, and the increase size.

Raw devices

You can create DMS table space containers on block devices directly. This approach offers the superior performance but also requires more administration.

Note: Prior to Version 9, direct disk access using a *raw controller* utility on Linux was used. This method is now deprecated, and its use is discouraged. The database manager will still allow you to use this method if the Linux operating system still supports it, however, there will be a message in the db2diag.log indicating that its use is deprecated.

DB2 also allows you to create normal SMS and DMS table spaces in the file system with the option **NO FILE SYSTEM CACHING**. The term “normal” means that they are created in the file system and not as raw devices. This option, same as for raw devices, will bypass the double buffering caused by the file system buffer and the buffer pools.

However, device files still have a little performance advantage over table spaces created with the **NO FILE SYSTEM CACHING** option. The reason is that they do not have any file system administration overhead. They also offer a little bit better availability as they are not dependent on file system failures.

Since long data bypasses the buffer pools, we do not recommended placing long table spaces in raw devices. For the same reason, it is not recommended to place the catalog table space on raw devices.

Now we demonstrate how to create a table space on a block device.

In Example 3-4 we use the **fdisk** command to create new disk partition on an empty disk with the name sdb. We then create one new extended disk partition called sdb1. After that, we create a logical disk partition called sdb5 within the extended disk partition sdb1. Note, under Linux, extended disk partitions are container which can contain several logical disk partitions.

Example 3-4 Use fdisk to create a block device

```
puget:~ # fdisk /dev/sdb
```

```
Command (m for help): p
```

```
Disk /dev/sdb: 214 MB, 214748160 bytes
64 heads, 32 sectors/track, 204 cylinders
Units = cylinders of 2048 * 512 = 1048576 bytes
```

Device	Boot	Start	End	Blocks	Id	System
--------	------	-------	-----	--------	----	--------

```
Command (m for help): n
```

```

Command action
  e   extended
  p   primary partition (1-4)
e
Partition number (1-4): 1
First cylinder (1-204, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-204, default 204):
Using default value 204

```

```

Command (m for help): p

```

```

Disk /dev/sdb: 214 MB, 214748160 bytes
64 heads, 32 sectors/track, 204 cylinders
Units = cylinders of 2048 * 512 = 1048576 bytes

```

Device	Boot	Start	End	Blocks	Id	System
/dev/sdb1		1	204	208880	5	Extended

```

Command (m for help): n
Command action
  l   logical (5 or over)
  p   primary partition (1-4)
l
First cylinder (1-204, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-204, default 204):
Using default value 204

```

```

Command (m for help): p

```

```

Disk /dev/sdb: 214 MB, 214748160 bytes
64 heads, 32 sectors/track, 204 cylinders
Units = cylinders of 2048 * 512 = 1048576 bytes

```

Device	Boot	Start	End	Blocks	Id	System
/dev/sdb1		1	204	208880	5	Extended
/dev/sdb5		1	204	208864	83	Linux

```

Command (m for help): w
The partition table has been altered!

```

```

Calling ioctl() to re-read partition table.
Syncing disks.

```

The instance owner has to have the write access to the device file for the disk partition. This can be done by executing the following command from the root user:

```
chown <instance user>.<instance group> /dev/sdb5
```

Now we can use this disk partition for the table space. First we need to calculate the size which we have to specify at the **CREATE TABLESPACE** command by using this formula:

```
num_pages = floor( (blocks * 1024) / 4096 )
```

On our example that will be

```
num_pages = floor( (208864 * 1024) / 4096 ) = 52216
```

This number can now be used to create the table space:

```
CREATE TABLE SPACE dms1 MANAGED BY DATABASE USING (DEVICE '/dev/sdb5' 52216)
```

Example 3-5 shows the result.

Example 3-5 List table space container for a raw device

```
db2inst1@puget:~> db2 list tablespace containers for 3 show detail
```

Table Space Containers for Table Space 3	
Container ID	= 0
Name	= /dev/sdb5
Type	= Disk
Total pages	= 52216
Useable pages	= 52160
Accessible	= Yes

If you create your table space with a size which smaller than the highest possible one for that device, you will see a message in the file db2diag.log as shown in Example 3-6.

Example 3-6 db2diag.log message if table space is too small

```
2008-02-11-11.07.25.876502-480 E288662G688      LEVEL: Warning
PID      : 4185                TID   : 2976902048  PROC  : db2sysc 0
INSTANCE: db2inst1            NODE   : 000          DB    : ITSODB
APPHDL   : 0-21               APPID: *LOCAL.db2inst1.080211185714
AUTHID   : DB2INST1
EDUID    : 25                 EDUNAME: db2agent (ITSODB) 0
FUNCTION: DB2 UDB, buffer pool services, sqlbDMSAddContainerRequest, probe:105
MESSAGE  : ADM6037W Container "/dev/sdb5" was created to be "208840" KB in size
           on a device that is "208864" KB in size. Extra storage will be
           wasted. The container can be extended to use the wasted space by
           using ALTER TABLE SPACE.
```

Automatic storage for databases

Automatic storage is a new feature introduced in DB2 9.1. It simplifies storage management for table spaces. When you create a database, you specify the storage paths where the database manager will place your table space data. The database manager will manage the container and space allocation for the table spaces as you create and populate them.

Databases are created with automatic storage *on* by default. If you do not want to enable it, create the database with the option **AUTOMATIC STORAGE NO**.

Note: You cannot change the automatic storage setting after a database was created.

For an automatic storage enabled database, you have the option to create table spaces to be managed automatically or not using **MANAGED BY** clause. **MANAGED BY AUTOMATIC STORAGE** specifies the storage of a table space to be managed by database manager automatically. **MANAGED BY SYSTEM** or **MANAGED BY DATABASE** define the storage of a table space to be managed manually. If you omit the **MANAGED BY** clause the table space is managed by automatic storage.

If you have not enabled your database for automatic storage you cannot specify the option **MANAGED BY AUTOMATIC STORAGE**.

In Example 3-7 we create a database with automatic storage enabled and add one path for automatic storage. The snapshot data shows the storage path for the database.

Example 3-7 Create a database with automatic storage enabled

```
db2inst1@puget:~> db2 "create database itsodb on /database/data1"
DB20000I The CREATE DATABASE command completed successfully.
db2inst1@puget:~> db2 connect to itsodb
```

Database Connection Information

```
Database server      = DB2/LINUX 9.5.0
SQL authorization ID = DB2INST1
Local database alias = ITSODB
```

```
db2inst1@puget:~> db2 get snapshot for database on itsodb | grep -i storage
Number of automatic storage paths      = 1
Automatic storage path                  = /database/data1
```

```
db2inst1@puget:~> db2 "alter database itsodb add storage on '/database/data2'"
DB20000I The SQL command completed successfully.
```



```
db2inst1@puget:~> db2 get snapshot for database on itsodb | grep -i storage
Number of automatic storage paths          = 2
Automatic storage path                     = /database/data1
Automatic storage path                     = /database/data2
```

Note: More information about the automatic storage feature can be found in the Information Center in Automatic storage section:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/topic/com.ibm.db2.luw.admin.dbo.doc/doc/c0052484.html>

3.2.3 Linux specific configuration

In order to access DB2, each Linux user must set up his or her Linux environment. During creation of an instance, DB2 creates a subdirectory named `sql1ib` in the instance home directory `$INSTHOME`. In this directory, there are important files such as *db2nodes.cfg*, *db2profile*, and *userprofile*. These files are used to set up the DB2 environment. To set up a user environment to use DB2 every time the user logs on to the Linux system, add the following command to the user profile under user's home directory `$INSTHOME/.profile`:

```
. $INSTHOME/sql1ib/db2profile
```

`db2profile` contains all settings that allow users to work with DB2 databases. Do not modify this file, because `db2profile` will be replaced after installing a FixPak. If you want to make changes specific to your environment, make changes in `$INSTHOME/sql1ib/userprofile`, which is called by `db2profile`. This file is useful for setting special environment variables such as TSM parameters.

Notes:

- *\$INSTHOME* should be replaced with the actual value of the home directory of the instance.
- DB2 provides `db2cshrc` under `$INSTHOME/sql1ib` for `csh` users to set up the DB2 environment.

3.2.4 Multiple partitioned specific configuration

In DB2 a *database partition* is a part of the database which consists of its own data, indexes, configuration files, and transaction logs. Partitions are used to create clustered databases where one logical database is split on several systems. The partitions can be on the same system (logical partitions) or on a different system (physical partitions).

Figure 3-4 shows the file system layout for our sample partitioned database.

`/db2home/<instance>` is the instance home directory. In the partitioned database environment `/db2home` is an NFS mount to `/export/db2home` on the first system. The other database files are located in local directories on each system. The file `db2nodes.cfg` defines the partitions which are part of the partitioned database.

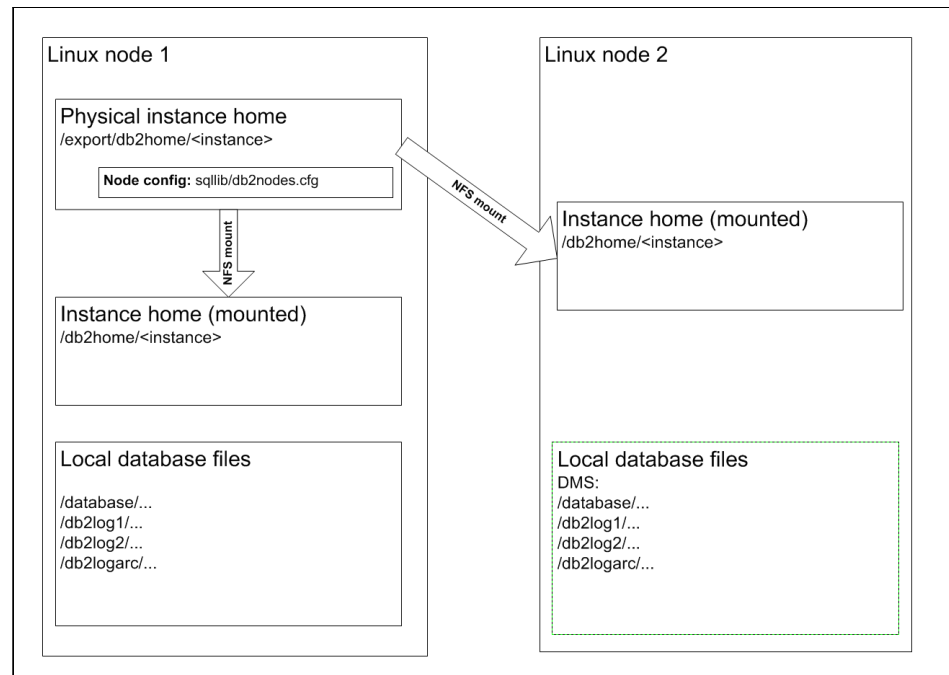


Figure 3-4 Partition and file system layout for partitioned databases

The setup of `/db2home/<instance>` and the file system structure for one partition has is shown in 2.2.3, “Storage planning” on page 39. This configuration of the file system structure must be done in the same way on the second system too.

For performance and availability reasons, we recommend that you avoid putting user data on the catalog partition. When restoring the catalog partition, the restore is faster when there is no data on it.

db2nodes.cfg

To define which hosts are participating in a partitioned environment, you have to modify the `db2nodes.cfg` file located in `$INSTHOME/sqlib`. The format of the `db2nodes.cfg` file is as follows:

```
nodenum hostname logical_port
```

Where, *nodenum* represents the partition number, *hostname* represents the host name or IP address of physical machine, and *logical_port* represents the logical port number of the partition on the same host.

In our example, we have two machines which each hold one database partition. The host *mensa* will be partition 0 and *gemin*i will be partition 1. Example 3-8 shows the contents of *db2nodes.cfg* for our setup.

Example 3-8 db2nodes.cfg file

```
0 mensa 0
1 gemini 0
```

To enable communications between the database partition servers that participate in your partitioned database system, you have to reserve TCP/IP ports in */etc/services* for the *Fast Communication Manager* (FCM). If you install DB2 with the *db2setup* wizard, a port range is automatically reserved on the primary (instance-owning) computer.

Log on to each participating computer as root user and add the identical entries into */etc/services*. Example 3-9 shows four ports reserved for four local partitions on this host.

Example 3-9 Ports in /etc/services

```
DB2_db2inst1 60000/tcp
DB2_db2inst1_1 60001/tcp
DB2_db2inst1_2 60002/tcp
DB2_db2inst1_END 60003/tcp
```

Now you are able to start DB2 in a multiple host, multi-partitioned environment. To do this, enter the **db2start** command on any host. Refer to Example 3-10.

Example 3-10 Start of a DB2 instance with four partition

```
$ db2start
02/14/2008 16:30:20    0    0    SQL1063N  DB2START processing was successful.
02/14/2008 16:30:21    1    0    SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.
```

If you encounter any errors at startup you can check the *db2diag.log* files in the different instances of the cluster for more detailed information.

3.3 Creating a database

In this chapter we show the creation of a database. We are going to demonstrate 2 cases:

- ▶ A single partitioned database. That chapter guides you through the steps to create a simple database by using the Create Database Wizard. We also provide you the scripts how to create a database from the command line.
- ▶ A multiple partitioned database. This chapter provides you the scripts and commands to setup a partitioned database with 2 physical partitions.

3.3.1 Creating a single partitioned database

When you create a database, the following tasks will be done for you:

- ▶ Setting up of all the system catalog tables that are required by the database.

During creation of a new database, DB2 allocates three table spaces: SYSCATSPACE, USERSPACE1, and TEMPSPACE1. The table space SYSCATSPACE holds all system catalog tables. These catalog tables are actually a repository that keeps information and statistics about all objects within the database. USERSPACE1 contains the user created tables, their indexes, and their long data. A system temporary table space (TEMPSPACE1) is required by DB2 to support activities like sorting data, join tables create indexes. If you do not specify them at the **CREATE DATABASE** command, DB2 will choose defaults for them.

- ▶ Allocation of the database recovery log.

The DB2 recovery log tracks all changes made against your database. It is also called the transaction log. It is used with the recovery history file to recover data in case of failure.

- ▶ Creation of the database configuration file and the default values are set.

The database configuration file is used to provide DB2 information such as how much memory it can use, where the log has to be stored, and DB2 optimizer relevant information.

- ▶ Binding of the database utilities to the database.

Programs which contain SQL statements have to be bound against the database. This binding process is used for DB2 utilities like import and load. The “Embedded SQL” on page 435 provides some background information about the bind process.

There are two methods to create databases in a existing instance:

- ▶ DB2 Create Database Wizard which is a Java-based graphical tool

- Use **create database** command from a command line processor window

You must have *sysadm* or *sysctrl* authorization to create a new database. Before creating a database, set the right permissions on all file systems or directories where you want to store the database files. By default, the database will be created and cataloged in the path that is determined by the database manager configuration (DBM cfg) parameter *DFTDBPATH*. You can change this to your desired default path by entering following command:

```
db2 update dbm cfg using DFTDBPATH /database
```

The creation of the file systems and mount points is explained in 2.2.3, “Storage planning” on page 39.

Create Database wizard

If you have DB2 installed in a single partition node, then the easiest way to create a database is to use the Create Database wizard. The DB2 instance is already created during DB2 installation as described in 2.3, “Installing DB2” on page 58.

The steps to create a single partitioned database using Create Database wizard is as follows:

1. Start the Create Database wizard from the Control Center. Right-click on **All Databases** then select **Create Database** → **Standard** as shown in Figure 3-5.

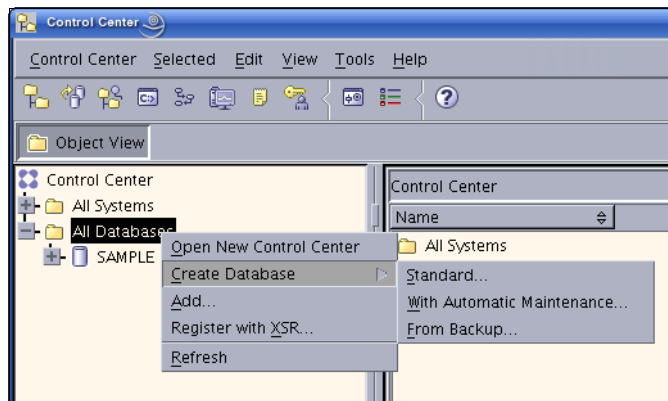


Figure 3-5 Start the create database wizard

2. Specify a name for your new database (Figure 3-6 on page 106): Enter the following:
 - Database name: ITS0DB.

- Database alias ITSODB.
- Comment for the database: Lab database for ITS0.
- Check **Let DB2 manage my storage (automatic storage)**.
- If you want to restrict user access to the system catalog tables, check **Restrict access to system catalogs**.
- and press **Next**.

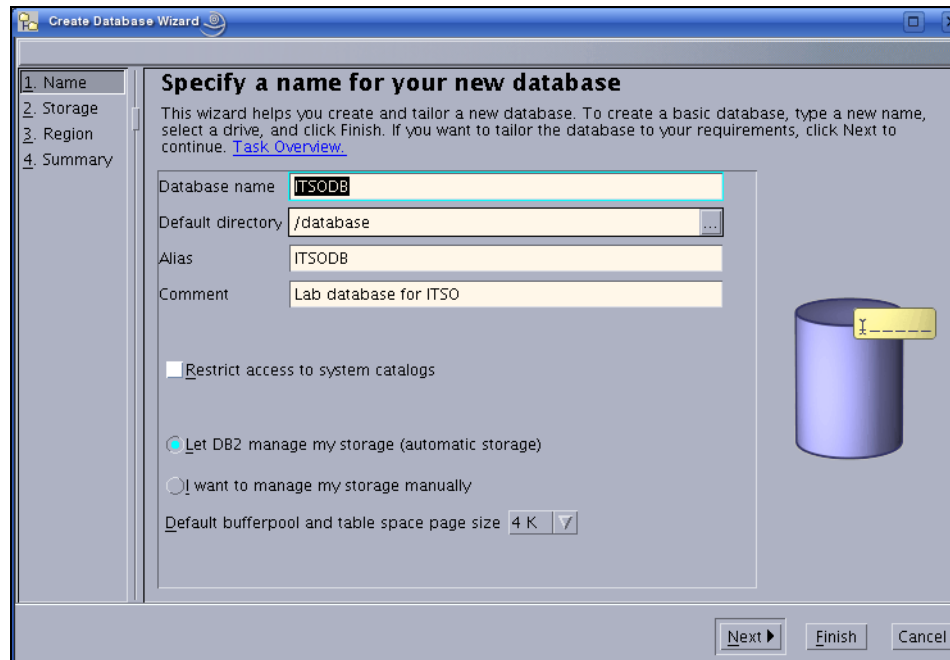


Figure 3-6 Specify the database name

3. The next panel allows to show additional storage paths (Figure 3-7 on page 107).

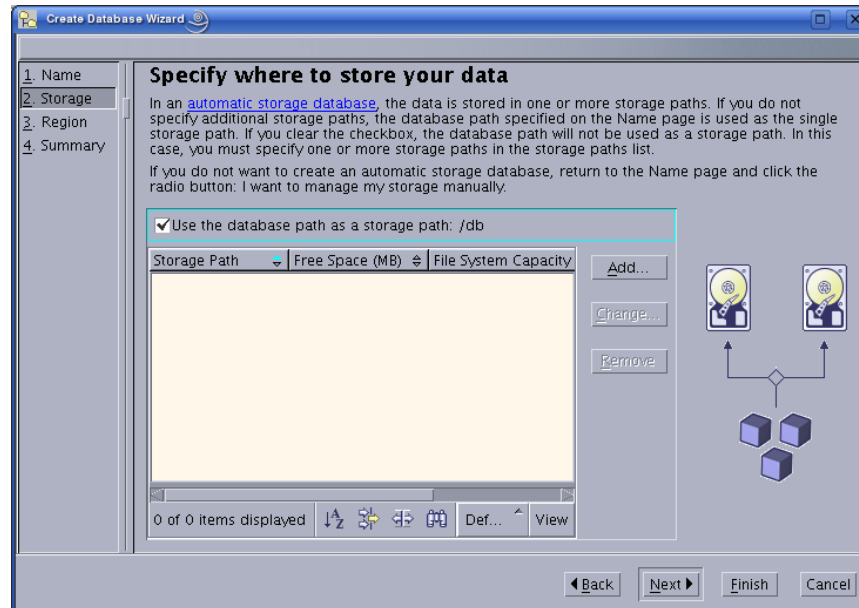


Figure 3-7 Specify the storage path

4. Press **Next** and you get to Figure 3-8.

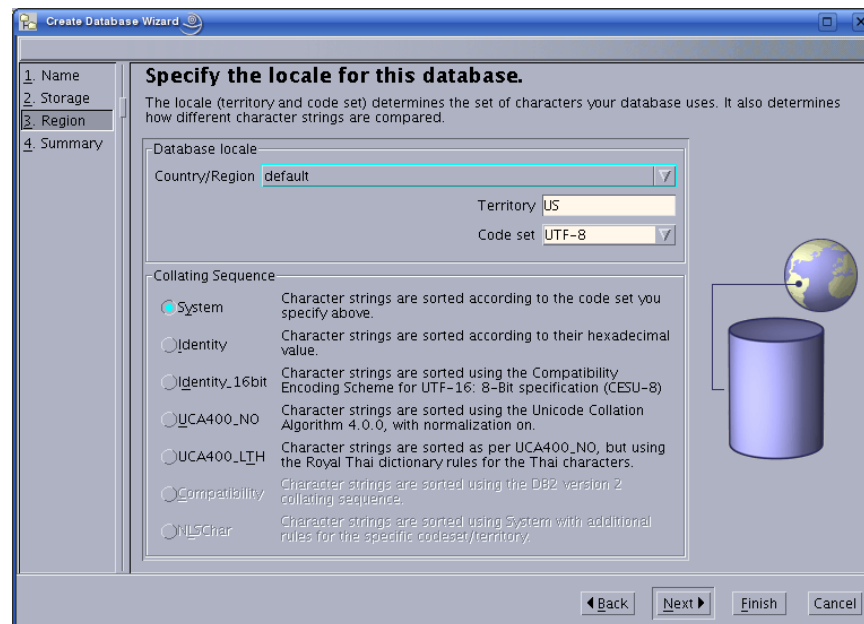


Figure 3-8 Specify the locale

5. Press **Next**.
6. Summary (Figure 3-9). Here you can review your settings, view or save the create database command. Press **Finish** to finally create the database.



Figure 3-9 Database wizard summary page.

Once the database has been created, it shows in the Control Center as shown in Figure 3-10. Expanding the database to view the database objects such as table spaces, containers, tables, and so on.

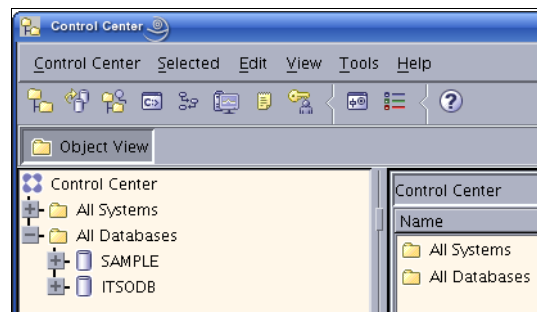


Figure 3-10 Database created

Command line

The command which was used to create the database is shown in Example 3-11:

Example 3-11 Create database statement with automated storage ON

```
CREATE DATABASE ITSODB AUTOMATIC STORAGE YES
ON '/database'
```



```
DBPATH ON '/database'
ALIAS ITSODB USING
CODESET UTF-8 TERRITORY US COLLATE
USING SYSTEM PAGESIZE 4096
WITH 'Lab database for ITS0';
```

The above script is a direct output of the wizard and can be used instead of the Create Database Wizard to create the same database.

If you want to control where to place your container files for the table spaces, Example 3-12 shows an example of a CREATE DATABASE statement with automatic storage off and manually defined table space containers:

Example 3-12 DDL to create the database with containers specified

```
CREATE DATABASE ITSODB AUTOMATIC STORAGE NO ON '/database' ALIAS ITSODB
USING CODESET UTF-8 TERRITORY US COLLATE
USING SYSTEM PAGESIZE 4096
CATALOG TABLESPACE MANAGED BY SYSTEM USING
    ('/db2catalog')
USER TABLESPACE MANAGED BY DATABASE USING
    (FILE '/db2user/cont1.tbsp' 89600)
TEMPORARY TABLESPACE MANAGED BY SYSTEM USING
    ('/db2temp')
WITH 'Lab database for ITS0';
```

Note: The **CREATE DATABASE** command is explained in more detail in *Command Reference*, SC23-5846-00.

3.3.2 Creating a multiple partitioned database

The Create Database Wizard can also be used to create a multi-partitioned database. However, for a multi-partitioned environment, we recommend that you use the DB2 command line, instead of the Create Database wizard. In this section, we provide an example of setting up a partitioned database over two physical hosts with one logical partitions on each host.

Before you create a multi-partitioned database, you should consider which partition will be the catalog partition. In a partitioned database environment, the **create database** command affects all database partitions that are listed in the `db2nodes.cfg` file. The database partition from which this command is issued becomes the catalog database partition for the new database.

Tip: On a host with more than one partition, every command or SQL statement runs against the first database partition on this host by default. To run the commands or SQL statements on the other database partitions, you need to specify the database partition by setting the DB2NODE variable or by using the DB2 command **SET CLIENT**.

Log on as the instance owner or any user with sysadm or sysctrl authority to the host where the database catalog should be created. Verify that you have initialized your DB2 environment as shown in Example 3-13.

Example 3-13 Verify DB2 environment

```
db2inst1@mensa:~> echo $DB2INSTANCE
db2inst1
db2inst1@mensa:~>
```

The rah and db2_all command

Before we start with the creation of the multi partitioned database we show you the **db2_all** and the **rah** command. Both are very useful to execute commands on all partitions.

The **db2_all** command can be used to execute commands on all database partitions. For instance, if you want to create a directory on all partitions, you can execute this command to create a directory on all nodes. For instance in Example 3-14.

Example 3-14 Example of the typeset command

```
db2inst1@mensa:~> db2_all 'typeset -Z4 DB2NODE; mkdir -p
/database/db2inst1/NODE$DB2NODE/ITSODB/SAMPLE_MPL'

mensa.itsosj.sanjose.ibm.com: typeset -Z4 DB2NODE completed ok

gemini.itsosj.sanjose.ibm.com: typeset -Z4 DB2NODE completed ok
```

The **rah** command can be used to execute commands on all physical hosts. The difference to the **db2_all** command is that the specified commands in **rah** are executed only once per physical host. That is important if you have several logical partitions on one physical system.

Tip: The command '**typeset -Z4 DB2NODE**' returns a four digit number for every partition defined in `db2nodes.cfg`, for example, 0 will be 0000, 1 will be 0001.

Create database

Example 3-15 shows the DDL to create multi partitioned database. This DDL is almost identical as the DDL for creating single partitoned database shown in Example 3-12 on page 109. We have only introduced the usage of the \$N variable which substitutes to the partition number.

Example 3-15 Create a partitioned database - crdb.sql

```
CREATE DATABASE ITSODB AUTOMATIC STORAGE NO ON '/database' ALIAS ITSODB
USING CODESET UTF-8 TERRITORY US COLLATE
USING SYSTEM PAGESIZE 4096
TEMPORARY TABLESPACE MANAGED BY SYSTEM USING
    ('/db2temp/db2inst1/NODE000 $N /ITSODB/TEMPSPACE1')
WITH 'Lab database for ITS0';
```

The argument “ \$N” ([blank]\$N) is a database partition expression that can be used anywhere in the storage path. Multiple database partition expressions can be specified. The partition expression is terminated by a space character. If there is no space character in the storage path after the database partition expression, it is assumed that the rest of the string is part of the expression. Table 3-1 shows you how \$N is used for the calculation of the path name.

Table 3-1 Calculation of \$N assuming the partition number is 10

Syntax	Example	Value
[blank]\$N	“ \$N”	10
[blank]\$N+[number]	“ \$N+100”	110
[blank]\$N#[number]	“ \$N%5”	0
[blank]\$N+[number]#[number]	“ \$N+1%5”	1
[blank]\$N#[number]+[number]	“ \$N%4+2”	4
% is the modulo function		

Use the following command to run the create database script:

```
db2 - tvf crdb.sql
```

Once the database is created, you can connect to the database as shown in Example 3-16. We also list the table spaces in the catalog node mensa.

Example 3-16 List table spaces on mensa

```
db2inst1@mensa:~> db2 connect to itsodb
```

Database Connection Information

```
Database server      = DB2/LINUX8664 9.5.0
SQL authorization ID = DB2INST1
Local database alias = ITSODB
```

```
db2inst1@mensa:~> db2 list tablespaces
```

Tablespaces for Current Database

```
Tablespace ID          = 0
Name                   = SYSCATSPACE
Type                   = System managed space
Contents               = All permanent data. Regular table
space.
State                  = 0x0000
  Detailed explanation:
    Normal

Tablespace ID          = 1
Name                   = TEMPSPACE1
Type                   = System managed space
Contents               = System Temporary data
State                  = 0x0000
  Detailed explanation:
    Normal

Tablespace ID          = 2
Name                   = USERSPACE1
Type                   = Database managed space
Contents               = All permanent data. Large table space.
State                  = 0x0000
  Detailed explanation:
    Normal
```

DB21011I In a partitioned database server environment, only the table spaces on the current node are listed.

Example 3-17 lists the table space in node gemini. The catalog table space exists only on mensa, the catalog node.

Example 3-17 List table spaces on gemini

```
db2inst1@gemini:~> db2 connect to itsodb
```

Database Connection Information

```
Database server      = DB2/LINUX8664 9.5.0
SQL authorization ID = DB2INST1
Local database alias = ITSODB
```

```
db2inst1@gemini:~> db2 list tablespaces
```

Tablespaces for Current Database

```

Tablespace ID          = 1
Name                   = TEMPSPACE1
Type                   = System managed space
Contents               = System Temporary data
State                  = 0x0000
  Detailed explanation:
    Normal

Tablespace ID          = 2
Name                   = USERSPACE1
Type                   = Database managed space
Contents               = All permanent data. Large table space.
State                  = 0x0000
  Detailed explanation:
    Normal

```

DB21011I In a partitioned database server environment, only the table spaces on the current node are listed.

Create table spaces and tables

The next step is to create tables for your application. When a database is created, DB2 creates three default table spaces. The USERSPACE1 is used for place the data tables. Based on the database design, more table spaces can be created. The tables usually will be placed to a particular table space based on the application and database design. It is common to create your own table spaces spread across all partitions or only a part of them.

Every table space has to be created in a database partition group. Here we demonstrate the creation of a table space over all partitions defined in db2nodes.cfg, using the default database partition group IBMDEFAULTGROUP.

First we create the directory for the table space container on both systems using the following command:

```
db2_all 'typeset -Z4 DB2NODE; mkdir -p
/db/db2inst1/NODE$DB2NODE/ITSODB/SAMPLE_MPL'
```

Example 3-18 on page 114 shows the creation of table space SAMPLE_MPL. We use the variable \$N in the **create tablespace** statement. It will be replaced by the partition number during runtime.

Example 3-18 Create multi partition table space

```
connect to itsodb;

create tablespace sample_mpl in IBMDEFAULTGROUP pagesize 4k
MANAGED BY SYSTEM
using ('/db/db2inst1/NODE000 $N /ITSODB/SAMPLE_MPL')
    on dbpartitionnum (0)
using ('/db/db2inst1/NODE000 $N /ITSODB/SAMPLE_MPL')
    on dbpartitionnum (1)
extentsize 8 prefetchsize 32;
```

Example 3-19 shows a creation of a database partition group sng_01 and a single partition table space on partition 1. We create a container on gemini (partition 1) before creating the table space:

```
mkdir /db/db2inst1/NODE0001/ITSODB/SAMPLE_SNG
```

Example 3-19 Create single partition table space

```
connect to itsodb;

create database partition group sng_01 on dbpartitionnum (1);

create tablespace sample_sng in sng_01 pagesize 4k
MANAGED BY SYSTEM
using ('/db/db2inst1/NODE0001/ITSODB/SAMPLE_SNG')
    on dbpartitionnum (1)
extentsize 8 prefetchsize 32;
```

Now, node 0 (mensa) should have table spaces SYSCATSPACE, TEMPSPACE1, USERSPACE1, and SAMPLE_MPL, and node 1 (gemini) has table spaces TEMPSPACE1, USERSPACE1, SAMPLE_MPL, and SAMPLE_SNG.

To create tables in the newly created table space, use the command line to execute DDL scripts like Example 3-20 using the **db2 -tvf** command.

Example 3-20 Create table DDL

```
connect to itsodb;

CREATE TABLE PART (P_PARTKEY INTEGER NOT NULL,
                    P_NAME    VARCHAR(55) NOT NULL,
                    P_MFGR    CHAR(25) NOT NULL,
                    P_BRAND    CHAR(10) NOT NULL,
                    P_TYPE     VARCHAR(25) NOT NULL,
                    P_SIZE     INTEGER NOT NULL,
                    P_CONTAINER CHAR(10) NOT NULL,
```

```
P_RETAILPRICE FLOAT NOT NULL,  
P_COMMENT VARCHAR(23) NOT NULL  
) IN SAMPLE_MPL;
```

If you prefer to use graphical tools, start the Control Center and expand the tree under the database until the tables folder appears. Right-click and select **Create** See Figure 3-11.

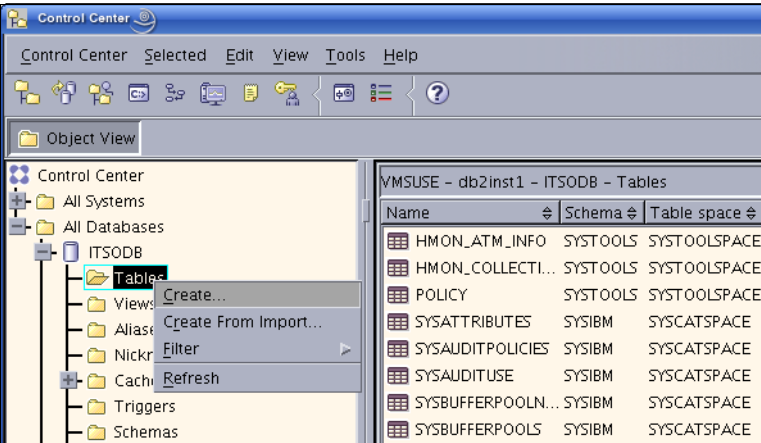


Figure 3-11 Start the Create Table wizard

That starts the Create Table Wizard (Figure 3-12). It guides you through many panels where you can create a new table using the predefined column definitions, or by defining your own column definitions.

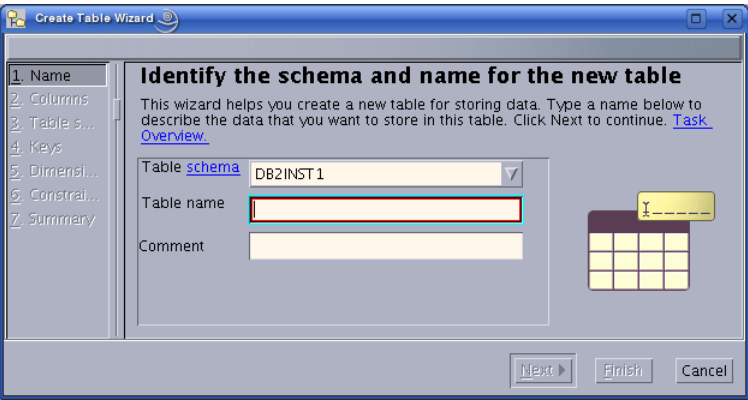


Figure 3-12 Create Table wizard in the Control Center

Table partitioning and row compression

DB2 offers numerous features and functions to support your enterprise databases. We call out here two table related features for you to consider when designing your tables.

Table partitioning feature provides you the capability to divide table data across multiple storage objects, called *data partitions* or *ranges*, according to values in one or more table partitioning key columns of the table. With table partitioning, you are able to roll-in and roll-out of table data efficiently, detach or reattach a data partition for back up, restore or reorganization, and potentially boost query performance through data partition elimination.

When you design your database, you may want to consider table partitioning feature for large tables. For more details, refer to *Database Partitioning, Table Partitioning, and MDC for DB2 9*, SG24-7467

The main idea of row compression is save disk storage space. Additional advantages of row compression including possible disk I/O saving, increasing buffer pool hit ratio, and backup image size decrease. For more details, refer to Information Center at:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/topic/com.ibm.db2.luw.admin.dbo.doc/doc/c0023489.html>

3.4 Further configuration

DB2 uses log files to record its transactions. To improve the performance and to ease the maintenance work, consider changing the location for online logs, archive logs, and db2diag.log.

3.4.1 Change online log path

By default the log path is configured as /database/db2inst1/NODE0000/SQL00001/SQLLOGDIR. To change the log path, use Control Center or command to update the database configuration parameter NEWLOGPATH to the new path. The new path will take effect once the database is in a consistent state and all users are disconnected from the database. Note that it is not necessary to issue a **db2stop** and **db2start**. When the first user reconnects to the database the log files are relocated.

The following is the procedure to change online log path by commands:

1. Create the directories. For a single partitioned database environment enter:

```
mkdir -p /db2log1/ITS0DB
```



```
mkdir -p /db2log2/ITSODB
```

and for the multiple partitioned environment, enter:

```
db2_all 'typeset -Z4 DB2NODE; mkdir -p /db2log1/ITSODB'
db2_all 'typeset -Z4 DB2NODE; mkdir -p /db2log2/ITSODB'
```

2. Set the database configuration manager parameter NEWLOGPATH to the new path. Example 3-21 shows the command.

Example 3-21 Set the NEWLOGPATH parameter

```
db2inst1@puget:~> db2 "UPDATE DB CONFIG FOR itsodb USING NEWLOGPATH
/db2log1/ITSODB"
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
SQL1363W One or more of the parameters submitted for immediate
modification
were not changed dynamically. For these configuration parameters, all
applications must disconnect from this database before the changes become
effective.
```

For the partitioned database, use the following command to set the NEWLOGPATH parameter in all database nodes:

```
db2_all 'db2 connect to itsodb;db2 update db cfg for itsodb using NEWLOGPATH
/db2log1/itsodb'
```

3. Disconnect all users from the database.
You can use the command **db2 "list applications"** to see all applications which are currently connected. You can then use the command **db2 "force application (<id>)"** to force individual users off. Note that you only need to force off users who are connected to the database where we want to change the log path.
4. Connect to the database. During the connection, the log will be relocated.
Example 3-22 shows that the log path has been changed.

Example 3-22 Change the online log path

```
db2inst1@puget:~> db2 terminate
DB20000I The TERMINATE command completed successfully.
db2inst1@puget:~> db2 connect to itsodb
```

Database Connection Information

```
Database server      = DB2/LINUX 9.5.0
SQL authorization ID = DB2INST1
Local database alias = ITSODB
```

```
db2inst1@puget:~> db2 get db config for itsodb | grep
"NEWLOGPATH\|MIRRORLOGPATH\|Path to log files"
```

Changed path to log files	(NEWLOGPATH) =
Path to log files	=
/db2log1/ITSODB/NODE0000/	
Mirror log path	(MIRRORLOGPATH) =

For the partitioned database, use the following command to check the path to log files entry:

```
db2_a11 'db2 connect to itsodb; db2 get db cfg for itsodb | grep Path'
```

This change is logged in db2diag.log as shown in Example 3-23. If you receive error messages during the change, you will see additional information here.

Example 3-23 Entry in db2diag.log for LOG change

```
2008-02-12-10.30.16.368903-480 I390990G483      LEVEL: Event
PID      : 10434                TID   : 2968513440  PROC  : db2sysc 0
INSTANCE: db2inst1            NODE   : 000          DB    : ITSODB
APPHDL   : 0-41                APPID: *LOCAL.db2inst1.080212175506
AUTHID   : DB2INST1
EDUID    : 61                  EDUNAME: db2agent (ITSODB) 0
FUNCTION: DB2 UDB, config/install, sqlfLogUpdateCfgParam, probe:20
CHANGE   : CFG DB ITSODB: "Newlogpath" From: "" To: "/db2log1/ITSODB"
```

```
2008-02-12-10.30.43.496729-480 I391474G474      LEVEL: Info
PID      : 10434                TID   : 2968513440  PROC  : db2sysc 0
INSTANCE: db2inst1            NODE   : 000          DB    : ITSODB
APPHDL   : 0-84                APPID: *LOCAL.db2inst1.080212183043
AUTHID   : DB2INST1
EDUID    : 61                  EDUNAME: db2agent (ITSODB) 0
FUNCTION: DB2 UDB, data protection services, sqlpgnlp, probe:1240
MESSAGE  : Active log path is moved to /db2log1/ITSODB/NODE0000/
```

We also activate the mirrored logs in order to enhance the availability as shown in Example 3-24 on page 118.

Example 3-24 Activate the mirror logs

```
db2inst1@puget:~> db2 "UPDATE DB CONFIG FOR itsodb USING MIRRORLOGPATH
/db2log2/ITSODB"
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
SQL1363W One or more of the parameters submitted for immediate modification
were not changed dynamically. For these configuration parameters, all
applications must disconnect from this database before the changes become
effective.
db2inst1@puget:~> db2 terminate
DB20000I The TERMINATE command completed successfully.
db2inst1@puget:~> db2 connect to itsodb
```

Database Connection Information

```

Database server      = DB2/LINUX 9.5.0
SQL authorization ID = DB2INST1
Local database alias = ITSODB

```

```

db2inst1@puget:~> db2 get db config for itsodb | grep
"NEWLOGPATH\|MIRRORLOGPATH\|Path to log files"
Changed path to log files      (NEWLOGPATH) =
  Path to log files              =
/db2log1/ITSODB/NODE0000/
  Mirror log path              (MIRRORLOGPATH) =
/db2log2/ITSODB/NODE0000/

```

This action is also logged in `db2diag.log`.

The logging parameters can also be changed in the Command Center. Right-click on the database under **All Databases**, select **Configure Database Logging**. Here you can change and apply the changes and save them.

3.4.2 Change the archive log path

By default, *circular logging* is set for a new database. With this type of logging, only full and offline backups of the database are allowed. For recovery, the database can only be restored to last offline backup not to the point of failure.

In order to enable full recovery of the database up to a point-in-time, *archive logging* has to be enabled. These archived logs are then used by the **restore** command to recover the database up to the current point in time.

Chapter 6, “Administering databases” on page 223 describes the setup of the archive logs path in detail.

3.4.3 Change the path for db2diag.log

DB2 logs information such as administrative event or errors in the diagnostic file `db2diag.log`. This is an instance level log file and is located in `<instance home>/sqllib/db2dump` by default. It is a good practice to examine this file from time to time. Very often this file contains additional information about some DB2 commands. Especially commands which interact with the operating system such as creating table spaces. You will find information about wrong path names, permission problems, errno codes, and so on.

The level of detailness logged in `db2diag.log` is specified in the `DIAGLEVEL` DBM configuration parameter ranging from 0 to 4. The default is 3. The higher

the number is the more information is logged. Setting it to 4 can have a performance impact on the database.

You need to maintain db2dump directory as the db2diag.log file keeps growing and other dump files can be added to the directory by DB2. This means you have to purge old files and roll over files or compress them. You should consider relocating this directory to a separate file system so that it will not impact the instance if it fills up.

Example 3-25 shows how to change the location for db2diag.log.

Example 3-25 Relocate db2diag.log

```
db2inst1@puget:~> db2 get dbm config | grep DIAGPATH
Diagnostic data directory path          (DIAGPATH) =
/home/db2inst1/sqllib/db2dump
db2inst1@puget:~> db2 update dbm config using DIAGPATH /db2diag
DB20000I The UPDATE DATABASE MANAGER CONFIGURATION command completed
successfully.
db2inst1@puget:~> db2 get dbm config | grep DIAGPATH
Diagnostic data directory path          (DIAGPATH) = /db2diag
```

Note: This is a configuration change on instance level. You only need to perform it once for the whole instance.

The DIAGPATH can also be changed from the Control Center. From **All Systems** → **<HOSTNAME>** → **Instances**, right click on **db2inst1**, select **Configure Parameters**. Change the DIAGPATH value in DBM Configuration pane (Figure 3-13).

Note: For a partitioned database the command is the same. The **UPDATE DBM CONFIG** applies to all nodes because the home directory is shared by an NFS mount.

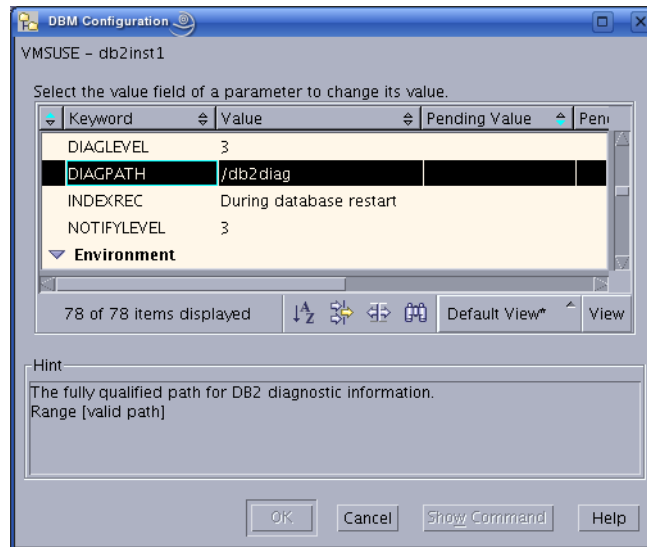


Figure 3-13 Change DIAGPATH in Control Center

3.5 Add database partition

If the load on a partitioned database becomes too high you can add a partition to take the advantage of the parallel processing. This section describes the steps to do that.

In a partitioned database environment you can use the **add dbpartitionnum** CLP command or use the **db2start** command with the option **dbpartitionnum** to add a logical database partition to a database which is already populated with data. The **add dbpartitionnum** command has two options:

- ▶ LIKE DBPARTITIONNUM
- ▶ WITHOUT TABLESPACES

The **add dbpartitionnum** command creates the database directory structure in the database path (for example, /database/...) on the partition where the new database partition has been created. Without using the **like dbpartitionnum** parameter, temporary table space definitions are retrieved from the catalog partition.

In our example, we add a new logical database partition (partition 2) on node 0 mensa. That means, we will have two logical partitions (0 and 2) on mensa and one partition (1) on gemini.

Check `/etc/services` that sufficient ports are reserved for the FCM.

Shutdown your DB2 instance (**db2stop**) so that you can modify `db2nodes.cfg` file. Add a new partition in `$INSTHOME/sql11b/db2nodes.cfg` as shown in Example 3-26.

Example 3-26 New db2nodes.cfg file

```
0 mensa 0
1 gemini 0
2 mensa 1
```

Log on to the host as instance owner where the new partition will reside and start the new partition with:

```
db2start dbpartitionnum 2
```

Alternatively if you start new database partition with the following command, DB2 will update the `db2nodes.cfg` automatically after the next stop of the instance.

```
db2start dbpartitionnum 2 add dbpartitionnum hostname mensa port 1 without tablespaces
```

To create the directory structure used by DB2 we have to issue the `add node` command if it has not already been done by starting the database manager. Issue the commands shown in Example 3-27.

Example 3-27 Add dbpartition utility command

```
db2inst1@mensa:~/sql11b> export DB2NODE=2
db2inst1@mensa:~/sql11b> db2 terminate
db2inst1@mensa:~/sql11b> db2 add dbpartitionnum without tablespaces
```

```
DB20000I The ADD NODE command completed successfully.
```

DB2 has now created the directory structure in `/database`, but with the keyword *without tablespaces*, the existing table space in partitions 0 and 1 has not been created in the new partition. The database partition node group for `IBMTMPGROUP` is expanded to the new database partition 2 and we have to add a table space container for `TEMPSPACE1` on the new partition. See Example 3-28 on page 122

Example 3-28 Add table space container for temp space

```
db2inst1@mensa:~> db2 "alter tablespace tempspace1 add
('db2temp/db2inst1/NODE0002/ITSODB/TEMPSPACE1') on dbpartitionnum (2)"
DB20000I The SQL command completed successfully.
```

All database partition groups that will benefit from the new partition have to be altered, so that the new partition can be used. First we create a table in the table space SAMPLE_SNG which is in the partition group SNG_01 (Example 3-29):

Example 3-29 Create table in SNG_01

```
connect to itsodb;

CREATE TABLE PART1(P_PARTKEY INTEGER NOT NULL,
                    P_NAME    VARCHAR(55) NOT NULL,
                    P_MFGR    CHAR(25) NOT NULL,
                    P_BRAND   CHAR(10) NOT NULL,
                    P_TYPE    VARCHAR(25) NOT NULL,
                    P_SIZE    INTEGER NOT NULL,
                    P_CONTAINER CHAR(10) NOT NULL,
                    P_RETAILPRICE FLOAT NOT NULL,
                    P_COMMENT VARCHAR(23) NOT NULL
                    ) IN SAMPLE_SNG;

ALTER TABLE PART1 ADD PARTITIONING KEY (P_PARTKEY);
```

Example 3-30 shows the statement for SNG_01 in our database ITSODB.

Example 3-30 Alter database partition group SNG_01

```
db2inst1@mena:~> db2 'alter database partition group sng_01 add
dbpartitionnum(2) without tablespaces'
SQL1759W  Redistribute database partition group is required to change database
partitioning for objects in nodegroup "SNG_01" to include some added database
partitions or exclude some dropped database partitions.  SQLSTATE=01618
```

The message SQL1759W is just a warning indicating that all tables in this database partition group have to be redistributed, but all tables are still accessible.

Then we add the table space container for SNG_01 (Example 3-31 on page 123):

Example 3-31 Att table space container for SNG_01

```
db2inst1@mena:~> db2 "alter tablespace SAMPLE_SNG add
('/db/db2inst1/NODE0002/ITSODB/SAMPLE_SNG') on dbpartitionnum (2)"
SQL1759W  Redistribute database partition group is required to change database
partitioning for objects in nodegroup "SNG_01" to include some added database
partitions or exclude some dropped database partitions.  SQLSTATE=01618
```

The **redistribute** command affects all table spaces and tables in the given database partition group. However, the redistribute occurs online and all tables and indexes are still fully accessible.

To redistribute all data in database partition group SNG_01, log on to the host where the catalog partition resides, connect to the database, and issue the **redistribute** command.

Example 3-32 Redistribute command for MPL_01

```
db2inst1@mena:~> db2 redistribute database partition group sng_01 uniform
DB20000I The REDISTRIBUTE NODEGROUP command completed successfully.
```

Note: You can also use the option **ADD DBPARTITIONNUM(S)** at the **REDISTRIBUTE** command. That combines the commands **ALTER DATABASE PARTITION GROUP** and **REDISTRIBUTE DATABASE PARTITION GROUP**.

For each table that has already been processed, the redistribution utility writes a message to file in \$INSTHOME/sqlib/redist/<dbmame>.<part-group>.<time>. See Example 3-33.

Example 3-33 Sample redistribution

Data Redistribution Utility :

```
The following options have been specified :
Database partition group name : SNG_01
Data Redistribution option : U
Redistribute database partition group : uniformly
No. of partitions to be added : 1
List of partitions to be added :
2
No. of partitions to be dropped : 0
List of partitions to be dropped :
```

Delete will be done in parallel with the insert.

The execution of the Data Redistribution operation on :

Begun at	Ended at	Table (poolID;objectID)
18.19.49		"DB2INST1"."PART1" (4;2)
	18.19.49	"DB2INST1"."PART1" (4;2)

--All tables have been successfully redistributed.--

Attention: The redistribute utility issues SQL insert and delete operations; therefore, you should ensure that sufficient log space is available. Redistributing is a very time consuming process and should be done when tables are not in use by applications.

DB2 packages which have a dependency on a table which was redistributed are set as invalid. They will be automatically be rebound at the first SQL request but to save delay time, rebind these packages manually with the **rebind** command.

Another recommendation is to update statistics by issuing the runstats table command and rebind affected packages.

Note: More information to add a database partition is described in DB2 manuel *Partitioning and Clustering Guide*, SC23-5860-00.

An alternate method to add a logical database partition is to use the Add Database Partition wizard (Figure 3-14).

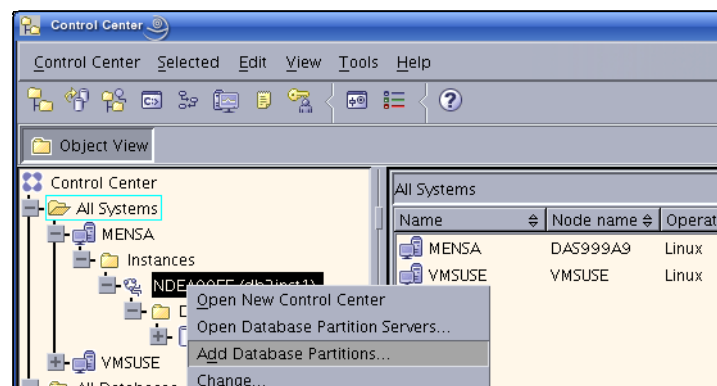


Figure 3-14 Add a partition to a database

The wizard guides you through the process. At the container tab (Figure 3-15 on page 126) you have to defined the container locations on the new partitions.

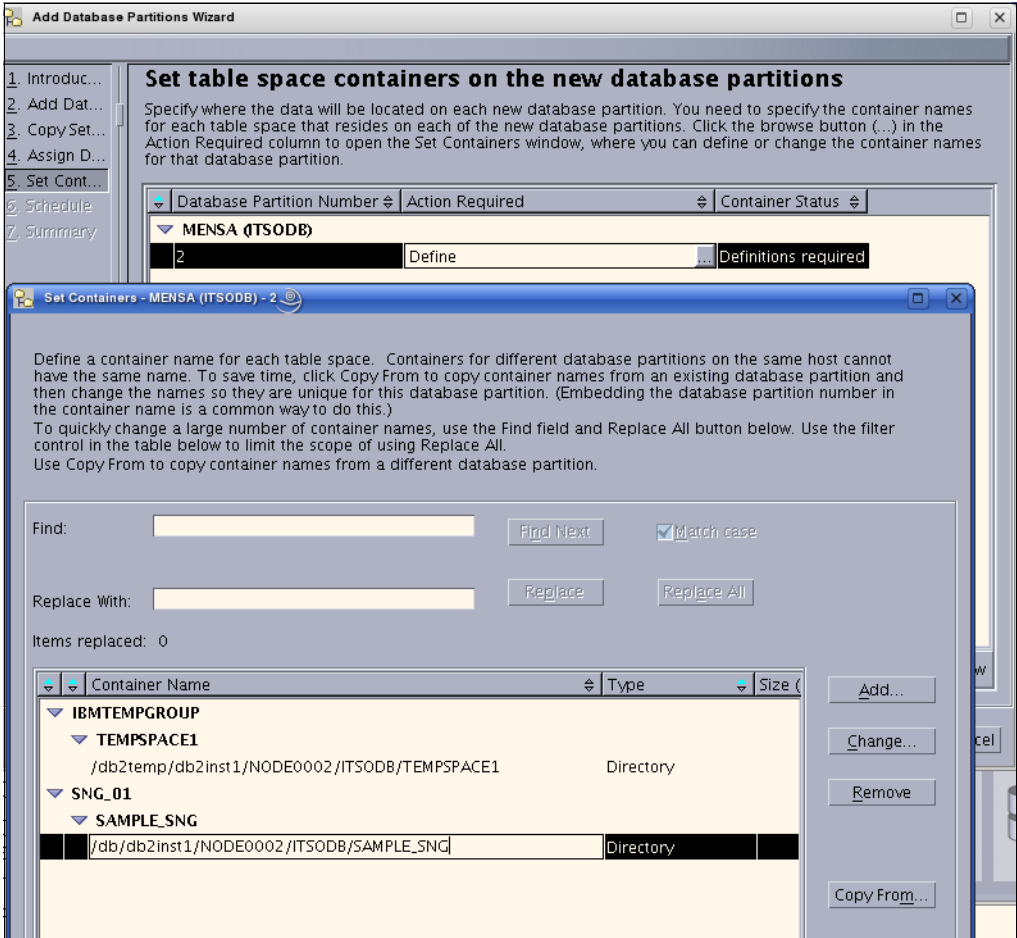


Figure 3-15

In Figure 3-16 on page 127 you can see the generated commands to add a database partition and the result of the command.

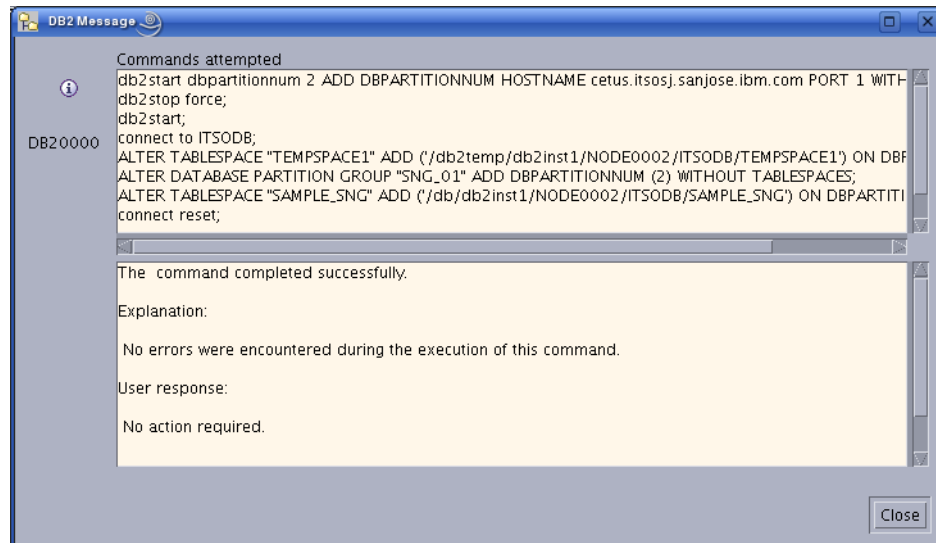


Figure 3-16 Messages after the partition has been created

After executing these commands, all tables within the affected table spaces have to be redistributed. Then click on **Redistribute** in Figure 3-17.

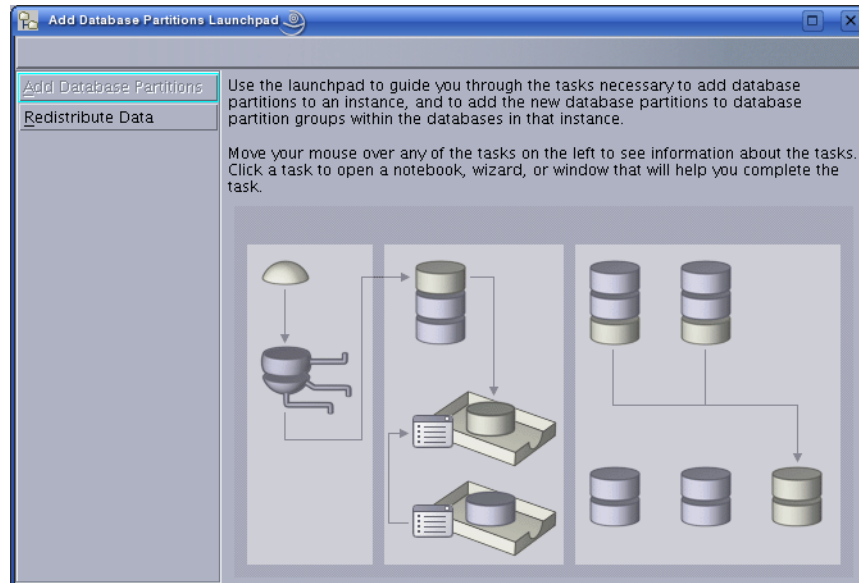


Figure 3-17 Redistribute partition

Then you get to the panel in Figure 3-18 on page 128.

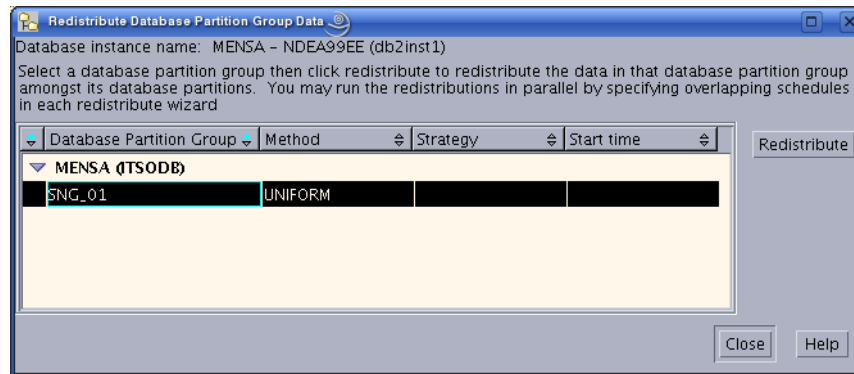


Figure 3-18 Redistribute the data

Here you can select the partition groups you want to redistribute and perform the redistribution. On the Redistribute Strategy window, select **Real time system analysis** to let DB2 gather information about how much log space is needed and how much is available.

Repeat this for all database partition groups that should be extended over the added partition. The redistribution messages are stored as well in:

```
$INSTHOME/sqllib/redist/<dbmame>.<part-group>.<time>
```

3.6 DB2 configuration

DB2 has a large number of configuration parameters that specify the allocation of system resources and the overall configuration of the instance and database. These parameters are stored in the *database configuration*, in the *database manager configuration* and in the *registry*.

Since the default values are set for machines with relatively small memory and disk storage, you may need to modify them to fit your environment. A good starting point for tuning your configuration is the *Configuration Advisor* or the **AUTOCONFIGURE** command.

If you create a new database many of the parameters are already set to **AUTOMATIC** so that STMM takes care of their tuning.

3.6.1 STMM - Self tuning memory management

In version 9 DB2 introduced a new feature, the *Self Tuning Memory Management* (STMM), that greatly simplifies the database memory configuration by

automatically setting values for several memory configuration parameters. The STMM manages memory for the following DB2 memory consumers:

- ▶ Buffer pools
- ▶ Package Cache
- ▶ Locking memory
- ▶ Sort memory

The tuning of these areas can be turned on and off individually. Tuning will be done across different databases and different instances on the same system. The memory available for tuning can be limited. If no more memory for tuning can be allocated from the operating system for or if it hits the limit you defined, the available memory is taken away from another area where the memory is not so much needed. For instance, if STMM figures out that more memory is needed for one buffer pool it can take away this memory from another one.

The frequency for the tuning is determined based on the workload and each tuning step is logged in `db2diag.log` and in the directory `stmmlog`.

STMM also works on partitioned databases but in the case each system should be of the same hardware and the workload and the data should be distributed equally because the tuning parameters are determined on one partition called the *tuning partition* and then distributed to the other partitions.

Figure 3-19 on page 130 illustrate how the space in the memory is distributed among the different areas.

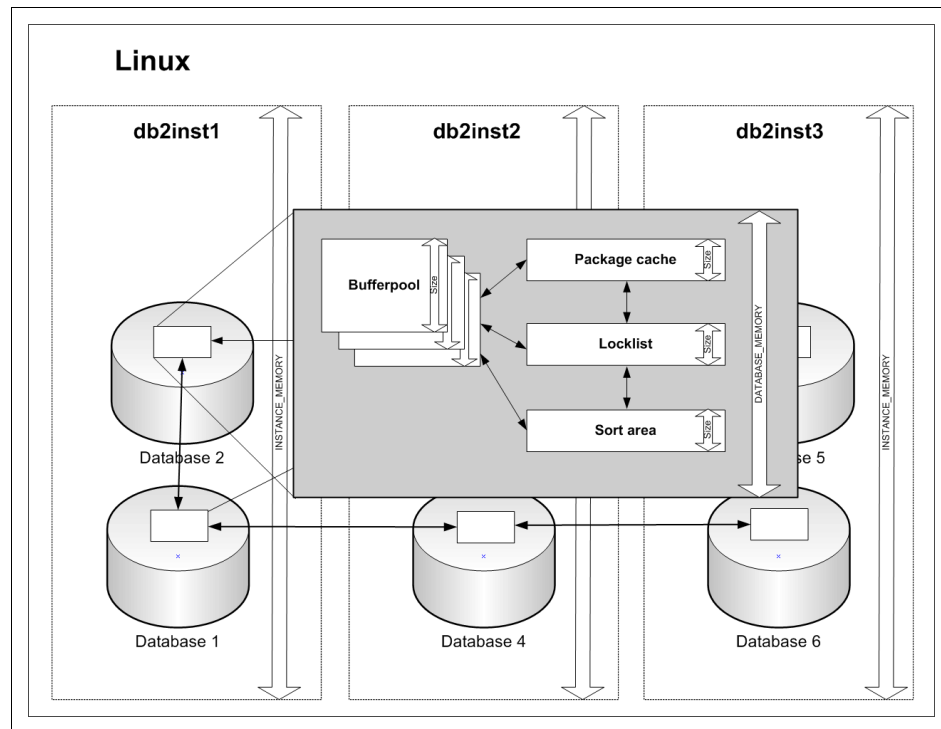


Figure 3-19 STMM

For newly created databases the self tuning feature is turned on by default. You can see that from the database configuration parameters as shown in Example 3-34.

Example 3-34 Parameters turned on for STMM

```
db2inst2@puget:~> db2 get db config for itsodb | grep -i automatic
Size of database shared memory (4KB) (DATABASE_MEMORY) = AUTOMATIC
Max storage for lock list (4KB) (LOCKLIST) = AUTOMATIC
Percent. of lock lists per application (MAXLOCKS) = AUTOMATIC
Package cache size (4KB) (PCKCACHESZ) = AUTOMATIC
Sort heap thres for shared sorts (4KB) (SHEAPTHRES_SHR) = AUTOMATIC
Sort list heap (4KB) (SORTHEAP) = AUTOMATIC
Database heap (4KB) (DBHEAP) = AUTOMATIC
SQL statement heap (4KB) (STMHEAP) = AUTOMATIC
Default application heap (4KB) (APPLHEAPSZ) = AUTOMATIC
Application Memory Size (4KB) (APPL_MEMORY) = AUTOMATIC
Statistics heap size (4KB) (STAT_HEAP_SZ) = AUTOMATIC
Number of asynchronous page cleaners (NUM_IOCLEANERS) = AUTOMATIC
Number of I/O servers (NUM_IOSERVERS) = AUTOMATIC
Default prefetch size (pages) (DFT_PREFETCH_SZ) = AUTOMATIC
```

Max number of active applications	(MAXAPPLS) = AUTOMATIC
Average number of active applications	(AVG_APPLS) = AUTOMATIC
...	

To turn off STMM for one particular area use the command:

```
UPDATE DATABASE CONFIGURATION USING <parameter> MANUAL
```

If you want to turn off STMM in general you can do that by executing this command:

```
UPDATE DATABASE CONFIGURATION USING SELF_TUNING_MEM OFF
```

Buffer pools can be enabled or disabled for STMM individually by using the option `SIZE AUTOMATIC` as follows:

```
ALTER BUFFERPOOL ibmdefaultbp SIZE AUTOMATIC
```

or

```
CREATE BUFFERPOOL <bpname> SIZE AUTOMATIC
```

STMM can run in two modes:

- ▶ You can limit the available size for memory tuning on database level with the parameter `DATABASE_MEMORY`. Then STMM will not increase the memory used by the database higher than this value. If one area needs to be increased because it has more demand, another area with less memory demand will be decreased accordingly.
- ▶ You can leave the available memory size for tuning unlimited by setting the `DATABASE_MEMORY` parameter to `AUTOMATIC`. In this case the available memory in the operating system limits the available memory for tuning. STMM leaves a buffer for the operating system processes so that no paging occurs.

You can also limit the memory used by the instance by setting the value of the DBM parameter `INSTANCE_MEMORY` to a particular value. If you do not want to limit it, you can leave the value at `AUTOMATIC`.

Note: Some Linux kernels do not allow setting the value of the `DATABASE_MEMORY` database parameter to `AUTOMATIC`. It is supported on RHEL5 and on SUSE 10 SP1 and newer. All other validated Linux distributions will return to `COMPUTED` if the kernel does not support this feature.

3.6.2 Database manager configuration parameters

The database manager configuration (DBM cfg) parameters are stored in a file named `db2system` in `$INSTHOME/sqllib/`. To modify these parameters, use the graphical interface, Control Center or use DB2 command line interface.

Important DBM cfg parameters

Here are the important DBM cfg parameters:

► **DFTDBPATH**

By default it is set to `$INSTHOME`. We recommend that you set it to your desired path. In a partitioned environment usually `$INSTHOME` is on a NFS file system, but DB2 does not support NFS for the database path.

► **DFT_MONSWITCHES**

Set all monitor switches to **ON** to let DB2 gather statistics data. Use the **get snapshot** command to show details about the behavior of DB2.

► **DIAGLEVEL**

For trouble determination, set to **4**. DB2 will write all errors, warnings and informational messages to the `db2diag.log` into the path defined by **DIAGPATH**. The default is **3**. Note that the log level 4 may have severe performance impacts.

► **MON_HEAP_SZ** - Can be set to **AUTOMATIC**.

The memory required for maintaining the private views of the database system monitor data is allocated from the monitor heap. Its size is controlled by the **MON_HEAP_SZ** configuration parameter. The amount of memory required for monitoring activity varies widely depending on the number of monitoring applications and event monitors, the switches set, and the level of database activity.

► **SHEAPTHRES**

For private sorts, this parameter is an instance-wide soft limit on the total amount of memory that can be consumed by private sorts at any given time.

For shared sorts, this parameter is a database-wide hard limit on the total amount of memory consumed by shared sorts at any given time. By reaching this limit, no more shared sorts are allowed.

► **NUM_POOLAGENTS** - Can be set to **AUTOMATIC**.

For a Decision Support System (DSS) environment with which few applications connect concurrently, set this value small. For Online Transaction Processing (OLTP) with many concurrent connected applications, set it larger to prevent extra creation of `db2agents`, which are time costs.

► **INTRA_PARALLEL**

For an OLTP environment set this value to NO, due to the fact that transaction throughput is more important than parallel work. OLTP queries read, update, or insert only one or a few rows and therefore it doesn't make sense to split this work over more processes.

In a DSS environment, typically most of all queries join many tables at once. We recommend that you set this value to YES, so DB2 can split the work over many processes.

► **MAX_QUERYDEGREE**

Because parallelism in an OLTP system is not desirable, set it to 1.

For DSS, set it to the number of CPUs of the database server to prevent users from setting an incorrect degree in their program, which can slowdown DB2 activities. Please note that in static package, if the query degree is set at the bind time, the max_querydegree is overridden by the query degree setting.

► **INSTANCE_MEMORY** - Can be set to **AUTOMATIC**.

If this variable is set to **AUTOMATIC**, it is calculated at the start of the instance. Setting it to a value allows you to define the memory used by the instance.

► **JAVA_HEAP_SZ**

Defines the heap size which is used for Java UDFs and stored procedures.

► **SVCENAME**

Defines the TCP/IP port number or name which is used by the instance. If you use a port name you have to define it in /etc/services.

Note: For a complete and detailed description of all the DBM parameters, refer to the Information Center at

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.config.doc/doc/c0004555.html>

Monitor and update DBM configuration

Configuring settings of a DB2 Instance can be done either through the Control Center or on DB2 Command line.

Control Center

Use the following steps to set DBM configuration:

- Start the Control Center and expand the tree up to the desired Instance.
- Right-click on it to open the pop-up window and choose **Configure Parameters ...** (Figure 3-20 on page 134).

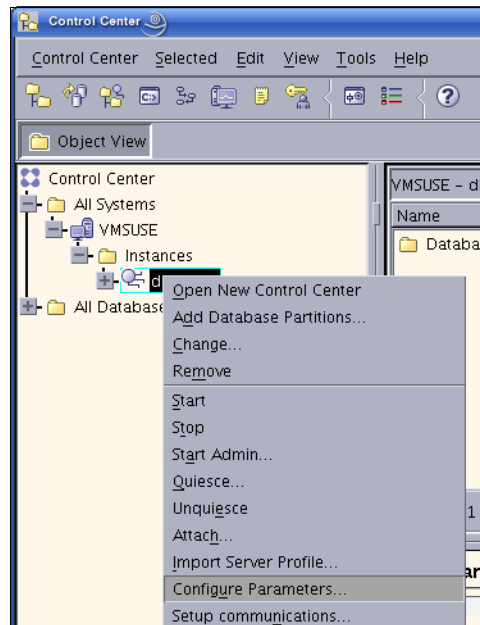


Figure 3-20 Configure DBM parameter in Control Center

- Make your updates and mark *Update when available* if it is there, for every parameter that you want to change. At the bottom of the window is a brief description about the selected parameter (Figure 3-21 on page 135).

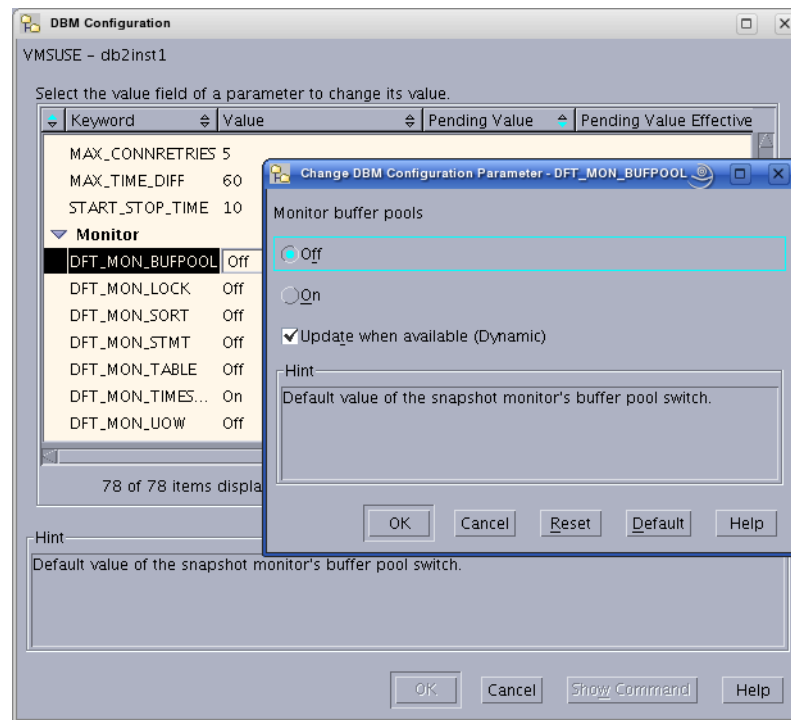


Figure 3-21 Change parameter

- Once all the desired changes have been made, click **OK** and these changes will be applied (Figure 3-22).

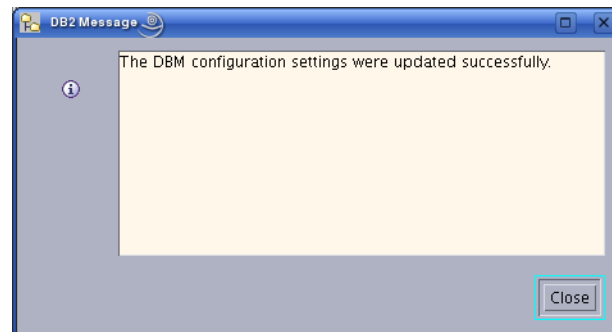


Figure 3-22 Changes of DBM parameters finished

The default behavior of the **UPDATE DBM CFG** command is to apply the change immediately. If you do not want the change to be applied immediately, use the **DEFERRED** option on the **UPDATE DBM CFG** command.

Command line

Commands to change the setting of the DBM configuration parameters can be quickly and conveniently entered by using the command line processor.

Use **GET DATABASE MANAGER CONFIGURATION** to show current values.

Use **UPDATE DATABASE MANAGER CONFIGURATION** **USING** **<parameter>** to change current values.

Use **RESET DATABASE MANAGER CONFIGURATION** to reset all database manager parameters to their default values.

Use **AUTOCONFIGURE** to let DB2 determine the values.

Example 3-35 shows some examples of DBM parameter changes.

Example 3-35 Example for DBM configuration change

```
db2 get dbm cfg
db2 update dbm cfg using DFT_MON_BUFPOOL on automatic
db2 get dbm cfg show detail
db2 autoconfigure using mem_precent 60 apply db and dbm
```

3.6.3 Database configuration

For allocation and control of system resources, DB2 uses a separate configuration file for every database in a DB2 instance. This file is named **SQLDBCON** and is stored in **/database_path/\$DB2INSTANCE/NODE0000/SQL00001**.

In partitioned environment, the **SQLDBCON** exists in every database partition. Therefore, it is possible to have different settings for each partition. To modify these database configuration (DB cfg) parameters, use the Control Center or use the DB2 command line.

db2 update db cfg for <database> using <parameter>

In a partitioned environment you have to issue this command on each partition. The **UPDATE DB CONFIG** command applies to all partitions. If you want to apply it on a particular partition, please use the parameter **DBPARTITIONNUM** **<number>**.

Important DB cfg parameters

This section describes some important database configuration parameters and the recommended settings. Some of the parameters are handled by **STMM**. **STMM** is discussed in 3.6, “DB2 configuration” on page 128.

- **AVG_APPLS** - Can be set to **AUTOMATIC**.

This lets DB2 know how many concurrent applications run in your system in average. When set to 1, DB2 gives every application all current available memory in the buffer pool. Changing this parameter can have a huge impact in the performance.

► **CATALOGCACHE_SZ**

This holds database, table space, table, and index information in memory. DB2 looks first in this cache when it prepares an execution of a plan. Sufficient memory avoids cost-intensive disk I/O. Check with **db2 get snapshot for database on db_name** and compute the hit ratio with:

$100 - ((\text{Catalog cache inserts} \times 100) / \text{Catalog cache lookups})$

► **MAXFILOP**

This is the number of files each application can open. The default value is mostly too small. Opening and closing files slows SQL response times and burns CPU cycles. If files are being closed, adjust the value of MAXFILOP until the opening and closing stops. Use the command:

db2 get snapshot for database on db_name

And, look at *Number of files closed*. If the value is more than 0, increase MAXFILOP value.

► **LOCKTIMEOUT**

The default is set to **-1**. This means a connection can wait until it gets all resources. Deadlock problems occur if a connection already holds locks which are required by another application. To prevent too many deadlock situations, for OTLP set it to **10**; and for DSS set it to **60**.

► **LOGBUFSZ**

DB2 uses a buffer for log records before writing the records to disk. If the value is set too small, it results in unnecessary I/O on the disk. When increasing the value of this parameter, you should consider increasing the **DBHEAP** parameter too, since the log buffer memory is allocated from the database heap.

► **SORTHEAP** - Can be set to **AUTOMATIC**.

Each sort has its own amount of memory that is allocated by the database manager. This parameter defines the maximum sort memory that can be used. When increasing this value you should examine whether the **SHEAPTHRES_SHR** parameter in the database manager configuration file also needs to be adjusted.

► **DATABASE_MEMORY** - Can be set to **AUTOMATIC**.

If it is set to **AUTOMATIC** STMM tunes it according to the available memory in the system and according to the needs of the different tuning areas. If it is set

to a fixed value, the memory for this particular database is limited to that value and STMM tunes the different areas based on that available memory.

► **DFT_DEGREE**

A value of 1 means no intra-partition parallelism. A value of -1 (or ANY) means the optimizer determines the degree of intra-partition parallelism based on the number of processors and the type of query.

On OLTP systems you should turn intra-partition parallelism off. On OLAP systems you should consider to turn it on so that large queries can take advantage of several CPUs in a multiprocessor system.

Note: For a complete and detailed description of all the Database configuration parameters, refer to the Information Center at

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.config.doc/doc/c0004555.html>

Monitor and update DB configuration

Configuring DB2 database settings can be done either through the Control Center or the DB2 Command line.

Control Center

Use these steps to update database configuration parameters:

- Start the Control Center and expand the tree up to the desired database.
- Right-click on it to open the pop-up window and choose **Configure Parameters** (Figure 3-23 on page 139).

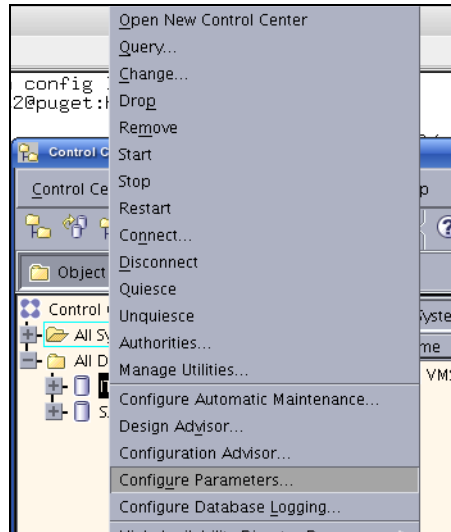


Figure 3-23 Change database parameter from the Control Center

There are two options to modify database configuration parameters:

- ▶ **Configure Parameters:** Use this option to change specific parameters.
- ▶ **Configuration Advisor:** This option is comparable with the **autoconfigure** command.

Using the Configure Parameter option

By selecting the **Configure Parameter** option in the Control Center, the following window appears (Figure 3-24 on page 140). After selecting a parameter, a brief description of this parameter is displayed at the bottom under **Hint**.

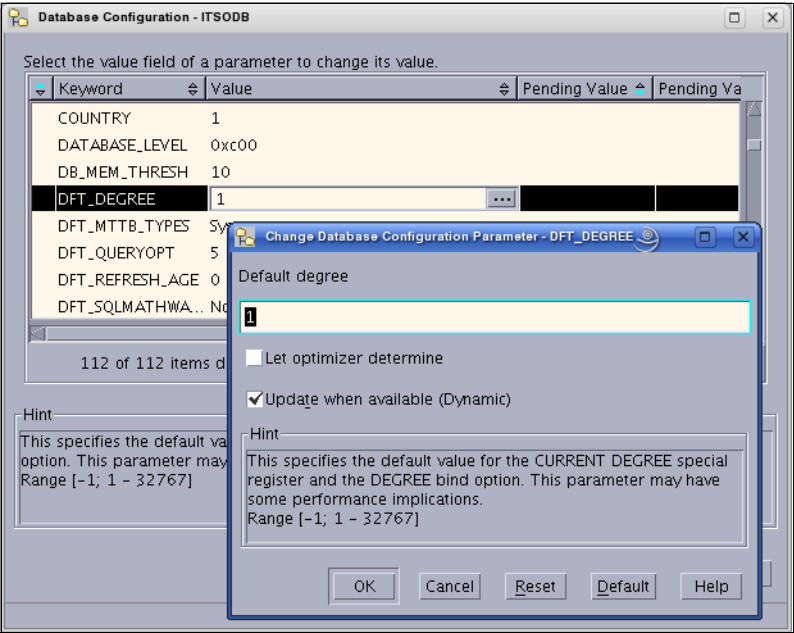


Figure 3-24 Change database parameter

After all desired changes are done, press **OK** and all changes will be applied to the database (Figure 3-25).

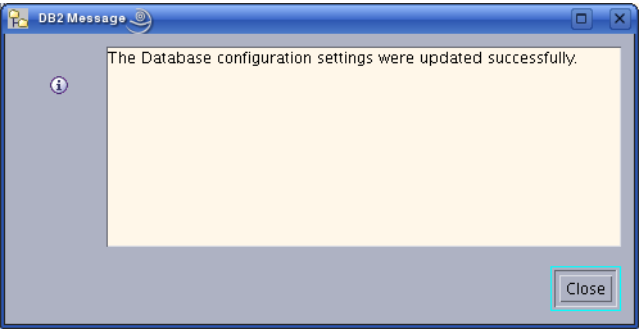


Figure 3-25 Confirmation after update database configuration

Using the Configuration Advisor option

The Configuration Advisor is a smart tool that will help you configure your databases easily. With the Configuration Advisor, you can tune the database without a strong knowledge of DB2. Simply provide the advisor with the requested information about your system, such as planned workload and

connection information, and the wizard will give you a recommendation of what changes you should make. Figure 3-26 shows the main window.

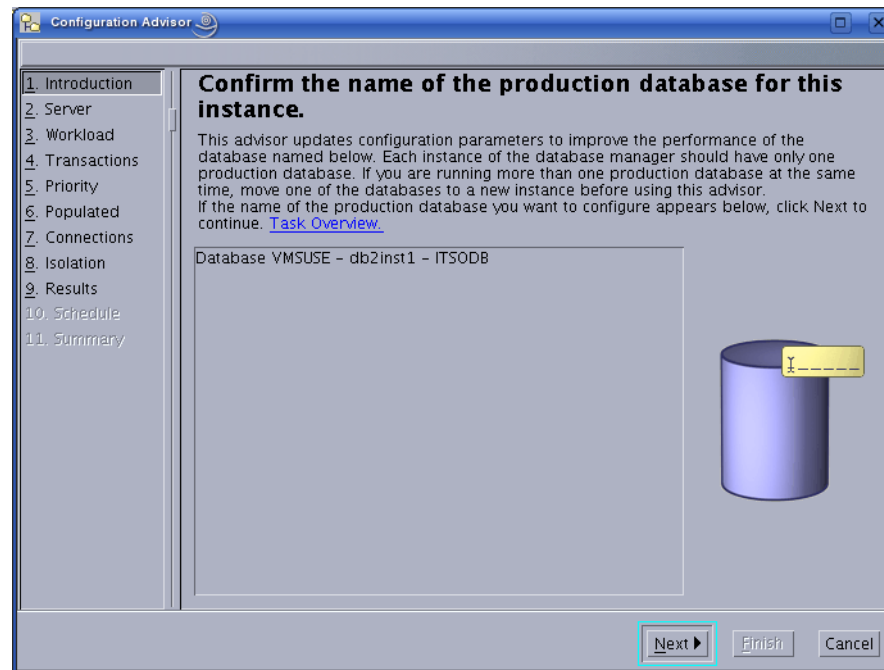


Figure 3-26 Configuration advisor main window

After finishing all sections, the wizard gives you suggestions of what to change. At the end, you have a choice to apply the change immediately or save the SQL to run later. Any parameter can require a reactivation of the database.

Command line

Commands to change the settings can be quickly and conveniently entered by using the command line processor.:

Use **GET DATABASE CONFIGURATION FOR <db_name>** to show current values.

Use **UPDATE DATABASE CONFIGURATION FOR db_name USING <parameter>** to change current values.

Use **RESET DATABASE CONFIGURATION FOR db_name** to reset all database parameters to their default values.

Example 3-36 on page 142 shows database configuration update command examples.

Example 3-36 Examples for the database configuration change

```
db2 get db cfg for itsodb
db2 update db cfg for itsodb using DBHEAP 1200
db2 get db cfg for itsodb show detail
```

3.6.4 DB2 registry and environment variables

All configuration settings in DB2 are stored as either environment variables or registry variables. On UNIX-based systems, all environment variables such as DB2INSTANCE, DB2NODE, DB2PATH, and DB2INSTPROF are set using the export command. Usually in file db2profile located in \$INSTHOME/sql1lib/.

Registry variables are set through the **db2set** command and require a DB2 restart. If a registry variable requires a Boolean argument, the values YES, 1, and ON are all equivalent and the values NO, 0, and OFF are also equivalent. To set registry values for instance level profile, issue:

```
db2set -i <instance_name> <command>
```

To set registry values for global level profile, issue:

```
db2set -g <command>
```

You can use the Configuration Assistant (**db2ca**) or **db2set** command to configure configuration parameters and registry variables. When updating the registry, changes do not affect the currently running DB2 applications or users. Applications started after the update has happened use the new values.

Important registry variables

Here is a list of some important registry variables.

► **DB2_DISABLE_FLUSH_LOG** Default=OFF

When an online backup has completed, the last active log file is truncated, closed, and made available to be archived. This ensures that your online backup has a complete set of archived logs available for recovery.

► **DB2DBDFT** Default=null

This specifies the database alias name to be used for implicit connects. If any application has no database connection defined but contains an SQL statement, it will connect to the database specified in DB2DBDFT.

► **DB2_PARALLEL_IO** Default=null

This enables DB2 to read and write I/O in parallel for a specific table space or all table spaces. The degree of parallelism is determined by the prefetch size and extent size for the containers in the table space.

► **DB2COMM** Default=null

This determines which protocol will be used to communicate with DB2. Usually TCP/IP between server and client accept communication to S390 systems. On Linux, TCP/IP is the only supported protocol for client/server communications.

► **DB2INSTANCE** set by the DB2 profile

After you have enabled your user for DB2 by executing the `db2profile` script this environment variable is set to the name of the current instance.

Note: Details of all the registry variables are described in the Information Center at

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.regvars.doc/doc/c0007340.html>

db2set on command line

db2set is the command to set registry variables. The command can be executed using the command prompt (Example 3-37).

Example 3-37 Setting and deleting a registry variable

```
db2inst1@puget:~> db2set
db2inst1@puget:~> db2set DB2COMM=tcPIP
db2inst1@puget:~> db2set
DB2COMM=tcPIP
db2inst1@puget:~> db2set DB2COMM=
db2inst1@puget:~>
```

db2ca graphical tool

To start the graphical tool, issue the command **db2ca &**. In the main window select **Configure** → **DB2 Registry** as shown in Figure 3-27 on page 144.

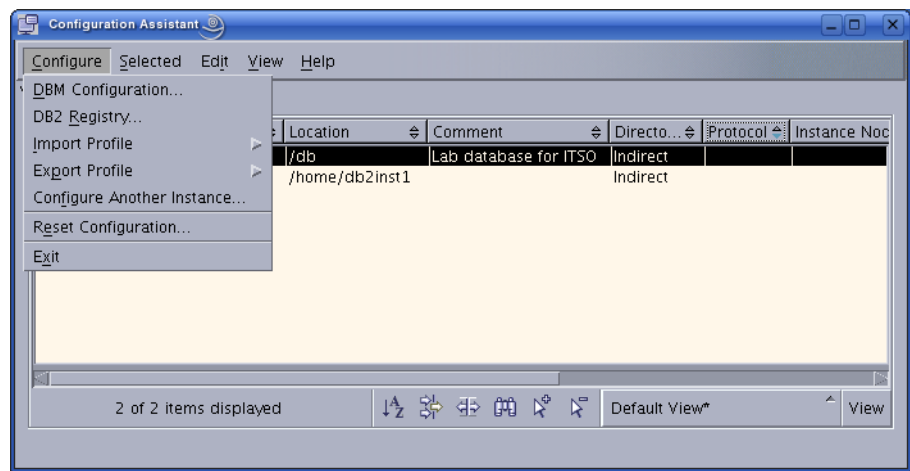


Figure 3-27 Change registry from db2ca

You will get to the panel shown in Figure 3-28, where you can change the registry variables.

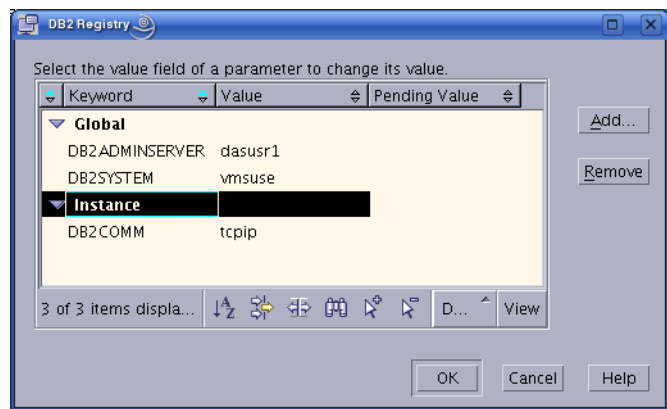


Figure 3-28 Change registry variables

Multi page allocation

For system managed table spaces (SMS) we recommend that you enable multi page file allocation. It reduces the time needed for formatting new pages. To enable this parameter, issue the command `db2empfa` for every database. Exclusive access is needed, because in existing table spaces all empty pages will be filled up to the last extent. To check this, issue the command:

```
db2 get db cfg for <dbname> | grep Multi-page
```

Multi-page file allocation enabled = YES.

In a multi-partition environment issue **db2empfa** on every database partition.

3.7 Security

In this section we briefly discuss the security in DB2. We introduce security in Linux, instance, database, and network levels. For detail information on this subject, refer to the following:

- ▶ *DB2 Security and Compliance Solutions for Linux, UNIX, and Windows*, SG24-7555
- ▶ *Database Security Guide*, SC23-5850-00
- ▶ Information Center under **Database fundamentals** → **Security**

Security on Linux level

On Linux level the security is managed in terms of users and groups. A user has one primary group and can be member of several more groups. The database instance is more or less a dedicated user ID which runs all DB2 processes that are part of this instance. It offers access from the network through a TCP/IP port. Security considerations for DB2 on Linux level are done based on that assumptions.

File System permissions

When an instance is created, DB2 places all DB2 under the home directory of the instance user ID. These files have to be accessible by users who want to use this instance.

All further files you create such as table space containers, transaction logs, archive logs, backups, diagnostics log, and so on. do not need to be accessible by the users directly. Therefore, you can restrict their permission to only the instance user.

Many Linux systems have set the *umask* to 0022 by default that makes all newly created files readable by everyone. If security is a concern, you should consider to set it to 0027 for the instance user.

Fenced user ID

UDFs and stored procedures can be configured so that they run under the fenced user ID to shield the UDFs from the core of the database manager. So, a crashing UDF cannot bring down the database manager. It adds some communication overhead to the call of the UDF, though.

The use of fenced user IDs can serve also another purpose. It allows them to run under a different user ID than the database manger. So they cannot access the files of the database instance directly. It also prevents buffer overflows from taking excessive database resources.

The fenced user ID is defined during the creation of the instance.

Authorities

DB2 instance level authorities, *SYSADM*, *SYSCTRL*, *SYSMAINT*, and *SYSMON*, are granted through operating system group. Each of them provides a certain level of access and permissions to the databases of the instance. In the database manager configuration you can define what Linux groups are assigned to what authorities. The authorized groups are specified in the database manager configuration as shown in Example 3-38.

Example 3-38 Instance level authorities

```
db2inst1@puget:~> db2 get dbm config | grep _GROUP
SYSADM group name          (SYSADM_GROUP) = DB2IADM1
SYSCTRL group name         (SYSCTRL_GROUP) =
SYSMAINT group name        (SYSMAINT_GROUP) =
SYSMON group name          (SYSMON_GROUP) =
```

Security on network level

If networking is enabled, the database manager opens a TCP/IP port which is used by DB2 clients to communicate to the server. Firewalls must be opened for that port and you can use proxies. For instance SOCKS Version 4.

By default all communication between database server and client is not encrypted. If you need to protect the data transfer you can enable SSL support or you can configure the server for *SERVER_ENCRYPT* or *DATA_ENCRYPT* authentication.

The authentication type is defined in the database manager configuration as shown in Example 3-39.

Example 3-39 Authentication parameter in the database manger configuration

```
db2inst1@puget:~> db2 get dbm config | grep AUTHENTICATION
Database manager authentication (AUTHENTICATION) = SERVER
```

Security on database level

On database level you can control data access using authorities and privileges DB2 provided. The data access can be controlled even in the row and column. The security mechanism DB2 offered includes:

- Authorities for the administrators

- ▶ Views for restricting access to a subset of the data
- ▶ Privileges to restrict access on the DB2 objects
- ▶ Packages to restrict access to certain functions
- ▶ UDFs and stored procedures to restrict access to certain functions
- ▶ Roles offer a more detailed way to implement security roles than offered by the Linux groups
- ▶ Encryption to even protect data from accessing by the DBA
- ▶ LBAC to restrict access to data of users on column and row level
- ▶ Trusted context to address the typical problems introduced in a multiple tier architecture

Privileges

On database level privileges can be granted on the following object types:

- ▶ Schemas
- ▶ table spaces
- ▶ Tables and views
- ▶ Packages
- ▶ Indexes
- ▶ Sequences
- ▶ Routines

Privileges can be granted to

- ▶ Linux users
- ▶ Linux groups
- ▶ DB2 roles
- ▶ PUBLIC

To grant privileges, use the **GRANT** statement.

Encryption

If you need to protect sensitive data like credit card information, you can use the built-in functions ENCRYPT, DECRYPT_BIN, and DECRYPT_CHAR. Encryption allows you to protect the sensitive data accessing even from the database administrator.

Defaults

When a database is initially created the following permissions are granted to PUBLIC:

- ▶ CREATETAB
- ▶ BINDADD

- ▶ CONNECT
- ▶ IMPLSCHEMA
- ▶ BIND on all packages created in the NULLID schema
- ▶ EXECUTE on all packages created in the NULLID schema
- ▶ CREATEIN on schema SQLJ
- ▶ CREATEIN on schema NULLID
- ▶ USE on table space USERSPACE1
- ▶ SELECT access to the SYSIBM catalog tables
- ▶ SELECT access to the SYSCAT catalog views
- ▶ SELECT access to the SYSSTAT catalog views
- ▶ UPDATE access to the SYSSTAT catalog views
- ▶ EXECUTE with GRANT on all procedures in schema SQLJ
- ▶ EXECUTE with GRANT on all functions and procedures in schema SYSPROC

If you do not want this you can specify the option **RESTRICTIVE** at the **CREATE DATABASE** command.

3.8 Client configuration

In this section we discuss how to install and configure DB2 clients using the graphical tool Configuration Assistant or the DB2 command line processor.

DB2 clients provide access to DB2 database server. In DB2 Version 9.5, there are three types of DB2 data server clients available:

▶ IBM Data Server Client

This is a full installation client. It includes all the components as well as graphical and non-graphical tools for administering DB2 systems and developing applications with DB2. Add-ins for Eclipse and Microsoft Visual Studio 2005 development environments are also included.

Use this client if you need database administration support, and application development using an application programming interface (API).

▶ IBM Data Server Runtime Client

The Runtime Client is a light-weight, non-graphical client that provides the functionality required for your application to access DB2 servers. It provides application support to run applications that use embedded SQL, CLI, JDBC, SQLJ, ODBC, OLE DB, .NET, and PHP interfaces.

Use this client if you need Command Line Processor (CLP) support and basic client functionality for application runtime and deployment support.

▶ IBM Data Server Driver for ODBC, CLI, and .NET

It is a light-weight deployment solution for Windows applications. Use it if you need runtime support for the DB2 CLI API, ODBC API, and .NET API for Windows applications without the need of installing the Data Server Client or the Data Server Runtime Client.

The client selection depends on your application environment. Only the IBM Data Server Client provides the BIND support. All client versions are available on the AIX, HP-UX, Linux, Solaris, and Windows platforms.

For more information about the IBM Data Server clients and installation requirement details, refer to *Quick Beginnings for IBM Data Server Clients*, GC23-5863-00.

3.8.1 Installing IBM Data Server Client

Once you have checked the prerequisites as specified in 2.1, “Basic requirements” on page 26, perform the following tasks to install IBM Data Server Client:

1. Insert and mount the product DVD. Change to the directory where the DVD is mounted or the install image resides.
2. Run `./db2setup` command to start the DB2 Setup Wizard.
3. Choose **Install a Product** when the DB2 launchpad opens.
4. Select the client you want to install.
5. Follow the DB2 Setup wizards instructions to install the client.

Note: With DB2 Version 9.5 you can install the IBM Data Server Client without root privileges. The DB2 installer automatically creates and configures a non-root instance during the non-root installation. Non-root installation includes most of the functionality of root installation. There are differences and limitations, such as DB2 Control Center and the Configuration Assistant are not available. Please see 2.3.1, “Non-root installation” on page 58 for more information.

If not already done during the installation, you have to create a Data Server Client instance using the following command as root user. In our case, we create a DB2 Data Server Client instance called db2vs.

```
/opt/ibm/db2/V9.5/instance/db2icrt -s client db2vs
```

If you have already installed a DB2 server product on your machine, i.e. DB2 Enterprise Server Edition, you can use the `db2icrt` command with the `-s client` option to create a separate client instance.

Now you should configure the client to access a remote DB2 server. To administer a DB2 database remotely, you must connect to the server using TCP/IP. Both TCP/IPv4 and TCP/IPv6 are supported.

3.8.2 Configuring IBM Data Server Client

In order to accept connection requests from a Data Server Client using TCP/IP, you have to configure the database server.

On the DB2 server, verify the following items:

- ▶ The db2 services name and port is set in the file */etc/services*, for example:

```
db2c_db2inst1 50001/tcp
```

- ▶ DB2COMM registry variable is set for TCP/IP:

Use the **db2set -a11** command and look for **DB2COMM=TCP/IP**

- ▶ TCP/IP port name in database manager configuration should be set:

```
db2inst1@mensa: /> db2 get dbm cfg | grep SVCE
TCP/IP Service name                (SVCENAME) = db2c_db2inst1
```

If changes are done, the DB2 server instance must be restarted.

Now you can configure the DB2 client. There are two tools available for configuring client-to-server communications:

- ▶ **Configuration Assistant**

This is a graphical tool provided with Data Server Client and DB2 server products on Windows and Linux. It is not provided with Data Server Runtime Client.

- ▶ **Command line tools**

The command line tools consist of the Command Line Processor (CLP), the configuration export command **db2cfexp**, and the configuration import command **db2cfimp**.

Configuring Client using the Configuration Assistant

First we show how to configure the client using the graphical tool:

1. Start the Configuration Assistant (Figure 3-29 on page 151) by running **db2ca** & as a valid DB2 user or open it from the DB2 folder.

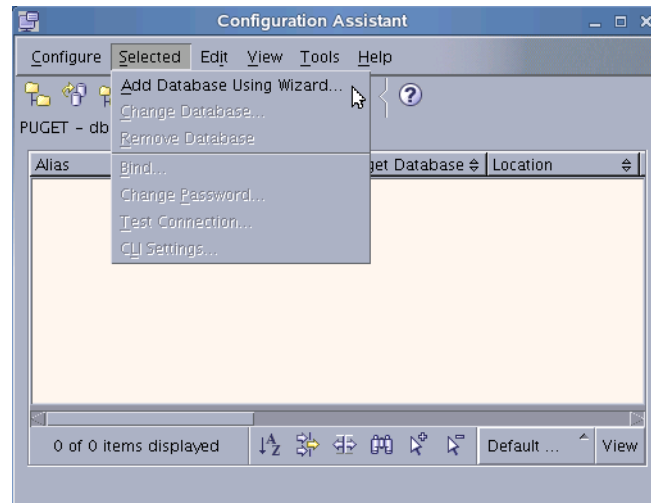


Figure 3-29 Add Database Using Wizard from CA

2. In the next panel (Figure 3-27 on page 126), select how you want to set up the connection to the database server.

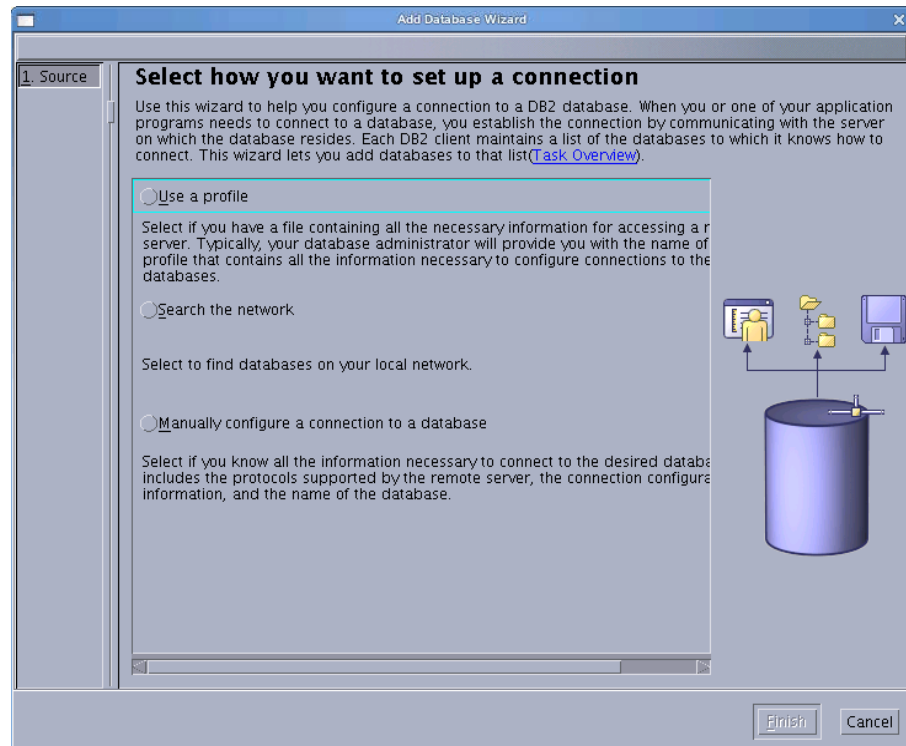


Figure 3-30 Add Database wizard

Select **Search the network** and click **Next**. Expand the tree under **Known systems** until you see all databases on this system. Select your desired database. You can also choose to manually configure the connection if you know all the information necessary to connect to the database. After filling out all required fields, you can test a connection to this database (Figure 3-31 on page 153).

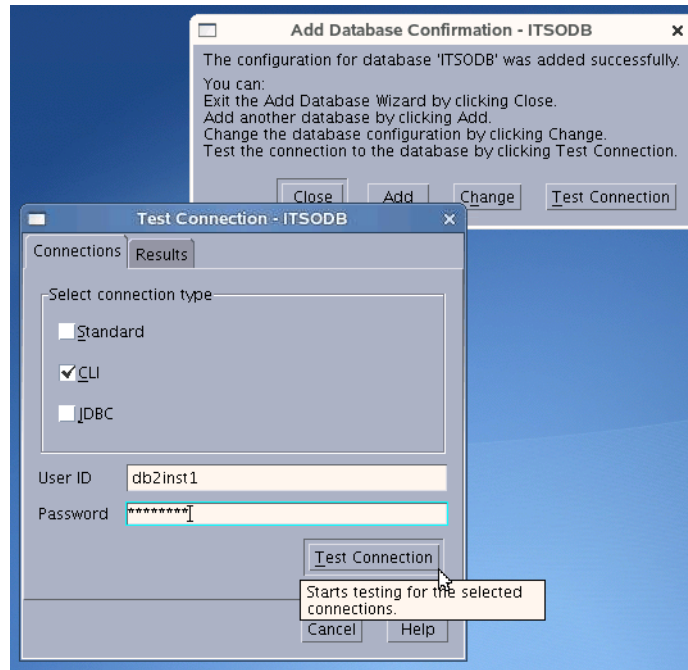


Figure 3-31 Test Connection panel

To configure many client machines, perform all configurations on one client machine and export the definition to a flat file using the Configuration Assistant (Figure 3-29 on page 128). Transfer the configuration profile to each client machine and import the definition using Configuration Assistant

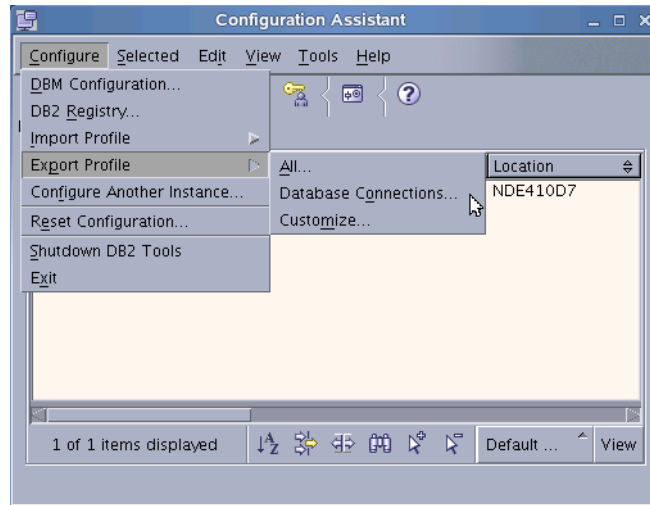


Figure 3-32 Export client configuration definition

Configuring Client using the command line

In addition to Configuration Assistant, you can use DB2 Command Line Processor (CLP) to configure DB2 clients. Example 3-40 shows the commands for configuring a client called `puget` to access our sample database `ITS0DB` on server called `mensa`.

Example 3-40 Configure client using the command line

```
db2vs@puget:~> db2 "catalog tcpip node mensa remote mensa server 50001 ostype
Linux with 'DB Server MENSA' "
DB20000I The CATALOG TCP/IP NODE command completed successfully.
DB21056W Directory changes may not be effective until the directory cache is
refreshed.
db2vs@puget:~> db2 "catalog database ITS0DB as ITS0DB at node mensa
authentication server with 'Test database for ITS0' "
DB20000I The CATALOG DATABASE command completed successfully.
DB21056W Directory changes may not be effective until the directory cache is
refreshed.
db2vs@puget:~> db2 list node directory
```

Node Directory

Number of entries in the directory = 1

Node 1 entry:

Node name	= MENSA
Comment	= DB Server MENSA

```

Directory entry type      = LOCAL
Protocol                  = TCPIP
Hostname                  = mensa
Service name              = 50001

```

```
db2vs@puget:~> db2 list db directory
```

```
System Database Directory
```

```
Number of entries in the directory = 1
```

```
Database 1 entry:
```

```

Database alias            = ITSODB
Database name             = ITSODB
Node name                 = MENSA
Database release level    = c.00
Comment                  = Test database for ITS0
Directory entry type      = Remote
Authentication            = SERVER
Catalog database partition number = -1
Alternate server hostname =
Alternate server port number =

```

```

db2vs@puget:~> db2 connect to itsodb user db2inst1
Enter current password for db2inst1:

```

```
Database Connection Information
```

```

Database server          = DB2/LINUX8664 9.5.0
SQL authorization ID     = DB2INST1
Local database alias     = ITSODB

```

Note: For more details about DB2 commands, refer to the *Command Reference*, SC23-5846-00.

3.9 Configuring licensing

Each DB2 product and DB2 feature comes with its own license key. To assist you in managing your licenses, a compliance report lists the compliance or noncompliance of DB2 features with your current product entitlement. To be in compliance with your license agreement, you should apply the license key. The management of all these licenses are done through one of the following tools:

- ▶ License Center within the Control Center
- ▶ **db2licm**, a license management tool command

This section provides instructions about how to configure licensing using both of these tools. We discuss the following topics:

- ▶ Installing a license key
- ▶ Setting up a license policy
- ▶ Changing the enforcement policy
- ▶ Generating a compliance report

If you installed a DB2 product with a *Try and Buy* license and now you want to upgrade to a full license, you do not need to reinstall the DB2 product. You simply upgrade your license.

3.9.1 DB2 License Center

The DB2 License Center is a graphical tool within the Control Center that manages licensing. You can use this tool to view license status and usage information for DB2 products installed on your system, add and remove license keys and specify user and enforcement policies. It can also generate usage statistics and display connection information for current users. To access the License Center, do the following:

1. Log in as instance owner.
2. From Control Center, select **License Center** from the **Tools** menu.
3. Select the system and product on which you want to configure licensing. You may configure a local or remote system. Select **DB2 Enterprise Server Edition** from the Installed products menu as shown in Figure 3-33 on page 157

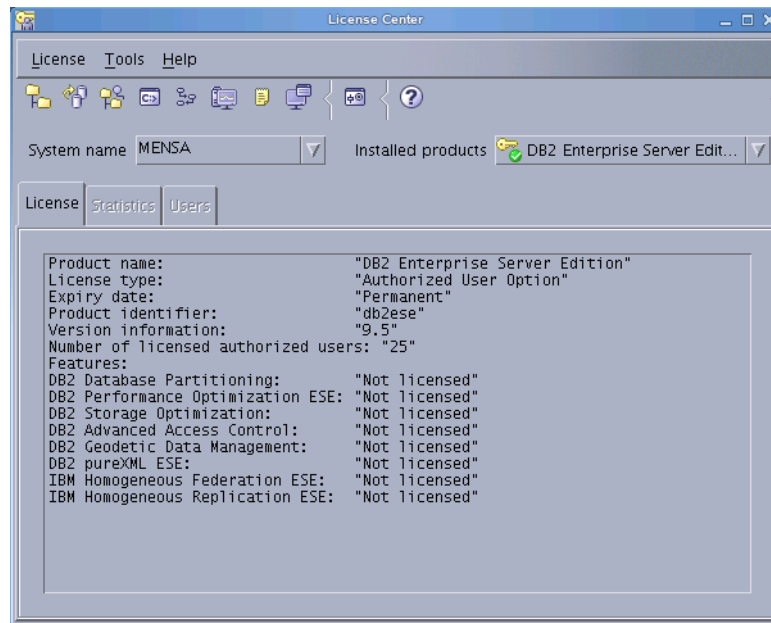


Figure 3-33 DB2 License Center

Installing a license key

To install a license key using the License Center, you must add a *db2*.lic* license file as follows:

1. Select **Add** from the License menu.
2. Select the **From a file** option and choose the appropriate directory. If you purchased a fully licensed DB2 product, the *db2ese.lic* license file is located in */cdrom/db2/license*, where */cdrom* is the root directory of your CD-ROM. Click **OK**.

To remove a license file, select **Remove** from the License menu.

Setting up a license policy

For *DB2 Connect Enterprise Server Edition* the license policy controls and monitors the number of users that can connect simultaneously to a DB2 Connect server. For *WebSphere Replication Server* or *WebSphere Federation Server*, the license policy controls and monitors the number of connectors to a non-DB2 data source.

To setup the license policy using the License Center, do this:

1. Select **Change** from the **License** menu.

2. In the Change License window, select the type of license that you have purchased for your product.

Changing the enforcement policy

The default enforcement policy is *Soft Stop*. This means that all connections to DB2 databases go through, even when the license policy is violated. When a violation occurs, a license violation message is written to `db2diag.log`. The *Hard Stop* enforcement policy does not permit any connections that violate the license policy. To specify the enforcement policy type, do the following:

1. Select **Change** from the License menu.
2. Select the appropriate radio button under *Enforcement policy*.

Generating a compliance report

To verify license compliance of DB2 features, you can generate a compliance report. It lists DB2 features out of compliance with your current product entitlement. Each DB2 feature status is listed as one of the following:

- In compliance: The feature has been used and is properly licensed.
- Violation: The feature is not licensed and has been used.

You can generate a compliance report using the License Center by selecting **Generate Compliance Report** from the License menu. Figure 3-34 shows the report.

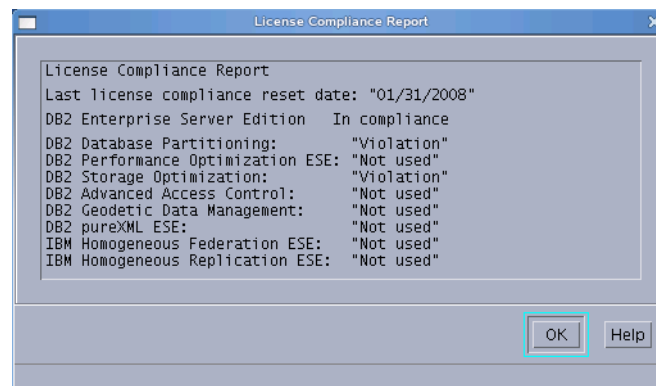


Figure 3-34 Compliance Report

The License Center can also be used to reset the license usage information by selecting **Reset Compliance Report** from the License menu.

3.9.2 db2licm tool

db2licm is a command line tool that manages licensing. It is used to display license information, add and remove license keys, and specify user and enforcement policies. It can also generate usage statistics and display connection information for current users.

To view all of your licensing information, enter **db2licm -l** as shown in Example 3-41.

Example 3-41 View DB2 licensing information using db2licm command

```
db2inst1@mensa: /> db2licm -l
Product name:                "DB2 Enterprise Server Edition"
License type:                 "Authorized User Option"
Expiry date:                  "Permanent"
Product identifier:           "db2ese"
Version information:           "9.5"
Number of licensed authorized users: "25"
Features:
DB2 Database Partitioning:    "Not licensed"
DB2 Performance Optimization ESE: "Not licensed"
DB2 Storage Optimization:     "Not licensed"
DB2 Advanced Access Control:  "Not licensed"
DB2 Geodetic Data Management: "Not licensed"
DB2 pureXML ESE:              "Not licensed"
IBM Homogeneous Federation ESE: "Not licensed"
IBM Homogeneous Replication ESE: "Not licensed"
```

The **ENV_FEATURE_INFO** administrative view and the **ENV_GET_FEATURE_INFO** table function also return information about all available features for which a license is required. For each feature, there is information about whether a valid license for the feature is installed. A sample output is shown in Example 3-42

Example 3-42 Using the ENV_FEATURE_INFO view

```
db2inst1@mensa: /> db2 "select * from sysibmadm.env_feature_info"
```

FEATURE_NAME	FEATURE_FULLNAME
LICENSE_INSTALLED	PRODUCT_NAME
FEATURE_USE_STATUS	
DPF	DB2_DATABASE_PARTITIONING_FEATURE
N	ESE
	NOT_USED
POESE	DB2_PERFORMANCE_OPTIMIZATION_FEATURE_FOR_ESE
N	ESE
	NOT_USED
SO	DB2_STORAGE_OPTIMIZATION_FEATURE
N	ESE
	NOT_USED
AAC	DB2_ADVANCED_ACCESS_CONTROL_FEATURE
N	ESE
	NOT_USED

GEO		DB2_GEODETTIC_DATA_MANAGEMENT_FEATURE
N	ESE	NOT_USED
XMLESE		DB2_PUREXML_FEATURE_FOR_ESE
N	ESE	NOT_USED
HFSE		IBM_HOMOGENEOUS_FEDERATION_FEATURE_FOR_ESE
N	ESE	NOT_USED
HRESE		IBM_HOMOGENEOUS_REPLICATION_FEATURE_FOR_ESE
N	ESE	NOT_USED

8 record(s) selected.

Installing a license key

To install a license key, you must add the *db2*.lic* license file using the **db2licm** command. To add a DB2 license file:

1. Login as instance owner.
2. Enter the following command:

```
db2licm -a <filename>
```

Where *filename* is the full path name and file name of the license file, *db2ese.lic*. The license file is available from either the DB2 product or feature image you downloaded from *Passport Advantage*, or from the Activation CD you received in the physical media pack.

On Linux servers, the license file is located in `/cd/db2/license`. If you are setting up a multi-partitioned database system, perform this task on each machine.

To remove a license file, use **db2licm -r**. For example:

```
db2licm -r db2ese
```

Setting up a license policy

For *DB2 Connect Enterprise Server Edition* the license policy controls and monitors the number of users that can connect simultaneously to a DB2 Connect server. For *WebSphere Replication Server* or *WebSphere Federation Server*, the license policy controls and monitors the number of connectors to a non-DB2 data source.

Perform the following task:

1. Log in as the instance owner.
2. If you purchased a *DB2 Connect Server Concurrent User* policy, enter the following command:

```
db2licm -p db2consv concurrent
```

If you purchased a *WebSphere Replication Server* or *WebSphere Federation Server Concurrent Connector* policy, enter the following command:

```
db2licm -c wsfs concurrent
```

or

```
db2licm -c wsrs concurrent
```

If you are not sure about the product identifier (wsfs or wsrs), use the following command to find it out:

```
db2licm -l
```

Changing the enforcement policy

The default enforcement policy is *Soft Stop*. This means that all connections to DB2 databases go through, even when the license policy is violated. When a violation occurs, a license violation message is written to db2diag.log. The *Hard Stop* enforcement policy does not permit any connections that violate the license policy. To change the enforcement policy from *Soft Stop* to *Hard Stop*, enter the following command:

```
db2licm -e db2ese HARD
```

Likewise, to change the enforcement policy from *Hard Stop* to *Soft Stop*, enter:

```
db2licm -e db2ese SOFT
```

Generating a compliance report

To verify license compliance of DB2 features, you can generate a compliance report. It lists DB2 features out of compliance with your current product entitlement. Each DB2 feature status is listed as one of the following:

- ▶ In compliance: The feature has been used and is properly licensed.
- ▶ Violation: The feature is not licensed and has been used.

To generate a compliance report, use the following command:

```
db2licm -g <filename>
```

where <filename> specifies the file name the output is to be stored. To reset the license usage information, use the **db2licm -x** command.



Migration and Fix Packs

When migrating DB2 from Version 8 or 9.1 to Version 9.5, there are several important steps to perform before and after installation to ensure a safe and successful migration.

In this chapter, we cover DB2 server migration from Version 8.2 and 9.1 to Version 9.5 as well as fix pack installation. We discuss:

- ▶ Migration planning
- ▶ Migrating multi-partitioned database
- ▶ Migrating single-partitioned database
- ▶ Post-migration tasks
- ▶ Client and server compatibility
- ▶ Fix pack installation

For complete information about migrating a DB2 server, and other components such as clients and applications in your DB2 environment, refer to the DB2 Information Center at

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.qb.migration.doc/doc/c0023662.html>

DB2 also provides migration tools and services from other databases, such as Oracle and Microsoft® SQL Server®. For further information, refer to this Web site:

<http://www.ibm.com/db2/migration/>

4.1 Migration planning

Prior to installing DB2 9.5, it is essential that you prepare your system and databases for the migration. Additional table space and log space is required for migration, as well as a recovery plan in case of failure. This section discusses migration requirements.

Figure 4-1 illustrate the migration roadmap. You have to consider the entire DB2 environment to determine an optimal migration strategy.

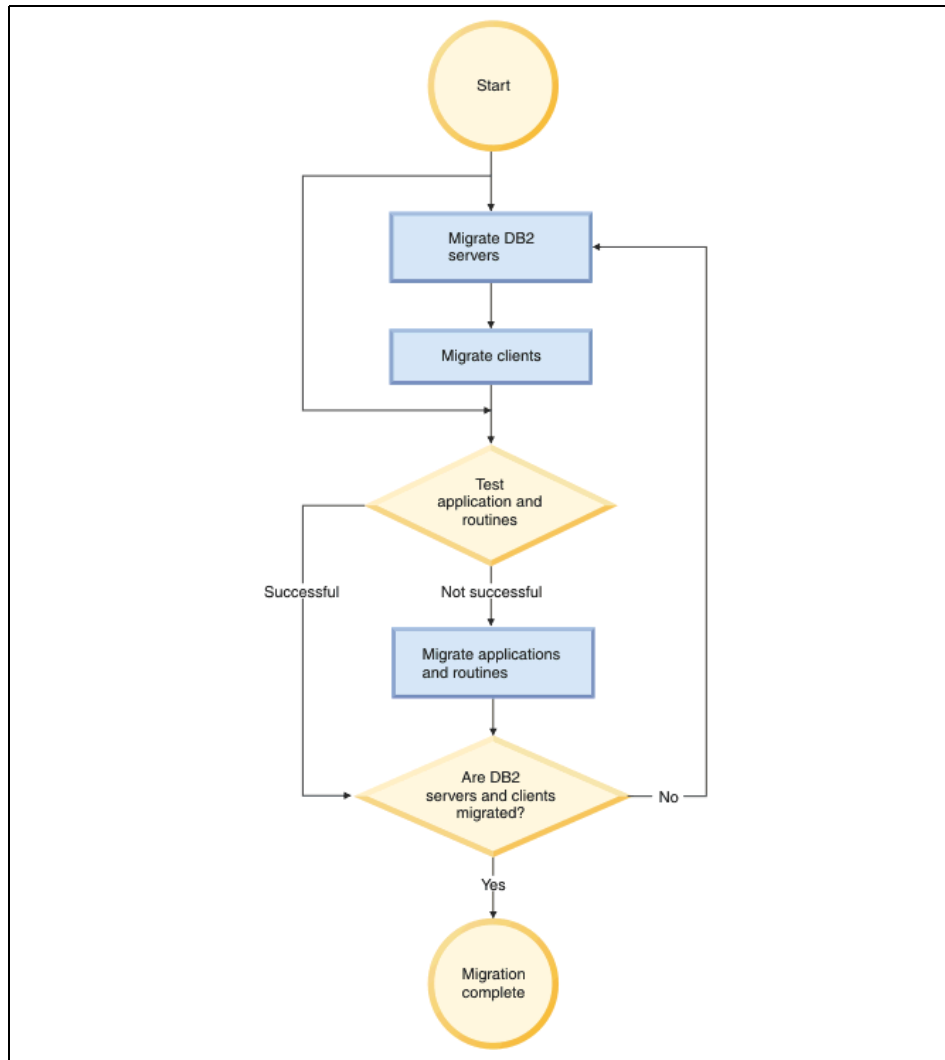


Figure 4-1 Migration roadmap

4.1.1 Migration requirements

Make sure your system satisfies the following software and space requirements prior to Version 9.5 migration.

- ▶ Migration is supported from DB2 UDB Version 8 and DB2 9.1. If you have DB2 UDB Version 7 or earlier, you need to migrate to DB2 UDB Version 8 before migrating to DB2 9.5.

If you are running DB2 UDB Version 8.1, we recommend you upgrade to DB2 UDB Version 8.2 first.

- ▶ You can only migrate to a root installation of DB2 9.5.
- ▶ DB2 9.5 Enterprise Server Edition is only supported on 64-bit architectures.

Check the recommended and validated environment for DB2 9.5 from the following Web site:

<http://www.ibm.com/software/data/db2/linux/validate/>

- ▶ Restoring full database offline backups from DB2 UDB Version 8 or DB2 Version 9.1 is supported. However, rolling forward of logs from a previous level is not possible.
- ▶ Perform hardware and operating system upgrades prior to DB2 database product migration.
- ▶ If you migrate DB2 server, you might also need to migrate your database applications and routines to support changes for 64-bit instances, SQL stored procedures, Java Virtual Machine (JVM™), and development software.
- ▶ Migrate all of your DB2 servers to Version 9.5 before you migrate any of your DB2 clients to Version 9.5. Some problems may occur if you migrate your clients first.
- ▶ If you have 32-bit instances and you migrate to DB2 9.5 on a 64-bit system, you need to manage incompatibilities due to the shared library path specification and discontinued features to run successfully your applications and routines.
- ▶ The files SQLSPCS.1 and SQLSPCS.2 contain table space information. During migration from DB2 UDB Version 8 to DB2 9.5, these files grow to four times their previous size but the total data size on disk does not exceed the new size of the SQLSPCS.1 and SQLSPCS.2 files.
- ▶ Ensure that you have sufficient free space in the /tmp directory on the machine where you will be migrating an instance. The instance migration trace file is written to /tmp. The minimum size required is 20MB.
- ▶ Ensure that you have sufficient space in the file system where the table spaces of the databases you are migrating are located. Additional space is required for both old and new database catalogs during migration. The

amount of required space depends on your system, but some general recommendations are provided in Table 4-1 on page 166.

Table 4-1 Space recommendations for migration

Table space	Space recommendation
System catalog space (SYSCATSPACE)	<ul style="list-style-type: none"> ► For SMS table spaces, you should allocate 2x the space currently occupied ► For DMS table spaces, free pages should be equal to or greater than used pages
Temporary table space (TEMPSPACE1, by default)	<ul style="list-style-type: none"> ► Total pages for the system temporary table space should be twice the amount of total pages for the system catalog table space. ► If you are upgrading from Version 8, check the row size in your result sets. If it is close to the maximum row length limit for your existing system temporary table spaces, you might need to create a system temporary table space with a larger page size. This is because DB2 Version 9.1 and Version 9.5 use larger record identifiers (RID) that increases the row size in the result sets from queries or positioned updates.

To check the size of your DMS table spaces, use the following commands as instance owner:

```
db2 list database directory
db2 connect to dbname
db2 list tablespaces show detail
```

To increase space in a DMS table space, you can add additional containers or increase the size of your existing containers.

To increase space in a SMS table space, you can free sufficient disk space on the corresponding file systems or increase the size of your file systems if you are using a volume manager.

- Ensure that you have sufficient log file space prior to migration. We recommend that you significantly increase the values of LOGFILSIZ, LOGPRIMARY, and LOGSECOND to prevent log file space from running out. The amount of required space depends on your system setup. Also on partitioned database environments, you only need to increase the log space in the catalog database partition server

For instance, if you want to change the size of LOGFILSIZ from 1000 to 2000, connect to the database and enter the following command:

```
db2 update db cfg using logfilsiz 2000
```

- ▶ Ensure that you have enough free pages in the corresponding index table space to account for one additional page per index on populated tables before:
 - Populating tables in new databases created in DB2 9.5, real-time statistics are enabled by default in these newly created databases.
 - Enabling deferred cleanup roll out by setting DB2_MDC_ROLLOUT to DEFER, or when DB2WORKLOAD is set to SAP®.
 - Reorganizing or recreating indexes on populated tables.

4.1.2 Planning considerations

DB2 9.5 brings new features to enhance performance and manageability. Some of the features and functionality are also discontinued. The default values for several configurations parameters and registry variables have changed and should be checked.

Creating database to enable new features

Migration time is a good opportunity to consider creating a new database to leverage the full advantages of DB2 9.5 features and functions or change the database settings which can only be done in a new database. The considerations include

- ▶ Automatic storage: This feature can only be enabled at the database creation time.
- ▶ Database code page: The database code page determines what characters you can store in the database. You can not change the code page once the database is created. In DB2 9.5, the default code page default is Unicode.
- ▶ Database territory: This determines the date and time formats. This option must be defined during database creation.
- ▶ Database collating sequence: The order in which character data is sorted in a database depends on the structure of the data and the collating sequence defined for the database. The database collating sequence is specified during database creation time and can not be changed afterward.

Discontinued and deprecated functionality

Here are some of the discontinued and deprecated functionality in Version 9.5:

- ▶ -w option for db2icrt, db2ilist, and db2iupdt
- ▶ db2undgp command
- ▶ Some registry and environment variables, like:
DB2LINUXAIO, DB2_LGPAGE_BP.

- ▶ IMPORT command options CREATE and REPLACE_CREATE are deprecated
- ▶ XML Extender is deprecated

When you are migrating DB2 UDB Version 8 you have to review changes that occurred in Version 9.1 too. The following are some of them:

- ▶ Alternate Fix Pack images are discontinued.
- ▶ Type 3 JDBC support is discontinued.
- ▶ The Data Warehouse Center and the Information Catalog Center are no longer included.
- ▶ Text Extender is no longer supported.
- ▶ Audio, Image, and Video (AIV) Extenders are no longer supported.
- ▶ DB2_SCATTERED_IO registry variable is discontinued.

4.1.3 Migration test consideration

The entire migration process consists of two parts: (1) Installing DB2 9.5 code, and (2) Migrating instances and databases. Installing DB2 9.5 does not require system down time. Migrating instances and databases is the actual step which converts your DB2 system from Version 8 or 9.1 to Version 9.5. You need to stop DB2 to perform the instances and databases migration.

We recommend that you install and test Version 9.5 before migrating any Version 8 or Version 9.1 instances to Version 9.5. If you have a test environment which is a mimic of the production system, you can install DB2 Version 9.5, migrate DB2 instances and databases, test all your applications on the test system, and then carry the same procedure to the production environment. However, if you have limited test environment or no test environment, you can utilize the DB2 features to test your applications under the new version.

Since multiple versions of DB2 can coexist on Linux, you can install DB2 9.5 while the application is still up and running under Version 8 or 9.1. You can then create Version 9.5 test instances and databases with a small amount of data to test your applications while production transactions are running.

When you are convinced to move your production systems up to Version 9.5, after making any required changes in your applications or environment, you can do the instances and databases migration during off-peak times to reduce system down time and reduce the number of people or systems affected by that downtime.

We do not recommend this migration in your production environment without test.

4.2 Migrating multi-partitioned database

This section provides a list of pre-migration tasks and the migration procedures for partitioned databases. The procedure to migration a multi-partitioned database from DB2 Version 8 or 9.1 to Version 9.5 are, in general, the same. The high level steps consist of the following:

- ▶ Migration preparation: This step prepares the instance and database for migration include backing up databases and checking if the database is ready for migration using the db2migr command,
- ▶ DB2 Version 9.5 installation.
- ▶ Migrating instance and databases: Actual migrating instance and database.
- ▶ Post migration: Tasks to ensure that the DB2 servers perform as expected.
- ▶ Enabling new DB2 9.5 functionality.
- ▶ Migrating Administration Server.
- ▶ Migrating DB2 clients.

4.2.1 Pre-migration tasks

Perform the following steps to prepare for a safe and successful migration:

1. The first step to migrate your environment is to determine if you can migrate your existing instances to a DB2 9.5 copy that you have.
 - a. Determine the node type using the following command:

```
db2 get dbm cfg | grep "Node type"
```

Example 4-1 shows the node type of an ESE DB2 system.

Example 4-1 Node type

```
dpfusr1@gemini:~> db2 get dbm cfg | grep "Node type"
Node type = Enterprise Server Edition with local and remote
clients
```

- b. Review Table 4-2 to determine the instance type and whether the instance migration is supported.

Table 4-2 Node types

Instance type	Node type	Migration support
client – default type for DB2 clients	Client	<ul style="list-style-type: none"> ► Migration to a client, a standalone, a wse, or an ese instance is supported.
standalone – default type for DB2 Personal Edition (PE)	Database server with local clients	<ul style="list-style-type: none"> ► Migration to a standalone, a wse, or an ese instance is supported. ► Migration to a client instance is unsupported.
wse – default type for DB2 Workgroup Server Edition (WSE)	Database server with local and remote clients	<ul style="list-style-type: none"> ► Migration to a wse or an ese instance is supported. ► Migration to a standalone instance creates a standalone instance. ► Migration to a client instance is unsupported.
ese – default type for DB2 Enterprise Server Edition (ESE)	Partitioned database server with local and remote clients or Enterprise Server Edition with local and remote clients	<ul style="list-style-type: none"> ► Migration to an ese instance is supported. ► Migration to a standalone or a wse instance from single database partition environments creates a standalone or wse instance ► Migration to a client instance is unsupported.

2. Save DBM and DB configuration settings. You should compare these settings before and after migration to check for any migration errors.

Save the DB configuration for each database partition, the DB configuration could be different.

3. Save table space and package information using the **db2 list tablespaces show detail** and **db2 list packages show detail** commands. You may also want to compare this information before and after migration.
4. Back up DB2 server configuration and diagnostic information.
 - a. Run the **db2support** command for all the databases that you are going to migrate in all of your instances. See Example 4-2.

Example 4-2 db2support

```
dpfusrl@gemini:~> db2support /dpfhome/dpfusr1 -d dpfsamp -cl 0
```

- b. As external routine libraries are not backed up with other database objects when a database backup is performed, back up all your external routines. Example 4-3 shows how to backup all external routines created using the default path.

Example 4-3 external routines backup

```
cp -R /dpfhome/dpfusr1/sqlllib/function /dpfhome/dpfusr1/routine_backup
```

- 5. If you use distributed transactions involving DB2 databases, ensure that the databases to be migrated do not contain any indoubt transactions by using the LIST INDOUBT TRANSACTIONS command to get a list of indoubt transactions and to interactively resolve any indoubt transactions.
- 6. During the database migration to DB2 Version 8 the EXECUTE privilege is granted to public for all existing functions, methods, and external stored procedures. You can use the **db2undgp** command to revoke this privilege on all these routines. **db2undgp** is no longer supported in DB2 9.5. If you want to revoke the EXECUTE privilege using this command you have to do it before the migration to DB2 9.5. If you ran the db2undgp command after migrating your databases to DB2 UDB Version 8 or DB2 9.1, you do not have to run this command again after migrating to DB2 9.5.

Example 4-4 shows a db2undgp execution. In this example, a file that contains all the REVOKE statements needed to remove the EXECUTE privilege from PUBLIC is created. You can review or edit this file.

Example 4-4 db2undgp

```
dpfusr1@mensa:~> db2undgp -d dpfsamp -o /dpfhome/dpfusr1/revoke.db2
db2undgp processing begins for database 'dpfsamp'.
db2undgp has completed successfully.
db2undgp complete successfully for database 'dpfsamp'.
```

Grant the EXECUTE privilege on all your routines to specific users or group only. The following statement shows how to grant this privilege on all functions under a specific schema:

```
db2 grant execute on function <schema-name>.* to <user>
```

- 7. If the DIAGLEVEL parameter is set to 2 or less, set this parameter to 3 or higher before migration. Issue the following command:
db2 update dbm cfg using diaglevel 3
- 8. Disconnect all the connections in Version 8 or Version 9.1 databases:
 - a. Stop the DB2 license service using **db2licd -end**
 - b. Stop all command line processor sessions by issuing **db2 terminate** on each session window.

- c. Disconnect all applications and users.

For a list of all database connections, enter the **db2 list applications** command. You can disconnect applications and users by entering **db2 force applications all**.

- 9. If you use replication, archive all of your DB2 log files using the **archive log** command. This command closes and truncates the active log file for your database. The command syntax is as follows:

```
db2 archive log for database <database_alias> user <user_name> using
<password>
```

- 10. Back up databases.

We suggest you do a full offline backup from each local database. You can use **DB2_ALL** command with the **BACKUP DATABASE** command to back up a database in a partitioned database environment. See Chapter 6, “Administering databases” on page 223 for more details about this option and others ways to perform a back up.

- 11. If you are using raw devices for database logs or table space containers, you have to change them to block devices.

- a. Ensure that backup taken at step 10 is successful.
- b. Shut down your database and consider putting the database in quiesce mode.
- c. Use the raw **-a** Linux command to see which raw bindings you defined.
- d. Create a configuration file for the **db2relocatedb** command for each database partition that requires changes. Use the clauses **CONT_PATH** and **LOG_DIR** to specify the old value with the new value. Do not forget to include **NODENUM** value in the configuration file.
- e. Execute **db2relocatedb** command using your definition created in previous step in each database partition that requires changes.
- f. Activate the database

- 12. As instance owner, stop the instance by issuing **db2stop**.

Login as root and issue **ps -ef | grep db2** to check for any outstanding processes on the instance you are migrating. If any processes are still running, such as *db2bp*, kill the process using the following command:

```
kill -9 <pid>
```

Where, *pid* represents the process ID number.

This process have to be done in all database partition servers. Pay attention if there are others instances before kill the processes.

13. Run **db2ckmig** in each database partition to verify that a database can be migrated. The **db2ckmig** command verifies that all of the following conditions are true:

- A catalogued database actually exists.
- A database is not in an inconsistent state.
- A database is not in a backup pending state.
- A database is not in a restore pending state.
- A database is not in rollforward pending state.
- Table spaces are in a normal state.
- A database does not contain user-defined types (UDTs) with the name ARRAY, BINARY, DATALINK, DECFLOAT, VARBINARY, or XML.
- A database does not have orphan rows in system catalog tables that would cause database migration to fail.
- A database enabled as an HADR primary database allows successful connections.
- A database is not in HADR standby role.
- If SYSCATSPACE is a DMS table space and AUTORESIZE is not enabled, SYSCATSPACE has at least 50% free pages of total pages.

The command syntax is as follows:

```
./db2ckmig -e -l logfile_name
```

In Example 4-5, we scanned all cataloged databases and saved the results to a text file **db2ckmig.log**.

Example 4-5 db2ckmig command

```
dpfusr1@mensa:/opt/ibm/db2/V9.5/bin> ./db2ckmig -e -l  
/dpfhome/dpfusr1/db2ckmig.log
```

```
db2ckmig was successful. Database(s) can be migrated.
```

We checked **db2ckmig.log** for any errors and to ensure that the version of **DB2CKMIG** being run was Version 9.5, and not a previous version. Example 4-6 shows the **db2ckmig.log** file. If **db2ckmig** encounter any error when checking the database, actions to fix the errors have to be done before the migration.

Example 4-6 db2ckmig output

```
dpfusr1@mensa:/opt/ibm/db2/V9.5/bin> more /dpfhome/dpfusr1/db2ckmig.log  
Version of DB2CKMIG being run: VERSION 9.5.
```

4.2.2 Install DB2 Version 9.5

DB2 should be installed in all machines of the partitioned environment. We provide the installation details in Chapter 2, “Installation” on page 25.

4.2.3 Migrating instances and databases

All the steps we have done up to now are to prepare for the instances and databases migration. The **db2imigr** command is used to migrate the instances to Version 9.5 format. The **db2 migrate database** command is used to migrate the databases.

The **db2imigr** command performs the following:

- ▶ Check cataloged databases to make sure they are ready for migration through db2ckmig.
- ▶ Migrate your instance to a Version 9.5 instance.
- ▶ Update system and local database directories to the Version 9.5 format.
- ▶ Merge the prior 9.5 version DBM configuration settings presented in your environment with new Version 9.5 DBM configuration settings.

In the remaining of this section, we guide you through a procedure to complete the migration. The steps are

1. Log in as the instance owner to the database partition server that owns the instance. The first entry of the db2node.cfg file shows the database partition server that owns the instance. Example 4-7 shows a db2nodes.cfg. In this example, mensa is the database partition server that owns the instance.

Example 4-7 Shows mensa as database partition server that owns the instance

```
dpfusr1@mensa:~> more /dpfhome/dpfusr1/sql1lib/db2nodes.cfg
0 mensa 0
1 gemini 0
```

2. Stop the instance by running the db2stop force command.
3. Login as root and run the **db2imigr** command to migrate your instance to Version 9.5. Use the following syntax:

```
$DB2DIR/instance/db2imigr [-u <fenced_user>] instance_name
```

Where DB2DIR is set to the location you specified during DB2 9.5 installation, fencedID is the user name under which the fenced user-defined functions and stored procedures will run, and InstName is the login name of the instance owner. The fenced user is only required if you are migrating from a client instance to a server instance. Our input and output is shown in Example 4-8.

Example 4-8 Migrating the instance

```
mensa:/opt/ibm/db2/V9.5/instance # ./db2imigr dpfusr1
```

```
db2ckmig was successful. Database(s) can be migrated.
```

```
DBI1070I Program db2imigr completed successfully.
```

4. At this point in time, you may want to check the DB2 level to ensure that your instance is now a Version 9.5 instance as shown in Example 4-9.

Example 4-9 Check db2level

```
dpfusr1@gemini:~> db2level
DB21085I Instance "dpfusr1" uses "64" bits and DB2 code release "SQL09050"
with level identifier "03010107".
Informational tokens are "DB2 v9.5.0.0", "s071001", "LINUXAMD6495", and Fix
Pack "0".
Product is installed at "/opt/ibm/db2/V9.5".
```

5. Optional: rename the db2diag.log using **db2diag -A** command. This is to give a fresh start of db2diag.log for easy error identification later.
6. Move any existing dump files, trap files, and alert log files in the directory indicated by the DIAGPATH parameter to another directory.
7. Migrate all databases to Version 9.5 format.

Log in as the instance owner and start instance using **db2start** as shown in Example 4-10.

Example 4-10 Starting the instance

```
dpfusr1@mena:~/sql1lib/db2dump> db2start
01/30/2008 17:15:27 0 0 SQL1063N DB2START processing was
successful.
01/30/2008 17:15:28 1 0 SQL1063N DB2START processing was
successful.
SQL1063N DB2START processing was successful.
```

Run the **db2 migrate database** command to migrate the database. The command syntax is as follows:

```
db2 migrate database <database_alias> user <user_name> using <password>
```

Where, *database_alias* represents the database you are migrating, *user_name* represents the name under which the database is migrated, and *password* is that user's password. The user and password options are not required if you want to use the current user for the command. See Example 4-11.

Example 4-11 Migrating the database

```
dpfusrl@gemini:~> db2 migrate database dpfsamp  
DB20000I The MIGRATE DATABASE command completed successfully.
```

Note: The catalog partition must be available when you issue the MIGRATE DATABASE regardless on what database partition you issue this command from.

8. Compare the database configuration settings after migration with the configuration settings before the database is migrated.
 - a. Database manager configuration settings
 - b. Database configuration parameter settings
 - c. Table spaces information
 - d. Application packages information. You do not need to check package information for system generated packages.

4.2.4 Post-migration tasks

After migrating your DB2 servers, you should perform several post-migration tasks to ensure that your DB2 servers perform as expected and at their optimum level.

1. If you set the DIAGLEVEL database manager configuration parameter to 3 or higher as recommended in pre-migration steps, reset this parameter to the value set before the migration.
2. If you changed your log space setting as recommended in “Migration requirements” on page 165 reset the LOGFILSIZ, LOGPRIMARY, and LOGSECOND database configuration parameters to the values they had before migration. Ensure that the amount of log space that you allocate is adequate for your DB2 server.
3. Start up your database with **ACTIVATE DATABASE** and review the administration notification log or the db2diag.log file to verify that all database services are running properly and all buffer pools are activated.
4. There are new registry variables, new configuration parameters, and new default values for registry variables and configuration parameters introduced in DB2 9.5 that can impact the behavior of DB2 server. Also, there are several deprecated and discontinued variables, more detail in:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/topic/com.ibm.db2.luw.qb.migration.doc/doc/t0050543.html>

5. Update statistics.

You must update the statistics for system catalog tables using the RUNSTATS command. You may want to run the **runstats** command on tables, particularly those tables that are critical to the performance of your SQL queries. The **runstats** command updates statistics about physical characteristics of a table and associated indexes. The optimizer uses these statistics to find the best access path to data.

6. Rebind packages.

Any packages stored in a database are invalidated during migration, but packages will be implicitly rebound the first time an application uses them after migrating your database. To avoid this overhead during the user execution time, after migration you can run the **db2rbind** command to rebind these packages as Example 4-12 shows. Review the log file to ensure that everything is fine.

Example 4-12 db2rbind

```
dpfusrl@gemini:~> db2rbind sample -l db2rbind.log all
```

```
Rebind done successfully for database 'SAMPLE'.
```

7. Migrate DB2 Explain tables.

The **migrate database** command does not migrate explain tables. Perform this task if you want to keep explain table information that you previously gathered. If you don't want to keep this information, you can later recreate the explain tables and gather new information. To migrate explain tables, use the **db2exmig** command, as follows:

```
db2exmig -d <database_name> -e <explain_schema> [-u userid password]
```

Where, *database_name* is the name of the database where the explain tables are stored, *explain_schema* represents the schema name of the explain tables to be migrated, and *userid* and *password* are the current user's ID and password. The user ID and password parameters are optional.

8. If you obtained customized code page conversion tables from the DB2 support service, copy all of the files for those tables from the DB2OLD/conv to DB2DIR/conv, where DB2OLD is the location of your DB2 9.1 or DB2 UDB Version 8 copy and DB2DIR is the location of your DB2 9.5 copy. You do not need to copy standard code page conversion tables.
9. After all steps are completed, perform back up for all of your databases.
10. An optional task can be performed now. Check if index or tables need to be reorganized using **REORGCHK** command.

4.2.5 Enabling new DB2 9.5 functionality

After databases and instance migrations you might want to enable important new features to enhance the functionality and improve the performance.

If you are coming from Version 8.2, consider using the following features:

- ▶ Self tuning memory
- ▶ Large RIDs (record identifiers)
- ▶ Data compression
- ▶ Label based access control
- ▶ Trusted context

If you are coming from Version 9 and have already using the features above, now you may want to enable the new DB2 9.5 features. The same way if you were from Version 8.

- ▶ Enable automatic agent configuration for your databases.
- ▶ Enable the use of non-buffered I/O operations
- ▶ Enable automatic statistics collection that use real-time statistics
- ▶ Enable the use of the Workload Manager functionality
- ▶ In partitioned database environments, take advantage of single system view backups by issuing the BACKUP DB command with the ON ALL DBPARTITIONNUMS clause

You can find more details in Chapter 6, “Administering databases” on page 223.

4.2.6 Migrating the DB2 Administration Server

In a partitioned database system, a separate DAS service typically runs on each machine in the cluster. This allows each machine to act as a coordinator node for requests issued to the instance from the Control Center or Configuration Assistant. This reduces the overhead that exists when one administrative coordinator node is spread across multiple partitions in an instance and helps to balance incoming connections. If you are running in a single partition you still have to migrate DAS to use the Control Center.

If you do not want to keep your existing configuration, you can just drop the DAS and create a new one.

If you do not have a DAS in any database partition server, create a new one. You can find more details about DAS creation in Chapter 2, “Installation” on page 25.

DAS migration

The following steps show how to migrate a DAS from Version 8 or 9.1 to DB2 9.5.

1. Log in as root user.
2. Go to: \$DB2DIR/instance
3. Enter the command:

```
./dasmigr
```

Example 4-13 shows a dasmigr execution:

Example 4-13 dasmigr

```
mena:/opt/ibm/db2/V9.5/instance # ./dasmigr
SQL4407W The DB2 Administration Server was stopped successfully.
SQL4406W The DB2 Administration Server was started successfully.
DBI1070I Program dasmigr completed successfully.
```

Tools catalog database migration

If you want to use your existing scripts and schedules created in Version 8 or Version 9, you have to perform the following steps:

1. Obtain the name of the instance and the tools catalog database through **db2 get admin cfg** command, see Example 4-14.

Example 4-14 Get tool catalog information

```
dpfusr1@gemini:~> db2 get admin cfg
Admin Server Configuration

...
Tools Catalog Database           (TOOLSCAT_DB) = TOOLSDDB
Tools Catalog Database Instance  (TOOLSCAT_INST) = dpfusr1
Tools Catalog Database Schema    (TOOLSCAT_SCHEMA) = CC
Scheduler User ID                =
```

2. Migrate the instance that owns the tools catalog database using the procedure described in 4.2.3, “Migrating instances and databases” on page 174. In Example 4-14, our instance is dpfusr1 which has already been migrated.
3. Migrate tools catalog database using the steps provided in 4.2.3, “Migrating instances and databases” on page 174.

Example 4-15 shows migrating the tools catalog database in our Lab.

Example 4-15 db2tdbmgr

```
dpfusr1@gemini:~> db2tdbmgr -d toolssdb -s cc
```

Tools catalog database migrated successfully to the current level

Tools catalog creation

If you do not have a tools catalog yet, you can create a new one now. You can use an existing database or create a new one, this database must be local.

The syntax for create a new database:

```
db2 "create tools catalog <catalog_name> create new database
<database_name> "
```

Where catalog_name is used to identify the DB2 tools catalog, and database_name is the new database.

Example 4-16 shows how to create a tools catalog database.

Example 4-16 Tools catalog tables creation in a new database

```
dpfusrl@gemini:~> db2 "create tools catalog cc create new database toolsdb"
DB20000I The CREATE TOOLS CATALOG command completed successfully.
```

4.2.7 Migrating DB2 clients

Once the DB2 server is migrated, we recommend that you migrate the DB2 Clients.

Considerations

Mixing the server and client version will reduce the capability of using DB2 server functionality.

Table 4-3 on page 180 shows DB2 9.5 connectivity support.

Table 4-3 Supported connectivity

Client	DB2 server	Client connectivity support
32-bit or 64-bit DB2 9.5 clients	32-bit or 64-bit DB2 9.5 server	Any DB2 9.5 clients can establish 32-bit or 64-bit connections and use full DB2 9.5 functionality.
	32-bit or 64-bit DB2 9.1 server	Only DB2 9.1 functionality is available.
	32-bit or 64-bit DB2 UDB Version 8 server	Only DB2 UDB Version 8 functionality is available.

Client	DB2 server	Client connectivity support
32-bit or 64-bit DB2 9.1 clients	32-bit or 64-bit DB2 9.5 server	Only DB2 Version 9.1 functionality is available.
32-bit or 64-bit DB2 Version 8 clients	32-bit or 64-bit DB2 9.5 server	Only DB2 UDB Version 8 functionality is available.

If you have a Version 8 client in the same machine as a DB2 9.5 server or a DB2 9.5 client in a Version 8 server, connections to the server from the client using local node (IPC) are *not* supported. We recommend you migrate this client to DB2 9.5.

You can not choose the bit size of a client instance, the size is determined by the operating system.

Migrating

DB2 server for Linux supports DB2 clients on UNIX and Windows platforms. In this section, we provide the migration methods for migrating DB2 clients for Linux and Windows.

Table 4-4 summaries how you can migrate various DB2 clients from Version 8 or Version 9.1 to DB2 9.5 in a Windows environment.

Table 4-4 Migration options in Windows

Migrating from	Migrating to	Migration methods summary
<ul style="list-style-type: none"> ▶ Version 8 DB2 Administration Client ▶ Version 8 DB2 Application Development Client ▶ Version 9.1 DB2 Client 	DB2 9.5 Data Server Client	<ul style="list-style-type: none"> ▶ Install the DB2 9.5 Data Server Client, and choose the migrate action in the Work with Existing panel. The client instance is then automatically migrated for you. ▶ Install a new copy of DB2 9.5 Data Server Client, then manually migrate Version 9.1 or Version 8 client instances.
<ul style="list-style-type: none"> ▶ Version 8 DB2 Run-Time Client ▶ Version 8 DB2 Run-Time Client Lite ▶ Version 9.1 DB2 Runtime Client 	DB2 9.5 Data Server Runtime Client	<ul style="list-style-type: none"> ▶ Install the Version 9.5 Data Server Runtime Client as a new copy. ▶ Manually migrate your Version 9.1 or Version 8 client instance.

For Linux environments the client migration process is the same for both DB2 Version 9.5 clients. The process is as follows:

1. Checked the client supported described in “Considerations” on page 180.
2. Back up the client configuration.
The following two commands create two files in your current directory that contain database manager configuration the information of cataloged databases.

```
db2 get dbm cfg > dbm_client.cfg
db2cfexp cfg_profile backup
```

3. Install DB2 9.5 client.
4. Migrate your client instance:

```
$DB2HOME/bin/db2imigr instname
```

Where *\$DB2HOME* is the location the DB2 9.5 client is installed and *instname* is the instance.

Instead of migrate your existing client instance, you can create a new one using the following command:

```
$DB2HOME/bin/db2icrt -s client instname
```

Then import the configuration collected in step 2.

4.3 Migrating single-partitioned database

This section describes single-partitioned database migration procedure. In general, single-partitioned database migration steps are a “subset” of the procedure for migrating multi-partitioned database. Here we provide the steps required and call out for the differences. For the details in each step, refer to 4.2, “Migrating multi-partitioned database” on page 169.

The following are single-partitioned database server migration steps:

1. Check if you can migrate your instance to your Version 9.5 that you have.
2. Save packages, and table spaces information.
3. Back up DB2 configuration and diagnostics information (db2support).
4. Ensure that the database do not contain any indoubt transaction.
5. Revoke the EXECUTE privilege on migrated routines from PUBLIC. This is an optional step.
6. Set DIAGLEVEL parameter to 3 or higher.
7. Take the DB2 Version 8 or Version 9.1 databases inactive.

8. Archive all of the DB2 log files if replication is used.
9. Back up databases. We suggest a full offline backup for each database.
10. If you are still using raw devices for database logging or table space containers, you have to change them to block devices.
11. Install DB2 9.5 Server.
12. Stop the instance.
13. Make sure there is no outstanding processes on the instance to be migrated.
14. Verify that the databases in the instance can be migrated using db2ckmig, and if necessary take actions to fix the problems.
15. Migrate instance to DB2 9.5.
16. Preserve db2diag.log by renaming the file.
17. Preserve files indicated in DIAGPATH parameter including dump files, trap files, and alert log files.
18. Start DB2 instance.
19. Migrate databases in the instance.
20. Compare database configuration settings after migration with the configuration settings before the migration and adjust them if required.
21. Reset the following configuration parameters to the value before the migration:
 - DIAGLEVEL
 - LOGFILSIZ
 - LOGPRIMARY
 - LOGSECOND
22. Start up your database and review the administration notification log or the db2diag.log file to verify that all database services are running properly and all buffer pools are activated.
23. Update the statistics for system catalog tables and user tables, particularly those tables that are critical to the performance of your SQL queries.
24. Rebind application packages to avoid implicit rebound overhead on the first time the packages are used.
25. Migrate DB2 Explain tables.
26. Copy customized code page conversion tables to the proper directories.
27. Back up the migrated databases.
28. Check and reorganize index or tables if required.
29. Migrate the DB2 administration server.

30.Migrate DB2 clients.

4.4 32-bit to 64-bit conversion

If you are running a DB2 Version 8 or Version 9 Enterprise edition under 32-bits Linux, be aware that Version 9.5 is only supported on 64-bit architecture. DB2 Workgroup Edition provides both 32-bit and 64-bit supports.

More details about recommended and validated environments for DB2 Version 9.5 on Linux, refer to Web site:

<http://www.ibm.com/software/data/db2/linux/validate/>

Even though you are running with a validated environment and version of 32-bit DB2 9.5, such as Workgroup edition, consider migrating to DB2 9.5 64-bit instead to avoid any 32-bit kernel limitations.

There are two considerations regarding 32-bit virtual memory address limits and the new multi threaded architecture:

- ▶ Agent private memory for all agent threads is now allocated within a single process. The process memory space might not be large enough to allocate the aggregate of all private memory for all agents. You might need to reduce the number of agents configured.
- ▶ Support for multiple databases is limited because all database shared memory segments for all databases are allocated in a single process memory space. You can reduce the memory usage for each database so that you can activate all databases successfully. However, the database server performance is impacted.

Note: You cannot specify the bit size for the instance when you create or migrate an instance. The bit size for new instances is determined by the operating system where DB2 9.5 is installed.

Table 4-5 shows some important details about available support for 32-bit and 64-bit DB2.

Table 4-5 Available support for 32-bit and 64-bit DB2 9.5

Size	Available support
32-bit	<ul style="list-style-type: none">▶ 32-bit instances only▶ 32-bit DB2 server, client, and GUI tools packages▶ 32-bit IBM Software Development Kit (SDK) for Java

Size	Available support
64-bit	<ul style="list-style-type: none"> ▶ 64-bit instances ▶ 32-bit and 64-bit DB2 libraries available ▶ 64-bit DB2 server and client ▶ 64-bit applications and routines ▶ 32-bit client side application support ▶ 32-bit fenced stored procedures/UDFs only (non- Java) ▶ Java fenced Stored Procedures/UDFs ▶ 64-bit IBM SDK for Java

If you are running in a validated 64-bit environment, converting from 32-bit to 64-bit is done automatically during the migration process. Example 4-17 shows in few details of a migration from Version 8.2 32-bit to Version 9.5 64-bit.

Example 4-17 Migrating from Version 8 32-bit to Version 9.5 64-bit

```
### with db2 instance owner ###
db2inst1@gemini:~> db2level
DB21085I Instance "db2inst1" uses "32" bits and DB2 code release "SQL08028"
with level identifier "03090106".
Informational tokens are "DB2 v8.1.3.136", "s070720", "MI00194", and FixPak
"15".
Product is installed at "/opt/IBM/db2/V8.1".

### with root user ###
gemini:/opt/ibm/db2/V9.5/instance # ./db2imigr db2inst1
db2ckmig was successful. Database(s) can be migrated.
DBI1070I Program db2imigr completed successfully.

### with DB2 instance owner ###
db2inst1@gemini:~> db2level
DB21085I Instance "db2inst1" uses "64" bits and DB2 code release "SQL09050"
with
level identifier "03010107".
Informational tokens are "DB2 v9.5.0.0", "s071001", "LINUXAMD6495", and Fix
Pack "0".
Product is installed at "/opt/ibm/db2/V9.5".
```

4.5 Migrating to a new database

In this section we demonstrate, by a simple example, how to migrate to a new database to leverage the DB2 9.5 features and functions enabled in database creation time. For example, Unicode is set when a database is created. To

convert from a non-Unicode database to a Unicode database, creating a new database is required.

The procedure to convert a DB2 Version 8 or Version 9.1 non-Unicode database to a DB2 9.5 Unicode database is the same. Be aware that the migration can take some time and requires large disk space if you have a huge database.

The following are the steps to convert a non-Unicode database to a Unicode database using **db2move**. You can find more detail about how **db2move** works in 6.3.4, “Using db2move utility” on page 330.

1. Export data using db2move command to a directory with enough space.

```
db2move ITSO export
```

2. Generate a DDL script for your existing database using the db2look command:

```
db2look -d itso -e -o unicode.ddl -l -x -f
```

3. Create a Unicode database, this is default in DB2 9.5.

```
db2 "create database unidb automatic storage yes on '/database/db2inst1'  
collate using SYSTEM_819_BR"
```

4. Edit DDL script, unicode.ddl, created by db2look for the new DB name.

- Replace all occurrences of the database name to the new Unicode database name. For example, change “connect to itso” to “connect to unidb”.
- Increase the column lengths for character columns in your tables, that is because characters converted to Unicode could be expanded.
- If you want to keep your existing database, you must also change the file name specification for table spaces in the unicode.ddl file. Otherwise, you can drop the existing database and use the same table space files.

5. Recreate the database structure by running the edited DDL script:

```
db2 -tvf unicode.ddl
```

6. Import data into the new Unicode database using the db2move command:

```
db2move unidb import
```

Note: The maximum LOB column size db2move moves is 32 KB. DB2move will truncate LOB over 32 KB.

Example 4-18 shows a conversion from a non-Unicode database in DB2 UDB Version 8.2 to a Version 9.5 conversion.

Example 4-18 Converting DB2 Version 8 database non-unicode to Version 9.5 unicode

```

### The steps bellow create a DB in DB2 UDB V8
#

db2inst1@gemini:~> export LANG=en_US.iso88591

db2inst1@gemini:~> db2 "CREATE DATABASE itso ON '/database/db2inst1' USING
CODESET 1252 TERRITORY BR COLLATE USING SYSTEM"
DB20000I The CREATE DATABASE command completed successfully.

db2inst1@gemini:~> db2 get db cfg for itso | grep Database
      Database Configuration for Database itso
Database configuration release level           = 0x0a00
Database release level                         = 0x0a00
Database territory                           = BR
Database code page                           = 1252
Database code set                            = 1252
Database country/region code                 = 55
Database collating sequence                  = UNIQUE
Database page size                           = 4096
Database is consistent                       = YES

db2inst1@gemini:~> db2 -tvf create_tables.sql
#
##### This script creates tables and inserts some lines
#
db2inst1@gemini:~> db2level
DB21085I Instance "db2inst1" uses "32" bits and DB2 code release "SQL08028"
with
level identifier "03090106".
Informational tokens are "DB2 v8.1.3.136", "s070720", "MI00194", and FixPak
"15".
Product is installed at "/opt/IBM/db2/V8.1".

### Here starts the steps to the non-Unicode to Unicode conversion

db2inst1@gemini:~> cd temp

db2inst1@gemini:~/temp> db2move ITS0 export
***** DB2MOVE *****

Action:      EXPORT
Start time:  Wed Feb  6 16:51:37 2008

Connecting to database ITS0 ... successful!  Server: DB2 Common Server V8.2.8

Binding package automatically ...
Bind file: /home/db2inst1/sqlllib/bnd/db2move.bnd

```

Bind was successful!

```
EXPORT:      2 rows from table "DB2INST1"  "."LOCATION"
EXPORT:      2 rows from table "DB2INST1"  "."BRAND"
EXPORT:      2 rows from table "DB2INST1"  "."CUSTOMER"
```

Disconnecting from database ... successful!

End time: Wed Feb 6 16:51:38 2008

```
db2inst1@gemini:~/temp> db2look -d itso -e -o unidb.ddl -l -x -f
-- No userid was specified, db2look tries to use Environment variable USER
-- USER is: DB2INST1
-- Creating DDL for table(s)
-- Output is sent to file: unidb.ddl
-- Binding package automatically ...
-- Bind is successful
-- Binding package automatically ...
-- Bind is successful
```

```
db2inst1@gemini:~/temp> vi unidb.ddl
---> This point we did the step 4 <---
```

```
db2inst1@gemini:~/temp> db2 terminate
DB20000I The TERMINATE command completed successfully.
```

```
db2inst1@gemini:~/temp> db2 drop db itso
DB20000I The DROP DATABASE command completed successfully.
```

```
db2inst1@gemini:~/temp> db2stop
02/06/2008 16:55:00 0 0 SQL1064N DB2STOP processing was successful.
SQL1064N DB2STOP processing was successful.
```

```
gemini:/opt/ibm/db2/V9.5/instance # ./db2imigr db2inst1
```

System Database Directory is empty. No database has been processed.

```
DBI1070I Program db2imigr completed successfully.
```

```
db2inst1@gemini:~> db2start
02/06/2008 16:58:39 0 0 SQL1063N DB2START processing was successful.
SQL1063N DB2START processing was successful.
```

```
db2inst1@gemini:~> db2level
DB21085I Instance "db2inst1" uses "64" bits and DB2 code release "SQL09050"
with
level identifier "03010107".
```


Informational tokens are "DB2 v9.5.0.0", "s071001", "LINUXAMD6495", and Fix Pack "0".

Product is installed at "/opt/ibm/db2/V9.5".

```
db2inst1@gemini:~/temp> db2 "create database itso automatic storage yes on
'/database/db2inst1' USING CODESET UTF-8 TERRITORY BR collate using
SYSTEM_819_BR"
```

DB20000I The CREATE DATABASE command completed successfully.

#

Create database objects using the edited DDL

#

```
db2inst1@gemini:~/temp> db2 -tvf unidb.ddl
```

```
db2inst1@gemini:~/temp> db2move itso import
```

Application code page not determined, using ANSI codepage 1208

***** DB2MOVE *****

Action: IMPORT

Start time: Wed Feb 6 17:07:16 2008

Connecting to database ITS0 ... successful! Server : DB2 Common Server V9.5.0

Binding package automatically ... /home/db2inst1/sqllib/bnd/db2common.bnd ...
successful!

Binding package automatically ... /home/db2inst1/sqllib/bnd/db2move.bnd ...
successful!

* IMPORT: table "DB2INST1" "."LOCATION"

```
-Rows read:      2
-Inserted:      2
-Rejected:      0
-Committed:     2
```

* IMPORT: table "DB2INST1" "."BRAND"

```
-Rows read:      2
-Inserted:      2
-Rejected:      0
-Committed:     2
```

* IMPORT: table "DB2INST1" "."CUSTOMER"

```
-Rows read:      2
-Inserted:      2
-Rejected:      0
-Committed:     2
```

Disconnecting from database ... successful!

End time: Wed Feb 6 17:07:20 2008

```
db2inst1@gemini:~/temp> db2 get db cfg for itso | grep Database
Database Configuration for Database itso
Database configuration release level      = 0x0c00
Database release level                  = 0x0c00
Database territory                      = BR
Database code page                      = 1208
Database code set                      = UTF-8
Database country/region code           = 55
Database collating sequence            = SYSTEM_819
Database page size                     = 4096
Database is consistent                 = YES
```

4.6 Fix pack installation

A DB2 fix pack is a collection of updates and fixes for Authorized Program Analysis Reports (APARs). We recommend that you keep your environment running up-to-date at the latest fix pack level.

Note: Fix packs are cumulative. This means that the latest fix pack for any given version of DB2 contains all of the updates from previous fix packs for the same version of DB2.

You can have multiple DB2 copies on the same system, those copies can be at different version and fix pack levels. If you want to apply a fix pack to one or more DB2 copies, you must install the fix pack on those DB2 copies one by one.

DB2 9.5 enhanced the procedure for applying DB2 fix packs. Now, updating DB2 instance and DAS (running db2iupdt and dasupdt) are automated. In addition, binding occurs automatically at the first connection. This means once you start the database manager, the DB2 product is ready to use immediately after fix pack installation.

To apply a DB2 fix pack perform the following steps:

1. Download the code of the latest fix pack from:
<http://www.ibm.com/software/data/db2/udb/support.html>
2. Read carefully the Readme file and release notes and perform all the necessary tasks.
3. Install the DB2 fix pack

If you are running a non-root installation and root based features are enabled, you must rerun **db2rfe** command each time a fix pack is applied in order to re-enable those features. Details about non-root installation see Chapter 2, “Installation” on page 25.

5



IBM Data Studio

IBM Data Studio is a comprehensive data management solution. It empowers you to effectively design, develop, deploy, and manage data, databases and database applications throughout the data management life cycle utilizing a consistent and integrated user interface.

In this chapter we briefly introduce IBM Data Studio. We cover the following topics:

- ▶ **Introduction**
A short introduction about Data Studio.
- ▶ **Installation**
This section describes the installation steps for Data Studio. We briefly talk about how to migrate projects from the Developer workbench into Data Studio.
- ▶ **Features and functions**
In this section we briefly introduce the features and functions of Data Studio.

5.1 Introduction

Enterprises of all sizes and across all industries share common drivers of the investments they are making in the management of their business data. The complexity and cost of supporting and managing data from initial design of the data structure throughout all phases until properly sunsetting the data continue increasing. The demand to deliver an increasing amount of data more rapidly across and beyond the business keeps growing. The expensive human resources, developers, architects, and administrators, work in silos that cause issues such as data models not shared across silos; Web services are difficult to develop and deploy; complex environments use different tools for each data server; life cycle of data is poorly understood and challenging to manage; and so on.

The innovation will be restrained when silos of information are disconnected, inaccurate, incomplete, and out of context. New approaches are needed for application delivery and support in order to reduce the time and cost of empowering a broader user audience with more powerful data-driven applications.

IBM Data Studio, a foundation for innovation and a comprehensive data management solution, is a family of integrated tools for database development and administration. It empowers you to effectively design, develop, deploy and manage your data, databases, and database applications throughout the data management life cycle utilizing a consistent and integrated user interface.

Figure 5-1 on page 195 illustrates the life cycle of data management. IBM Data Studio provides the capability for architects, developers, and administrators to

- ▶ Increase productivity for all roles throughout the data life cycle:
 - Slash development time up to 50% with an integrated data management environment.
 - Promote collaboration across roles to optimize data server and application performance.
- ▶ Simplify and speed development of new skills:
 - Learn once, use with all supported data servers.
 - Extensible with Eclipse plug-ins to customize the environment for each team member.
- ▶ Accelerate data as a service for Service Oriented Architecture:
 - Develop and publish data as a Web service without programming.
 - Info 2.0 Ready - support for Web 2.0 protocols and format.

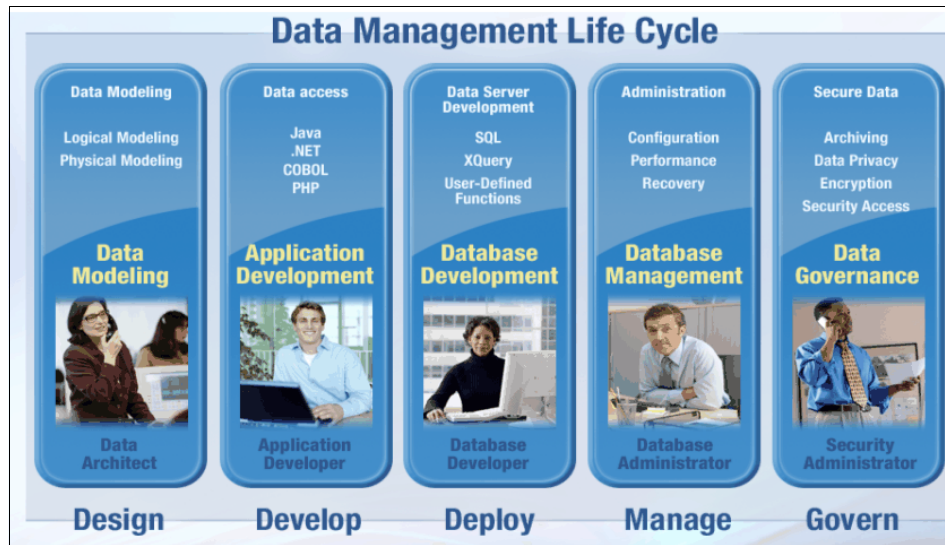


Figure 5-1 Data management life cycle

IBM Data Studio supports for DB2 and IDS on all platforms including i5/OS. It is a full replacement of DB2 Developer Workbench plus much more. The following are features provided:

- ▶ Common features for DB2 and IDS
 - ER diagramming
 - Data Distribution Viewer
 - Integrated Query Editor
 - SQL Builder
 - XML Editor
 - XML Schema Editor
 - Data Web Services
 - Schema Management
 - Data Management
 - Security Access Controls
 - Project Management
- ▶ Common Features for DB2
 - SQL Routine Debugger
 - Java Routine Debugger
- ▶ Common features for DB2 LUW, z/OS
 - Update Statistics
 - Visual Explain

IBM Data Studio Administration Console (DSAC), introduced in Version 1.2, is a Web based interface to perform operational database management tasks. DSAC provides quick analysis and resolution capabilities to identified data server conditions and scenarios. With DSAC, you can immediately access to critical data server information and functions from anywhere, anytime. In Version 1.2, DSAC functions include

- ▶ Health and availability monitoring
 - Problem determination and recommendations
 - 72 hours of history
 - Ability to monitor up to 100 databases
- ▶ Q Replication monitoring and administration

For more detailed information refer to the following sources:

- ▶ General information about Data Studio:
<http://www.ibm.com/software/data/studio>
- ▶ Online documentation of the Data Studio:
<http://publib.boulder.ibm.com/infocenter/dstudio/v1r1m0/index.jsp>
- ▶ For general information about the Eclipse platform:
<http://www.eclipse.org>

5.2 Installation

This section provides the procedures to install IBM Data Studio. There are three methods by which you can install IBM Data Studio on Linux:

- ▶ Installing from the Launchpad program
- ▶ Using the IBM Installation Manager graphical interface
- ▶ Installing silently, that is, response file installation

5.2.1 Installing IBM Data Studio

You can download IBM Data Studio free of charge from

<http://www.ibm.com/software/data/studio/>

In this section, we show you how to install IBM Data Studio from an electronic image. For installing Data Studio from other media, refer to the *Data Studio installation Guide*.

The installation steps are as follows:

1. Review the *Preinstallation tasks* in the IBM Data Studio Installation Guide to make sure that the installation requirements are met.
2. Unzip the Data Studio zip file:

```
unzip ibm_data_studio_v111_linux.zip
```
3. Run the setup executable provided:

```
./setup
```

Data Studio uses IBM Installation Manager to install the product. If you haven't had the IBM Installation manager yet, the setup program will install the IBM Installation Manager for you.
4. Figure 5-2 shows the first panel presented. You can browse the installation guide and release note. Select **Install IBM Data Studio** to install the product.

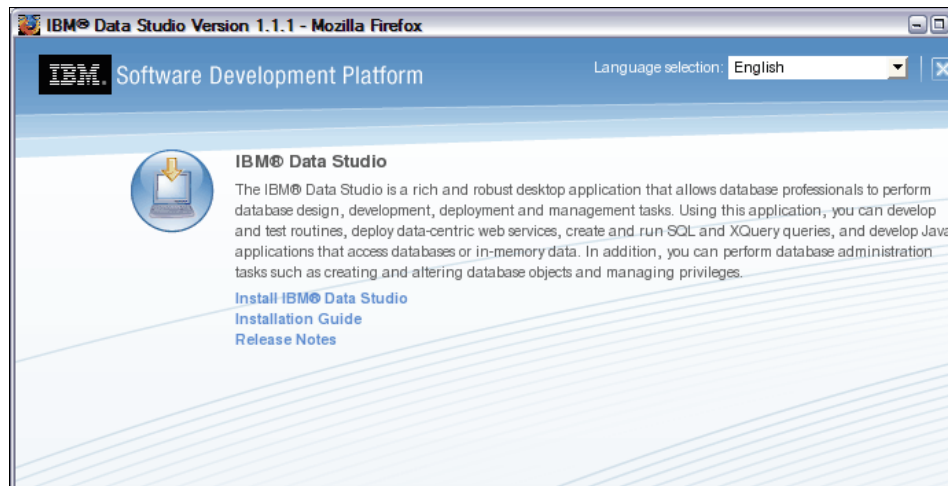


Figure 5-2 Installing IBM Data Studio

5. In the Install Packages panel (Figure 5-3 on page 198), the IBM Installation Manager is also selected if it has not been installed in the system yet.

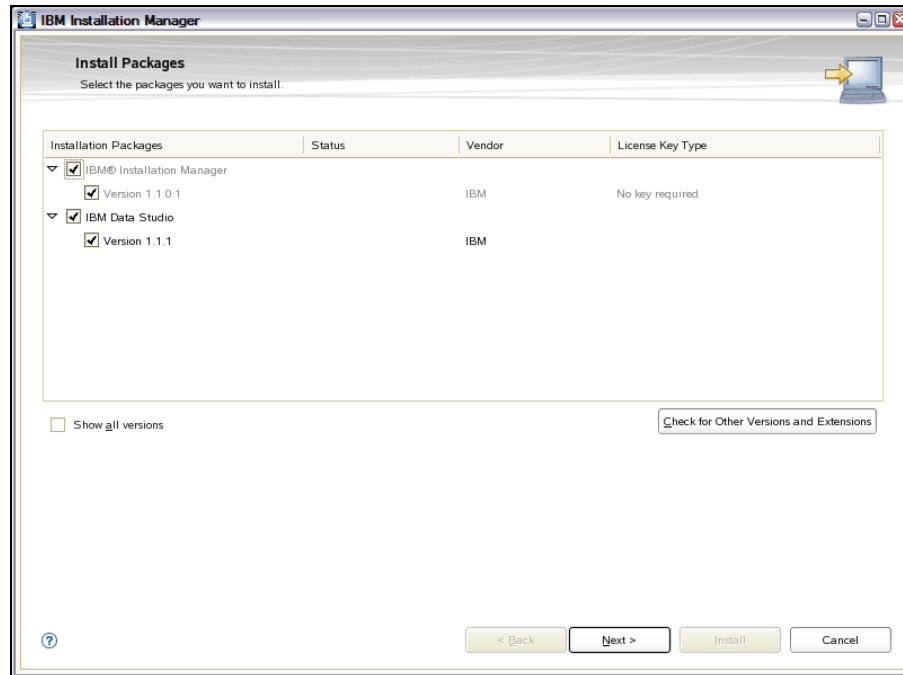


Figure 5-3 Select install package

6. On the Licenses panel, accept the license agreement.
7. The location tab is for specifying the locations for shared resources, IBM Installation Manager, and package groups.

On the first panel of the Location tab (Figure 5-4 on page 199), specify the paths for the shared resources directory and the IBM Installation Manager. The shared resources directory contains resources that can be shared by one or more package groups. We use the default locations.

Note: You can specify the shared resources directory only at the first time that you install a package. You cannot change the directory location unless you uninstall all packages.

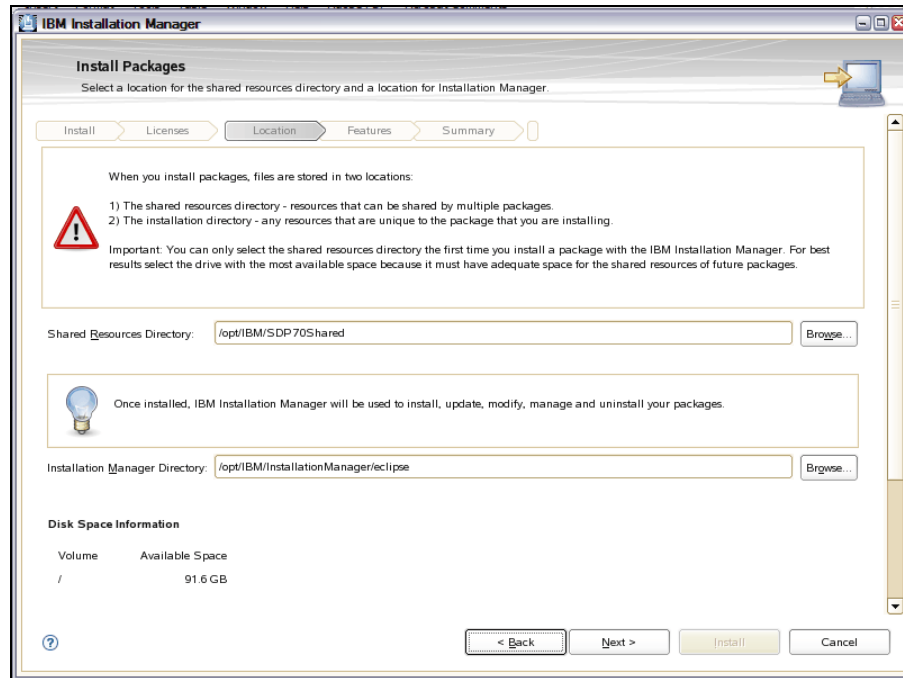


Figure 5-4 Location for shared resources and Installation Manager

Next, you can specify if you want to use an existing package group or create a new one. We create a new package group using the default location. See Figure 5-5 on page 200.

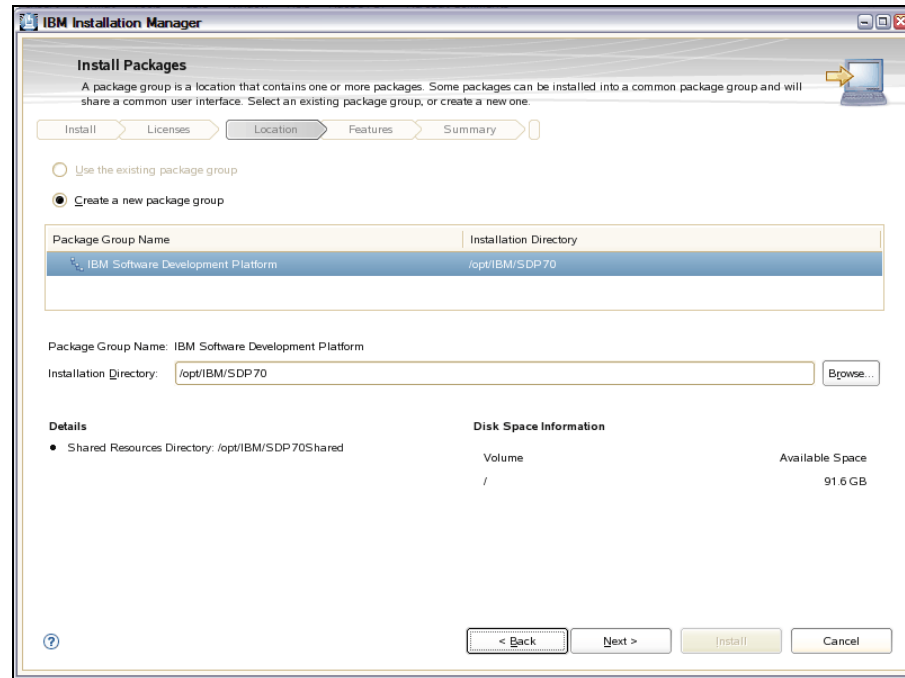


Figure 5-5 Package group

IBM Data Studio is an Eclipse based application. A version of Eclipse integrated development environment (IDE) is bundled in the installation package. The next panel allows you to extend an existing version of Eclipse IDE instead of installing a new one. See Figure 5-6 on page 201.

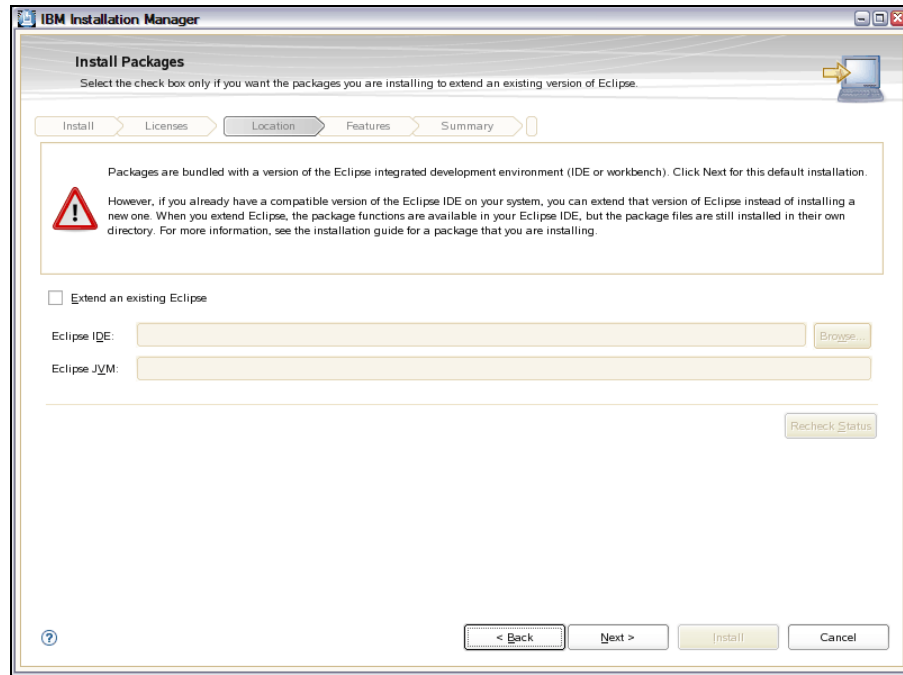


Figure 5-6 Eclipse version

8. Feature tab: You can select the Data Studio features to be installed under this tab. The first panel is for selecting the language. Other features are listed in the second panel for your selection. See Figure 5-7 on page 202.

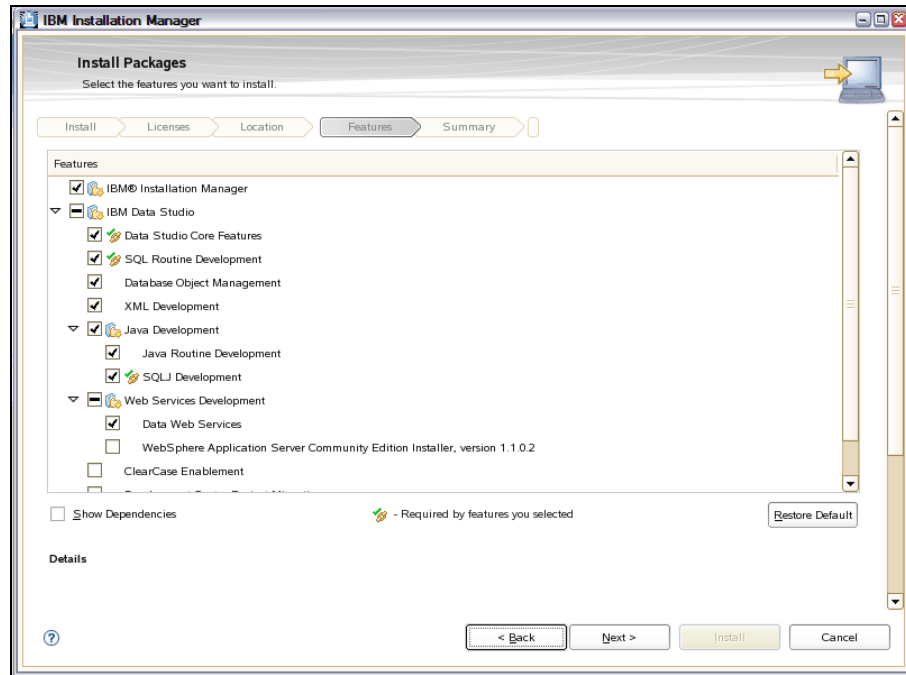


Figure 5-7 Data Studio features

9. The Summary tab (Figure 5-8 on page 203) shows all your installation specifications. Review your choices before installing the IBM Data Studio Developer package. If you want to change the choices you made on previous pages, click **Back** and make your changes. Once you are satisfied with your installation choices, click **Install** to install the package.

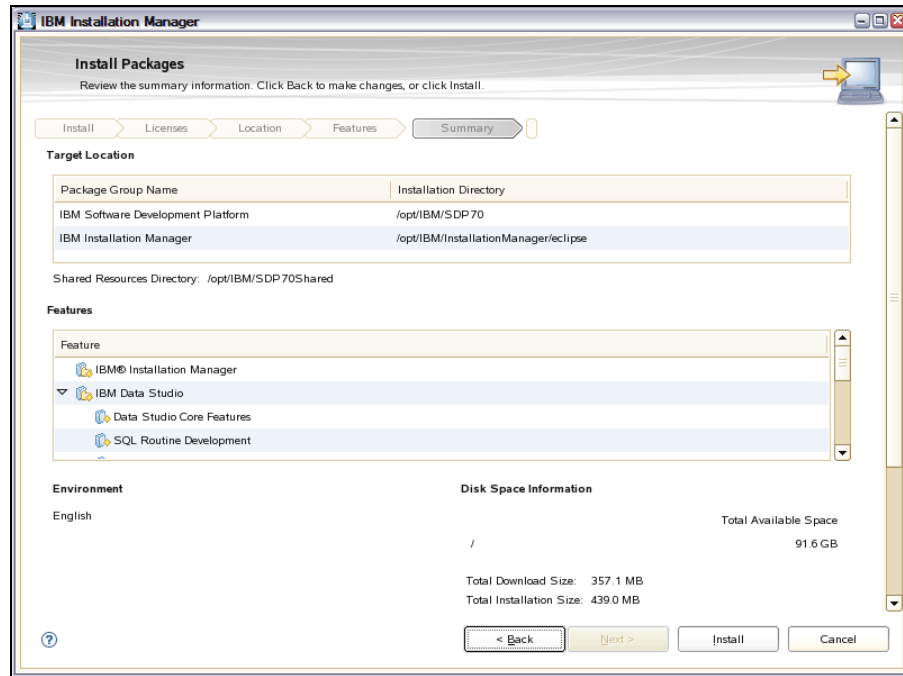


Figure 5-8 Summary of features to be installed

10. When the update process is completed, a message confirms the success of the process. Click **View Log File** if you want to open the update log file, or click **Finish**.

5.2.2 Installing IBM Data Studio Administration Console

You can install IBM Data Studio Administration Console (DSAC) using the following steps:

1. Log in as root.
2. Enter the following command to start the DSAC installation program.

```
sh ./IBM_DSAC_V1.1.2_1.0.0.0-b199_Linux_x86_32_64.bin
```
3. On the *Introduction* panel, click **Next**.
4. On the license agreement panel, read the license agreement and then select **I accept the terms in the license agreement** and click **Next**.
5. On the *Choose install Directory* panel (Figure 5-9 on page 204), specify the directory where you want to install DSAC or accept the default setting. Click **Next**.



Figure 5-9 Choose Install directory

6. On the *Specify Server Details* panel (Figure 5-10), specify the host name and ports for the Web server, then click **Next**.

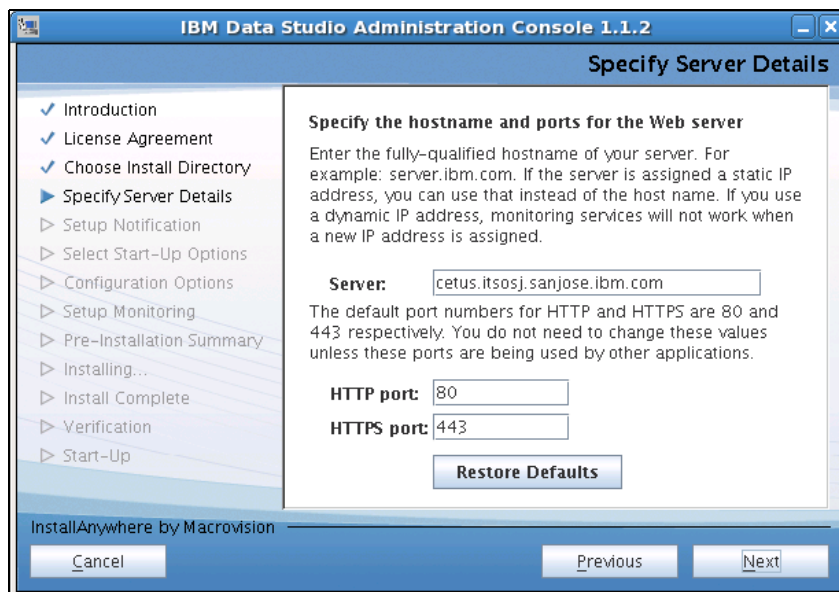


Figure 5-10 Specify the host name and ports for the Web server

7. On Setup Notification panel (Figure 5-11), you can choose **Enable e-mail functions** and configure SMTP server. Click **Next**.

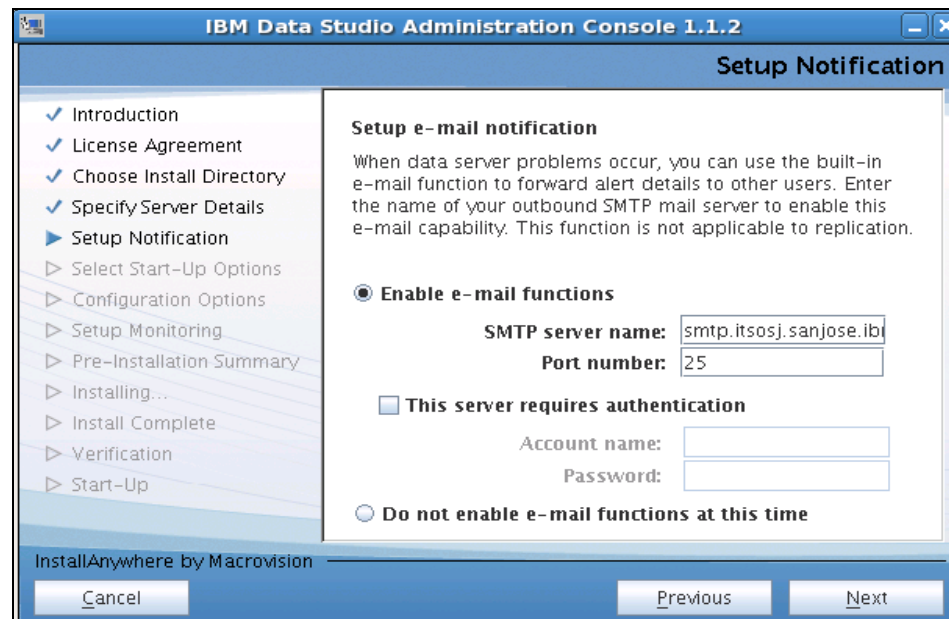


Figure 5-11 Setup Notification

8. On the *Select Start-Up Options* panel, select **Start services automatically**, then click **Next**.
9. On the *Configuration Options* panel, you can choose *Allow or sending of logs* or **Do not allow the sending of logs**, click **Next**.
10. On the next panel, enter the password for *sysadmin* which will be created by automatically by installer, then click **Next**.
11. On the *Pre-Installation Summary* panel, click **Next**.
12. On the Start-Up (Figure 5-12 on page 206), DSAC server has started successfully and a Web address is given. You can access client console by pointing your Web browser to this address. Click **Done**.



Figure 5-12 Start-UP

5.2.3 Migration from the Developer Workbench

As IBM Data Studio is a full replacement of the Developer Workbench, there is no direct migration path from the Developer Workbench to the Data Studio. You can install the Data Studio as a separate package and import the existing projects from the Developer Workbench to Data Studio.

5.3 Features and functions

In this section, we introduce some frequently used terminologies in Data Studio and give a briefly discussion of some features and functions that support you in designing and developing your database and your database applications.

In 8.5.3, “Application development with the Data Studio” on page 460, we demonstrate the creation of a query using the Query Editor. We also create a Java stored procedure and demonstrate how to debug it on the database server.

5.3.1 Terminology

IBM Data Studio is an Eclipse based application. It uses the same terminologies as other Eclipse based applications such as workspace, perspectives, views, and so on.

Workspace

A workspace is a logical collection of projects that you define. It contains all files of your current local working environment. When you start the Data Studio, it prompts you for a workspace location. This Welcome panel contains several useful links for learning the Data Studio. You can omit this prompt in the future by checking the check box on that panel.

You can use as many different workspaces as you want. There is no direct relationship between them. Different workspaces can be used to organize projects for different applications or different versions of a project.

Note: You can always get back to the Welcome screen by selecting **Help → Welcome**.

If you want to share a workspace in your team, you should use the *Team functions* described in 5.3.2, “Team function” on page 212.

A workspace is presented through the views which are laid out within a perspective. When you close the Welcome panel, a perspective that has information from the workspace is presented. See Figure 5-13.

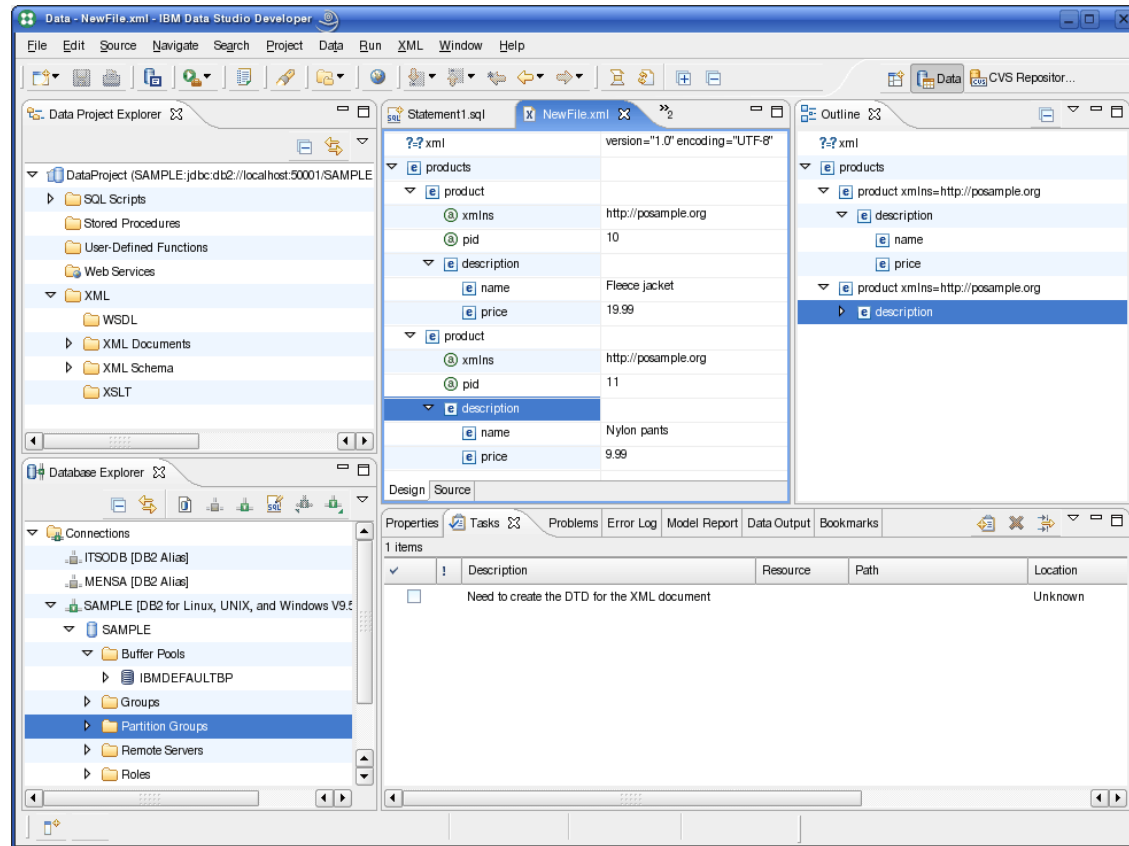


Figure 5-13 Sample perspective for a workspace

Perspectives

A *perspective* is a collection of views and editors grouped into a particular layout. Perspectives reflect different roles or tasks within your workspace, for instance, debug perspective, data perspective, and version control perspective.

You can switch to a different perspective by selecting **Window → Open Perspective → ...** and then one of perspectives in the sub menus. You can also create your own perspectives or modify existing ones. Figure 5-13 on page 208 is the *Data perspective*.

At the right top you can see the recently used perspectives (Figure 5-14). Clicking on them switches to the related perspective.



Figure 5-14 Buttons to switch between recently used perspectives

Here are some of the available perspectives:

- ▶ *Data*
This is the default perspective you get if you enter the Data Studio. It allows you to manipulate your database, database objects, and data.
- ▶ *Debug*
You are asked to switch to this perspective if you start debugging a Java program. It supports you in debugging code.
- ▶ *J2EE*
Provides a layout that is more useful if you develop J2EE applications. You might use it if you create web services which access your data.
- ▶ *Java*
This perspective is for developing Java applications.
- ▶ *Team Synchronizing*
Offers you the tools for synchronizing your code with the code repository.

Views

A view is a representation of a resource in a project. A resource can be a certain type of file or an object in the database. All views of a certain resource are synchronized between each other automatically. This means that if you change anything in one view, you can see the result immediately in all other views that represent the same resource.

The sample perspective shows in Figure 5-13 has the following views:

- ▶ *Data Project Explorer*: This view lists the projects you are working on in an hierarchical structure.
- ▶ *Database Explorer*: This view shows the defined database connections. It shows the database objects of the related connection in a hierarchical structure.
- ▶ *Editor*: Located in the center, the Editor is the main working pane. In our example, you see an XML editor which currently has an XML file opened. It shows the XML file as a tree structure and allows you to modify it. At the top of the editor view you can see several tabs where you can switch between different editors.

- *Outline*: The outline view shows the current object in the editor pane in an outline mode and allows you to quickly navigate within it.

At the right bottom pane you can see other views organized by tabs and the *Tasks* view is open. The Tasks view is a Notepad where you can keep track of your open tasks. It also shows tasks which you have annotated in your source files. Use the tab to switch to other views.

The editor pane shows this pane shows a tree representation of the file content. The one shows in the figure is the XML file `NewFile.xml` file. The Outline view at the right side shows the contents of the same file in a different way. If you change anything in the editor, the change will be reflected in the Outline view immediately.

Figure 5-15 shows that we have added a new product element.

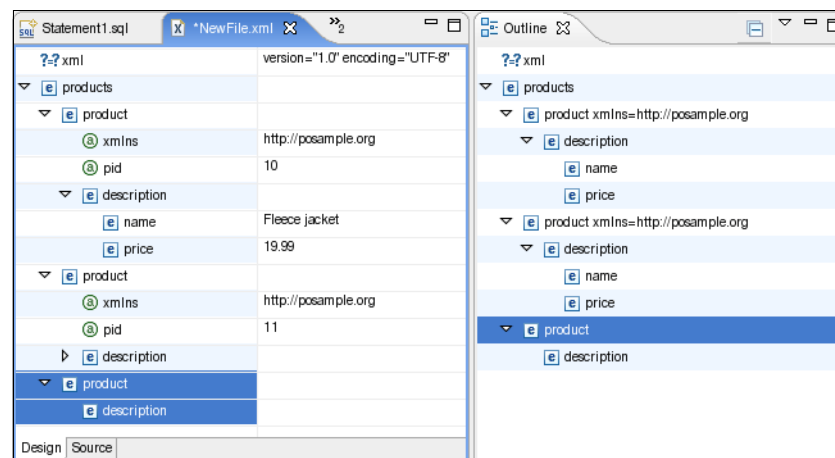


Figure 5-15 Views

You can open new views by selecting **Window** → **Show View** → ... and then one of the sub menus. The new view will be added somewhere in your perspective and you can rearrange it afterwards.

Here is a list of some other useful views:

- *Navigator*

Shows you the real file structure. You can enable the filters to hide files. Turn the filters off to see all files.

- *Servers*

You can run and debug J2EE applications within the debugging framework. In this view you can manage the servers for running and deploying the applications.

► *Problems*

This view is usually already open in one of the tabs. It shows you the compilation or syntax problems of the currently open projects.

► *Error Log*

This view is usually already open in one of the tabs. It shows you general error messages in your environment.

► *Console*

Here you can see console output from Java programs or from application servers while running or debugging applications. In the console view you can switch between the outputs of the concurrently running programs.

► *Bookmarks*

Here you can bookmark sections in your code and quickly navigate through them.

► *Search*

Contains a list of your search results and provides meanings to navigate through them.

Editors

Editors are associated with the type of the file you edit. For instance, in Figure 5-13 on page 208 the XML editor is used to edit the XML file. For other file types there are other editors available. If there is no editor for a particular file type, the text editor is used. The editor tool bar (Figure 5-16) allows you to move around between the annotations in your file.



Figure 5-16 Editor tool bar

Projects

Projects are used to organize your resources. For instance, a Java project contains the source files, all built files, and meta information files for your Java application. A *Data Project* is related to a database connection and stores all information about this project. In the file system a project is represented as a subdirectory in your workspace.

You can create a new project by selecting **File** → **New** → **Project...** A panel lists wizards for creating different types of project shown. Select the wizard suitable for your application. The Data Studio will ask you a couple of project specific questions and switch to a particular perspective that is useful for that project type.

You can create as many projects as you want in your workspace. Projects can be opened or closed as you need to work on them. If you want to share projects between different workspaces or between different members in your team, use the *Team* functions as described in 5.3.2, “Team function” on page 212. Just copying the project subdirectory is not supported.

5.3.2 Team function

The team function introduces a powerful facility to share projects among team members or workspaces. This function supports all version control applications that are available as plug-ins for the Eclipse framework. Examples are CVS, SVN, CMVC, and Rational ClearCase®.

Share a project

A project can be shared by right-clicking the project in the project explorer and select **Team** → **Share Project...** (Figure 5-17).

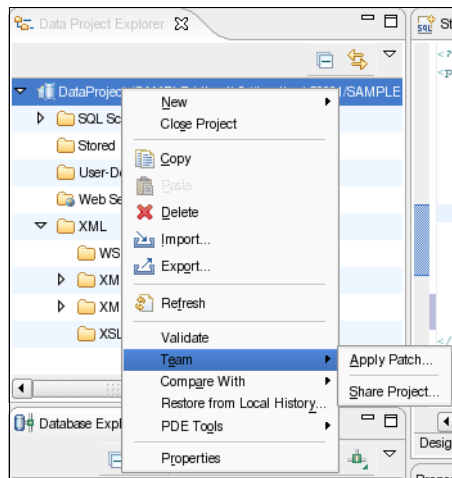


Figure 5-17 Share a project

Data Studio asks a few information, such as repository location, about the source control application you use, then synchronizes the resources from the workspace with the repository.

Note: To use the team functions, a code repository like SVN or CVS is required. You need to install and configure SVN or CVS separately as these repositories are not part of the Data Studio. Linux provides packages for SVN and CVS.

Once a workspace is shared, you can synchronize it with the repository at any time, you can check in the code at certain checkpoints and perform all operations which are supported by your repository. You have access to all previous check-in versions of your files.

Checkout a shared project

If you want to use a project which is checked in into a code repository you have to check it out to your local workspace. In order to do that you can switch to the related perspective for that repository.

For instance, if you want to check out a project from a CVS repository, switch to the CVS perspective by selecting **Window → Open Perspective → Other...** In the Open Perspective panel, select **CVS Repository Exploring**. This perspective allows you to specify a repository location and check out a project from there.

Once a project is checked out to your local workspace, it is connected with that repository location and you can check in or synchronize your content at any time with the repository. In that way, all your work is shared between your team members too.

5.3.3 XML editing

The Data Studio contains a powerful graphical XML and XSD editor. A GUI tool is provided to guide you through the creation steps if you are not familiar with the XML and XSD syntax. This helps both experienced and inexperienced application developers to create error free XML and XSD documents.

Example 5-1 shows the source of an XSD document.

Example 5-1 Example XSD document

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.example.org/TestSchema"
xmlns:tns="http://www.example.org/TestSchema" elementFormDefault="qualified">

  <element name="products" type="tns:productsType"></element>

  <complexType name="descriptionType">
```

```

<sequence>
  <element name="name" type="string"></element>
  <element name="price" type="double"></element>
</sequence>
</complexType>

<complexType name="productType">
<sequence>
  <element name="description" type="tns:descriptionType" minOccurs="1"
maxOccurs="1"></element>
</sequence>
<attribute name="pid" type="int"></attribute>
</complexType>

<complexType name="productsType">
<sequence>
  <element name="product" type="tns:productType" minOccurs="1"
maxOccurs="unbounded"></element>
</sequence>
</complexType>

</schema>

```

Figure 5-18 shows how it looks like in the XML schema editor.

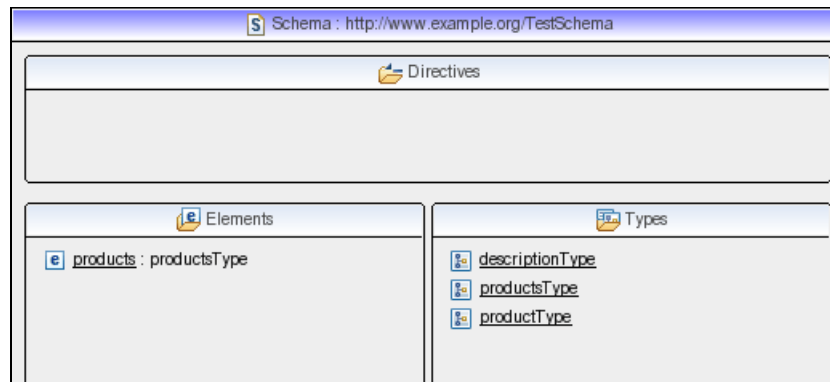


Figure 5-18 Top view of the XSD

If you double click the **products** element at the left side, you can see its definition (Figure 5-19).

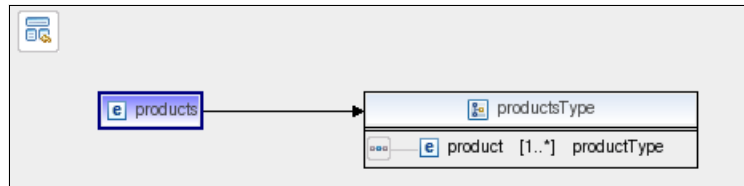


Figure 5-19 Definition of the products element

Double-clicking the title bar of the productsType element presents you its definition as shown in Figure 5-20 on page 215.

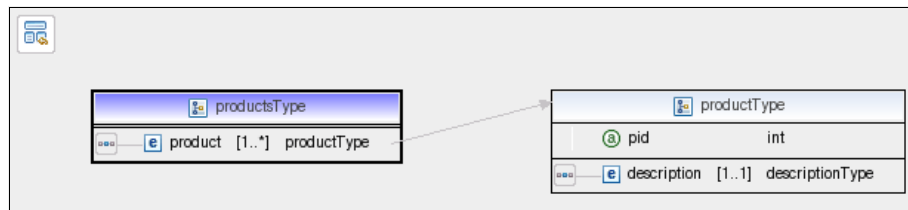


Figure 5-20 Definition of the product element

To see the definition of descriptionType element, double click it (Figure 5-21).

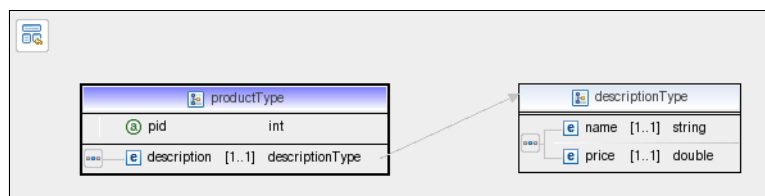


Figure 5-21 Definition of the description element

The XML schema editor GUI navigates through the XML structure and allows you to add or modify definitions easily. With the guidance you are able to avoid syntax errors. You also have the choice to switch to the source and edit the file directly.

You can create an XML document from the XML schema definitions you have created. In the Data Project Explorer right click **XML Documents** and select **New** → **XML**. At the Create XML File panel, select **Create XML file from an XML schema file**. Provide the name for the project, the new XML file, and the schema file. After going through the rest of panels, you get to the XML editor for that file.

The XML editor guides you based on the schema definition (Figure 5-22 on page 216). Just right click the elements and it provides you with only the valid options. The options come from the previously defined XSD document.

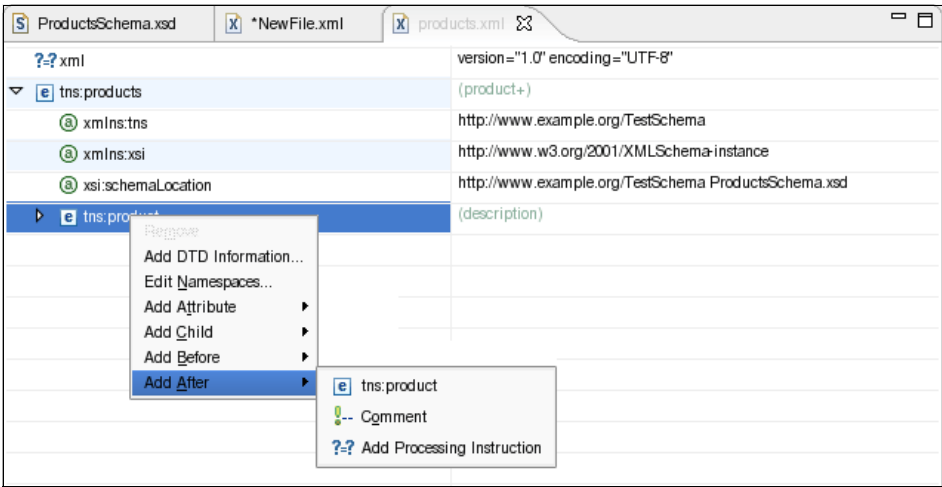


Figure 5-22 Sample in the XML editor

5.3.4 ER Diagramming

The Data Studio allows you to easily create ER diagrams from the database schema. In the Database Explorer, open the tree **Connections** → **SAMPLE** → **SAMPLE** → **Schemas** → **DB2INST1** → **Tables** for the list of tables. Select the tables you want to create the ER diagram. In our example, we select the EMP_RESUME and the EMPLOYEE tables and right-click on them (Figure 5-23).

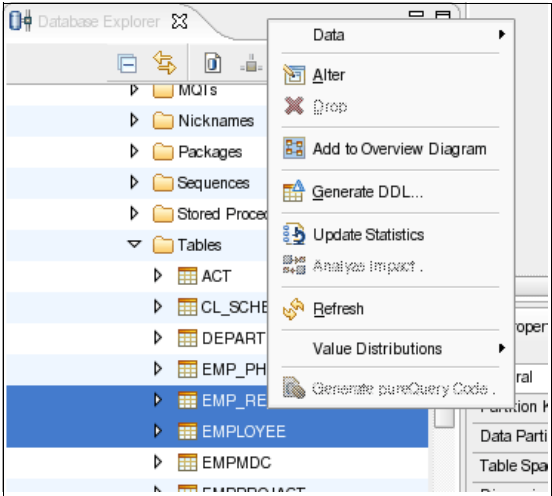


Figure 5-23 Add tables to a diagram

Select **Add to Overview Diagram**. enter OverviewDiagram as the diagram name. You get an ER diagram for that two tables (Figure 5-24 on page 217).

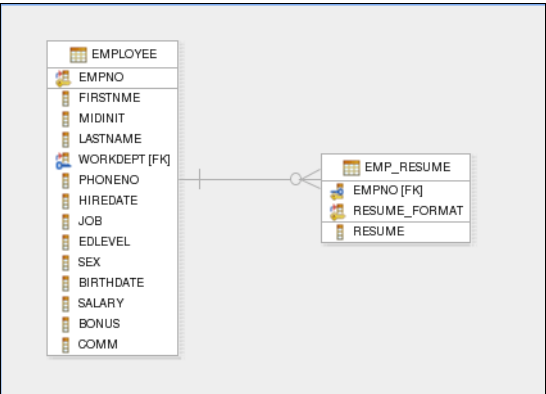


Figure 5-24 ER diagram of the EMPLOYEE and the EMP_RESUME table

5.3.5 Data and object management

You can manage the data of your tables directly from the workbench of the Data Studio. In order to do that, open the tree **Connections** → **SAMPLE** → **SAMPLE** → **Schemas** → **DB2INST1** → **Tables** → **EMPLOYEE**. Right-click on the table and select **Data** → **Edit**. You get an editor panel that allows you to manipulate the data (Figure 5-25).

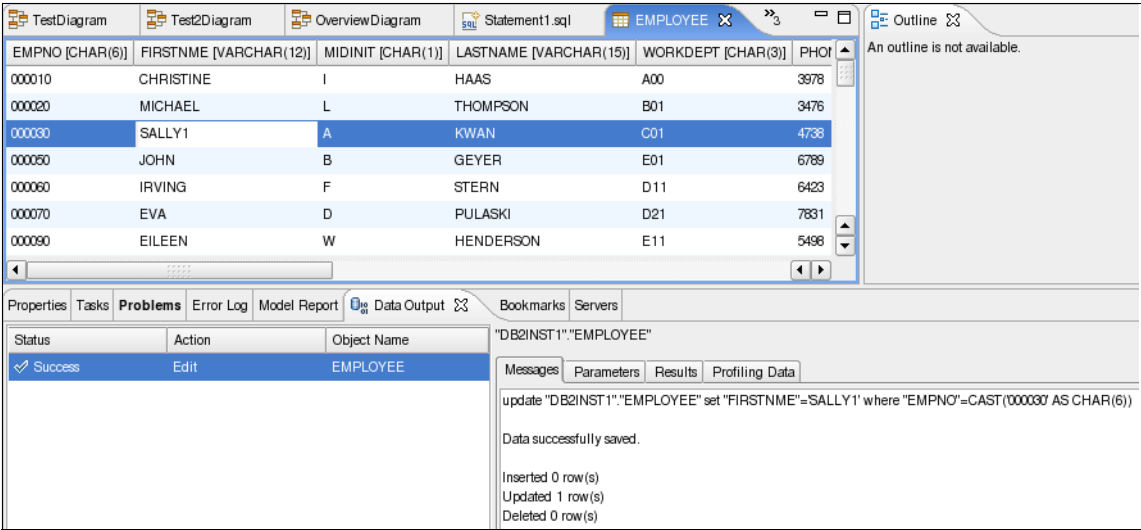


Figure 5-25 Update table data in Data Studio

The data distribution viewer can be selected by a right-click the EMPLOYEE table in the Database Explorer and select **Value Distributions** → **Multivariate**. You get the viewer panel where you can analyze your data (Figure 5-26).

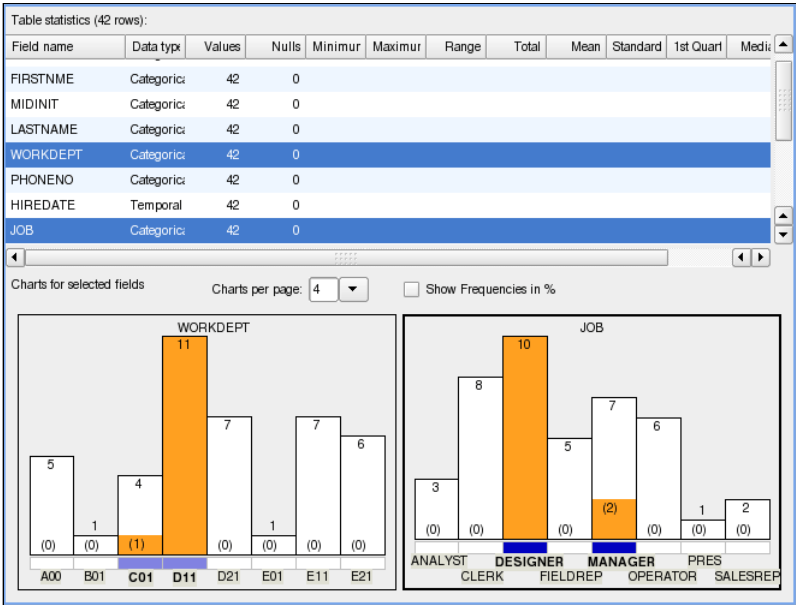


Figure 5-26 Data distribution viewer

If you double click on any of the database objects in the Database Explorer you get an editor panel which allows you to manipulate the related object. For instance, if you double-click on the EMPLOYEE table you get the panel as shown in Figure 5-27 on page 219.

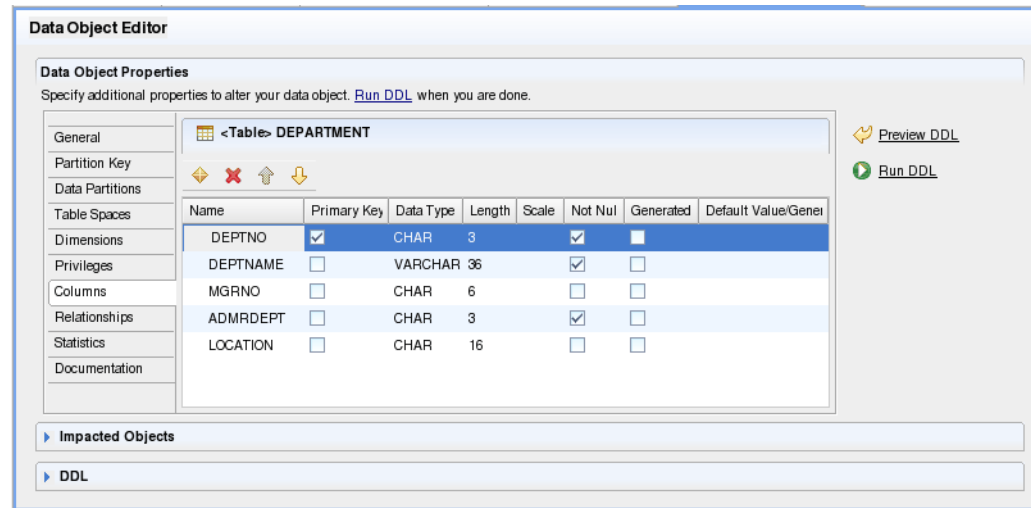


Figure 5-27 table editor

In a similar way you can manipulate the other database objects such as aliases, buffer pools, table spaces, roles, privileges.

5.3.6 Data Web Services

Data Web Services provides you the capability to generate a Web Service from a query without any coding. Right-click on the statement in the Data Project Explorer and select **Add to Web Service** (Figure 5-28).

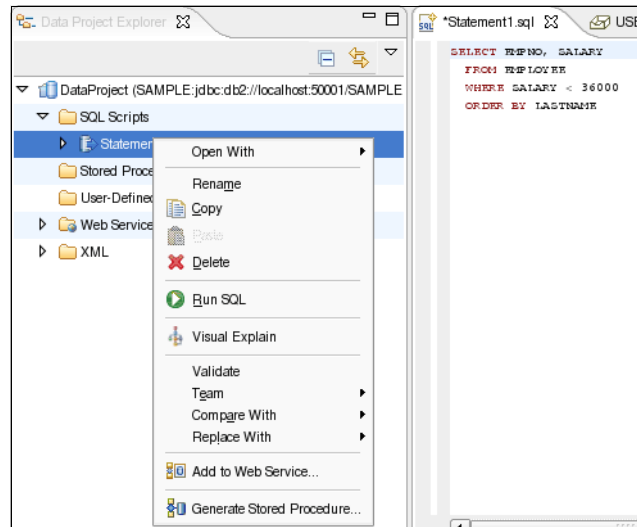


Figure 5-28 Create a web service from a query

This creates a Web service from that query. You can build the WAR file and deploy it to an application server. You also can use it as a framework for your further development.

If you have installed the runtimes of the application server (WebSphere, Tomcat, and son on) you can debug and run them from within your workspace.

5.3.7 Tips

Here are a few useful tips for using the Data Studio:

- ▶ If you double click on the title bar of a view or tab, the related panel expands to full screen. Another double-click resizes it back to the previous size.
- ▶ You can rearrange your layout freely. You can move any view into another part of the pane by just using drag and drop.
- ▶ Keyboard shortcut saves key strokes and can increase your productivity. use **Help** → **Key Assist** or **Shift+Ctrl+L** to shows the available keyboard shortcuts.
- ▶ In the editor view, a very helpful function *Content Assist* (keyboard shortcut **Ctrl+Space**) can help you complete the name of the object you are currently typing in.
- ▶ Tips and tricks provides useful information for working efficiently with the Data Developer. Use **Help** → **Tips and Tricks** to show the available tips and tricks.

- ▶ If you select **Help** → **Tutorial Gallery**, you get to tutorial sessions. They introduce you the various features of Data Studio in a guided tour.

6

Administering databases

This chapter contains information regarding some important database maintenance tasks for DB2 on Linux platforms, such as database backup and recovery, table and index reorganization, and data movement via utilities, such as export, import and load, and so on. These topics are discussed:

- ▶ Database backup and recovery
 - Considerations for choosing database and table space recovery methods
 - How to enable the automatic backup feature
 - Basic usage of the utilities (BACKUP, RESTORE, ROLLFORWARD)
- ▶ Table and index reorganization and statistics collection
 - Automatic table maintenance feature
 - Table reorganization
 - Statistic collection
 - Package rebinding
- ▶ Data movement using EXPORT, IMPORT and LOAD
- ▶ Job scheduling using the Task Center

6.1 DB2 database backup and recovery

A database can become unusable or corrupted in certain situations, such as hardware failure, power interruption or application mistakes. To protect your database from losing any of your data, it is important to have a good recovery strategy. This includes regular schedule for taking backups of the entire database, as well as saving the recovery log files. Your overall strategy should also include procedures for recovering command scripts, applications, user-defined functions (UDFs), stored procedure code in operating system libraries, and load copies.

DB2 distinguishes between two types of databases, non-recoverable and recoverable. *Non-recoverable* means that DB2 can only restore the entire database to the point in time where the backup was taken. This is also called version recovery. Data that is easily recreated can be stored in a non-recoverable database. Data that cannot be easily recreated should be stored in a recoverable database. For a *recoverable* database, DB2 can restore the entire database or a part of the database to the point in time where the interrupt occurred (to end of log) or to a specific point in time. We call it rollforward recovery.

Database logging

A key element of any high availability strategy is database logging. DB2 is logging every transaction, unless told not to, or in case of using the Load utility. The database log files are important if it is needed to recover the database. Recovery log files and the recovery history file are created automatically when a database is created. DB2 has two types of logging mechanisms:

- ▶ Circular logging

Circular logging is the default logging method for a newly created database. If both the LOGARCHMETH1 and LOGARCHMETH2 database configuration parameters are set to OFF then the database is *non-recoverable* and circular logging is used. A set of log files is used in round-robin fashion. These log files are used by crash recovery if needed. Crash recovery is the process where all uncommitted changes are rolled back to a consistent state of the data.

- ▶ Archive logging

By using archive logging, the database is considered as a recoverable database. To enable archive logging, set the LOGARCHMETH1 or LOGARCHMETH2 database configuration parameters to a value other than OFF. Active logs are still available for crash recovery, but you also have the archived logs, which contain committed transaction data. After a successful restore, you can roll the database *forward* (that is, past the time when the

backup image was taken) by using the active and archived logs to either a specific point in time, or to the end of the active logs.

DB2 provides different backup mechanisms. You need to decide between the following:

Offline or Online backup

During an offline backup, applications cannot connect to the database. This is the default backup method when you do not specify any parameter in the **BACKUP** command. After an offline backup, you have a consistent image copy of the entire database. Recoverable database backup operations can also be performed *online*. Online mean that other applications can connect to the database during the backup operation. During an online backup operation, rollforward recovery ensures that *all* table changes are captured and reapplied if that backup is restored. If you have a recoverable database, you can back up, restore, and roll individual table spaces forward, rather than the entire database. When you back up a table space online, it is still available for use, and simultaneous updates are recorded in the logs. When you perform an online restore or a roll forward operation on a table space, the table space itself is not available for use until the operation completes, but users are not prevented from accessing tables in other table spaces.

Full or Incremental backup

A full backup is a image copy of the entire database (Figure 6-1). In a partitioned environment, each database partition backup is written in a separate output file.

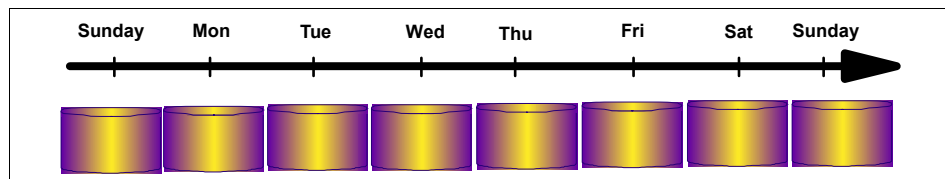


Figure 6-1 Full backup

An incremental backup takes an image copy of changed data since the most recent successful full backup. This is a cumulative backup as shown in Figure 6-2 on page 226. The TRACKMOD parameter (Track modified pages) in the database configuration must be set to ON.

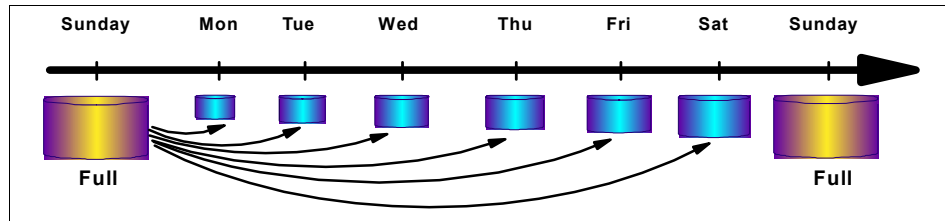


Figure 6-2 Incremental backup

A delta backup is basically an incremental backup. It creates a copy of all data pages that have changed since the last successful backup of any types. This is also known as a non-cumulative backup (Figure 6-3).

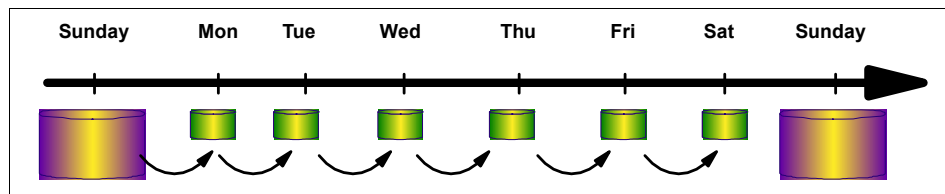


Figure 6-3 Incremental delta backup

The backup strategy depends on your environment. Database workload, availability, and recovery level are all decision factors. A good backup strategy may be a daily offline backup if the database does not have a 24x7 availability requirement. Another strategy can be to do an offline backup every weekend and an incremental backup during the week.

6.1.1 Enable roll forward recovery and log archiving

To use all functionality available in DB2 to prevent data loss, we enable both roll forward recovery and log archiving. This allows us to recover the database to the point in time of failure. The following example walks you through this task in both single and multi-partitioned database environments.

Setup log file management

Starting with DB2 Version 8.2, a new log file manager which is fully integrated in the DB2 engine is provided. The configuration is done using the database configuration parameters called *LOGARCHMETH1* and *LOGARCHMETH2*.

Note: The user exit application **db2uext2** is no longer needed. The database configuration parameters **LOGRETAIN** and **USEREXIT** are deprecated in DB2 9.5, but are still being used by the data servers and clients with prior versions. Any value specified for these configuration parameters will be ignored by the DB2 9.5 database manager.

In our example we show you how to archive DB2 logs to disk and TSM storage. Our first task is to create the file system and directory that will hold the archived log files.

Use **fdisk** or any preferred tool to create the file system **/db2archive** and mount it on each host of the database partitioned environment.

Create directory **/db2archive/db2inst1** and set read/write permissions for the instance owner user ID, so that DB2 can access the directory.

```
mkdir /db2archive/db2inst1
chown db2inst1:db2iadml /db2archive/db2inst1
```

Turn on log archiving by updating database configuration.

```
db2 update db cfg for itsodb using logarchmeth1 disk:/db2archive
```

Note: The command applies to both a single and a multi partitioned database environment. DB2 9.5 provides a single view of all database configuration elements across multiple partitions. You can update the database configuration across all database partitions by invoking the **UPDATE DATABASE CONFIGURATION** command against one database partition. You no longer need to use the **db2_a11** command.

To verify, use the following command:

```
db2_a11 'db2 get db cfg for itsodb | grep LOGARCHMETH1'
```

These steps can also be done through the DB2 Control Center. Right click on the database name shows you the context menu (Figure 6-4 on page 228):

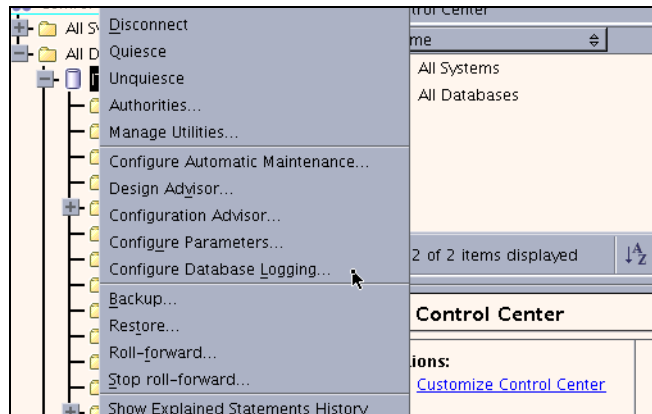


Figure 6-4 Database context menu

Choosing **Configure Database Logging** opens the Wizard as shown in Figure 6-5.

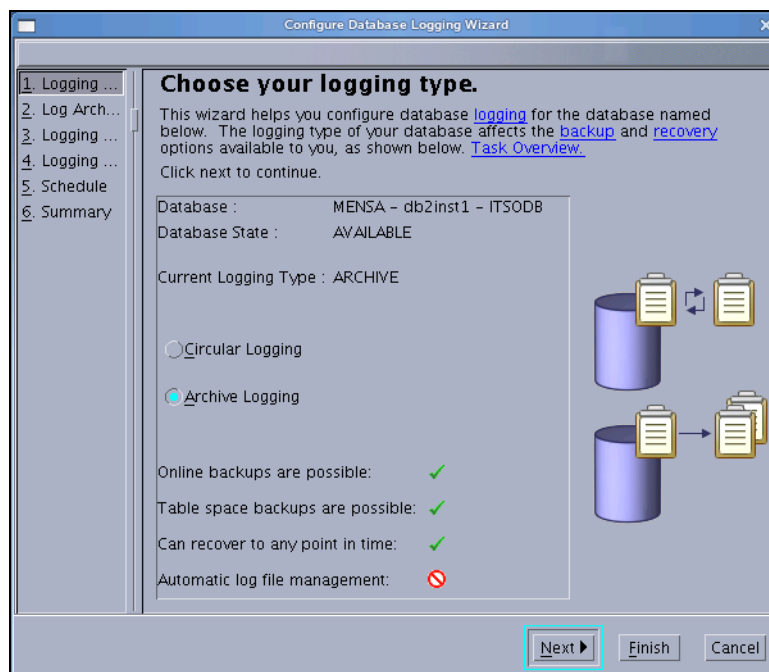


Figure 6-5 Configure Database Logging Wizard

Choose **Archive Logging** and on the next page let DB2 automatically archive the log files. You need to update the primary archive log path as shown in Figure 6-6

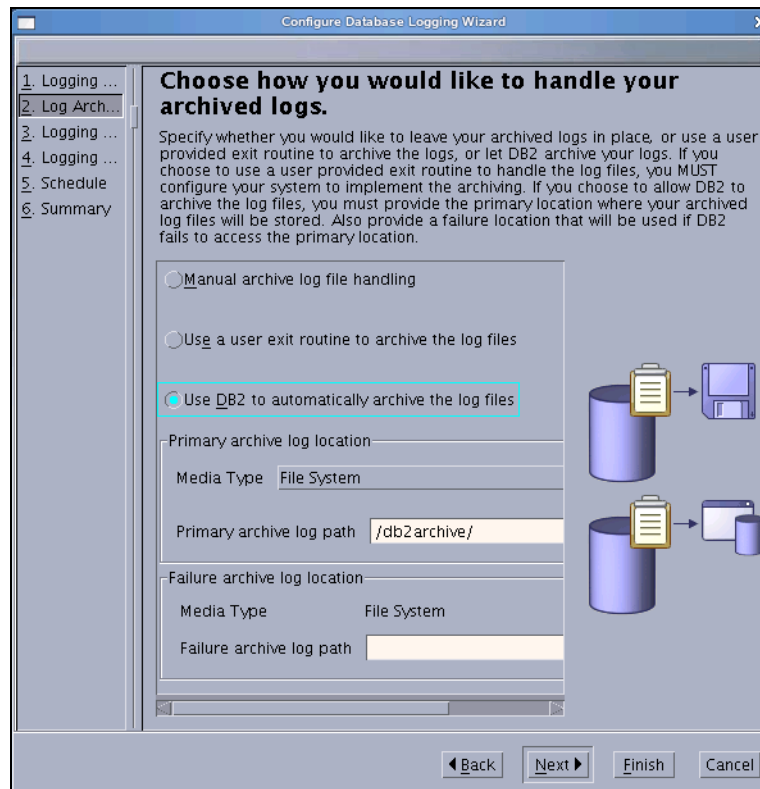


Figure 6-6 Primary archive log path

You can go through all the pages to configure all the related database logging parameters.

After enabling log archiving, DB2 sets the database in backup pending state to ensure a full offline backup is taken before starting with log archiving. To do the backup in a single database partition environment, use the following command

```
db2 backup db itsodb to /db2backup
```

In a partitioned database environment, there are different ways to take the backup. We explain in more detail under 6.1.4, “Backup utility” on page 235. For now, Example 6-1 on page 230 shows one way of backing up a partitioned database.

Example 6-1 Back up all database partitions

```
db2inst1@mensa:~> db2 "backup db itsodb on all dbpartitionnums to /db2backup"
Part  Result
-----
0000  DB20000I  The BACKUP DATABASE command completed successfully.
0001  DB20000I  The BACKUP DATABASE command completed successfully.

Backup successful. The timestamp for this backup image is : 20080201170437
```

Now the database is ready for log archiving. To verify it, use the following command:

```
db2 archive log for itsodb
```

The active log files will be archived from the log directory to /db2archive directory. DB2 uses the following subdirectory structure for the archived log files:

```
/<logarchpath>/instance/DBNAME/PARTITION#/CHAIN#
```

In our case, we see the archived log files for the partitioned database under:

```
/db2archive/db2inst1/ITSODB/NODE0000/C0000000:
-rw-r----- 1 db2inst1 db2grp1 12288 2008-02-01 17:38 S0000000.LOG
/db2archive/db2inst1/ITSODB/NODE0001/C0000000:
-rw-r----- 1 db2inst1 db2grp1 12288 2008-02-01 17:38 S0000000.LOG
```

The database is now enabled to take database backups in online mode as well as to backup individual table spaces.

More parameters for log file management

With the setup we did, the log files archived automatically by the DB2 log manager (*db2logmgr*) to another file system called /db2archive. Depending to your company's needs about log archiving, you may consider to set some other database configuration parameters.

For log archiving, check the following parameters:

- ▶ **LOGARCHMETH2**
This parameters specifies a second location or method to archive the log files. It can be set like LOGARCHMETH1, but points to a different location like another file system or TSM. If both LOGARCHMETH1 and LOGARCHMETH2 are specified, each log file is archived twice. This means that you will have two copies of archived log files in two different locations.
- ▶ **ARCHRETRYDELAY, NUMARCHRETRY, and FAILARCHPATH**
It may be possible that the log archive method specified fails, such as the archive file system full or the TSM server is not available. In such a case, DB2 will retry the archive after the time specified by the ARCHRETRYDELAY

parameter. NUMARCHRETRY specifies the number of attempts that will be made to archive the log files before they are archived to the path specified by the FAILARCHPATH configuration parameter. FAILARCHPATH is a temporary storage area for the log files until the log archive method that failed becomes available again at which time the log files will be moved from this directory to the log archive method. By moving the log files to this temporary location, log directory full situations might be avoided. This parameter must be a fully qualified existing directory.

The log files are critical for database recovery. You should manage the log files and the log space carefully.

Note: To monitor the current log file usage for a running database, use the **GET SNAPSHOT** command or the **LOG_UTILIZATION** administrative view as follows:

```
db2 "SELECT * FROM SYSIBMADM.LOG_UTILIZATION"
```

It shows the percentage of total log space used, the available, and the used log space for each database partition.

Starting with DB2 9.5, DB2 maintains two copies of the log control file SQLLOGCTL.LFH.1 and SQLLOGCTL.LFH.2. You can configure DB2 to also mirror the database log files by setting the following database parameter:

- ▶ **MIRRORLOGPATH**
DB2 provides the capability to mirror the active log files to prevent log data loss in case of a disk crash. We recommend that you place the secondary log path on a physically separate disk.

Another related parameter is

- ▶ **BLK_LOG_DSK_FUL**
The database configuration parameter can be set to prevent disk full errors for the transaction when DB2 cannot create a new log file in the active log path because of a file system full condition. If the parameter is set to YES, DB2 will attempt to create the log file every five minutes until it succeeds.

Note: For a detailed overview of the configuration parameters for database logging, refer to *Data Recovery and High Availability Guide and Reference*, SC23-5848-00

6.1.2 Recovery history file

DB2 log manager and the DB2 utilities such as backup, restore, load, export, or reorg write detailed information into the recovery history file. In a

multi-partitioned database, this file exists in every database partition. To see the entries in the recovery history file, use the LIST HISTORY command.

Example 6-2 illustrates a sample output with the backup information. It shows the backup timestamp and the log files that were active during the backup was taken.

Example 6-2 Backup history

```
db2inst1@mensa:~> db2 list history backup all for itsodb

List History File for itsodb

Number of matching file entries = 2

Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log Backup ID
-----
B D 20080201170437001 F D S0000000.LOG S0000000.LOG
-----
Contains 2 tablespace(s):

00001 SYSCATSPACE
00002 USERSPACE1
-----
Comment: DB2 BACKUP ITSODB OFFLINE
Start Time: 20080201170437
End Time: 20080201170449
Status: A
-----
EID: 1 Location: /db2backup
```

You can also retrieve the history entries for ROLLFORWARD, REORG, CREATE TABLESPACE, ALTER TABLESPACE, DROPPED TABLE, LOAD, RENAME TABLESPACE, and ARCHIVE LOG. To see the archived log files, you have to use the command shown in Example 6-3.

Example 6-3 Archive log history

```
db2inst1@mensa:/> db2 list history archive log all for db itsodb
...
Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log Backup ID
-----
X D 20080207090530 1 D S0000005.LOG C0000001
-----

Comment:
Start Time: 20080207090530
End Time: 20080207090543
```

Status: A

 EID: 25 Location: /db2archive/db2inst1/ITS0DB/NODE0000/C0000001/S0000005.LOG

In a multi-partitioned database environment you have to run the **list history** command on each database partition. You can use the *DB_HISTORY* administrative view or *SYSPROC.ADMIN_LIST_HIST()* function to get the information from the history file in all database partitions. Sample commands are:

```
db2 "select * from sysibmadm.DB_HISTORY"
db2 "select * from table(SYSPROC.ADMIN_LIST_HIST())"
```

Over the time, the recovery history file will increase in size. So when an entry is no longer relevant, because the associated recovery objects would no longer be needed to recover the database, you might want to remove, or prune, those entries from the recovery history file. This can be done by the **prune history** command or a call of the *ADMIN_CMD* procedure with the *PRUNE_HISTORY* parameter. By setting the database configuration parameters *NUM_DB_BACKUPS* or/and *REC_HIS_RETENTN* the DB2 database manager automatically prunes the database history file.

Note: In Version 9.5, if you set the new configuration parameter *AUTO_DEL_REC_OBJ* to ON, the database manager will also delete backup images, load copy images, and log files associated with any history file entries that it automatically prunes. Once *AUTO_DEL_REC_OBJ* is enabled, the system will only perform this maintenance when both the *NUM_DB_BACKUPS* and *REC_HIS_RETENTN* values are exceeded.

6.1.3 Enable usage of TSM

DB2 supports database backup and log archiving using TSM storage. The TSM system consists of TSM server and TSM client. The database server uses the TSM client API to communicate with the TSM server. Our test was based on TSM server version 5.5.0 and TSM client and client API version 5.4.0.

After installing the TSM client and TSM API on each database server, we have to configure DB2 so that DB2 can communicate with TSM server. Edit the file *\$INSTHOME/sqllib/userprofile* to add the TSM variables as shown in Example 6-4:

Example 6-4 TSM definition in the DB2 instance userprofile

```
export DSMI_DIR=/opt/tivoli/tsm/client/api/bin64
export DSMI_CONFIG=/opt/tivoli/tsm/client/api/bin64/dsm.opt
export DSMI_LOG=$HOME/sqllib/db2dump
```

Note: If the DSMI variables are changed, you need to restart the DB2 instance. To be sure that the right values are used during db2start time, run the following commands:

```
db2stop
export | grep DSMI
db2start
```

If you want to verify the DSMI variables for a running DB2, use the **ps** command as follows:

```
ps -fu db2inst1 | grep db2sysc
ps eww <PID_of_db2sysc>
```

The TSM client must be configured as well. In the following file, define the TSM server to be used:

```
/opt/tivoli/tsm/client/api/bin64/dsm.sys
```

Example 6-5 shows the file in our test environment.

Example 6-5 Sample dsm.sys configuration file on the database server

SErvername	TSM_SERVER1
NodeName	db2mensa
passwordaccess	generate
COMMmethod	TCPip
TCPServeraddress	9.43.86.125
TCPPort	1500

The value specified for SErvername has to be specified in the dsm.opt file as shown in Example 6-6. The location of the dsm.opt file is specified with the *DSMI_CONFIG* variable.

Example 6-6 dsm.opt file

```
SErvername TSM_SERVER1
tapeprompt no
```

Each *NodeName* has to be registered at the TSM server. The TSM server administrator can do this using the `register node` command. Tivoli requires certain storage pools to store the archived log files and the database backup copy. By default, an archive storage pool (ARCHIVEPOOL) is used for storing DB2 archive logs sent by the DB2 log manager and a backup storage pool (BACKUPPOOL) is used to store all DB2 backup files.

The TSM admin user will also set up a password for the node. We need to save this TSM password at our database server which is the TSM client. Change to the directory \$INSTHOME/sql1lib/adsm and do the steps shown in Example 6-7.

Example 6-7 set the password at the TSM client

```
db2inst1@mensa:/db2home/db2inst1/sql1lib/adsm> su root
root's Password:
db2inst1@mensa:/db2home/db2inst1/sql1lib/adsm> ./dsmapiw

*****
* Tivoli Storage Manager                                *
* API Version = 5.4.0                                  *
*****
Enter your current password:
Enter your new password:
Enter your new password again:

Your new password has been accepted and updated.
```

DB2 provides the *db2adutl* tool to manage the backup files in TSM. You can view and delete DB2 backup files as well as bring backup files from TSM storage to the database server. All options are shown if you type **db2adutl**.

Note: A complete description for using db2adutl is in *Command Reference*, SC23-5846-00

6.1.4 Backup utility

We have completed all the steps, as described in 6.1.1, “Enable roll forward recovery and log archiving” on page 226. Now we describe how to take different types of database backup.

Backing up databases

DB2 provides several database backup options:

- **Full offline:** A full offline database backup allows you to restore the database image without applying any log records. Example 6-8 shows the offline backup command.

Example 6-8 Full offline backup to disk

```
db2inst1@mensa:/> db2 backup db itsodb to /db2backup
```

Backup successful. The timestamp for this backup image is : 20080204081423

- **Full online:** During online backup, the database is still accessible for applications. Rollforward recovery has to be enabled. To be able to recover the database from an online backup, at least you need the range of active log files from the online backup time frame. To make sure that these log files available, you can use the *INCLUDE LOGS* clause within the backup command (the default behavior in V9.5). This function includes the required log files for recovery with the database backup image. If you need to ship the backup images to a disaster recovery site, you do not have to send the log files separately or package them together with database backup copy manually. Further, you do not have to decide which log files are required to guarantee the consistency of an online backup. Example 6-11 provides a sample of a full online backup.

Example 6-9 Full online backup to disk with included logs

```
db2inst1@mena:/> db2 backup db itsodb online to /db2backup include logs
```

Backup successful. The timestamp for this backup image is : 20080204090001

- **Table space:** A table space backup allows you to back up only the specified table spaces. In Example 6-10 we back up only table space TBSPDMS_0.

Example 6-10 Table space backup online

```
TBSPDMS_0
```

```
db2inst1@mena:/> db2 "backup db itsodb tablespace (TBSPDMS_0) online to /db2backup"
```

Backup successful. The timestamp for this backup image is : 20080204090637

- **Full incremental:** An incremental backup saves backup time and resources. Example 6-11 provides an incremental backup example.

Example 6-11 Incremental backup to disk

```
db2inst1@mena:/> db2 backup db itsodb online incremental to /db2backup
```

Backup successful. The timestamp for this backup image is : 20080204091101

- **Delta backup:** Example 6-12 shows the delta database backup after taking a full database backup.

Example 6-12 Delta backup to TSM

```
db2inst1@mena:/> db2 backup db itsodb online incremental delta use tsm
```

Backup successful. The timestamp for this backup image is : 20080204091912

Note: To enable incremental or delta backup, you need to set the TRACKMOD database configuration parameter to ON as follows:

```
db2 update db cfg for itsodb using TRACKMOD ON
```

Once the TRACKMOD is set to ON, you need to do a full database backup once before starting incremental or delta backup.

Backing up partitioned databases

In a partitioned database environment, there are three possible ways to back up the database:

- ▶ Back up each database partition one at a time.
- ▶ Use the db2_all command with the BACKUP DATABASE command to back up all the database partitions that you specify.
- ▶ Run a single system view (SSV) backup to back up some or all of the database partitions simultaneously.

Backup each database partition one at a time is time consuming and error prone. Using the db2_all command to issue the backup database command on all the database partitions simplifies the process and saves some time. However, since the database partition containing the catalog can not be backed up simultaneously with non-catalog database partitions, you still have to back up the catalog database partition first, then backup the rest of the database partitions. Assuming that database partition 0 is the catalog partition, the commands are:

```
db2_all ' <<+0< db2 backup db itsodb to /db2backup'
db2_all ' <<-0< db2 backup db itsodb to /db2backup'
```

Whether you back up the database partition one at a time or using db2_all, managing backup images can be difficult due to the timestamp difference in each database partition backup image. The log files can not be included in the backup image.

In Version 9.5, when you perform a backup operation from the catalog node of a partitioned database, you can specify which database partitions to include in the backup, or specify that all the database partitions should be included. The specified database partitions will be backed up simultaneously, and the backup timestamp associated with all specified database partitions will be the same. Also, you can include database logs with a SSV backup; including logs in backup images is the default behavior for SSV backup (that is, if you specify the ON DBPARTITIONNUM parameter). Finally, when you restore from a SSV backup image, you can specify to roll forward to end of logs, which is the minimum recovery time calculated by the database manager.

Example 6-13 shows the command to backup all database partitions. The ON DBPARTITIONNUMS options specifies the database partition to be backup. If you like to exclude the log files from the backup image, use the EXCLUDE LOGS command parameter.

Example 6-13 SSV backup of all database partitions

```
db2inst1@mena:/> db2 "backup db itsodb on all dbpartitionnums online to
/db2backup compress"

Part  Result
-----
0000  DB20000I  The BACKUP DATABASE command completed successfully.
0001  DB20000I  The BACKUP DATABASE command completed successfully.

Backup successful. The timestamp for this backup image is : 20080201170437
```

Monitoring the backup database utility

The backup database utility writes an entry into the database recovery history file for each backup. You can query it using the **LIST HISTORY** command. When the database backup is running, you can use the **LIST UTILITY** command to monitor the progress of the backup utility as shown in Example 6-14. The option **WITH DETAILS** is used in this example.

Example 6-14 LIST UTILITIES to monitor the backup

```
db2inst1@mena:/> db2 list utilities show detail

ID                                     = 327
Type                                  = BACKUP
Database Name                         = ITSODB
Partition Number                     = 0
Description                           = online db
Start Time                           = 02/21/2008 14:28:40.181501
State                                = Executing
Invocation Type                      = User
Throttling:
  Priority                             = Unthrottled
Progress Monitoring:
  Estimated Percentage Complete = 29
    Total Work                    = 304720105 bytes
    Completed Work                 = 87135441 bytes
    Start Time                     = 02/21/2008 14:28:40.181510

ID                                     = 135
Type                                  = BACKUP
Database Name                         = ITSODB
```

Partition Number = 1
Description = online db
Start Time = 02/21/2008 14:28:41.835045
State = Executing
Invocation Type = User
Throttling:
 Priority = Unthrottled
Progress Monitoring:
 Estimated Percentage Complete = 98
 Total Work = **224002009 bytes**
 Completed Work = **220510145 bytes**
 Start Time = 02/21/2008 14:28:41.835055

The same information can be retrieved using the SNAPUTIL administrative view or the SNAP_GET_UTIL table function. Used in conjunction with the SNAPUTIL_PROGRESS administrative view or the SNAP_GET_UTIL_PROGRESS table function, the same information as the **LIST UTILITIES SHOW DETAIL** command are provided. Example 6-15 shows how to use the administrative views.

Example 6-15 administrative views to monitor the backup

```
db2inst1@mena:/> db2 "select UTILITY_TYPE, substr(UTILITY_DESCRIPTION,1,15),  
UTILITY_STATE, DBPARTITIONNUM from SYSIBMADM.SNAPUTIL"
```

UTILITY_TYPE	2	UTILITY_STATE	DBPARTITIONNUM

BACKUP	online db	EXECUTE	0
BACKUP	online db	EXECUTE	1

2 record(s) selected.

```
db2inst1@mena:/> db2 "select SNAPSHOT_TIMESTAMP, UTILITY_STATE,  
PROGRESS_START_TIME, PROGRESS_TOTAL_UNITS, PROGRESS_COMPLETED_UNITS,  
DBPARTITIONNUM from SYSIBMADM.SNAPUTIL_PROGRESS"
```

SNAPSHOT_TIMESTAMP	UTILITY_STATE	PROGRESS_START_TIME
PROGRESS_TOTAL_UNITS	PROGRESS_COMPLETED_UNITS	DBPARTITIONNUM

2008-02-21-14.28.43.442266	EXECUTE	2008-02-21-14.28.40.181510
304720105		88446673
		0
2008-02-21-14.28.45.099408	EXECUTE	2008-02-21-14.28.41.835055
224002009		220510145
		1

2 record(s) selected.

Check backup image

You can use `db2ckbkp` to test the integrity of a backup image and to determine whether or not the image can be restored. Example 6-16 shows how to use the tool to verify a backup image.

Example 6-16 db2ckbkp output

```
db2inst1@mena:/db2backup> db2ckbkp
ITS0DB.0.db2inst1.NODE0000.CATN0000.20080204081423.001
```

```
[1] Buffers processed: #####
```

```
Image Verification Complete - successful.
```

The `db2ckbkp` utility can also be used to display the metadata stored in the backup header, such as table space and container information or automatic storage paths. The backup media header information is shown using the `-h` option as illustrated in Example 6-17. The output shows that we used the `INCLUDE LOGS` clause and also the `COMPRESS` option to take the backup.

Example 6-17 Using db2ckbkp to show the backup media header

```
db2inst1@mena:/db2backup> db2ckbkp -h
ITS0DB.0.db2inst1.NODE0000.CATN0000.20080205141153.001
```

```
=====
MEDIA HEADER REACHED:
=====
      Server Database Name      -- ITS0DB
      Server Database Alias    -- ITS0DB
      Client Database Alias    -- ITS0DB
      Timestamp                -- 20080205141153
      Database Partition Number -- 0
      Instance                 -- db2inst1
      Sequence Number          -- 1
      Release ID               -- C00
      Database Seed            -- 50B96082
      DB Comment's Codepage (Volume) -- 0
      DB Comment (Volume)      --
      DB Comment's Codepage (System) -- 0
      DB Comment (System)      --
      Authentication Value      -- -1
      Backup Mode              -- 1
      Includes Logs            -- 1
      Compression              -- 1
      Backup Type              -- 0
      Backup Gran.             -- 0
```

```

Status Flags                -- 21
System Cats inc             -- 1
Catalog Partition Number   -- 0
DB Codeset                  -- UTF-8
DB Territory                --
LogID                       -- 1202239916
LogPath                     --
/database/db2inst1/NODE0000/SQL00002/SQLLOGDIR/
Backup Buffer Size          -- 1048576
Number of Sessions         -- 1
Platform                   -- 1E

```

The proper image file name would be:
 ITSODB.0.db2inst1.NODE0000.CATN0000.20080205141153.001

[1] Buffers processed: #####

Image Verification Complete - successful.

For more options of the db2ckbkp utility, see the *Command Reference*, SC23-5846-00.

Backup database with Control Center

The backup utility is fully integrated within the DB2 Control Center. Figure 6-7 shows where you can invoke the backup utility.

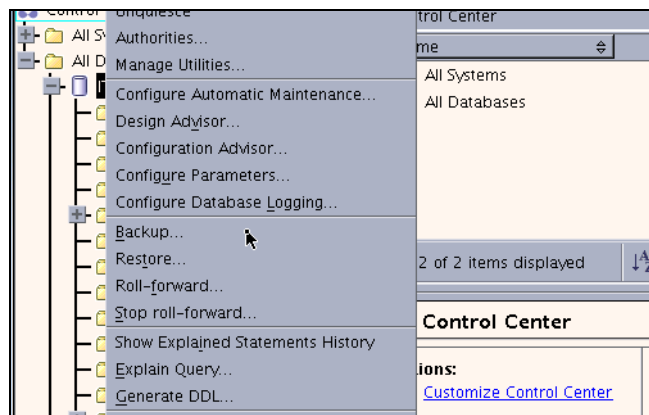


Figure 6-7 Backup in Control Center

Go through all of the panels and fill out all needed parameters. At the end you can list the generated backup statements as shown in Figure 6-8 on page 242.

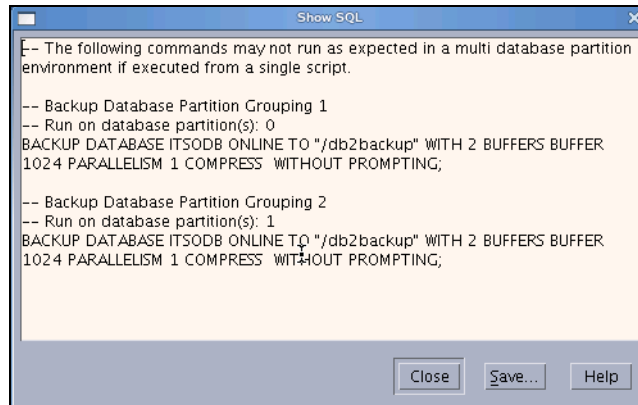


Figure 6-8 Generated backup statements

You have two options, either run this generated script immediately or let it run under control of the scheduler, which is integrated in the Tools catalog. See Figure 6-9 on page 243.

Another way to backup the database is to generate the scripts through the Control Center and use your own scheduler to run this backup script periodically.

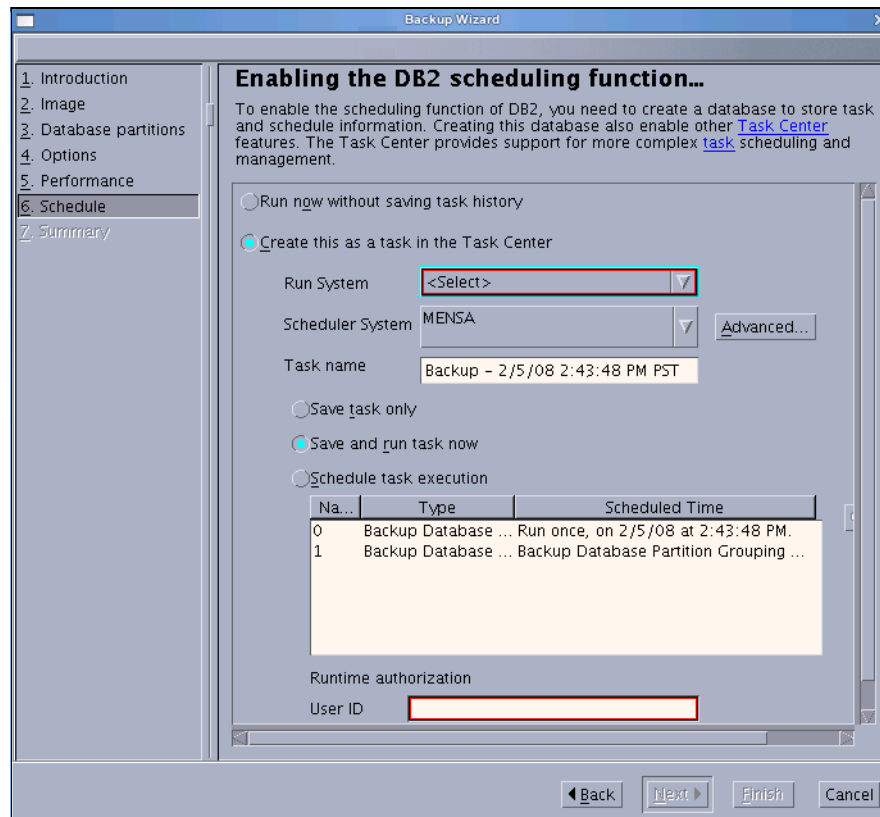


Figure 6-9 Scheduling backup scripts

Backup database sample script

We already showed various backup commands in 6.1.4, “Backup utility” on page 235. In Example 6-18 we show you a script to backup our sample database ITS0DB, which is partitioned. Call this script with the db2_all command:

```
db2_all '/db2home/db2inst1/database_backup.ksh db2inst1 itsodb'
```

Example 6-18 Backup script database_backup.ksh

```
#!/usr/bin/ksh
#
# Script: backing up database
#-----

echo running against instance: ${1}
echo running against database: ${2}
INSTANCE=${1}
DB_ALIAS=${2}
```

```

#-----
# check for db2profile and execute it, if available
#-----

DBHOME=`finger -l $INSTANCE | grep Directory | awk '{print $2}'`

if [[ ! -f /$DBHOME/sqllib/db2profile ]]
then

    echo "File not found: " /$DBHOME/sqllib/db2profile
    echo "please check host- and instancename"
    echo "          ***** script abnormally ended *****"
    "

    exit 1

fi

. /$DBHOME/sqllib/db2profile

for i in 1 2 3
do
    echo ""
    echo "Start backing up database $DB_ALIAS on node $DB2NODE "
    echo ""

    db2 backup db ${DB_ALIAS} to /db2backup with 4 buffers parallelism 2

    if [ $? = '0' ]
    then
        szDate=$(date)
        echo ""
        echo "backing up database $DB_ALIAS on node $DB2NODE successfully $szDate" |
tee -a /$DBHOME/$DB_ALIAS.log
        echo ""
        exit 0
    fi

    if [ "$i" -le "3" ]
    then
        echo "Start another trial"
    fi
done
szDate=$(date)
echo ""
echo "backing up database $DB_ALIAS on node $DB2NODE faild after 3 trials
$szDate " | tee -a /$DBHOME/$DB_ALIAS.log
echo ""

```


exit 8

Modify this sample so that it fits your environment and requirements.

The new DB2 9.5 fewture single system view (SSV) backup is the recommended way for taking backup for all database partitions (Example 6-13 on page 238).

Note: For a detailed explanation of backing up databases, refer to Chapter 10 “Database Backup” in *Data Recovery and High Availability Guide and Reference*, SC23-5848-00.

Enable automatic backup

DB2 provides automatic maintenance capabilities for performing database backup, keeping statistics current, and reorganizing tables and indexes as necessary. When enabling automatic backup, you can specify the following criteria for determining when a backup is required:

- ▶ Maximum time between backups
- ▶ Maximum log space used between backups

Using these settings, DB2 determines if a backup is required and runs the backup job on the next available maintenance window (a user-defined time period for running the automatic maintenance activities).

Note: You can still perform manual backup operations when automatic maintenance is configured. DB2 will perform the automatic backup operations only if they are required.

You can configure automatic backup using the graphical user interface tools, the command line interface, or the AUTOMAINT_SET_POLICY stored procedure.

Configure automatic backup using the wizard

In this section, we demonstrate how to enable automatic database backup for the ITS0DB database using the Configure Automatic Maintenance wizard. The wizard can be launched from the DB2 Control Center as shown in Figure 6-10 on page 246

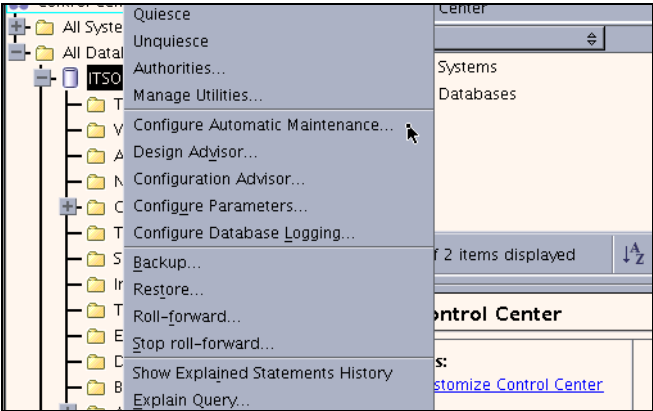


Figure 6-10 Configure Automatic Maintenance in Control Center

Figure 6-11 shows the panel where you can specify the time frame in which the automatic maintenance can be performed.

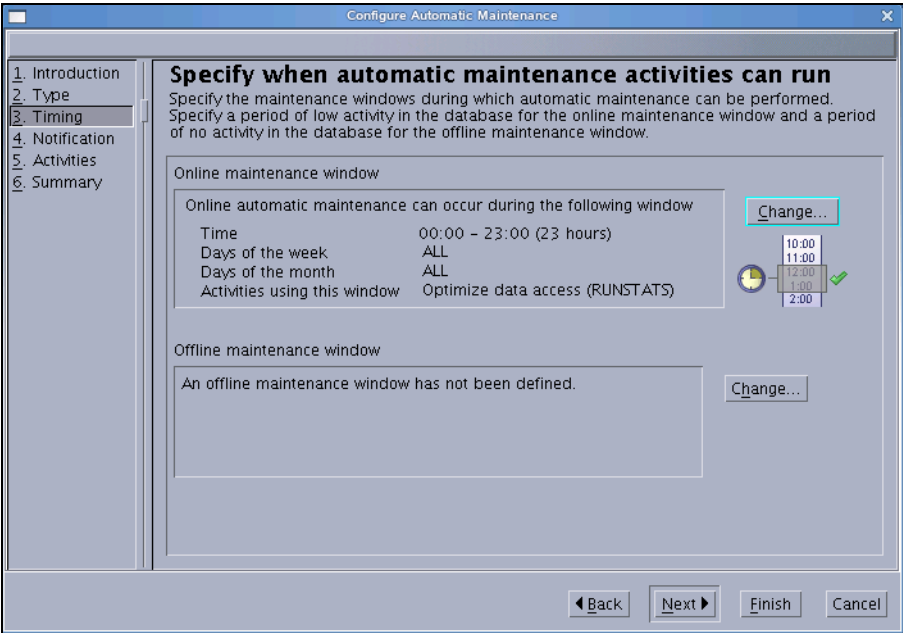


Figure 6-11 Specify maintenance panel

At the activities page, you have to select the check box for automatic backup, see Figure 6-12 on page 247. Use the **Configure Settings** button to specify policy settings details such as backup location.

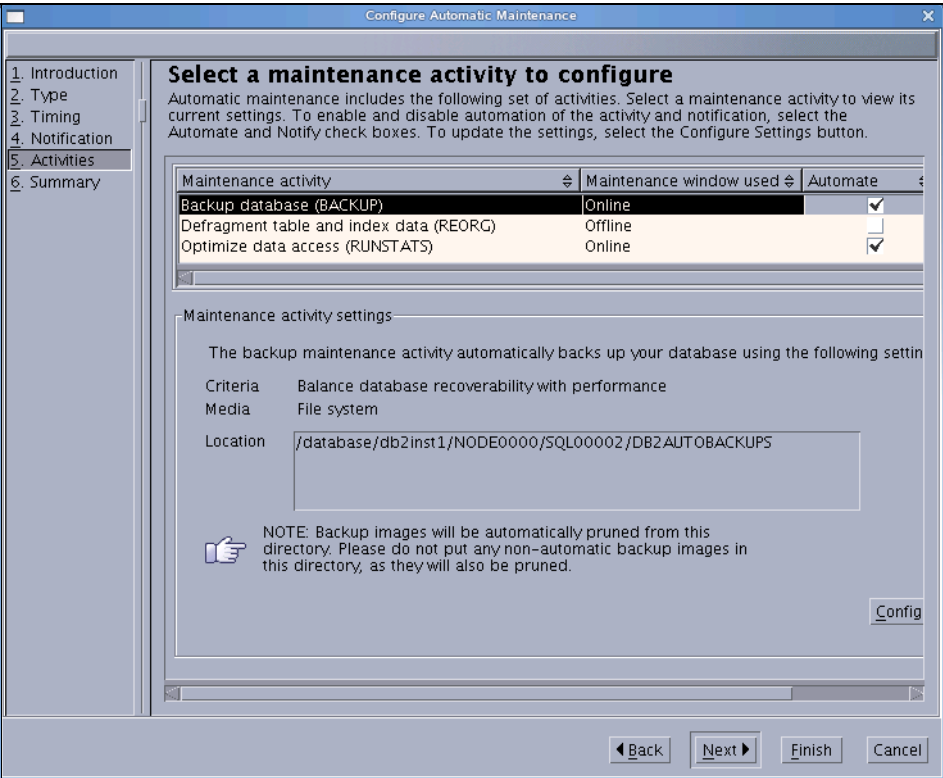


Figure 6-12 select maintenance activity

Configure automatic backup using the CLP

Enabling the automatic maintenance features is controlled by the automatic maintenance database configuration parameters as shown in Example 6-19.

Example 6-19 Automatic maintenance parameters

```
db2inst1@mena:/db2home/db2inst1> db2 get db cfg for itsodb | grep Automatic
Automatic maintenance                (AUTO_MAINT) = ON
Automatic database backup             (AUTO_DB_BACKUP) = OFF
Automatic table maintenance           (AUTO_TBL_MAINT) = ON
Automatic runstats                    (AUTO_RUNSTATS) = ON
Automatic statement statistics         (AUTO_STMT_STATS) = OFF
Automatic statistics profiling         (AUTO_STATS_PROF) = OFF
Automatic profile updates              (AUTO_PROF_UPD) = OFF
Automatic reorganization              (AUTO_REORG) = OFF
```

To configure automatic backup using the command line interface, you have to set AUTO_MAINT and AUTO_DB_BACKUP to ON.

Configure automatic backup using AUTOMAINT_SET_POLICY

DB2 provides a stored procedure AUTOMAINT_SET_POLICY to specify your automated maintenance policy for automatic backup (also for AUTO_REORG, and AUTO_RUNSTATS, as well as to set the days and times for the online and offline maintenance windows).

Note: The DB2 built-in administrative routines administrative views provide an easy to use programmatic interface for performing a variety of DB2 administrative tasks. These routines and views can be invoked from an SQL-based application, a DB2 command line, or a command script.

To configure automatic backup using the AUTOMAINT_SET_POLICY system stored procedure, you have to create an XML configuration input specifying details like backup media, whether the backup should be online or offline, and frequency of the backup.

The *AUTOMAINT_SET_POLICY* stored procedure takes two input parameters:

- ▶ Maintenance type: AUTO_BACKUP, AUTO_REORG, AUTO_RUNSTATS, or MAINTENANCE_WINDOW
- ▶ Configuration: A BLOB type argument or an XML file specifying the configuration in XML format.

The procedure *AUTOMAINT_GET_POLICYFILE* allows you to obtain details on the policies, in XML format. The XML file can be edited as needed, and the changes are re-imported by using the *AUTOMAINT_SET_POLICYFILE* stored procedure.

In Example 6-20, we obtain the current settings from the database, edit the file to define the duration to four hours, and set settings for the database.

Example 6-20 Change maintenance window

```
db2inst1@mena:/> db2 "call
AUTOMAINT_GET_POLICYFILE('MAINTENANCE_WINDOW','window.xml')"

db2inst1@mena:/> cat /db2home/db2inst1/sqllib/tmp/window.xml
<?xml version="1.0" encoding="UTF-8"?>
<DB2MaintenanceWindows
xmlns="http://www.ibm.com/xmlns/prod/db2/autonomic/config" >

  <!-- Online Maintenance Window -->
  <OnlineWindow Occurrence="During" startTime="00:00:00" duration="24" >
    <DaysOfWeek>All</DaysOfWeek>
    <DaysOfMonth>All</DaysOfMonth>
    <MonthsOfYear>All</MonthsOfYear>
```

```
</OnlineWindow>
</DB2MaintenanceWindows>
```

```
-----
-- now edit the file /db2home/db2inst1/sqllib/tmp/window.xml
-- We will set the maintenance window with a duration of 4 hours only, so
-- we change duration="24" to duration="04"
-- Now import the new settings.
-----
```

```
db2inst1@mena:/> db2 "CALL
AUTOMAINT_SET_POLICYFILE('MAINTENANCE_WINDOW','window.xml')"
```

```
Return Status = 0
```

The procedures can also be used to configure automatic backup, reorg, and runstats by replacing MAINTENANCE_WINDOW in the commands above with one of the following:

- ▶ AUTO_BACKUP
- ▶ AUTO_REORG
- ▶ AUTO_RUNSTATS

There are four sample policy files (one for each type) shipped with DB2 under directory \$HOME/sqllib/samples/automaintcfg.

Note: For more information, see the topic “Configuring an automated maintenance policy using SYSPROC.AUTOMAINT_SET_POLICY or SYSPROC.AUTOMAINT_SET_POLICYFILE” in *Data Recovery and High Availability Guide and Reference*, SC23-5848-00.

6.1.5 DB2 database recovery

Database recovery means rebuilding of a database or table space after a problem such as media or storage failure, or to rebuild the database on another system.

There are three types of recovery:

- ▶ *Crash recovery* protects a database from being left in an inconsistent, or unusable, state when transactions (also called units of work) are interrupted unexpectedly.
- ▶ *Version recovery* is the restoration of a previous version of the database, using an image that was created during a backup operation.

- *Rollforward recovery* can be used to reapply changes that were made by transactions that were committed after a backup was made.

Crash recovery

A crash recovery may be needed after a severe error or condition that causes the database or the database manager to end abnormally. Unit of work (UOW) that have not been flushed to disk at the time of failure leave the database in an inconsistent state. The database needs to be recovered.

By default, DB2 starts crash recovery automatically to attempt to recover the database. If you do not want the automatic restart behavior, set the `AUTORESTART` database configuration parameter to `OFF`. You will need to issue the **db2 restart database** command manually when a database failure occurs. The administration notification log records information about crash recovery.

Version recovery

Version recovery means to restore the database to the point in time when the database backup was taken. The database backup image file used was created during a backup operation. Every unit of work after the time of the backup is lost. Version recovery can be used with non-recoverable databases (databases without archived logs). It also can be used with recoverable databases by using the `WITHOUT ROLLING FORWARD` option on the `RESTORE DATABASE` command.

Full database restore from an offline backup

A full database restore example from an *offline* backup is:

```
db2 restore db itsodb from /db2backup taken at 20080204081423
```

If the offline database backup was taken from a recoverable database, you have to use the `WITHOUT ROLLING FORWARD` clause within the restore command to do the version recovery.

Note: In a partitioned database environment, you must restore all database partitions, and the backup images that you use for the restore database operation must all have been taken at the same time.

```
db2_all "db2 restore db itsodb from /db2backup taken at 20080201170437"
```

Full database restore from an online backup

Version recovery of a database using backup taken *online* means that you restore the database to the point in time when the online backup was finished. During the online backup, transactions were still running against the database. So the log files written during the backup time need to be applied. Once the

restore command is run successfully, you have to roll forward to the end-of-backup point in time. This can be done with the V9.5 command

```
db2 rollforward db itsodb to end of backup and stop
```

If you used the backup command with the INCLUDE LOGS clause, you can specify the LOGTARGET option within the restore command. Example 6-21 shows how this works for a single partition database:

Example 6-21 Version recovery with online backup and included logs

```
db2inst1@mena:/> db2 "restore db itsodb from /db2backup taken at
20080206143831 logtarget /db2backup/logs"
```

```
DB20000I The RESTORE DATABASE command completed successfully.
```

```
db2inst1@mena:/> db2 "rollforward db itsodb to end of backup overflow log path
(/db2backup/logs)"
```

Rollforward Status

```
Input database alias           = itsodb
Number of nodes have returned status = 1

Node number                    = 0
Rollforward status             = DB working
Next log file to be read       = S0000001.LOG
Log files processed            = S0000000.LOG - S0000000.LOG
Last committed transaction     = 2008-02-06-22.38.32.000000 UTC
```

```
DB20000I The ROLLFORWARD command completed successfully.
```

```
db2inst1@mena:/> db2 "rollforward db itsodb stop"
```

Rollforward Status

```
Input database alias           = itsodb
Number of nodes have returned status = 1

Node number                    = 0
Rollforward status             = not pending
Next log file to be read       =
Log files processed            = S0000000.LOG - S0000000.LOG
Last committed transaction     = 2008-02-06-22.38.32.000000 UTC
```

```
DB20000I The ROLLFORWARD command completed successfully.
```

Full database restore with incremental

The best way to restore an incremental backup image is to use the **AUTO** option. DB2 will collect the needed backup images, full and incremental, to restore the database.

```
db2 restore db itsodb incremental auto from /db2backup taken at 20080204091912
```

If you want to know which backup images are required to restore an incremental backup, you can use the **db2ckrst** command as follows:

```
db2ckrst -d itsodb -t 20080204091912
```

The command queries the database history and generates a list of timestamps for the backup images and suggested restore order.

Rollforward recovery

To use the rollforward recovery method, the database must be set up as a recoverable database. At the end of the *restore* operation, the database will be in *rollforward pending* state which gives you the possibility to roll forward the database to a specific point in time or to the end of log.

Note: Restore and roll forward of the database is done in two steps, using the **RESTORE** and the **ROLLFORWARD** commands. You can use the **RECOVER** command to restore and roll forward the database with only one command.

```
db2 recover database itsodb
```

The **RECOVER** command reads the database history file to get all needed information to restore and roll forward the database.

You can check the progress of the rollforward with the command

```
db2 rollforward db itsodb query status
```

Using the **roll forward** command you can specify a *timestamp* or *end of logs*. It tells the database how far the roll forward should be done.

There are two types of rollforward recovery to consider:

- ▶ Database roll forward recovery

This process applies transactions recorded in the database logs to the database. The rollforward utility reads the needed log files, archived or active logs. Already archived logs will be retrieved using the log archive method. The simplest way to roll forward the database as far as possible is using the command

```
db2 "rollforward db itsodb to end of logs and stop"
```


If you need to roll forward the database to a specific point in time, use the following commands:

```
db2 "rollforward db itsodb to 2008-02-06-20.15.30 using local time"
db2 "rollforward db itsodb query status"
db2 "rollforward db itsodb stop"
```

► **Table space roll forward recovery**

If the database is enabled for forward recovery, you have the option of backing up, restoring, and rolling forward table spaces instead of the entire database. Table space rollforward can be done to the end of logs or to a point in time. The time must be greater or equal to the minimum recovery time for that table space. Example 6-22 shows you how to determine that time.

Example 6-22 Restore to minimum point in time

```
db2inst1@mena:/> db2 "restore db itsodb tablespace (userspace1) online
from /db2backup taken at 20080206145105"
DB20000I The RESTORE DATABASE command completed successfully.
```

```
----- From another login shell, find Minimum recovery time -----
db2inst1@mena:/> db2 connect to itsodb
db2inst1@mena:/> db2 list tablespaces show detail
```

Tablespaces for Current Database

Tablespace ID	= 0
Name	= SYSCATSPACE
Type	= Database managed space
Contents	= All permanent data. Regular table
space.	
State	= 0x0000
Detailed explanation:	
Normal	
Total pages	= 16384
Useable pages	= 16380
Used pages	= 12288
Free pages	= 4092
High water mark (pages)	= 12288
Page size (bytes)	= 4096
Extent size (pages)	= 4
Prefetch size (pages)	= 4
Number of containers	= 1
Tablespace ID	= 1
Name	= TEMPSPACE1
Type	= System managed space
Contents	= System Temporary data
State	= 0x0000

Detailed explanation:

Normal

Total pages	= 1
Useable pages	= 1
Used pages	= 1
Free pages	= Not applicable
High water mark (pages)	= Not applicable
Page size (bytes)	= 4096
Extent size (pages)	= 32
Prefetch size (pages)	= 32
Number of containers	= 1
Tablespace ID	= 2
Name	= USERSPACE1
Type	= Database managed space
Contents	= All permanent data. Large table

space.

State = 0x0080

Detailed explanation:

Roll forward pending

Total pages	= 8192
Useable pages	= 8160
Used pages	= 0
Free pages	= 8160
High water mark (pages)	= 96
Page size (bytes)	= 4096
Extent size (pages)	= 32
Prefetch size (pages)	= 32
Number of containers	= 1
Minimum recovery time	= 2008-02-06-22.51.24.000000

---- Minimum recovery time (UCT) is **2008-02-06-22.51.24**

---- so continue the roll forward

```
db2inst1@mensa: /> db2 "rollforward db itsodb stop tablespace (userspace1)
online "
```

```
SQL1275N The stoptime passed to roll-forward must be greater than or equal
to "2008-02-06-22.51.24.000000 UTC", because database "ITSODB" on node(s)
"0" contains information later than the specified time.
```

```
db2inst1@mensa: /> db2 "rollforward db itsodb to 2008-02-06-22.51.24
tablespace (userspace1) online "
```

Rollforward Status

Input database alias	= itsodb
Number of nodes have returned status	= 1
Node number	= 0
Rollforward status	= TBS working

```
Next log file to be read      = S0000003.LOG
Log files processed          = -
Last committed transaction   = 2008-02-06-22.52.06.000000 UTC
```

```
DB20000I The ROLLFORWARD command completed successfully.
db2inst1@mensa: /> db2 "rollforward db itsodb stop tablespace (userspace1)
online "
```

Rollforward Status

```
Input database alias          = itsodb
Number of nodes have returned status = 1

Node number                   = 0
Rollforward status            = not pending
Next log file to be read      =
Log files processed           = -
Last committed transaction     = 2008-02-06-22.52.06.000000 UTC
```

```
DB20000I The ROLLFORWARD command completed successfully.
```

The minimum recovery time is updated by DB2 when data definition language (DDL) statements are run against the table space, or against tables in the table space. The table space must be rolled forward to at least the minimum recovery time, so that it is synchronized with the information in the system catalog tables.

During a table space rollforward, each log file needs to be read, even if they do not contain log records that affect that table space. This may be time consuming. To skip the log files known not containing any log records affecting the table space, set the DB2_COLLECT_TS_REC_INFO registry variable to ON (the default value). The table space change history file (db2tschg.his) keeps track of which logs should be processed for each table space.

Note: DB2 supports the recovery of a dropped table by using table space level restore, if the table space where the table resides is enabled for dropped table recovery. This can be done during table space creation or by altering an existing table using the DROPPED TABLE RECOVERY ON option. An example is shown in section 6.1.7, “Recovering a dropped table sample scenario” on page 264.

Monitoring restore and rollforward utilities

Similar to the backup utility, the RESTORE and ROLLFORWARD write entries into the database recovery history file for each backup. You can query it using the LIST HISTORY command. During the runtime of the utilities, you can use the LIST UTILITY command to monitor RESTORE and ROLLFORWARD. The same information

can be retrieved using the administrative views SNAPUTIL and SNAPUTIL_PROGRESS or the table functions SNAP_GET_UTIL and SNAP_GET_UTIL_PROGRESS. Example 6-23 shows the output for a table space restore on database partition 0 and a following rollforward.

Example 6-23 Monitor restore and rollforward

```
db2inst1@mensa: /> db2 list utilities

ID                                = 331
Type                              = RESTORE
Database Name                     = ITSODB
Partition Number                  = 0
Description                       = tablespace TSDATA2...
Start Time                       = 02/21/2008 15:23:01.033971
State                            = Executing
Invocation Type                   = User

db2inst1@mensa: /> db2 list utilities show detail

ID                                = 331
Type                              = RESTORE
Database Name                     = ITSODB
Partition Number                  = 0
Description                       = tablespace TSDATA2...
Start Time                       = 02/21/2008 15:23:01.033971
State                            = Executing
Invocation Type                   = User
Progress Monitoring:
    Completed Work                 = 54837248 bytes
    Start Time                    = 02/21/2008 15:23:01.033976

db2inst1@mensa: /> db2 "select UTILITY_TYPE, substr(UTILITY_DESCRIPTION,1,15),
UTILITY_STATE, DBPARTITIONNUM from SYSIBMADM.SNAPUTIL"

UTILITY_TYPE      2      UTILITY_STATE DBPARTITIONNUM
-----
RESTORE           tablespace TSDA EXECUTE              0

1 record(s) selected.

db2inst1@mensa: /> db2 "select SNAPSHOT_TIMESTAMP, UTILITY_STATE,
PROGRESS_START_TIME, PROGRESS_TOTAL_UNITS, PROGRESS_COMPLETED_UNITS,
DBPARTITIONNUM from SYSIBMADM.SNAPUTIL_PROGRESS"

SNAPSHOT_TIMESTAMP      UTILITY_STATE      PROGRESS_START_TIME
PROGRESS_TOTAL_UNITS PROGRESS_COMPLETED_UNITS DBPARTITIONNUM
```

```
-----
-----
2008-02-21-15.23.03.523409 EXECUTE      2008-02-21-15.23.01.033976
-                                     54837248                      0
```

1 record(s) selected.

db2inst1@mensa: /> db2 list utilities

```
ID                                = 332
Type                              = ROLLFORWARD RECOVERY
Database Name                     = ITSODB
Partition Number                  = 0
Description                       = Online Tablespace Rollforward Recovery: 7
Start Time                       = 02/21/2008 15:25:20.435220
State                             = Executing
Invocation Type                   = User
Progress Monitoring:
    Estimated Percentage Complete = 0
```

db2inst1@mensa: /> db2 list utilities show detail

```
ID                                = 332
Type                              = ROLLFORWARD RECOVERY
Database Name                     = ITSODB
Partition Number                  = 0
Description                       = Online Tablespace Rollforward Recovery: 7
Start Time                       = 02/21/2008 15:25:20.435220
State                             = Executing
Invocation Type                   = User
Progress Monitoring:
    Estimated Percentage Complete = 0
    Phase Number                  = 1
        Description               = Forward
        Total Work                 = 38614 bytes
        Completed Work             = 38614 bytes
        Start Time                 = 02/21/2008 15:25:20.435226

    Phase Number [Current]        = 2
        Description               = Backward
        Total Work                 = 5627893427 bytes
        Completed Work             = 0 bytes
        Start Time                 = 02/21/2008 15:25:20.988645
```

```
db2inst1@mensa: /> db2 "select UTILITY_TYPE, substr(UTILITY_DESCRIPTION,1,15),
UTILITY_STATE, DBPARTITIONNUM from SYSIBMADM.SNAPUTIL"
```

UTILITY_TYPE	2	UTILITY_STATE	DBPARTITIONNUM

ROLLFORWARD_RECOVERY	Online Tablespace	EXECUTE	0

1 record(s) selected.

```
db2inst1@mensa: /> db2 "select SNAPSHOT_TIMESTAMP, UTILITY_STATE,
substr(PROGRESS_DESCRIPTION,1,10) desc, PROGRESS_START_TIME,
PROGRESS_TOTAL_UNITS, PROGRESS_COMPLETED_UNITS, DBPARTITIONNUM from
SYSIBMADM.SNAPUTIL_PROGRESS"
```

SNAPSHOT_TIMESTAMP	UTILITY_STATE	DESC	PROGRESS_START_TIME
PROGRESS_TOTAL_UNITS	PROGRESS_COMPLETED_UNITS	DBPARTITIONNUM	

2008-02-21-15.25.21.767401	EXECUTE	Forward	2008-02-21-15.25.20.435226
38614		38614	0
2008-02-21-15.25.21.767401	EXECUTE	Backward	2008-02-21-15.25.20.988645
5627893427		0	0

2 record(s) selected.

For the rollforward recovery there are two phases, *Forward* and *Backward*. During the forward phase, log files are read and the log records are applied to the database. During the backward phase, any uncommitted changes applied during the forward phase are rolled back.

Database recovery using the Control Center

Restore and roll forward are fully integrated in the DB2 Control Center. You can access the wizard from the context menu of the database as shown in Figure 6-13 on page 259.

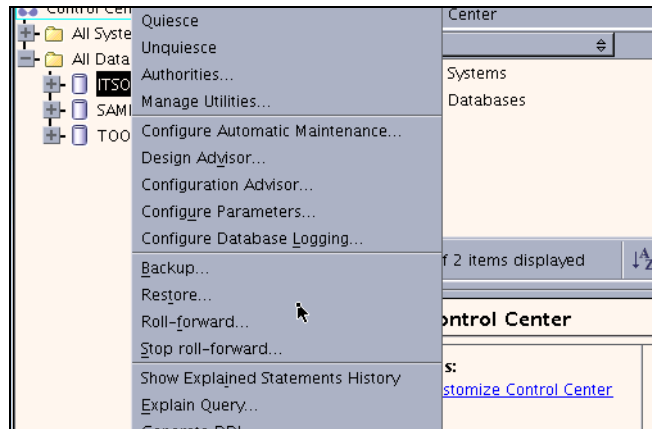


Figure 6-13 Database context menu

The Restore Data Wizard (Figure 5-16 on page 177) guides you through different steps and options to recover your data.

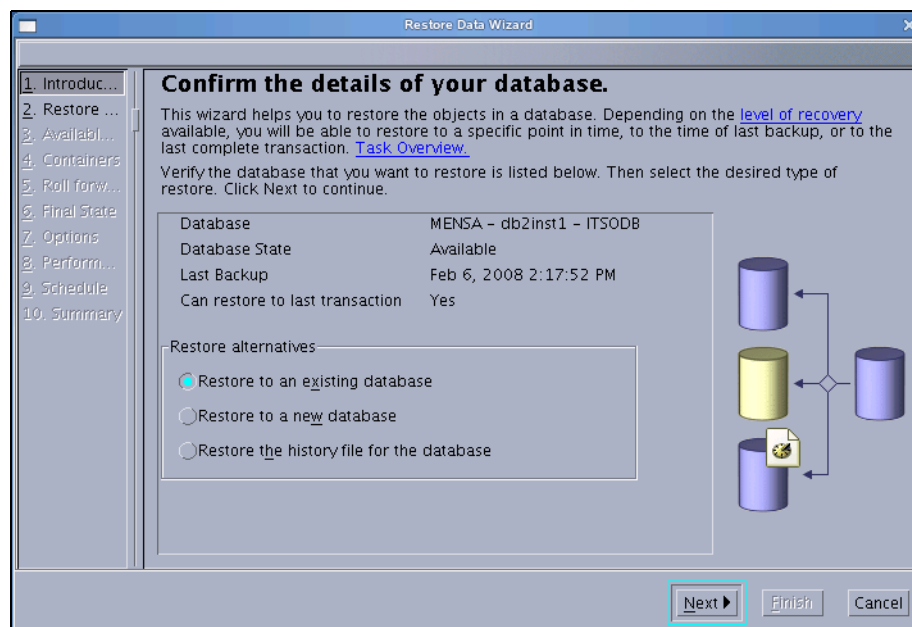


Figure 6-14 Restore Data Wizard

You can choose, for instance, if you want to do a full or table space restore, which backup image to restore, and how far the roll forward should be done.

Note: For a detailed explanation of the database recovery options, refer to Part 2 “Data recovery” in *Data Recovery and High Availability Guide and Reference*, SC23-5848-00.

6.1.6 Redirected restore sample scenario

Using a redirected restore, you can change the physical location of table space containers stored in the backup image when restoring the database into same database or into another database. Redirected restore is one using the RESTORE DATABASE command with the REDIRECT parameter, or by using the Restore Database wizard in the Control Center. You have to redefine each table space container you want to modify with the SET TABLESPACE CONTAINER command or in the wizard. With Version 9.5, alternatively, you can generate a redirected restore script from a backup image and modify it as needed.

The steps are as follows:

1. Use the restore utility to generate the redirected restore script. The RESTORE DATABASE command must be called with the REDIRECT option and the GENERATE SCRIPT option.
2. Edit the generated redirected restore script as required. You can modify:
 - Restore options
 - Automatic storage paths
 - Container layout and paths
3. Run the modified redirected restore script.

In a partitioned database environment, you have to perform these steps for each database partition because the table space container definitions may be different.

For our sample database ITS0DB we will change the location and the size of a table space using a redirected restore with a generated script. Example 6-24 shows you first, how we generate the scripts for our partitioned database.

Example 6-24 Prepare database and generate scripts for redirected restore

```
db2inst1@mensa:/db2home/db2inst1> db2 connect to itsodb
```

Database Connection Information

```
Database server      = DB2/LINUX8664 9.5.0
SQL authorization ID = DB2INST1
Local database alias = ITS0DB
```



```
db2inst1@mena:/db2home/db2inst1> db2 -tvf tscrt.sql
create tablespace TSDATA1 in IBMDEFAULTGROUP managed by database using (file
'/tablespaces/db2inst1/itsodb/NODE0000/TSDATA1.000' 40M) on dbpartitionnum(0)
using (file '/tablespaces/db2inst1/itsodb/NODE0001/TSDATA1.000' 40M) on
dbpartitionnum(1)
DB20000I The SQL command completed successfully.
```

```
db2inst1@mena:/db2home/db2inst1> db2 "create table TAB1 (id int not null
primary key, name char(10)) in tsdata1"
DB20000I The SQL command completed successfully.
```

```
db2inst1@mena:/db2home/db2inst1> db2 "insert into tab1 values (1, 'Tina')"
DB20000I The SQL command completed successfully.
```

```
db2inst1@mena:/db2home/db2inst1> db2 connect reset
DB20000I The SQL command completed successfully.
```

```
db2inst1@mena:/db2home/db2inst1> db2 backup db itsodb on all dbpartitionnums
to "/db2backup"
```

```
Part Result
```

```
-----
0000 DB20000I The BACKUP DATABASE command completed successfully.
0001 DB20000I The BACKUP DATABASE command completed successfully.
```

Backup successful. The timestamp for this backup image is : 20080206173239

```
db2inst1@mena:/db2home/db2inst1> db2_all "<<+0< db2 restore db itsodb from
/db2backup taken at 20080206173239 redirect generate script
itsodb_node0000.clp"
```

```
DB20000I The RESTORE DATABASE command completed successfully.
```

```
mena: db2 restore db itsodb ... completed ok
```

```
db2inst1@mena:/db2home/db2inst1> db2_all "<<+1< db2 restore db itsodb from
/db2backup taken at 20080206173239 redirect generate script
itsodb_node0001.clp"
```

```
DB20000I The RESTORE DATABASE command completed successfully.
```

```
gemini: db2 restore db itsodb ... completed ok
```

```
db2inst1@mena:/db2home/db2inst1> ls *.clp
itsodb_node0000.clp itsodb_node0001.clp
```

Now we need to modify the two generated scripts, one for each database partition. In our case, we change the table space container path and the size for TSDATA. Example 6-25 on page 262 shows a part of the script which is related to table space TSDATA on database partition 0 with the changes we did.

Example 6-25 Part of the changed redirected restore script

```
-- ** Tablespace name                = TSDATA1
-- ** Tablespace ID                  = 4
-- ** Tablespace Type                = Database managed space
-- ** Tablespace Content Type        = All permanent data. Large
table space.
-- ** Tablespace Page size (bytes)   = 8192
-- ** Tablespace Extent size (pages) = 32
-- ** Using automatic storage        = No
-- ** Auto-resize enabled            = No
-- ** Total number of pages          = 5120
-- ** Number of usable pages         = 5088
-- ** High water mark (pages)        = 96
-- *****
SET TABLESPACE CONTAINERS FOR 4
-- IGNORE ROLLFORWARD CONTAINER OPERATIONS
USING (
-- FILE  '/tablespaces/db2inst1/itsodb/NODE0000/TSDATA1.000' 5120
  FILE  '/tablespaces/ITS00000/TSDATA1.000' 4000
);
```

Once both scripts are changed, we create the new container directory on each database partition and run the scripts against their database partition. Example 6-26 shows the output for database partition 0.

Example 6-26 Execute the script for database partition 0

```
db2inst1@mena:/db2home/db2inst1> mkdir /tablespaces/ITS00000/
db2inst1@mena:/db2home/db2inst1> export DB2NODE=0
db2inst1@mena:/db2home/db2inst1> db2 terminate
DB20000I The TERMINATE command completed successfully.
db2inst1@mena:/db2home/db2inst1> db2 -tvf itsodb_node0000.clp
UPDATE COMMAND OPTIONS USING S ON Z ON ITSODB_NODE0000.out V ON
DB20000I The UPDATE COMMAND OPTIONS command completed successfully.

SET CLIENT ATTACH_DBPARTITIONNUM 0
DB20000I The SET CLIENT command completed successfully.

SET CLIENT CONNECT_DBPARTITIONNUM 0
DB20000I The SET CLIENT command completed successfully.

RESTORE DATABASE ITSODB FROM '/db2backup' TAKEN AT 20080206173239 INTO ITSODB
REDIRECT
SQL2539W Warning! Restoring to an existing database that is the same as the
backup image database. The database files will be deleted.
Do you want to continue ? (y/n) y
SQL1277W A redirected restore operation is being performed. Table space
configuration can now be viewed and table spaces that do not use automatic
```

```
storage can have their containers reconfigured.
DB20000I The RESTORE DATABASE command completed successfully.

SET TABLESPACE CONTAINERS FOR 4 USING ( FILE
'/tablespaces/ITS00000/TSDATA1.000'          4000 )
DB20000I The SET TABLESPACE CONTAINERS command completed successfully.

RESTORE DATABASE ITSODB CONTINUE
DB20000I The RESTORE DATABASE command completed successfully.
```

We also run script `itsodb_node0001.clp` against database partition 1. Because the database is enabled for rollforward recovery, we have to do the rollforward before connecting to the database. Example 6-27 shows the steps and the new table space container definition.

Using the generated script, we were able to redefine the table space container path and size very easy.

Example 6-27 Roll forward after redirected restore

```
db2inst1@mena:/db2home/db2inst1> db2 rollforward db itsodb stop
```

Rollforward Status

Input database alias		= itsodb	
Number of nodes have returned status		= 2	
Node number	Rollforward	Next log	Log files
processed	Last committed transaction		
	status	to be read	

0	not pending		
S0000002.LOG-S0000002.LOG	2008-02-07-01.32.41.000000	UTC	
1	not pending		-
2008-02-07-01.36.12.000000	UTC		

```
DB20000I The ROLLFORWARD command completed successfully.
db2inst1@mena:/db2home/db2inst1> db2 connect to itsodb
```

Database Connection Information

Database server	= DB2/LINUX8664 9.5.0
SQL authorization ID	= DB2INST1
Local database alias	= ITSODB

```
db2inst1@mena:/db2home/db2inst1> db2 list tablespace container for 4
SQL0104N An unexpected token "container" was found following "TABLESPACE".
```

```
Expected tokens may include: "CONTAINERS".  SQLSTATE=42601
db2inst1@mensa:/db2home/db2inst1> db2 list tablespace containers for 4
```

Tablespace Containers for Tablespace 4	
Container ID	= 0
Name	= /tablespaces/ITS00000/TSDATA1.000
Type	= File

Note: If you want to perform a redirected restore from one machine to another one using **db2adut1** and TSM, you need to set `logarchopt1` and `vendoropt` database configuration parameters. A detailed sample can be find under section 'Recovering data using db2adut1' in Chapter 11 'Recover overview' of the *Data Recovery and High Availability Guide and Reference*, SC23-5848-00.

6.1.7 Recovering a dropped table sample scenario

It may be possible, that user dropped a database table by mistake or the table data needs to be recovered. In such a case, DB2 provides a dropped table recovery feature. In this section, we demonstrate how to recover a dropped table through table space restore. The table space will be restored from an older backup image and during the roll forward of the database, the dropped table data will be exported.

To set up the scenario environment, we first alter the table space TSDATA1 to allow table space level recovery, take a database backup, and drop table `tab1` which is in table space TSDATA1. See Example 6-28.

Example 6-28 Prepare for dropped table recovery

```
db2inst1@mensa:/> db2 connect to itsodb

Database Connection Information

Database server          = DB2/LINUX8664 9.5.0
SQL authorization ID     = DB2INST1
Local database alias     = ITSODB

db2inst1@mensa:/> db2 alter tablespace TSDATA1 dropped table recovery on
DB20000I The SQL command completed successfully.
db2inst1@mensa:/> db2 connect reset
DB20000I The SQL command completed successfully.
db2inst1@mensa:/> db2 backup db itsodb on all dbpartitionnums to "/db2backup"
Part  Result
```

```
-----
0000 DB20000I The BACKUP DATABASE command completed successfully.
0001 DB20000I The BACKUP DATABASE command completed successfully.
```

Backup successful. The timestamp for this backup image is : 20080207095407

```
db2inst1@mensa:/> db2 connect to itsodb
```

Database Connection Information

```
Database server      = DB2/LINUX8664 9.5.0
SQL authorization ID = DB2INST1
Local database alias = ITSODB
```

```
db2inst1@mensa:/> db2 drop table tab1
DB20000I The SQL command completed successfully.
db2inst1@mensa:/> db2 "select * from tab1"
SQL0204N "DB2INST1.TAB1" is an undefined name.  SQLSTATE=42704
db2inst1@mensa:/> db2 connect reset
DB20000I The SQL command completed successfully.
```

To recover the dropped table tab1 t, follow the following procedure.

1. Restore the table space in which the dropped table is stored from a database or table space backup. Please note that the whole table space is restored, means also the other tables in that table space are restored and later rolled forward to the current state.

In our example, the dropped table was in table space TSDATA1 which resides in database partitions 0 and 1. In Example 6-29 we restore the table space from the last full backup image.

Example 6-29 Table space restore for all database partitions

```
db2inst1@mensa:/> db2_all 'db2 restore db itsodb "tablespace(tsdatal)" online
from /db2backup taken at 20080207090432'
```

```
DB20000I The RESTORE DATABASE command completed successfully.
mensa: db2 restore db itsodb ... completed ok
```

```
DB20000I The RESTORE DATABASE command completed successfully.
gemin: db2 restore db itsodb ... completed ok
```

2. Create a directory in a shared file system available for all database partitions or create the same directory on every physical database partition. We need the directory for DB2 to export the data of the dropped table. There needs to be sufficient space available in that file system for the exported table data.

```
db2_all 'mkdir /db2backup/rest_table'
```

3. Determine the backup ID by looking in the history file (Example 6-30). In a multi-partitioned environment this has to be done on the catalog partition.

Example 6-30 List history for dropped table

```
db2inst1@mensa: /> db2 list history dropped table all for itsodb

List History File for itsodb

Number of matching file entries = 1

Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log Backup ID
-- --
D T 20080207095643
000000000000350800040004
-----
"DB2INST1"."TAB1" resides in 1 tablespace(s):

00001 TSDATA1
-----
Comment: DROP TABLE
Start Time: 20080207095643
End Time: 20080207095643
Status: A
-----
EID: 38

DDL: CREATE TABLE "DB2INST1"."TAB1" ( "ID" INTEGER NOT NULL , "NAME" CHAR(10)
) DISTRIBUTE BY HASH("ID") IN "TSDATA1" ;
-----
```

4. Roll forward the database with the correct table ID. The table ID is listed in the backup ID column (Example 6-31).

Example 6-31 Roll forward with the table ID

```
db2 'rollforward db itsodb to end of logs on all dbpartitionnums and complete
tablespace(tsdatab1) online recover dropped table 000000000000350800040004 to
/db2backup/rest_table'

Rollforward Status

Input database alias           = itsodb
Number of nodes have returned status = 2

Node number Rollforward      Next log      Log files
processed   Last committed transaction
              status          to be read
```

```

-----
-----
0 not pending -
2008-02-07-01.32.41.000000 UTC
1 not pending -
2008-02-07-01.36.12.000000 UTC

```

DB20000I The ROLLFORWARD command completed successfully.

5. Upon finishing the roll forward, DB2 restores the table data to the export directory in the subdirectory NODEnnnn in a file called data. Now get DDL from the history file to create the table and load all data from the export file into the table. The **CREATE TABLE** DDL was put into the recovery history file during the **DROP TABLE** command as seen in Example 6-30 on page 266. The commands we need to do now to recover the table are shown in Example 6-32:

Example 6-32 Recreate the table and Import the data

```
db2inst1@mena:/> db2 connect to itsodb
```

Database Connection Information

```

Database server      = DB2/LINUX8664 9.5.0
SQL authorization ID = DB2INST1
Local database alias = ITSODB

```

```
db2inst1@mena:/> db2 'CREATE TABLE "DB2INST1"."TAB1" ( "ID" INTEGER NOT NULL ,
"NAME" CHAR(10) )  DISTRIBUTE BY HASH("ID")  IN "TSDATA1"'
```

DB20000I The SQL command completed successfully.

```
db2inst1@mena:/> db2_all '<<+0< db2 connect to itsodb; db2 import from
/db2backup/rest_table/NODE0000/data of del insert into tab1*
```

Database Connection Information

```

Database server      = DB2/LINUX8664 9.5.0
SQL authorization ID = DB2INST1
Local database alias = ITSODB

```

SQL3109N The utility is beginning to load data from file
 ///db2backup/rest_table/NODE0000/data".

SQL3110N The utility has completed processing. "0" rows were read from the
 input file.

SQL3221W ...Begin COMMIT WORK. Input Record Count = "0".

SQL3222W ...COMMIT of any database changes was successful.

SQL3149N "0" rows were processed from the input file. "0" rows were successfully inserted into the table. "0" rows were rejected.

```

Number of rows read      = 0
Number of rows skipped   = 0
Number of rows inserted  = 0
Number of rows updated   = 0
Number of rows rejected  = 0
Number of rows committed = 0

```

```

mensa: db2 connect to itsodb completed ok
db2inst1@mensa:/> db2_all '<<+1< db2 connect to itsodb; db2 import from
/db2backup/rest_table/NODE0001/data of del insert into tab1*

```

Database Connection Information

```

Database server      = DB2/LINUX8664 9.5.0
SQL authorization ID = DB2INST1
Local database alias = ITSODB

```

SQL3109N The utility is beginning to load data from file
 "//db2backup/rest_table/NODE0001/data".

SQL3110N The utility has completed processing. "2" rows were read from the
 input file.

SQL3221W ...Begin COMMIT WORK. Input Record Count = "2".

SQL3222W ...COMMIT of any database changes was successful.

SQL3149N "2" rows were processed from the input file. "2" rows were
 successfully inserted into the table. "0" rows were rejected.

```

Number of rows read      = 2
Number of rows skipped   = 0
Number of rows inserted  = 2
Number of rows updated   = 0
Number of rows rejected  = 0
Number of rows committed = 2

```

gemini: db2 connect to itsodb completed ok

```

db2inst1@mensa:/> db2 "select * from tab1"

```

```

ID          NAME
-----

```



```
1 Tina
2 Jonas
```

```
2 record(s) selected.
```

The table was recreated and the table data are available.

6.2 Table/index reorganization and statistics collection

When considerable changes have been made to a table, the physical data pages for the table and its indexes will become discontinuous or fragmented. If a clustering index exists for a specific table, it may become badly clustered after a lot of changes have been made against the table and index key entries. Additional cost such as CPU time and I/O operations will be required to deal with similar workload under these conditions. It may lead to obvious performance degradation. Performing maintenance activities on your databases is essential in ensuring that they are optimized for performance and recover ability.

DB2 offers the configurable automatic table maintenance function. If you are unsure about when and how to reorganize your tables and indexes and collect statistics information, you can incorporate automatic reorganization as part of your overall database maintenance plan.

If you want to manually reorganize the tables and indexes, DB2 provides a *REORG* utility to do this. After finishing the reorganization tasks, use the *RUNSTATS* utility to gather information about the physical data storage characteristics of a table and the associated indexes. Such information includes, for example, the number of records the table has, how many pages are occupied by the table and the associated indexes, the cluster ratio of the associated clustering index, and so forth. The DB2 optimizer will take advantage of these statistics when determining the access path to the data to gain better performance for subsequent database operations.

If you create a database in Version 9.1 or 9.5, automatic statistics collection is enabled by default. With automatic statistic collection enabled, the DB2 database manager automatically runs the *RUNSTATS* utility in the background to ensure that the correct statistics are collected and maintained.

6.2.1 Automatic table maintenance

Starting with Version 8.2, DB2 provides automatic maintenance capabilities for performing database backup, keeping statistics current, and reorganizing tables and indexes as necessary. It can be time-consuming to determine whether and when to run maintenance activities, automatic maintenance removes the burden

from you. You can manage the enablement of the automatic maintenance features simply and flexibly by using the automatic maintenance database configuration parameters. The default values for a DB2 Version 9.5 database are shown in Example 6-33.

Example 6-33 Automatic maintenance database configuration parameters

```
db2inst1@mensa: /> db2 get db cfg for itsodb | grep Automatic
Automatic maintenance                (AUTO_MAINT) = ON
Automatic database backup            (AUTO_DB_BACKUP) = OFF
Automatic table maintenance          (AUTO_TBL_MAINT) = ON
Automatic runstats                   (AUTO_RUNSTATS) = ON
Automatic statement statistics       (AUTO_STMT_STATS) = OFF
Automatic statistics profiling       (AUTO_STATS_PROF) = OFF
Automatic profile updates            (AUTO_PROF_UPD) = OFF
Automatic reorganization             (AUTO_REORG) = OFF
```

The hierarchy of the parameters allows you to tune all automatic maintenance on or off without affecting the respective activity settings. For instance, if `AUTO_TBL_MAINT` parameter is set to OFF, the parameter `AUTO_RUNSTATS` is not of interest.

Enable automatic table maintenance

Automatic table maintenance can be enabled by using the command line or the Configure Automatic Maintenance wizard.

To use the command line, you have to set each of the following configuration parameters to ON:

- ▶ `AUTO_MAINT`
- ▶ `AUTO_TBL_MAINT`
- ▶ `AUTO_RUNSTATS` for automatic statistics collection
- ▶ `AUTO_REORG` for automatic reorganization

To use the Configure Automatic Maintenance wizard, open the wizard from the Control Center by right-clicking a database object as shown in Figure 6-15 on page 271:

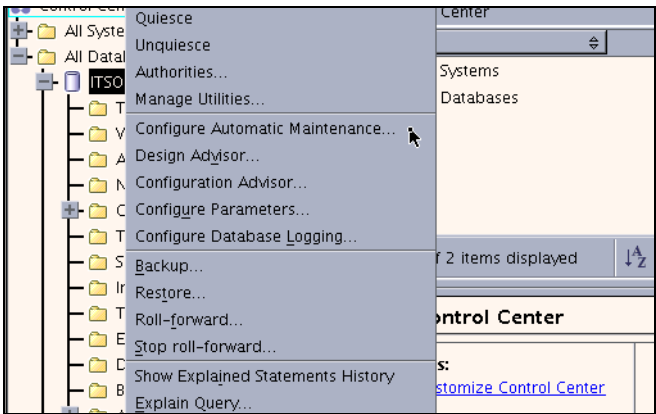


Figure 6-15 Open Configure Automatic Maintenance wizard

Select **Configure Automatic Maintenance** from the pop-up panel. You can also open the wizard from the detail panel of the Control Center. To do so, expand the All databases tree in the Control Center and select the database. On the bottom-right of the Control Center is the detail panel. Connect to the database from there and you will see more details about the database. See Figure 6-16.

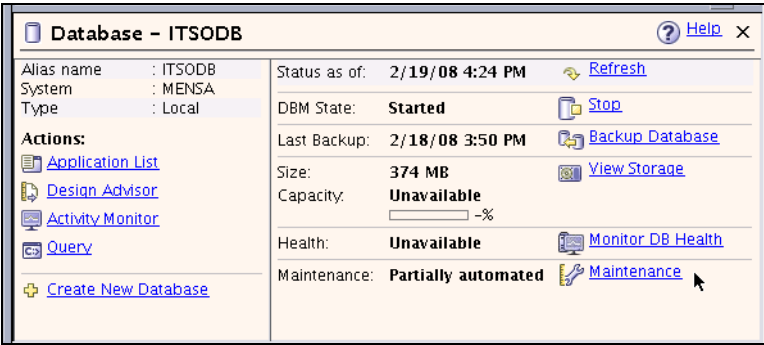


Figure 6-16 database detail panel of the Control Center

Select **Maintenance** to start the wizard.

The wizard open with general information (Figure 6-17 on page 272)

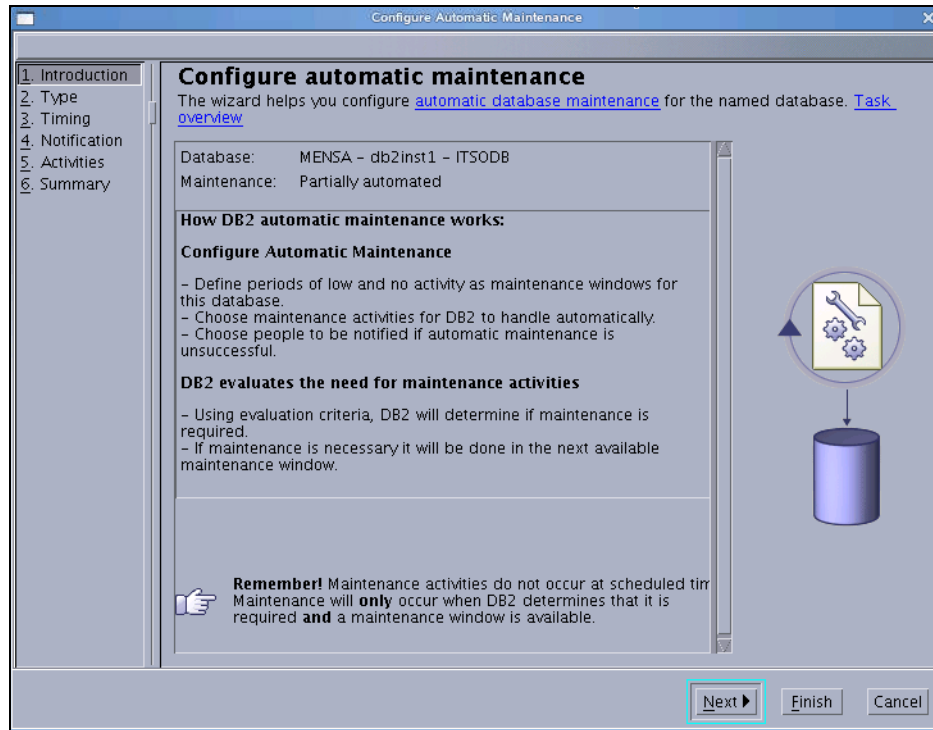


Figure 6-17 Configure Automatic Maintenance wizard

You can specify maintenance windows for the execution of the online activities like the RUNSTATS utility and for the offline activities like the REORG utility. Figure 6-18 on page 273 shows a sample.

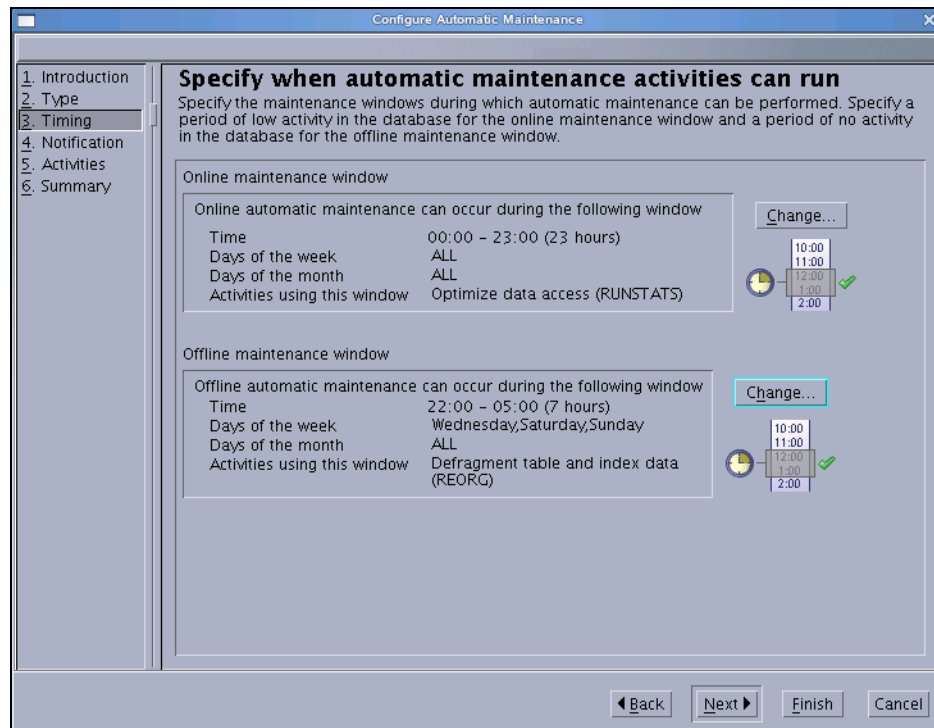


Figure 6-18 Specify maintenance panel

To enable the automatic table maintenance activities, you have to select the related check-box at the activities page as shown in Figure 6-19 on page 274:

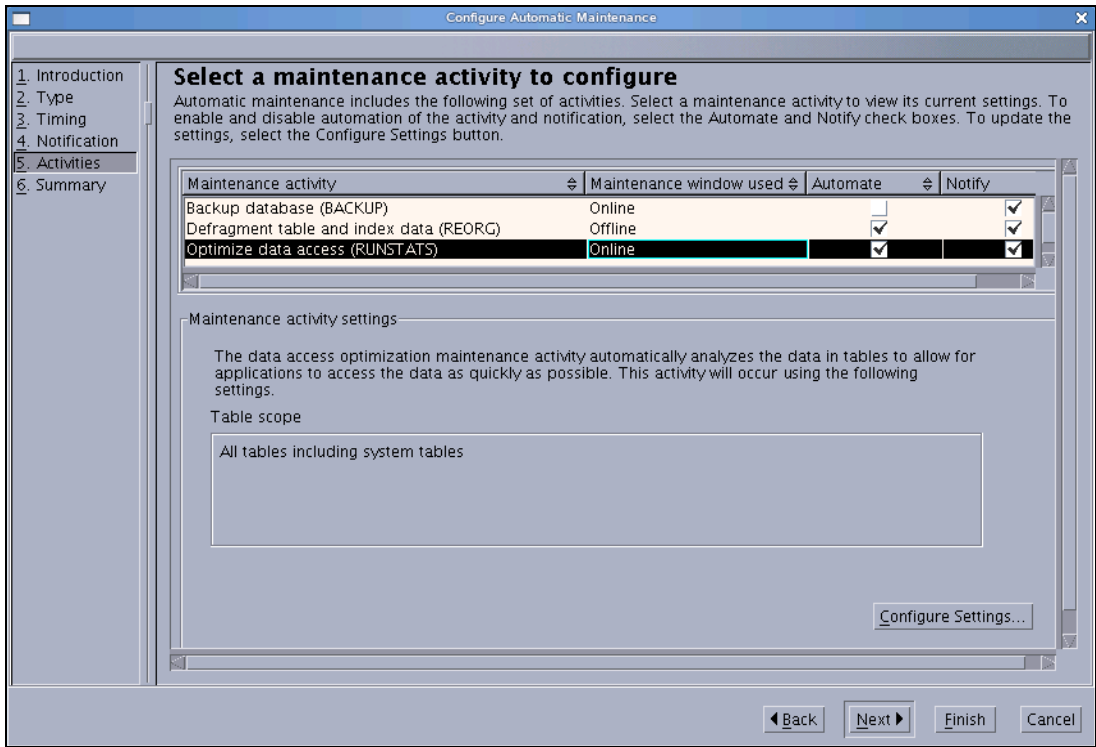


Figure 6-19 Select maintenance activity

Use the *Configure Settings* button to specify the tables from which you want to automatically collect statistics or the tables that you want to automatically reorganize.

Note: Using the Configure Automatic Maintenance wizard you can specify your maintenance objectives. The database manager uses these objectives to determine whether the maintenance activities need to be done and runs only the required ones during the next available maintenance window (a time period that you define).

Automatic statistics collection

Automatic statistics collection helps improve database performance by ensuring that you have up-to-date table statistics. The database manager determines which statistics are required by your workload and which statistics need to be updated. The DB2 optimizer uses these statistics to determine which path to use to access the data.

Statistics can be collected either asynchronously (in the background) or synchronously by gathering runtime statistics when SQL statements are compiled. By default, automatic statistics collection is set to ON. Real time statistics gathering can be enabled only when automatic statistics collection is enabled. Real time statistics gathering is controlled by the AUTO_STMT_STATS configuration parameter. If this configuration parameter is set to ON, table statistics are automatically compiled at statement compilation time, whenever they are needed to optimize a query.

Optional, you can enable the automatic statistics profile generation by setting the following two configuration parameters to ON:

- ▶ AUTO_STATS_PROF
- ▶ AUTO_PROF_UPD

Automatic statistics profiling advises when and how to collect table statistics by detecting outdated, missing, or incorrect statistics, and by generating statistical profiles based on query feedback.

Automatic reorganization

After many changes to table data, the table and indexes can become fragmented. Logically sequential data may be on non-sequential physical pages and so the database manager has to perform additional read operations to access data.

Among other information, the statistical information collected by RUNSTATS shows the data distribution within a table. In particular, analysis of these statistics can indicate when and what kind of reorganization is necessary. Automatic reorganization determines the need for reorganization on tables and indexes by using the REORGCHK formulas. It periodically evaluates tables and indexes that have had their statistics updated to see if reorganization is required. If so, it internally schedules an index reorganization or a classic table reorganization for the table. This requires that your applications function without write access to the tables being reorganized.

In a partitioned database environment, the determination to carry out automatic reorganization and the initiation of automatic reorganization, is done on the catalog partition. The reorganization runs on all of the database partitions on which the target tables reside.

6.2.2 Manual table maintenance

If you are not using the automatic table maintenance feature, you need to be sure that your table statistics are up-to-date and the tables are reorganized from time to time so that the performance of your applications can be kept optimal.

DB2 provides the following utilities to do this job manually:

- ▶ REORG
- ▶ RUNSTATS
- ▶ REBIND

Table reorganization

The REORG TABLE option reorganizes a table to match their index and to reclaim space. You can reorganize the system catalog tables as well as user tables. DB2 provides two methods for reorganizing tables:

▶ Inplace (online) reorganization

Inplace table reorganization allows applications to access the table during the reorganization. In addition, online table reorganization can be paused and resumed later by anyone with the appropriate authority by using the schema and table name. A simple command for an online reorganization is:

```
db2 reorg table db2inst1.item inplace
```

You can also use the ADMIN_CMD procedure to do the reorg:

```
db2 "call sysproc.admin_cmd ('reorg table item inplace')"
```

Table db2inst1.item will be reorganized online. The INPLACE option of the REORG command specifies an online reorganization. If this is not specified, an offline REORG is run.

▶ Offline (classic) reorganization

The offline method provides the fastest table reorganization, especially if you do not need to reorganize LOB or LONG data. LOBS and LONG data are not reorganized unless specifically requested. In addition, indexes are rebuilt in order after the table is reorganized. Read-only applications can access the original copy of the table except during the last phases of the reorganization, in which the shadow copy replaces the original copy and the indexes are rebuilt. A simple command for an offline reorganization is:

```
db2 reorg table db2inst1.item
```

You can also use the ADMIN_CMD procedure:

```
db2 "call sysproc.admin_cmd ('reorg table db2inst1.item')"
```


Note: In general, classic table reorganization is faster, but can be used only if your applications function without write access to tables during the reorganization. In addition, classic table reorganization will require the creation of a shadow copy of the table, so approximately twice as much space as the original table will be required. If your environment does not allow this restriction, although inplace reorganization is slower, it can occur in the background while normal data access continues. Consider the features of each method and decide which method is more appropriate for your environment. For both online and offline reorganization, you can reorganize all database partitions, or only one or a set of database partitions.

You can use the DB2 Control Center for table reorganization. Expand the object tree in the Control Center until you find the Tables folder. Existing tables are displayed in the contents pane on the right side of the panel. Right-click the table or view, and select **Reorganize ...**. See Figure 6-20:

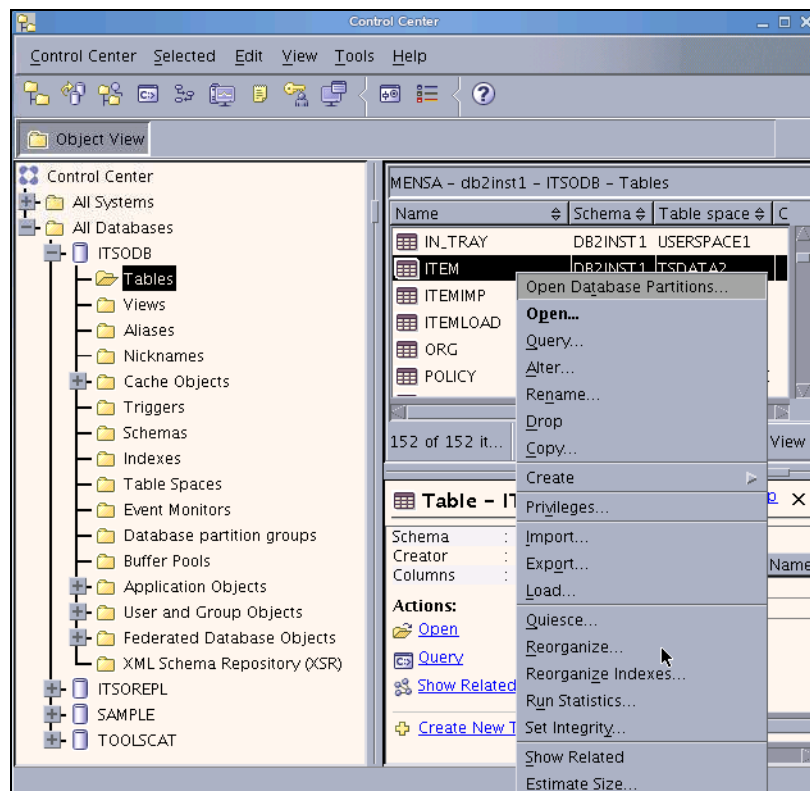


Figure 6-20 Table context menu

In the Reorganize Table panel you can select the REORG method and different options (Figure 6-21):

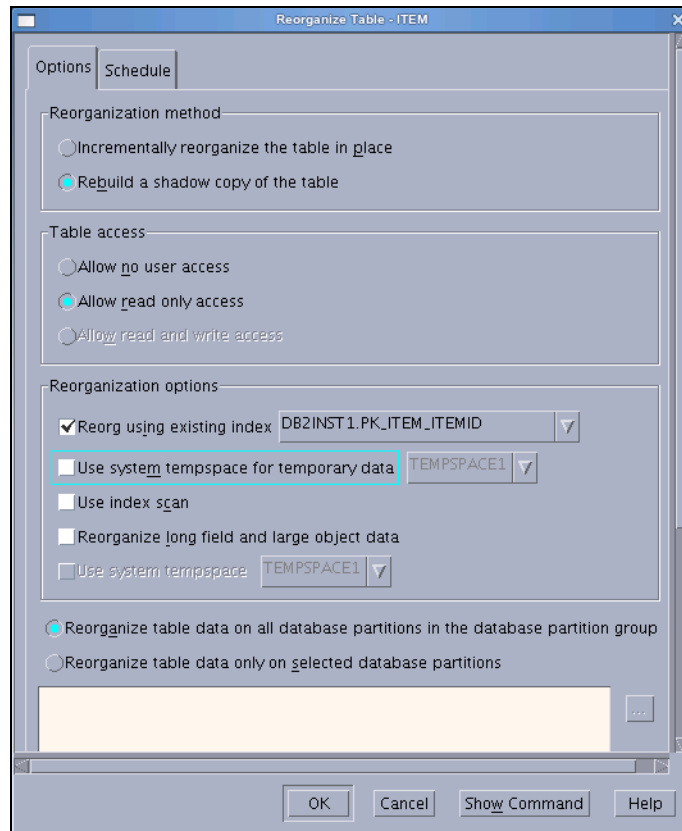


Figure 6-21 Reorganize Table panel

Note that you can also schedule the REORG as a task. Figure 6-22 shows you the generated command.

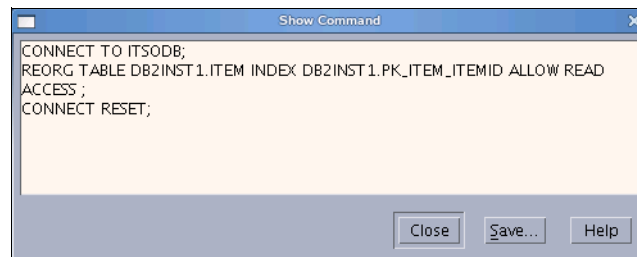


Figure 6-22 Generated REORG TABLE command

There are several methods available to reduce the need of the reorganization of table and indexes, for example, setting PCTFREE for a table to preserve a certain percentage of each data page for subsequent insert and update operations. For more information, refer to *Tuning Database Performance*, SC23-5867-00.

Analyzing the statistics produced by RUNSTATS can indicate when and what kind of reorganization is necessary. In addition, the REORGCHK command can be used to calculate statistics on the database or use current statistics information to determine if tables or indexes, or both, need to be reorganized or cleaned up.

Important: After reorganizing a table, you should collect statistics on the table so that the optimizer has the most accurate data for evaluating query access plans.

Index reorganization

To get the best performance from your indexes, consider reorganizing your indexes periodically, because table updates can cause index page prefetch to become less effective.

The REORG INDEX option reorganizes all indexes that are defined on a table by rebuilding the index data into unfragmented, physically contiguous pages. If you specify the Cleanup Only option while reorganizing an index, cleanup is performed without rebuilding the indexes.

During online index reorganization, using the REORG INDEXES command with the ALLOW WRITE ACCESS option, all indexes on the specified table are rebuilt while read and write access to the table is allowed. Any changes made to the underlying table that would affect indexes while the reorganization is in progress are logged in the database logs. The reorganization will process the logged changes while rebuilding the index.

Note: The default behavior of the REORG INDEXES command is ALLOW NO ACCESS, which places an exclusive lock on the table during the reorganization process. You can specify ALLOW READ ACCESS or ALLOW WRITE ACCESS to permit other transactions to read from or update the table.

Same as table reorganization, index reorganization is also integrated into the DB2 Control Center. See Figure 6-23 on page 280.

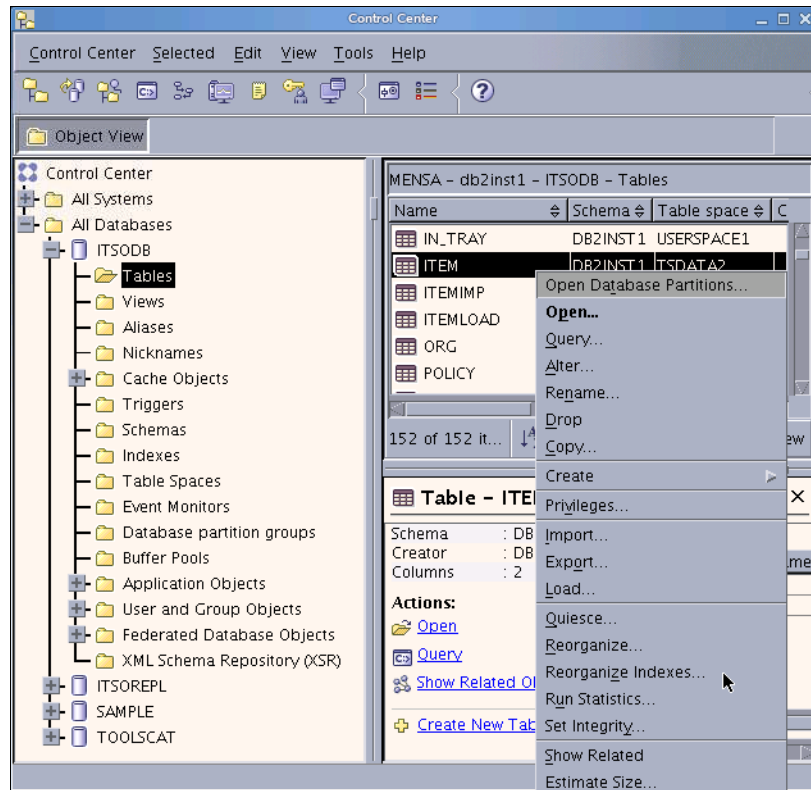


Figure 6-23 Table context menu

In the Reorganize Indexes panel you can configure the REORG method and different options (Figure 6-24 on page 281).

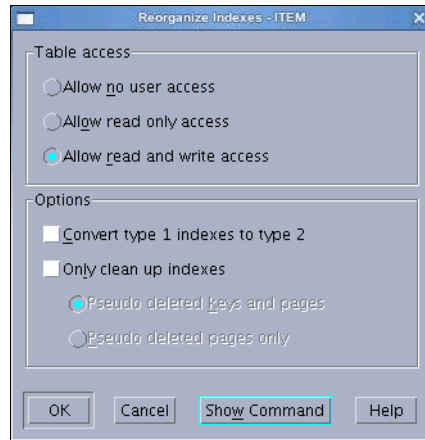


Figure 6-24 Reorganize Indexes panel

Figure 6-25 shows you the generated commands.

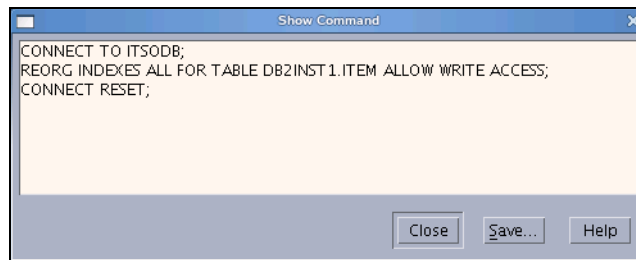


Figure 6-25 Generated REORG INDEX command

You can reduce the need of index reorganization through a variety of means, for example, setting PCTFREE for indexes, or enabling the online index defragmentation by setting MINPCTUSED. Setting MINPCTUSED will trigger online index leaf page merging when specific conditions are matched. This means it may reduce the likelihood of causing index page splits when future activities such as inserts happen on the table.

Statistics information collection

The RUNSTATS option updates the statistics about the characteristics of a table and/or associated indexes, or statistical views. These characteristics include number of records, number of pages, and average record length. The optimizer uses these statistics when determining access paths to the data.

For a table, this utility should be called when the table has had many updates, or after reorganizing the table. For a statistical view, this utility should be called

when changes to underlying tables have substantially affected the rows returned by the view. The view must have been previously enabled for use in query optimization using the ALTER VIEW command.

The RUNSTATS command can be issued from any database partition in the db2nodes.cfg file. It can be used to update the catalogs on the catalog database partition. In a partitioned database, the RUNSTATS command collects the statistics on only a single node. If the database partition from which the RUNSTATS command is executed has a part of the table, then the command will execute on that database partition directly. Otherwise, the command executes on the first database partition in the database partition group across which the table is partitioned.

It is possible to perform read or write operations to the table where RUNSTATS is taking place by using different RUNSTATS command parameters. For example, ALLOW READ ACCESS specifies that other users can have read-only access to the table while statistics are calculated and ALLOW WRITE ACCESS specifies that other users can read from and write to the table while statistics are calculated.

Note: Because reorganizing a table usually takes more time than running statistics, you might execute RUNSTATS to refresh the current statistics for your data and rebind your applications. If refreshed statistics do not improve performance, then reorganization may help.

The RUNSTATS utility is also available in the DB2 Control Center as seen in Figure 6-26 on page 283.

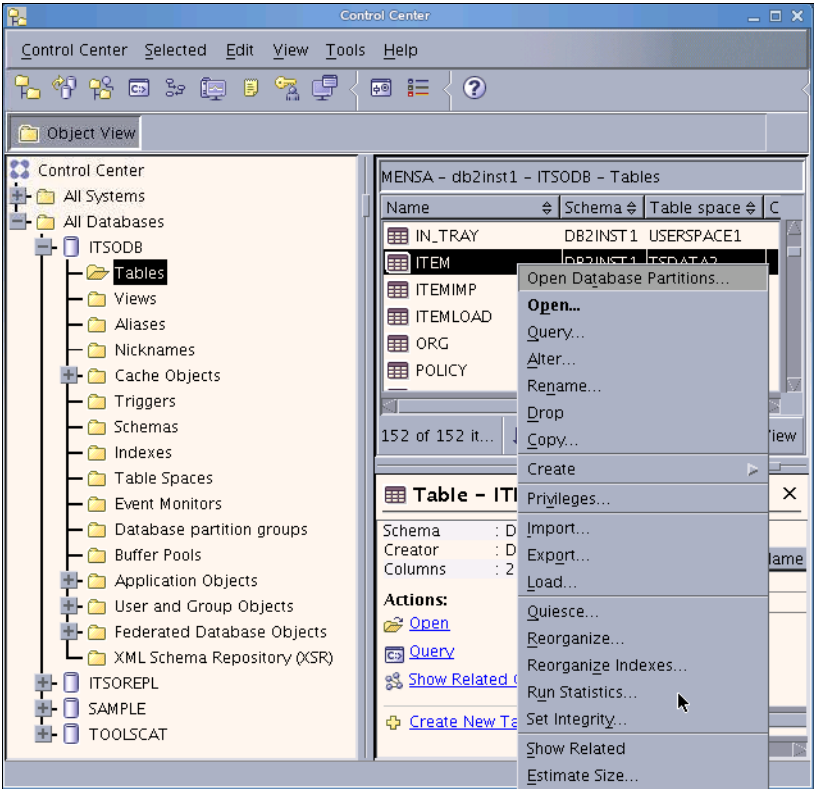


Figure 6-26 Table context menu

In the Run Statistics panel you can select different options for the RUNSTATS command (Figure 6-27 on page 284).

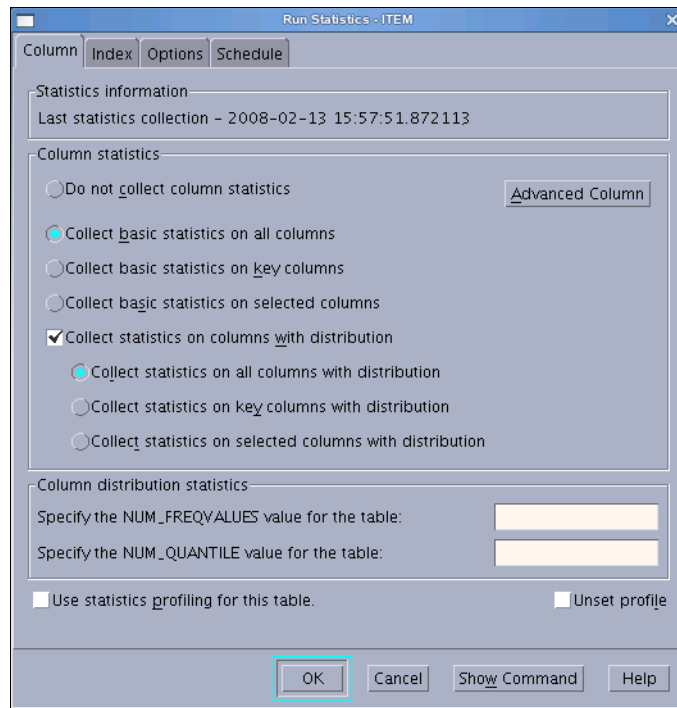


Figure 6-27 Run Statistics panel

You can schedule the RUNSTATS command as a task in the Task Center as well.

Figure 6-28 shows a sample of the generated commands.

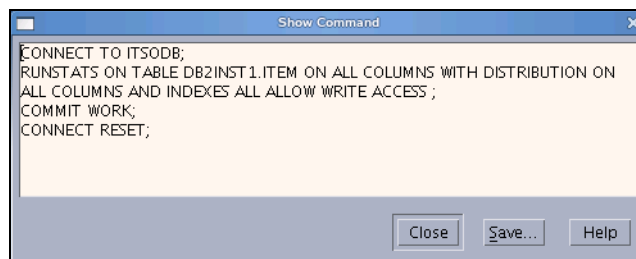


Figure 6-28 Generated RUNSTATS command

Packages rebinding

The command REBIND enables the user to take advantage of a change in the system without a need for the original bind file. It is likely that a particular SQL

statement can take advantage of a newly created index. The REBIND command can be used to recreate the package. REBIND can also be used to recreate packages after RUNSTATS is executed, thereby taking advantage of the new statistics.

Important: After a RUNSTATS, you have to use the REBIND or BIND command to recreate the package stored in the database, otherwise the static SQL application will not be able to utilize the new DB2 statistics.

Additionally, you can use the db2rbind utility to rebind all invalid packages or all packages within the database.

Examples

This section provides examples demonstrating the usage of the utilities discussed in this section: REORG TABLE, REORG INDEXES, REORGCHK and RUNSTATS. We have created a table named ITEM across two database partitions and populated with 3.000.000 rows of data. An unique index named PK_ITEM_ITEMID is associated with this table.

Example 6-34 shows some details for ITEM table.

Example 6-34 More details for sample table ITEM

```
db2inst1@mensa:/> db2 describe table item
```

Column name	Data type schema	Data type name	Column Length	Scale	Nulls
ITEMID	SYSIBM	INTEGER	4	0	No
AMOUNT	SYSIBM	DECIMAL	8	2	Yes

2 record(s) selected.

```
db2inst1@mensa:/> db2 describe indexes for table item
```

Index schema	Index name	Unique rule	Number of columns
DB2INST1	PK_ITEM_ITEMID	P	1

1 record(s) selected.

```
db2inst1@mensa:/> db2 "select dbpartitionnum(itemid) as part_no, count(*) as  
Counter from item group by dbpartitionnum(itemid)"
```

PART_NO	COUNTER
---------	---------

0	1500364
1	1499636

2 record(s) selected.

The steps performed in this example are as follows:

1. Run RUNSTATS to gather current statistics information.
2. Delete half of the total rows within the ITEM table.
3. Use REORGCHK with UPDATE STATISTICS option to update statistics and check the table again.
4. REORG TABLE and REORG INDEXES with different options.
5. Use REORGCHK with UPDATE STATISTICS option to update statistics and check the table again.
6. Re-binding packages if required.

Here are the details of each step:

1. Use “RUNSTATS” to gather current statistics information for sample table ITEM, see Example 6-35.

Example 6-35 Using RUNSTATS and REORGCHK to gather and check statistics

```
db2inst1@mena:/> db2 "runstats on table db2inst1.item with distribution on all columns and
detailed indexes all allow read access"
DB20000I The RUNSTATS command completed successfully.
db2inst1@mena:/> db2 "reorgchk current statistics on table db2inst1.item"
```

Table statistics:

```
F1: 100 * OVERFLOW / CARD < 5
F2: 100 * (Effective Space Utilization of Data Pages) > 70
F3: 100 * (Required Pages / Total Pages) > 80
```

SCHEMA.NAME	CARD	OV	NP	FP	ACTBLK	TSIZE	F1	F2	F3	REORG

Table: DB2INST1.ITEM	3000728	0	7394	7394	-	60014560	0	99	100	---

Index statistics:

```
F4: CLUSTERRATIO or normalized CLUSTERFACTOR > 80
F5: 100 * (Space used on leaf pages / Space available on non-empty leaf pages) > MIN(50, (100 -
PCTFREE))
```

F6: (100 - PCTFREE) * (Amount of space available in an index with one less level / Amount of space required for all keys) < 100
F7: 100 * (Number of pseudo-deleted RIDs / Total number of RIDs) < 20
F8: 100 * (Number of pseudo-empty leaf pages / Total number of leaf pages) < 20

SCHEMA.NAME	INDCARD	LEAF	ELEAF	LVLS	NDEL	KEYS	LEAF_RECSIZE	NLEAF_RECSIZE
LEAF_PAGE_OVERHEAD	NLEAF_PAGE_OVERHEAD	F4	F5	F6	F7	F8	REORG	

Table: DB2INST1.ITEM								
Index: DB2INST1.PK_ITEM_ITEMID								
	3000728	6176	0	3	0	3000728	4	4
1340	1340	100	92	6	0	0	-----	

CLUSTERRATIO or normalized CLUSTERFACTOR (F4) will indicate REORG is necessary for indexes that are not in the same sequence as the base table. When multiple indexes are defined on a table, one or more indexes may be flagged as needing REORG. Specify the most important index for REORG sequencing.

Tables defined using the ORGANIZE BY clause and the corresponding dimension indexes have a '*' suffix to their names. The cardinality of a dimension index is equal to the Active blocks statistic of the table.

The parameters in the RUNSTATS command can be modified based on your situation. Here we use “ALLOW READ ACCESS” to allow read-only access to the table from other users while statistics are calculated. We use REORGCHK CURRENT STATISTICS to show the statistics information for the table ITEM. You can also use the REORGCHK_TB_STATS and REORGCHK_IX_STATS procedures, which retrieve table and index statistics for reorganization evaluation directly from the system catalog views, for example, SYSSTAT.TABLES and SYSSTAT.INDEXES.

Attention: The RUNSTATS utility is run against one database partition only due to performance considerations, so some global statistics information may be inaccurate here. For example, the CARD should be 3,000,000 in reality, but here it says 3,000,728 (number of records on database partition 0 multiple by the number of database partitions). Another command INSPECT CHECK can be used to generate accurate information about the pages occupied by the table and index objects.

2. Delete rows from the table ITEM (Example 6-36).

Example 6-36 delete rows from table

```
db2inst1@mena:/> db2 "select count(*) from item"

1
-----
      3000000

      1 record(s) selected.

db2inst1@mena:/> db2 -tvf /tmp/itemdel.sql
delete from item where itemid > 250000 and itemid <= 500000
DB20000I  The SQL command completed successfully.

delete from item where itemid > 750000 and itemid <= 1000000
DB20000I  The SQL command completed successfully.

delete from item where itemid > 1250000 and itemid <= 1500000
DB20000I  The SQL command completed successfully.

delete from item where itemid > 1750000 and itemid <= 2000000
DB20000I  The SQL command completed successfully.

delete from item where itemid > 2250000 and itemid <= 2500000
DB20000I  The SQL command completed successfully.

delete from item where itemid > 2750000 and itemid <= 3000000
DB20000I  The SQL command completed successfully.

db2inst1@mena:/> db2 "select count(*) from item"

1
-----
      1500000

      1 record(s) selected.
```

3. Use REORGCHK with UPDATE STATISTICS option to obtain the new statistics and check the table again (Example 6-37).

Example 6-37 Using REORGCHK to update statistics for table

```
db2inst1@mena:/> db2 reorgchk update statistics on table db2inst1.item

Doing RUNSTATS ....

Table statistics:

F1: 100 * OVERFLOW / CARD < 5
```

F2: 100 * (Effective Space Utilization of Data Pages) > 70
F3: 100 * (Required Pages / Total Pages) > 80

SCHEMA.NAME	CARD	OV	NP	FP	ACTBLK	TSIZE	F1	F2	F3	REORG

Table: DB2INST1.ITEM										
	1502010	0	3714	7394	-	30040200	0	50	50	-**

Index statistics:

F4: CLUSTERRATIO or normalized CLUSTERFACTOR > 80
F5: 100 * (Space used on leaf pages / Space available on non-empty leaf pages) > MIN(50, (100 - PCTFREE))
F6: (100 - PCTFREE) * (Amount of space available in an index with one less level / Amount of space required for all keys) < 100
F7: 100 * (Number of pseudo-deleted RIDs / Total number of RIDs) < 20
F8: 100 * (Number of pseudo-empty leaf pages / Total number of leaf pages) < 20

SCHEMA.NAME	INDCARD	LEAF	ELEAF	LVLS	NDEL	KEYS	LEAF_RECSize	NLEAF_RECSize
LEAF_PAGE_OVERHEAD	NLEAF_PAGE_OVERHEAD	F4	F5	F6	F7	F8	REORG	

Table: DB2INST1.ITEM								
Index: DB2INST1.PK_ITEM_ITEMID								
	1502010	3106	2	3	2912	1502010		4
1340		1340	100	91	12	0	0	-----

...								

From the output we can see that DB2 has recommended that we reorganize the table (based on the output “-**” underneath “REORG” in table statistics part. The “**” means REORG is recommended). For more information regarding the explanation of the output from REORGCHK, refer to *Command Reference*, SC23-5846-00.

4. Reorganize the index and table and monitor the procedure of reorganization using GET SNAPSHOT command.

In the “Index statistics” part of the output shown in Example 6-37 on page 288, we can see that both ELEAF and NDEL are greater than 0.

Tip: The meaning of ELEAF and NDEL:

► **ELEAF:** Number of pseudo empty index leaf pages (NUM_EMPTY_LEAFS)
A pseudo empty index leaf page is a page on which all the RIDs are marked as deleted, but have not been physically removed.

► **NDEL:** Number of pseudo deleted RIDs (NUMRIDS_DELETED)
A pseudo deleted RID is a RID that is marked deleted. This statistic reports pseudo deleted RIDs on leaf pages that are not pseudo empty. It does not include RIDs marked as deleted on leaf pages where all the RIDs are marked deleted.

We can use the commands shown in Example 6-38 to physically remove the leaf pages occupied by pseudo delete.

Example 6-38 REORG INDEXES and REORGCHK to update statistics

```
db2inst1@mena:/> db2 "reorg indexes all for table item allow write access
cleanup only all on all dbpartitionnums"
DB20000I The REORG command completed successfully.
db2inst1@mena:/> db2 reorgchk update statistics on table db2inst1.item

Doing RUNSTATS ....

Table statistics:
...

Index statistics:
...

SCHEMA.NAME      INDCARD  LEAF  ELEAF  LVLS  NDEL      KEYS  LEAF_RECSIZE  NLEAF_RECSIZE
-----
Table: DB2INST1.ITEM
Index: DB2INST1.PK_ITEM_ITEMID
              1502010  3096      0    3      0 1502010              4              4

...
```

ELEAF and NDEL goes back to 0 as the pseudo deleted pages are physically removed.

Now we reorganize the table as recommended. We demonstrate both classic and inplace table reorganization. In Example 6-39 on page 291 we use classic table reorganization method with ALLOW READ ACCESS option.

Example 6-39 REORG TABLE with ALLOW READ ACCESS option

```
db2inst1@mensa: /> db2 "reorg table item index PK_ITEM_ITEMID allow read
access on all dbpartitionnums"
DB20000I The REORG command completed successfully.
```

For the inplace table reorganization, we use ALLOW WRITE ACCESS option. If you want to see more informations about a running REORG you can monitor the REORG process using different methods:

- GET SNAPSHOT command
- db2pd tool
- SNAPTAB_REORG administrative view and SNAP_GET_TAB_REORG table function

The monitoring can be done manually or with a shell script. To monitor the progress of the REORG job, we run the monitor commands to collect the information in another terminal session while the REORG job is running. The time needed to collect the REORG information will depend on the data volume you are operating against and your system's capability.

Example 6-40 shows the simple shell script we use.

Example 6-40 Start a script to monitor table reorganization procedure

```
db2inst1@mensa: > while true
> do
> db2 get snapshot for tables on itsodb >> snap.out; db2pd -reorgs -d
itsodb >> snap.out; sleep 1
> done
```

Following the startup of the monitor program, we start inplace table reorganization by using the command shown in Example 6-41.

Example 6-41 REORG TABLE with INPLACE and ALLOW WRITE ACCESS mode

```
db2inst1@mensa: /> db2 "reorg table item index PK_ITEM_ITEMID inplace allow
write access on all dbpartitionnums"
DB20000I The REORG command completed successfully.
DB21024I This command is asynchronous and may not be effective
immediately.
```

From the output you can find that the inplace table reorganization is asynchronous and may not be effective immediately. You have a few options to determine if the inplace table reorganization is finished or what the current status is. While the REORG and our monitor script were running, we checked the current status of the REORG command using the SNAPTAB_REORG administrative view as seen in Example 6-42 on page 292.

Example 6-42 SNAPTAB_REORG administrative view

```
db2inst1@mena: /> db2 "select substr(tabname, 1, 10) as tab_name, substr(reorg_type, 1, 30) as
reorg_type, reorg_status, reorg_completion, dbpartitionnum from sysibmadm.snaptab_reorg"
```

TAB_NAME	REORG_TYPE	REORG_STATUS	REORG_COMPLETION	DBPARTITIONNUM
ITEM	RECLUSTER+ONLINE+ALLOW_WRITE	STARTED	SUCCESS	0
ITEM	RECLUSTER+ONLINE+ALLOW_WRITE	STARTED	SUCCESS	1

2 record(s) selected.

...

```
db2inst1@mena: /> db2 "select substr(tabname, 1, 10) as tab_name, substr(reorg_type, 1, 30) as
reorg_type, reorg_status, reorg_completion, dbpartitionnum from sysibmadm.snaptab_reorg"
```

TAB_NAME	REORG_TYPE	REORG_STATUS	REORG_COMPLETION	DBPARTITIONNUM
ITEM	RECLUSTER+ONLINE+ALLOW_WRITE	STARTED	SUCCESS	0
ITEM	RECLUSTER+ONLINE+ALLOW_WRITE	COMPLETED	SUCCESS	1

2 record(s) selected.

...

```
db2inst1@mena: /> db2 "select substr(tabname, 1, 10) as tab_name, substr(reorg_type, 1, 30) as
reorg_type, reorg_status, reorg_completion, dbpartitionnum from sysibmadm.snaptab_reorg"
```

TAB_NAME	REORG_TYPE	REORG_STATUS	REORG_COMPLETION	DBPARTITIONNUM
ITEM	RECLUSTER+ONLINE+ALLOW_WRITE	COMPLETED	SUCCESS	0
ITEM	RECLUSTER+ONLINE+ALLOW_WRITE	COMPLETED	SUCCESS	1

2 record(s) selected.

After both database partitions completed, we stop our monitoring script. Example 6-43 is the output collected by the monitor program which was started prior to REORG TABLE. The output contains table reorganization information, such as Reorg Type, Start Time, and Status (Started, Truncate, Completed) and so forth. These informations can be useful to detect performance problems or to see, if a REORG is still in progress on a specific database partition and how long it is running.

Example 6-43 The snapshot and db2pd information for table reorganization

Table Snapshot

First database connect timestamp = 02/13/2008 09:32:59.904159


```

Last reset timestamp      =
Snapshot timestamp       = 02/13/2008 14:35:37.023048
Database name            = ITSODB
Database path            = /database/db2inst1/NODE0000/SQL00002/
Input database alias     = ITSODB
Number of accessed tables = 21

```

Table List

```

Table Schema      = DB2INST1
Table Name        = ITEM
Table Type        = User
Data Object Pages = 3697
Index Object Pages = 3101
Rows Read         = Not Collected
Rows Written      = 8251179
Overflows         = 0
Page Reorgs       = 8234
Table Reorg Information:
  Node number     = 0
  Reorg Type      =
    Reclustering
    Inplace Table Reorg
    Allow Write Access
  Reorg Index     = 1
  Reorg Tablespace = 7
  Start Time      = 02/13/2008 14:35:11.064496
  Reorg Phase     =
  Max Phase       =
  Phase Start Time =
  Status         = Started
  Current Counter = 543
  Max Counter     = 3696
  Completion      = 0
  End Time        =

```

Database Partition 0 -- Database ITSODB -- Active -- Up 0 days 05:02:38

Table Reorg Information:

Address	TbspaceID	TableID	PartID	MasterTbs	MasterTab	TableName	Type
IndexID	TempSpaceID						
0x00002B6A7E8F1B28	7	4	n/a	n/a	n/a	ITEM	Online
1	7						

Table Reorg Stats:

Address	TableName	Start	End	PhaseStart
MaxPhase	Phase	CurCount	MaxCount	Status Completion

0x00002B6A7E8F1B28	ITEM	02/13/2008 14:35:11	n/a	n/a
n/a	n/a	550	3696	Started 0

...

Table Snapshot

First database connect timestamp	= 02/13/2008 09:32:59.904159
Last reset timestamp	=
Snapshot timestamp	= 02/13/2008 14:36:33.595307
Database name	= ITSODB
Database path	= /database/db2inst1/NODE0000/SQL00002/
Input database alias	= ITSODB
Number of accessed tables	= 21

Table List

Table Schema	= DB2INST1
Table Name	= ITEM
Table Type	= User
Data Object Pages	= 3697
Index Object Pages	= 3101
Rows Read	= Not Collected
Rows Written	= 8251179
Overflows	= 0
Page Reorgs	= 8921
Table Reorg Information:	
Node number	= 0
Reorg Type	=
Reclustering	
Inplace Table Reorg	
Allow Write Access	
Reorg Index	= 1
Reorg Tablespace	= 7
Start Time	= 02/13/2008 14:35:11.064496
Reorg Phase	=
Max Phase	=
Phase Start Time	=
Status	= Truncate
Current Counter	= 1850
Max Counter	= 3696
Completion	= 0
End Time	=

Database Partition 0 -- Database ITSODB -- Active -- Up 0 days 05:03:34

Table Reorg Information:

Address	TbSpaceID	TableID	PartID	MasterTbs	MasterTab	TableName	Type
IndexID	TempSpaceID						
0x00002B6A7E8F1B28	7	4	n/a	n/a	n/a	ITEM	Online
1	7						

Table Reorg Stats:

Address	TableName	Start	End	PhaseStart
MaxPhase	Phase	CurCount	MaxCount	Status
Completion				
0x00002B6A7E8F1B28	ITEM	02/13/2008 14:35:11	n/a	n/a
n/a	n/a	1850	3696	Truncat 0

Table Snapshot

First database connect timestamp = 02/13/2008 09:32:59.904159
 Last reset timestamp =
 Snapshot timestamp = 02/13/2008 14:36:34.887245
 Database name = ITSODB
 Database path = /database/db2inst1/NODE0000/SQL00002/
 Input database alias = ITSODB
 Number of accessed tables = 21

Table List

Table Schema = DB2INST1
 Table Name = ITEM
 Table Type = User
 Data Object Pages = 1851
 Index Object Pages = 3101
 Rows Read = Not Collected
 Rows Written = 8251179
 Overflows = 0
 Page Reorgs = 8921
 Table Reorg Information:
 Node number = 0
 Reorg Type =
 Reclustering
 Inplace Table Reorg
 Allow Write Access
 Reorg Index = 1
 Reorg Tablespace = 7
 Start Time = 02/13/2008 14:35:11.064496
 Reorg Phase =
 Max Phase =
 Phase Start Time =
Status = Completed
Current Counter = 1850

Max Counter = 1850
Completion = 0
End Time = 02/13/2008 14:36:34.691349

Database Partition 0 -- Database ITS0DB -- Active -- Up 0 days 05:03:36

Table Reorg Information:

Address	TbspaceID	TableID	PartID	MasterTbs	MasterTab	TableName	Type
IndexID	TempSpaceID						
0x00002B6A7E8F1B28	7	4	n/a	n/a	n/a	ITEM	Online
1	7						

Table Reorg Stats:

Address	TableName	Start	End	PhaseStart
MaxPhase	Phase	CurCount	MaxCount	Status
Completion				
0x00002B6A7E8F1B28	ITEM	02/13/2008 14:35:11	02/13/2008 14:36:34	n/a
n/a	n/a	1850	1850	Done 0

5. Use the REORGCHK with UPDATE STATISTICS option to obtain the new statistics after the index and table reorganization (Example 6-44).

Example 6-44 Using REORGCHK to update statistics after table reorganization

db2inst1@mena: /> db2 reorgchk update statistics on table db2inst1.item

Doing RUNSTATS

Table statistics:

F1: 100 * OVERFLOW / CARD < 5
F2: 100 * (Effective Space Utilization of Data Pages) > 70
F3: 100 * (Required Pages / Total Pages) > 80

SCHEMA.NAME	CARD	OV	NP	FP	ACTBLK	TSIZE	F1	F2	F3	REORG
Table: DB2INST1.ITEM	1502010	0	3702	3702	-	30040200	0	100	100	---

Index statistics:

F4: CLUSTERRATIO or normalized CLUSTERFACTOR > 80
F5: 100 * (Space used on leaf pages / Space available on non-empty leaf pages) > MIN(50, (100 - PCTFREE))
F6: (100 - PCTFREE) * (Amount of space available in an index with one less level / Amount of space required for all keys) < 100

F7: 100 * (Number of pseudo-deleted RIDs / Total number of RIDs) < 20
F8: 100 * (Number of pseudo-empty leaf pages / Total number of leaf pages) < 20

SCHEMA.NAME	INDCARD	LEAF	ELEAF	LVLS	NDEL	KEYS	LEAF_RECSize	NLEAF_RECSize
LEAF_PAGE_OVERHEAD	NLEAF_PAGE_OVERHEAD	F4	F5	F6	F7	F8	REORG	

Table: DB2INST1.ITEM								
Index: DB2INST1.PK_ITEM_ITEMID								
	1502010	3128	0	3	0	1502010	4	4
1340	1340	100	91	12	0	0	----	

...								

Here you may find that FP (File Pages) for table and LEAF (Leaf Pages) for index are lowered, if the data were fragmented. For table statistics, you may find that “----” is reported. It means no REORG is required at present.

6. Rebinding packages.

Rebinding is required to take advantage of the new statistics information. For example, if statistics information for the tables or indexes related to the package has been changed, then, in general, rebinding is required for better performance. An example of using the rebind command is shown in Example 6-45.

Example 6-45 Re-binding packages

```
db2inst1@mensa: /> db2 rebind MYTEST
DB20000I The REBIND PACKAGE command completed successfully.
```

Attention: Be cautious when using the db2rbind utility, especially when the ALL parameter is used. By default, db2rbind will rebind invalid packages only. But when ALL is specified, all the packages inside the database, both valid and invalid, will be rebound. If you know exactly what packages are related to the tables or indexes, and the statistics information for those tables or indexes have just been changed, then using the **REBIND** with a specific package name is recommended.

6.3 Moving data using EXPORT, IMPORT, LOAD, and db2move

In this section, we discuss data movement using DB2 EXPORT, IMPORT and LOAD commands. EXPORT allows you to export data from database tables into files with different file formats. IMPORT writes data into database tables from a source data file. LOAD provides the capability to fast move large volume of data

into the database tables. We describe the basic usage of these utilities as well as the considerations when using the load utility under a multiple database partition environment. If you are dealing with a lot of tables, the db2move utility can be helpful for such a situation.

6.3.1 Export data to files from database tables and views

The DB2 export utility can be used to write data from a DB2 database to one or more files stored outside of the database. The exported data can then be imported or loaded into another DB2 database using the DB2 import or the DB2 load utility, respectively; or it can be imported into another application, for example, a spreadsheet. The user specifies the data to be exported by supplying a SELECT statement or by providing hierarchical information for typed tables.

Before invoking EXPORT, you need to establish a connection to the database, explicitly or implicitly. You can invoke the export utility through the DB2 Command Line Processor (CLP), the DB2 Control Center, by calling the ADMIN_CMD stored procedure, or through an Application Programming Interface (API).

Export table data through CLP

Example 6-46 is a sample to export data from a table named ITEM which is spread across two database partitions by using the DB2 CLP.

Example 6-46 Export all data from table ITEM

```
db2inst1@mena:/db2backup/datamove> db2 'export to item.ixf of ixf select *
from item'
SQL3104N The Export utility is beginning to export data to file "item.ixf".

SQL3105N The Export utility has finished exporting "3000000" rows.
```

Number of rows exported: 3000000

The data across all the two database partitions has been exported. If you want to export data for only one database partition, for example, data residing on current database partition, you can do it as in Example 6-47.

Example 6-47 Export data for current database partition from table ITEM

```
db2inst1@mena:/db2backup/datamove> db2 'export to itempart0.del of del
messages exppart0.msg select * from item where dbpartitionnum(itemid) = current
dbpartitionnum'
```

Number of rows exported: 1500364

We specified a messages file called exppart0.msg. The export utility writes error, warning, and informational messages into this file. For all interfaces except the CLP, you must specify the name of the messages file in advance with the **MESSAGES** parameter.

If you want to export data in parallel across different database partitions, you can use the **db2_a11** tool.

Export table data using the Control Center

You can also use the DB2 Control Center to invoke EXPORT on a Linux platform. Expand the object tree in the Control Center until you find the Tables or Views folder. Any existing tables or views are displayed in the contents pane on the right side of the panel. Right-click the table or view, a menu listed all the activities for the table shows. See Figure 6-29.

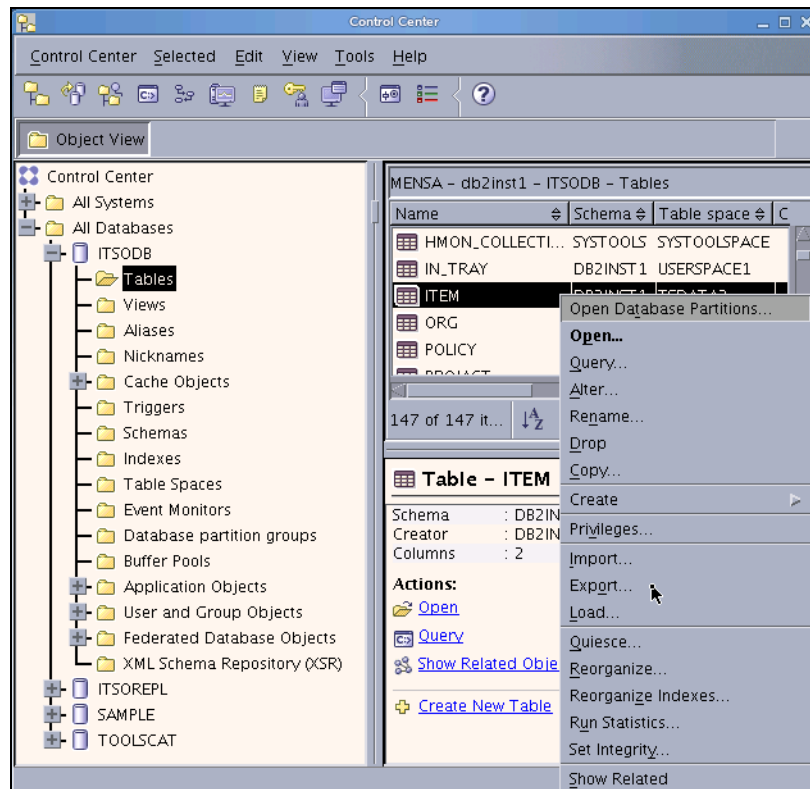


Figure 6-29 Table context menu

You can choose EXPORT to open the export table notebook. See Figure 6-30 on page 300. You can choose the output file format of export under the File Format

options; input SQL statement manually or by SQL Assist Wizard under the SELECT statement area. In the Messages file field, you can specify the messages file to be created for the error, warning, and informational messages associated with the EXPORT operation. In addition, if you want to know the command DB2 generated for this task, click **Show Command**. A pop-up panel appears with the command generated shown. Furthermore, you may use the Schedule function provided by DB2 to schedule a time to run this export task by choosing the Schedule tab on the panel.

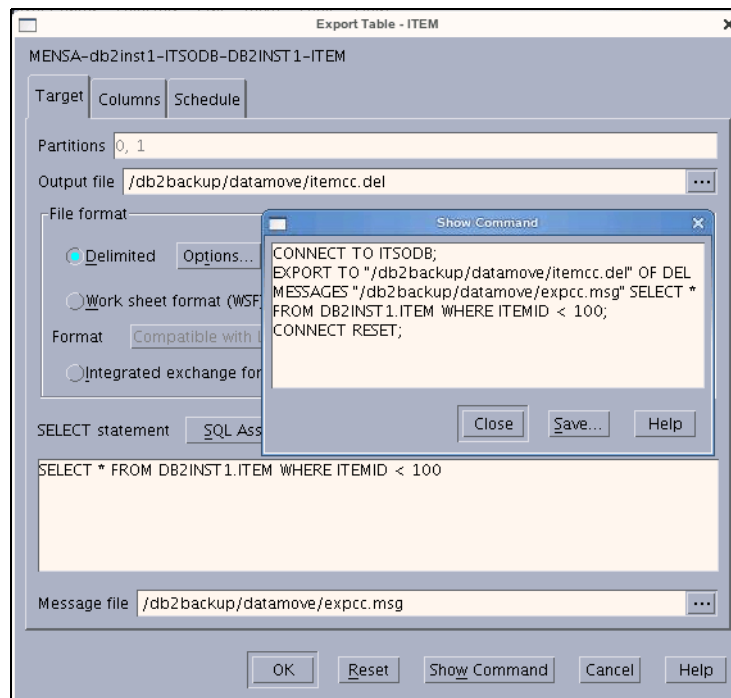


Figure 6-30 Using EXPORT in DB2 Control Center

The export operation will fail if the data you want to export exceeds the space available on the file system on which the exported file will be created. To avoid this, you can limit the amount of data selected by specifying conditions on the WHERE clause and invoke the export utility multiple times to export all of the data.

For information about using the ADMIN_CMD stored procedure, or the EXPORT API, and more details regarding the EXPORT utility, refer to *Data Movement Utilities Guide and Reference*, SC23-5847-00.

6.3.2 Import data from files into database tables or views

The import utility inserts data from an input file into a table or updatable view. If the table or view has already have data, you can either replace or append to the existing data.

You can also use import to create a new table, if the format of input file is Integrated eXchange Format (IXF), using the IMPORT command options CREATE and REPLACE_CREATE. Not all of the properties of the table are re-created when use CREATE and REPLACE_CREATE. Note that these options are deprecated and might be removed in a future release.

Note: Instead of using the IMPORT options to create table, use the db2look command to capture the original table definitions and re-create the table. Once the table is created, issue the LOAD or IMPORT command to add the data to the table. The db2look command preserves all of the properties of a table and is a superior option for table re-creation.

The authority and privileges required for running import with and without creating a new table option is different. To use the import utility to create a new table, you must have SYSADM authority, DBADM authority, or CREATETAB privilege for the database. To replace data in an existing table or view, you must have SYSADM authority, DBADM authority, or CONTROL privilege for the table or view. To append data to an existing table or view, you must have SELECT and INSERT privileges for the table or view.

The import utility performs the following steps to import the data:

1. Locking tables
Import acquires either an exclusive (X) lock or a nonexclusive (IX) lock on existing target tables, depending on whether you allow concurrent access to the table.
2. Locating and retrieving data
Import uses the FROM clause to locate the input data.
3. Inserting data
Import either replaces existing data or adds new rows of data to the table.
4. Checking constraints and firing triggers
As the data is written, import ensures that each inserted row complies with the constraints defined on the target table. Information about rejected rows is written to the messages file. Import also fires existing triggers.
5. Committing the operation
Import saves the changes made and releases the locks on the target table. You can also specify that commit is taken place periodically during the import.

The import utility can be invoked through the command line processor (CLP), the Import notebook in the Control Center, by calling the ADMIN_CMD stored procedure, or by calling the application programming interface (API) db2Import. You can specify the MESSAGES parameter for the IMPORT utility to record errors, warnings, and informational messages associated with the IMPORT operation.

Attention: If the volume of output messages generated by an import operation against a remote database exceeds 60 KB, the utility will keep the first 30 KB and the last 30 KB.

Import table data through CLP

Example 6-48 shows the IMPORT command issued through the CLP.

Example 6-48 Using IMPORT through DB2 CLP

```
db2inst1@mensa:/db2backup/datamove> db2 'create table itemimp like item'
DB20000I The SQL command completed successfully.
db2inst1@mensa:/db2backup/datamove> db2 'import from from ./item.ixf of ixf
commitcount 100000 messages itemimp.msg insert into itemimp'
```

Number of rows read	= 3000000
Number of rows skipped	= 0
Number of rows inserted	= 3000000
Number of rows updated	= 0
Number of rows rejected	= 0
Number of rows committed	= 3000000

In Example 6-48, the COMMITCOUNT parameter is used. It means that DB2 will perform a COMMIT after every n records are imported. With the COMMITCOUNT option, if the import is interrupted for some reason, the committed rows will remain in the target table despite the failure of the import procedure. You can continue the import with the option INSERT with RESTARTCOUNT parameter to import the reset of data. Another choice is to use the REPLACE or REPLACE_CREATE option.

The messages file is a convenient way of monitoring the progress of an import as you can access it during the import. In the event of a failed import operation, message files can be used to determine a restarting point by indicating the last row that was successfully imported. Example 6-49 shows a part of the messages file during our import.

Example 6-49 Monitor IMPORT progress with the messages file

```
db2inst1@mensa:/db2backup/datamove> tail -10 itemimp.msg
SQL3222W ...COMMIT of any database changes was successful.
```

SQL3221W ...Begin COMMIT WORK. Input Record Count = "2500000".

SQL3222W ...COMMIT of any database changes was successful.

SQL3221W ...Begin COMMIT WORK. Input Record Count = "2600000".

SQL3222W ...COMMIT of any database changes was successful.

Import table data using the Control Center

Similar to EXPORT, you also can use the DB2 Control Center to invoke the import utility.

Use these steps to perform import from the Control Center:

1. From the Control Center, expand the object tree until you find the Tables folder.
2. Click the Tables folder. Any existing tables are displayed in the pane on the right side of the panel (the contents pane).
3. Right-click the table you want in the contents pane, and select **Import** from the pop-up menu. The Import notebook opens.

See Figure 6-31 is a sample for using IMPORT from Control Center.

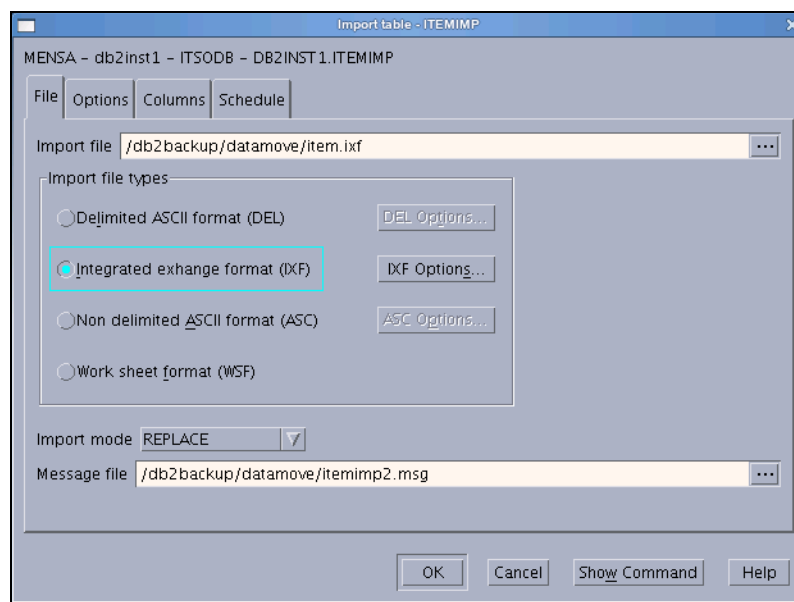


Figure 6-31 Using IMPORT within DB2 Control Center

Within the Control Center, we set the COMMITCOUNT parameter to AUTOMATIC (-1) as shown in Figure 6-32

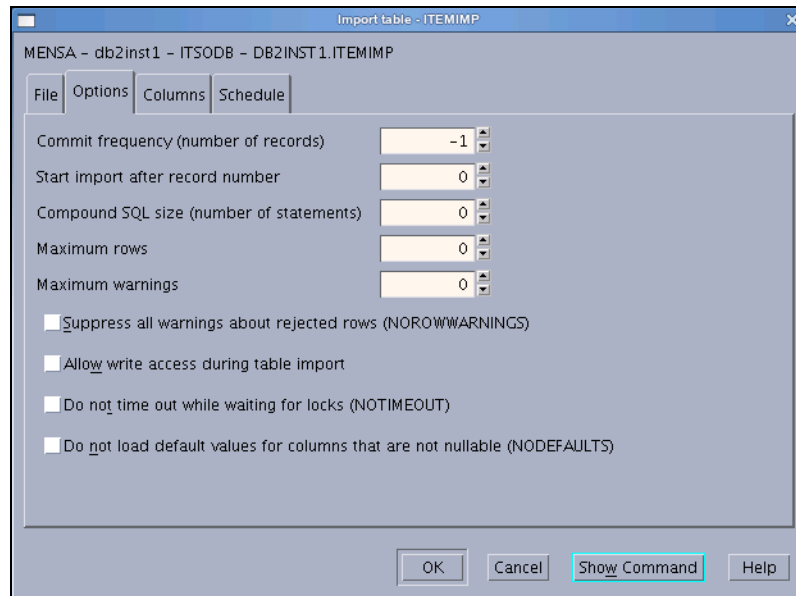


Figure 6-32 Import Options in the Control Center

When AUTOMATIC is specified, import internally determines when a commit needs to be performed. The utility will commit for either one of two reasons:

- ▶ To avoid running out of active log space
- ▶ To avoid lock escalation from row level to table level

Figure 6-33 shows the command generated by the Control Center.

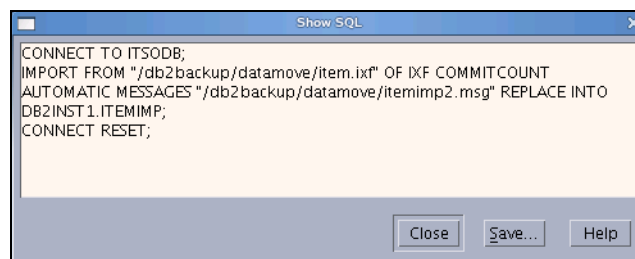


Figure 6-33 Generated IMPORT command

In a partitioned database environment, the import utility can be enabled to use buffered inserts. This reduces the messaging that occurs when data is imported, resulting in better performance. However, since details about a failed buffered

insert are not returned, this option should only be enabled if you are not concerned about error reporting.

Use the DB2 bind utility to request buffered inserts capability. The import package, db2uimpb.bnd, must be rebound against the database using the INSERT BUF option. See Example 6-50.

Example 6-50 Enable INSERT BUF for IMPORT in a partitioned database

```
db2inst1@mena:/db2home/db2inst1> db2 connect to itsodb
```

Database Connection Information

```
Database server      = DB2/LINUX8664 9.5.0
SQL authorization ID = DB2INST1
Local database alias = ITSODB
```

```
db2inst1@mena:/db2home/db2inst1> db2 bind sqllib/bnd/db2uimpb.bnd insert buf
```

```
LINE    MESSAGES FOR db2uimpb.bnd
```

```
-----
SQL0061W  The binder is in progress.
SQL0091N  Binding was ended with "0" errors and "0" warnings.
```

For more information regarding using IMPORT, for example, using IMPORT with Large Objects, XML data, or identity columns, refer to *Data Movement Utilities Guide and Reference*, SC23-5847-00 and *Command Reference*, SC23-5846-00.

6.3.3 Load data from files into database tables

The load utility provides an efficient way to move large quantity data into new or existing tables. The utility can handle most data types including XML, large objects (LOBs), and user-defined types (UDTs). In general, the load utility is faster than the import utility because it writes formatted pages directly into the database while the import utility performs SQL INSERTs. The load utility does not fire triggers, and does not perform referential or table constraints checking (other than validating the uniqueness of the indexes).

The load process consists of four distinct phases:

- ▶ *Load*: During which the data is written to the table.
- ▶ *Build*: During which indexes are produced.
- ▶ *Delete*: During which the rows that caused a unique or primary key violation are removed from the table and stored in the load exception table, if one was specified.

- *Index copy*: During which the index data is copied from a system temporary table space to the original table space. This will only occur if a system temporary table space was specified for index creation during a load operation with the READ ACCESS option specified.

To use the LOAD utility, you must have SYSADM authority, DBADM authority, or LOAD authority on the database together with the needed privilege depending on the load mode you choose, i.e. for REPLACE mode you need INSERT and DELETE privilege on the table.

The load utility can be invoked through the command line processor (CLP), the Load wizard in the Control Center, by calling the ADMIN_CMD stored procedure, or an application programming interface (API), db2Load.

Data for load can be in the form of a file, tape, or named pipe. Data can also be loaded from a cursor defined from a query running against the currently connected database or a different database under the same instance, or by using a user-written script or application.

You can use the LIST UTILITIES or LOAD QUERY command to monitor the progress of the load operation.

Load table data through CLP

Example 6-51 shows a simple LOAD command issued through the CLP:

Example 6-51 LOAD from command line

```
db2inst1@mena:/db2backup/datamove> db2 create table itemload like item in
TSDATA2
DB20000I The SQL command completed successfully.
db2inst1@mena:/db2backup/datamove> db2 'export to item.del of del select *
from item'
SQL3104N The Export utility is beginning to export data to file "item.del".

SQL3105N The Export utility has finished exporting "3000000" rows.
```

Number of rows exported: 3000000

```
db2inst1@mena:/db2backup/datamove> db2 'load from item.del of del insert into
itemload'
```

Agent Type	Node	SQL Code	Result
LOAD	000	+00000000	Success.
LOAD	001	+00000000	Success.

PARTITION	001	+00000000	Success.
PRE_PARTITION	000	+00000000	Success.
RESULTS:	2 of 2 LOADs completed successfully.		

Summary of Partitioning Agents:
Rows Read = 3000000
Rows Rejected = 0
Rows Partitioned = 3000000

Summary of LOAD Agents:
Number of rows read = 3000000
Number of rows skipped = 0
Number of rows loaded = 3000000
Number of rows rejected = 0
Number of rows deleted = 0
Number of rows committed = 3000000

db2inst1@mensa:/db2backup/datamove> db2 'select dbpartitionnum(itemid) as Partitionnumber, count(*) as Counter from db2inst1.itemload group by dbpartitionnum(itemid)'

NODENUMBER	COUNTER
-----	-----
0	1500364
1	1499636

2 record(s) selected.

As seen in Example 6-51 on page 306, the data were loaded into table `itemload` which is spread across two database partitions. To verify the load result and distribution, within the same example, we use an SQL statement to count the number of rows distributed on different database partitions, from the output of SQL execution, we find that the rows are distributed to the database partitions evenly.

Note: When the COPY NO option is specified for a recoverable database, the table space will be placed in the backup pending state when the load operation begins. This is to ensure that you can recreate your database with all the data, because the load is not logged in the database log files.

Since we set up log archiving for our database and we did not specify COPY YES with the LOAD command, the default COPY NO is used. The table space of

table itemload is now in backup pending state. To remove the table space state, we have to run a database or table space backup as shown in Example 6-52:

Example 6-52 Table space backup pending state

```
db2inst1@mensa:/db2backup/datamove> db2 list tablespaces show detail | tail -22

Tablespace ID                = 7
Name                          = TSDATA2
Type                          = Database managed space
Contents                       = All permanent data. Large table space.
State                         = 0x0020
    Detailed explanation:
        Backup pending
Total pages                    = 51200
Useable pages                  = 51168
Used pages                     = 10720
Free pages                     = 40448
High water mark (pages)       = 10720
Page size (bytes)              = 8192
Extent size (pages)           = 32
Prefetch size (pages)         = 32
Number of containers           = 1
Minimum recovery time          = 2008-02-08-23.20.58.000000
```

DB21011I In a partitioned database server environment, only the table spaces on the current node are listed.

```
db2inst1@mensa:/db2backup/datamove> db2 'backup db itsodb on all
dbpartitionnums tablespace(TSDATA2) online to /db2backup'
Part  Result
-----
0000 DB20000I The BACKUP DATABASE command completed successfully.
0001 DB20000I The BACKUP DATABASE command completed successfully.

Backup successful. The timestamp for this backup image is : 20080208155111
```

When loading data in a partitioned database environment, only non-delimited ASCII (ASC) and delimited ASCII (DEL) files can be partitioned. PC/IXF files cannot be partitioned. To load a PC/IXF file into a multiple database partitioned table, you can first load it into a single-partition table, and then perform a load operation using the CURSOR file type to move the data into a multiple database partition table (Example 6-53 on page 309). If you try to load source file in IXF format to a database partitioned table directly, it results in error SQL3004N. It means the filetype parameter is not valid, because IXF files cannot be used to load into a table spanning multiple database partitions.

Example 6-53 Load data from CURSOR into a database partitioned table

```
db2inst1@mensa:/db2backup/datamove> db2 'load from item.ixf of ixf replace into
itemload'
SQL3004N  The filetype parameter is not valid.

db2inst1@mensa:/db2backup/datamove> db2 "declare mycursor cursor for select *
from item"
db2inst1@mensa:/db2backup/datamove> db2 'load from mycursor of cursor replace
into itemload copy yes to /db2backup/datamove/copydata'
```

Agent Type	Node	SQL Code	Result
LOAD	000	+00000000	Success.
LOAD	001	+00000000	Success.
PARTITION	001	+00000000	Success.
RESULTS:	2 of 2 LOADs completed successfully.		

Summary of Partitioning Agents:
Rows Read = 3000000
Rows Rejected = 0
Rows Partitioned = 3000000

Summary of LOAD Agents:
Number of rows read = 3000000
Number of rows skipped = 0
Number of rows loaded = 3000000
Number of rows rejected = 0
Number of rows deleted = 0
Number of rows committed = 3000000

Options for loading table data into partitioned databases

In a multi-partition database, large amount of data are located across many database partitions. Distribution keys are used to determine on which database partition each portion of the data resides. The data must be distributed before it can be loaded at the correct database partition.

We have demonstrated using the LOAD command with default settings to load data into a partitioned database table. When loading data into a multi-partition database you can also use one of the following modes:

- ▶ *PARTITION_AND_LOAD*
Data is distributed and loaded to all database partitions in parallel. This is the default for the LOAD command.

- ▶ *PARTITION_ONLY*
Data is distributed and the output is written to files in a specified location on each loading database partition. These files can be loaded into the database using the `LOAD_ONLY` mode.
- ▶ *LOAD_ONLY*
The distribution process is skipped, the data is loaded simultaneously on the corresponding database partitions.
- ▶ *LOAD_ONLY_VERIFY_PART*
Like `LOAD_ONLY`, additional each row is checked to verify that it is on the correct database partition.
- ▶ *ANALYZE*
An optimal distribution map with even distribution across all database partitions is generated.

To use these different modes, there are some options that are specific for loading in a partitioned database environment, for example:

- ▶ `PART_FILE_LOCATION`: The fully qualified location of the partitioned files.
- ▶ `OUTPUT_DBPARTNUMS`: A list of partition numbers which represent the database partitions on which the load operation is to be performed.
- ▶ `PARTITIONING_DBPARTNUMS`: A list of partition numbers that will be used in the partitioning process.
- ▶ `MODE`: The mode in which the load operation will take place when loading a partitioned database, such as `PARTITION_AND_LOAD`, `PARTITION_ONLY`, `LOAD_ONLY` and so on.
- ▶ `PARTITIONED DB CONFIG`: Allows you to specify partitioned database-specific configuration options in a `LOAD` command.

Example 6-54 demonstrates how to use the `PARTITIONED DB CONFIG` parameter to specify partitioned database specific options for `LOAD`. With the `PARTITION_ONLY` option, the data are distributed to each database partition, but not loaded. The output is written to files in the specified location on each of the database partitions. The data can then be loaded at later time. We load the data from the files into the table `ITEM` using the `LOAD_ONLY` mode.

Example 6-54 Using PARTITIONED DB CONFIG parameter in LOAD command

```
db2inst1@mensa:/db2backup/datamove> db2 "load from item.del of del replace into
itemload partitioned db config mode partition_only part_file_location
/db2backup/datamove/load partitioning_dbpartnums (0,1)"
```

Agent	Type	Node	SQL Code	Result
LOAD_TO_FILE		000	+00000000	Success.

LOAD_TO_FILE	001	+00000000	Success.
PARTITION	000	+00000000	Success.
PARTITION	001	+00000000	Success.
PRE_PARTITION	000	+00000000	Success.

Summary of Partitioning Agents:

Rows Read = 3000000
Rows Rejected = 0
Rows Partitioned = 3000000

db2inst1@mena:/db2backup/datamove> db2 "select count(*) from
db2inst1.itemload"

1

0

1 record(s) selected.

db2inst1@mena:/db2backup/datamove/load> ls -la
total 27348
drwxr-xr-x 2 db2inst1 db2iadm1 4096 2008-02-11 11:32 .
drwxr-xr-x 4 db2inst1 db2iadm1 4096 2008-02-11 11:30 ..
-rw-r----- 1 db2inst1 db2iadm1 27959550 2008-02-11 11:32 item.del.000
db2inst1@mena:/db2backup/datamove> db2_all "ls -la /db2backup/datamove/load"

total 27348
drwxr-xr-x 2 db2inst1 db2iadm1 4096 2008-02-11 11:32 .
drwxr-xr-x 4 db2inst1 db2iadm1 4096 2008-02-11 11:30 ..
-rw-r----- 1 db2inst1 db2iadm1 27959550 2008-02-11 11:32 item.del.000
mena: ls -la /db2backup/datamove/load completed ok

total 27332
drwxr-xr-x 2 db2inst1 db2iadm1 4096 2008-02-11 11:36 .
drwxr-xr-x 4 db2inst1 db2iadm1 4096 2008-02-11 11:35 ..
-rw-r----- 1 db2inst1 db2iadm1 27946236 2008-02-11 11:36 item.del.001
gemini: ls -la /db2backup/datamove/load completed ok

db2inst1@mena:/db2backup/datamove> db2 "load from item.del of del replace
into itemload copy yes to /db2backup/datamove/copydata partitioned db config
mode load_only part_file_location /db2backup/datamove/load
partitioning_dbpartnums (0,1)"

Agent Type	Node	SQL Code	Result
------------	------	----------	--------

LOAD	000	+00000000	Success.
LOAD	001	+00000000	Success.
RESULTS:	2 of 2 LOADs completed successfully.		

Summary of LOAD Agents:

Number of rows read	= 3000000
Number of rows skipped	= 0
Number of rows loaded	= 3000000
Number of rows rejected	= 0
Number of rows deleted	= 0
Number of rows committed	= 3000000

db2inst1@mensa:/db2backup/datamove> db2 "select count(*) from itemload"

```
1
-----
3000000
```

1 record(s) selected.

Examples for **ALLOW READ ACCESS** when loading

The **ALLOW READ ACCESS** option is very useful when loading large amount of data because it allows users to access table data when the load operation is in progress or after a load operation has failed. The behavior of a load operation in **ALLOW READ ACCESS** mode is independent of the isolation level of the application. That is, readers with any isolation level can always read the pre-existing data, but they will not be able to read the newly loaded data until the load operation has finished.

Read access is provided throughout the load operation except at the very end. Before data is committed the load utility acquires an exclusive lock (Z-lock) on the table. The load utility will wait until all the locks held by all the applications have been released. This may cause a delay before the data can be committed. The **LOCK WITH FORCE** option may be used to force off conflicting applications, and allow the load operation to proceed without having to wait.

In Example 6-55 on page 313, we create a table named **ITEMLOAD** and insert three rows as the pre-existing data. We then use *ALLOW READ ACCESS* to load additional rows from a source file in **DEL** format. While the load is taking place, we try to access the table from another session.

Example 6-55 Load data with ALLOW READ ACCESS

```
db2inst1@mensa:/db2backup/datamove> db2 drop table itemload
DB20000I The SQL command completed successfully.
db2inst1@mensa:/db2backup/datamove> db2 "create table itemload (itemid integer
not null primary key, amount decimal (8,2) ) in TSDATA2"
DB20000I The SQL command completed successfully.
db2inst1@mensa:/db2backup/datamove> db2 "insert into itemload values
(4000001,1.1), (4000002,2.2), (4000003,3.3)"
DB20000I The SQL command completed successfully.
db2inst1@mensa:/db2backup/datamove> db2 "load from item.del of del insert into
itemload allow read access"
```

Agent Type	Node	SQL Code	Result
LOAD	000	+00000000	Success.
LOAD	001	+00000000	Success.
PARTITION	001	+00000000	Success.
PRE_PARTITION	000	+00000000	Success.
RESULTS:	2 of 2 LOADs completed successfully.		

Summary of Partitioning Agents:
Rows Read = 3000000
Rows Rejected = 0
Rows Partitioned = 3000000

Summary of LOAD Agents:
Number of rows read = 3000000
Number of rows skipped = 0
Number of rows loaded = 3000000
Number of rows rejected = 0
Number of rows deleted = 0
Number of rows committed = 3000000

When the load command is issued and load is running, we can access the same table with read-only operation from another terminal session. Before the very end of the load operation, the SELECT statement returns the answer set very quickly. But at the last stage of the load operation till the job is finished, the SELECT statement cannot succeed due to resource contention with load operation. At this time the load utility has to acquire an exclusive lock (Z-lock) on the table before data is committed. After the load operation is finished, the newly loaded data is visible to other applications. Example 6-56 on page 314 shows the SELECT output during LOAD is running and after the data is committed.

Example 6-56 Concurrent access to the table where load is taking place

-- LOAD is running, we can access the data:

```
db2inst1@mensa:/db2backup/datamove> db2 "select * from itemload" | head -10
```

ITEMID	AMOUNT
4000001	1.10
4000002	2.20
4000003	3.30

3 record(s) selected.

-- after the LOAD finished and committed the data:

```
db2inst1@mensa:/db2backup/datamove> db2 "select * from itemload" | head -10
```

ITEMID	AMOUNT
5	954.25
7	534.75
10	417.87
11	313.62
14	419.25
15	679.75
16	347.25

The LOCK WITH FORCE option may be used to force off conflicting applications with load operation to allow the load operation to proceed without waiting. The forced application will receive an error SQL1224N.

The ALLOW READ ACCESS option is not supported if the REPLACE option is specified, otherwise error code SQL3340N with reason code 1 will be returned. Since a load replace operation deletes the existing table data before loading the new data, there is no pre-existing data to query until after the load operation is complete.

For partitioned database, the message files will not be displayed to the console or retained for the load operations initiated from the CLP. To save or view the contents of these files after a partitioned database load has completed, the MESSAGES option of the LOAD command must be specified. A fully qualified file name to the MESSAGES file is recommended, for example, /db2backup/datamove/myload.msg.

If the MESSAGES option is used, once the load operation has completed the message files on each database partition will be transferred to the client machine and stored in files with the base name indicated by the MESSAGES option. For

partitioned database load operations, the name of the file corresponding to the load process that produced it is listed in Table 6-1.

Table 6-1 Message file types generated by partitioned LOAD operation

Process Type	File Name
Load Agent	<message-file-name>.load.<partitionnumber>
Partitioning Agent	<message-file-name>.part.<partitionnumber>
Pre-partitioning Agent	<message-file-name>.prep.<partitionnumber>

Note: We strongly recommend that the MESSAGES option be used for partitioned database load operations initiated from the CLP.

Monitoring the LOAD utility

The LOAD QUERY command can be used to check the status of a load operation during processing and returns the table state. You can connect to individual database partitions during a load operation and issue the LOAD QUERY command against the target table. When issued from the CLP, this command displays the contents of all the message files that currently reside on that database partition for the table that is specified in the LOAD QUERY command.

The output of the LOAD QUERY command in Example 6-57 is gathered when Example 6-55 on page 313 is running. If a load job has finished, the command returns the table state as shown in the last command in the example.

Example 6-57 Using LOAD QUERY to monitor LOAD status

```
db2inst1@mensa:/db2backup/datamove> db2 "load query table itemload"
SQL3530I The Load Query utility is monitoring "LOAD" progress on partition
"0".

SQL3109N The utility is beginning to load data from file
"/db2backup/datamove/item.del".

SQL3500W The utility is beginning the "LOAD" phase at time "02/11/2008
15:25:41.102272".

SQL3519W Begin Load Consistency Point. Input record count = "0".

SQL3520W Load Consistency Point was successful.

SQL3532I The Load utility is currently in the "LOAD" phase.
```

```

Number of rows read      = 1467592
Number of rows skipped   = 0
Number of rows loaded    = 1467592
Number of rows rejected  = 0
Number of rows deleted   = 0
Number of rows committed = 0
Number of warnings       = 0

```

```

Tablestate:
  Load in Progress

```

```

db2inst1@mensa:/db2backup/datamove> db2 "load query table itemload"
Tablestate:
  Normal

```

Another way to monitor the LOAD utility is using the LIST UTILITY command. Example 6-58 shows the output which was taken when loading the data into table ITEMLOAD. The table resides on two database partitions.

Example 6-58 Using LIST UTILITIES to monitor LOAD status

```

db2inst1@mensa:/db2backup/datamove> db2 list utilities

```

```

ID                        = 136
Type                      = LOAD
Database Name             = ITSODB
Partition Number          = 0
Description               = OFFLINE LOAD DEL AUTOMATIC INDEXING REPLACE
COPY NO DB2INST1.ITEMLOAD
Start Time                = 02/11/2008 15:57:02.886822
State                     = Executing
Invocation Type           = User

```

```

ID                        = 88
Type                      = LOAD
Database Name             = ITSODB
Partition Number          = 1
Description               = OFFLINE LOAD DEL AUTOMATIC INDEXING REPLACE
COPY NO DB2INST1.ITEMLOAD
Start Time                = 02/11/2008 16:00:40.095780
State                     = Executing
Invocation Type           = User

```

```

db2inst1@mensa:/db2backup/datamove> db2 list utilities show detail

```

```

ID                        = 136
Type                      = LOAD
Database Name             = ITSODB
Partition Number          = 0

```


Description	= OFFLINE LOAD DEL AUTOMATIC INDEXING REPLACE
COPY NO DB2INST1.ITEMLOAD	
Start Time	= 02/11/2008 15:57:02.886822
State	= Executing
Invocation Type	= User
Progress Monitoring:	
Phase Number	= 1
Description	= SETUP
Total Work	= 0 bytes
Completed Work	= 0 bytes
Start Time	= 02/11/2008 15:57:02.886827
Phase Number [Current]	= 2
Description	= LOAD
Total Work	= 1552469 rows
Completed Work	= 643221 rows
Start Time	= 02/11/2008 15:57:04.189448
Phase Number	= 3
Description	= BUILD
Total Work	= 1 indexes
Completed Work	= 0 indexes
Start Time	= Not Started
ID	= 88
Type	= LOAD
Database Name	= ITS0DB
Partition Number	= 1
Description	= OFFLINE LOAD DEL AUTOMATIC INDEXING REPLACE
COPY NO DB2INST1.ITEMLOAD	
Start Time	= 02/11/2008 16:00:40.095780
State	= Executing
Invocation Type	= User
Progress Monitoring:	
Phase Number	= 1
Description	= SETUP
Total Work	= 0 bytes
Completed Work	= 0 bytes
Start Time	= 02/11/2008 16:00:40.095786
Phase Number [Current]	= 2
Description	= LOAD
Total Work	= 1552469 rows
Completed Work	= 174190 rows
Start Time	= 02/11/2008 16:00:41.415394
Phase Number	= 3
Description	= BUILD

Total Work	= 1 indexes
Completed Work	= 0 indexes
Start Time	= Not Started

Using the SHOW DETAIL clause, you get information about the LOAD phases and their progress.

Each LOAD writes an entry into the database recovery history file. If you want to see the processed LOAD commands, please use the command as shown in Example 6-59. The output shows the entries for a LOAD we did in Example 6-53 on page 309. Please note the two entries, one for the LOAD (Op=L) and one for the COPY YES (Op=C).

Example 6-59 List history for LOAD operation

```
db2inst1@mensa: /> db2 list history load all for itsodb
...
Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log Backup ID
-----
L T 20080221114949001 R I S0000876.LOG S0000876.LOG
-----
"DB2INST1"."ITEMLOAD" resides in 1 tablespace(s):

00001 TSDATA2
-----
Comment: DB2
Start Time: 20080221114949
End Time: 20080221115029
Status: A
-----
EID: 1209 Location: DB2 Autoloader. Data was partitioned on 2 node(s), and
loaded on node 0.

Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log Backup ID
-----
C T 20080221114949002 R S S0000876.LOG S0000876.LOG
-----
"DB2INST1"."ITEMLOAD" resides in 1 tablespace(s):

00001 TSDATA2
-----
Comment: DB2
Start Time: 20080221114949
End Time: 20080221115029
Status: A
-----
```

EID: 1210 Location:
/db2backup/datamove/copydata/ITS0DB.4.db2inst1.NODE0000.CATN0000.20080221114949
.001

Load table data using the Control Center

The Load wizard is available in DB2 Control Center, to invoke the Load wizard:

1. From the Control Center, expand the object tree until you find the Tables folder.
2. Click the Tables folder. Any existing tables are displayed in the pane on the right-hand side of the panel (the contents pane).
3. Right-click the table you want in the contents pane, and select **Load** from the pop-up menu. The Load wizard panel opens.

Here we demonstrate how to use the Load wizard to load data into a sample table named ITEMLOAD which is defined on database partition 0 through 1.

Detailed steps are as follows:

1. Invoke the Load Wizard from DB2 Control Center (Figure 6-34 on page 320).

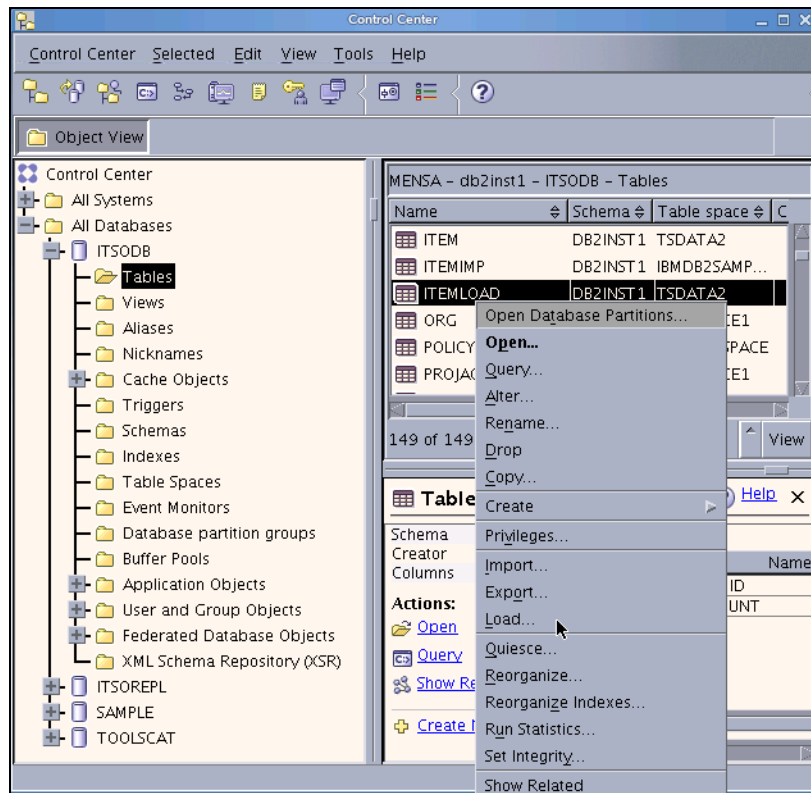


Figure 6-34 Invoking Load Wizard from DB2 Control Center

2. Choose the load operation (Figure 6-35 on page 321)

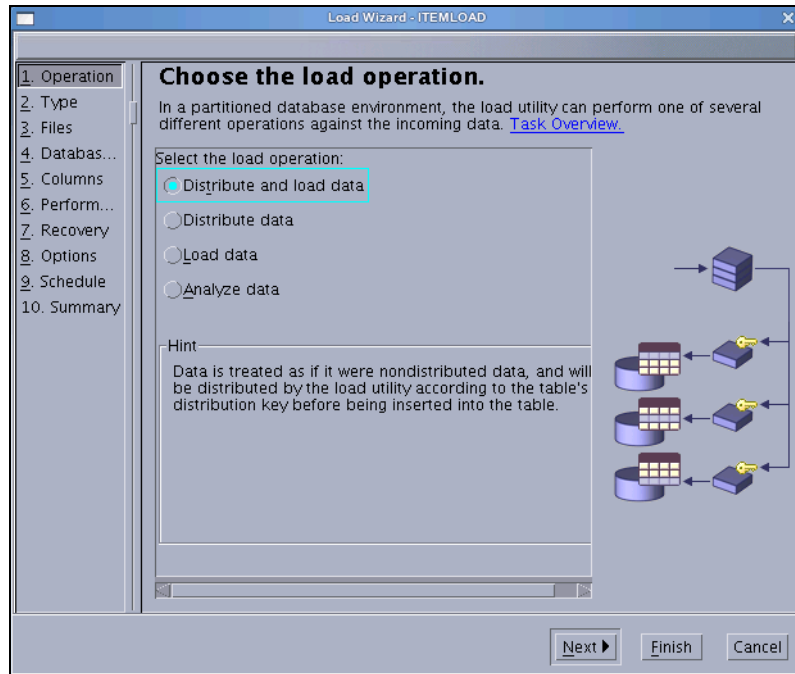


Figure 6-35 Choose the load operation mode

3. Choose whether the original table data will be kept (Figure 6-36 on page 322).

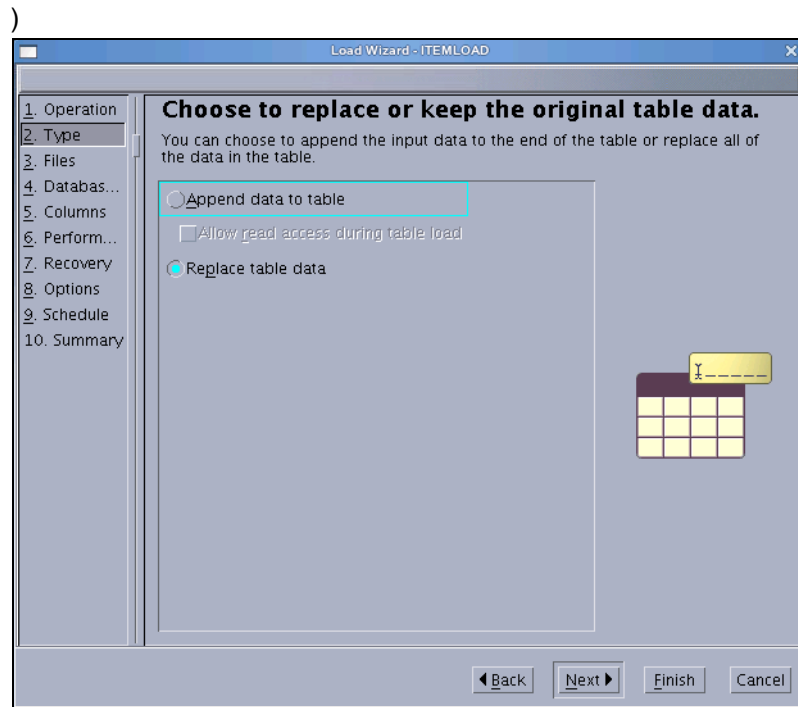



Figure 6-36 Choose if the original table will be kept

4. Specify input and output files (Figure 6-37 on page 323).

In this panel, you can specify the input file format, and options specific for the selected format. For example, you can click **DEL Options** to set up your source DEL file related options. You also can specify where the source resides, Server or Remote host. In addition, you can specify the input file name and output message file name manually or by browsing files using  button.

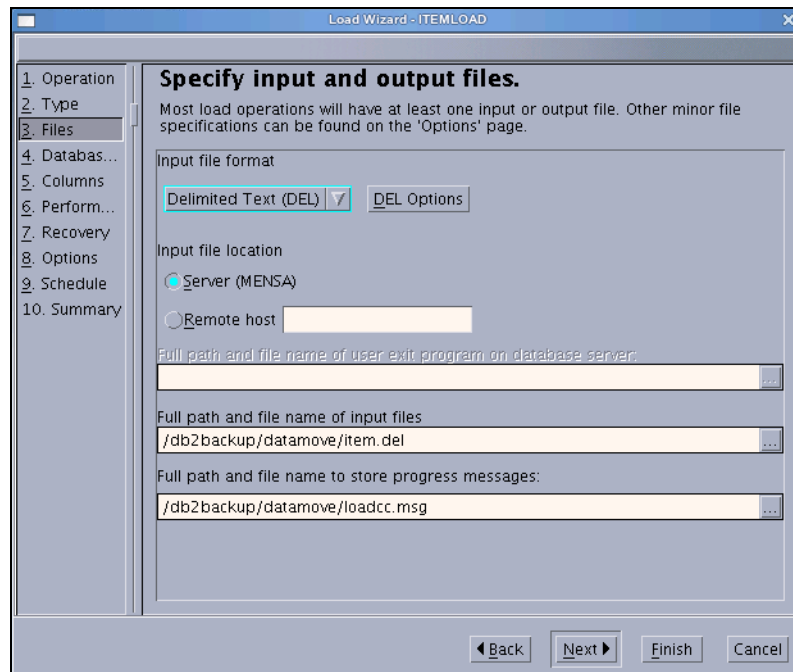


Figure 6-37 Specify input and output files

5. Specify which database partitions participate in the load (Figure 6-38 on page 324).

You can use the default choice of “Load data into all database partitions” if you just want to load data into all the database partitions where the table resides. Under some conditions, you may want to load data for a specific database partition or partitions, then you can choose to “Load data into selected database partitions”. Under “Distributing partitions”, you can specify which database partitions will be used to do the data partitioning work. In general, you can just leave as “Let DB2 decide which database partitions participate”.

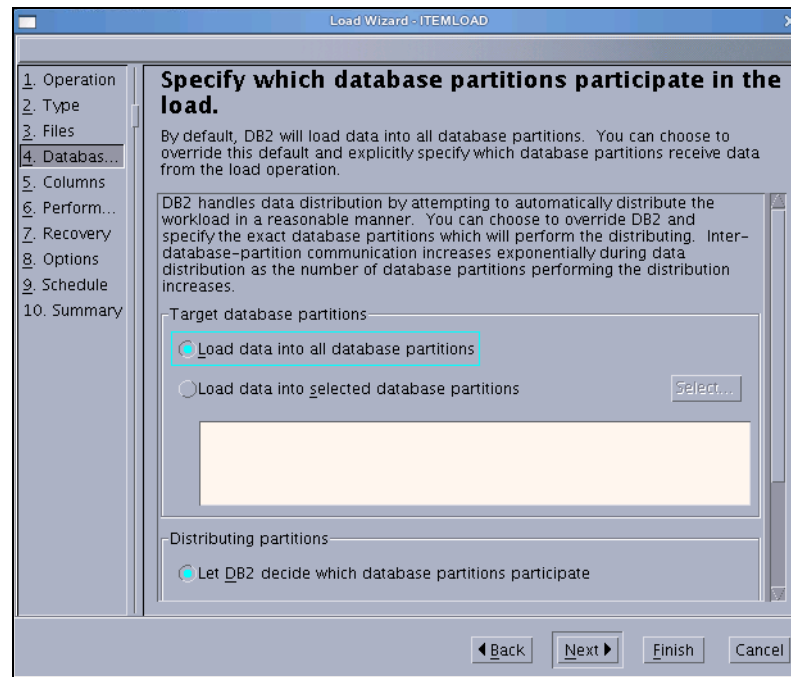


Figure 6-38 Specify which database partitions participate in the load

6. Define the input columns and their mapping to the output columns (Figure 6-39 on page 325).

You can simply use the default mapping if you think this is appropriate for your situation, or you can make some modifications in this panel if needed. For example, if large object (LOB) is involved in your load, you may need to specify LOB related options here.

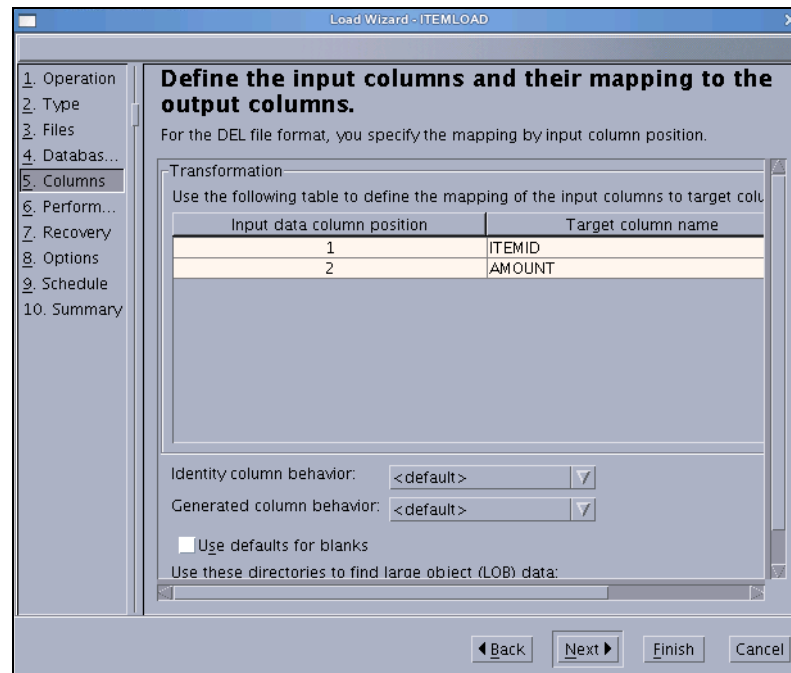


Figure 6-39 Define column related properties for load

7. Specify performance and statistics collection options (Figure 6-40 on page 326).

You can specify options which will impact load performance. For example, you can specify how the load utility will update the existing indexes, rebuild all indexes or rebuild indexes incrementally. A full index rebuild is faster than an incremental rebuild. If the index is very large, it will be quicker to do an incremental rebuild. In addition, options for statistics collection are also provided in this panel. If you do not want to make any modifications, take the defaults.

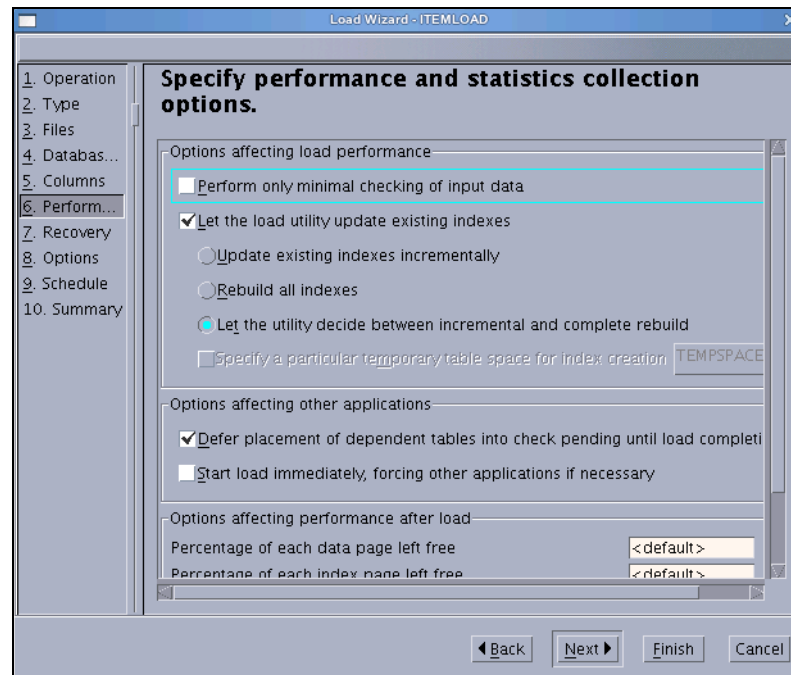


Figure 6-40 Specify performance and statistics collection options

8. Specify failure options and recovery strategy.

Options regarding failure handling and recovery strategy are covered in this panel. For example, if you want the whole load operation to halt when any error occurs during the load operation, then you can check both boxes for error isolation, load job fails, and load rollback in the Crash recovery section as shown in Figure 6-41 on page 327.

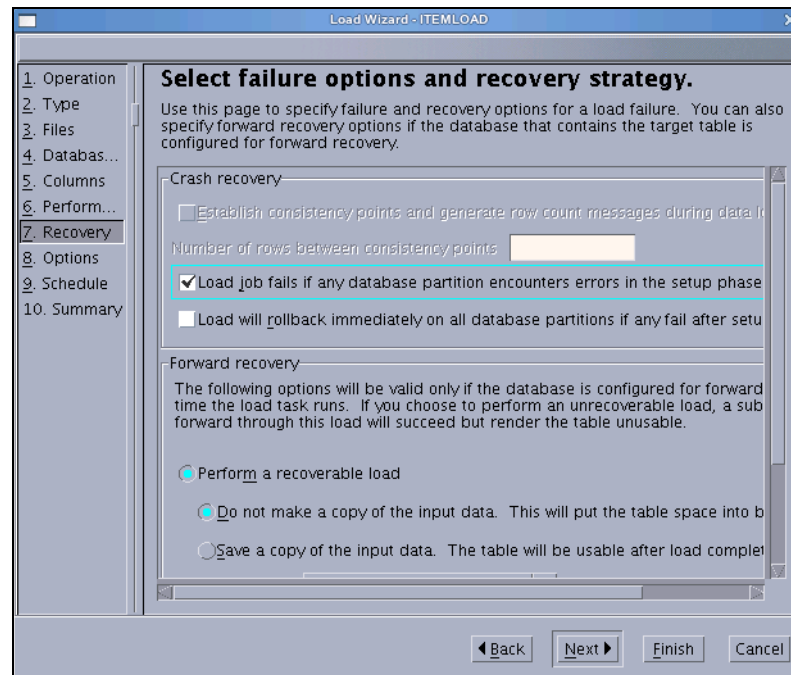


Figure 6-41 Specify failure options and recovery strategy

9. Set advanced options.

Here you can specify additional options associated with the load operation which are not covered by the preceding panels, and you can get hints by clicking on the option listed in the panel (Figure 6-42 on page 328).

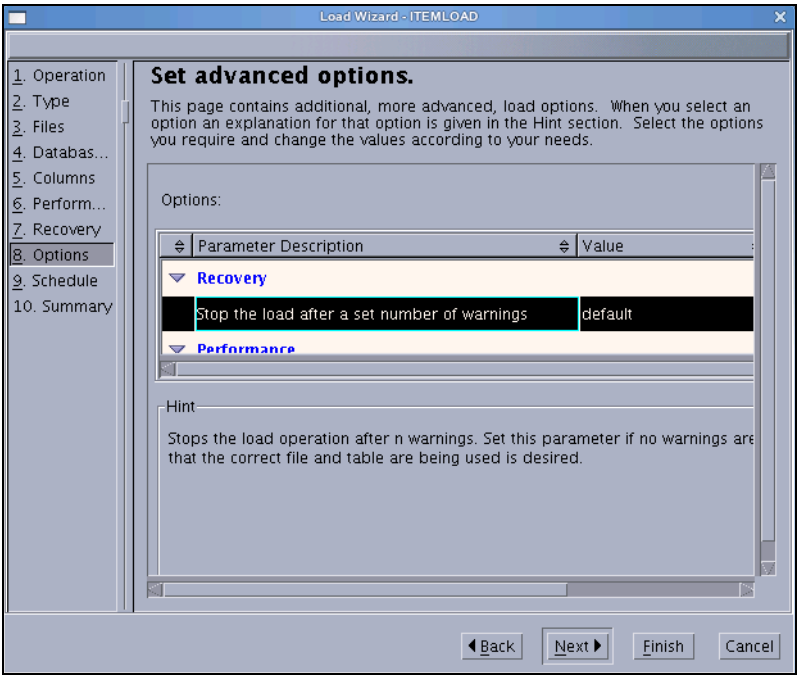


Figure 6-42 Set advanced options for load operation

10. Scheduling task execution.

If you want to run the load operation without saving task history, choose **Run now without saving task history** (Figure 6-43 on page 329). If you want to create it as a task in the Task Center, you can specify the details here.

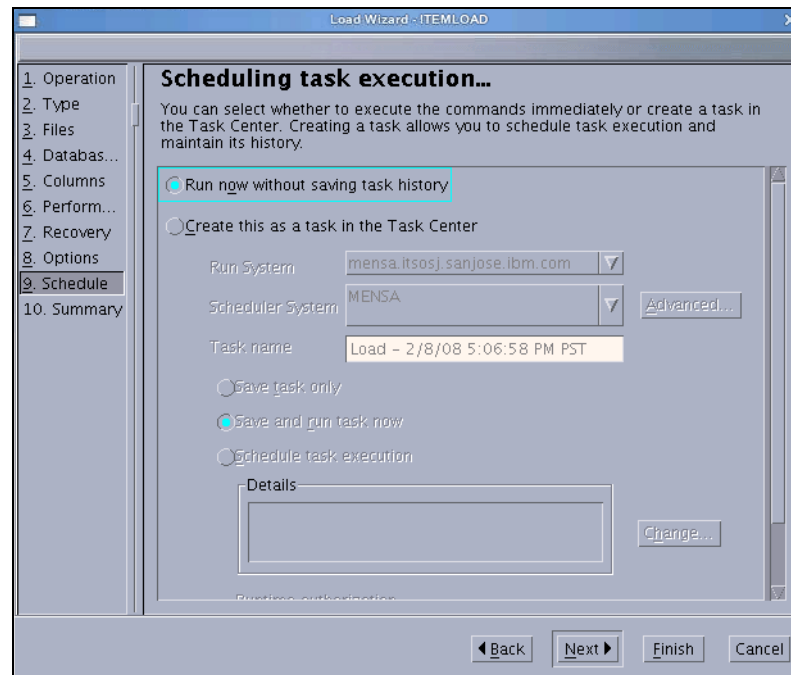


Figure 6-43 Scheduling task execution

11. View the task summary and execute it.

In Figure 6-44 on page 330, the DB2 CLP commands which were generated based on your choices are displayed. You may click **Finish** to submit the command, or click **Back** for further modification of your choices.

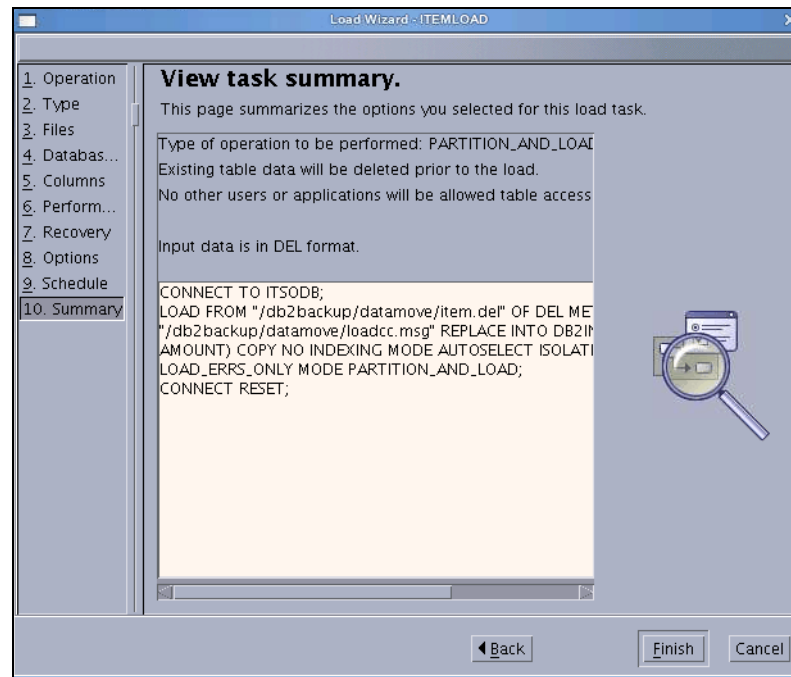


Figure 6-44 View task summary

After running the load, you can check the message files for more details regarding the load operation within the DB2 Control Center. The message files in our case are:

- ▶ loadcc.msg.prep.000 for pre-partitioning,
- ▶ loadcc.msg.part.001 for partitioning,
- ▶ loadcc.msg.load.000, and loadcc.msg.load.001 for loading.

All these files are located under the directory /db2backup/datamove which was specified in the Specifying input and output files panel.

For detailed information about using load, refer to *Data Movement Utilities Guide and Reference*, SC23-5847-00 and *Command Reference*, SC23-5846-00

6.3.4 Using db2move utility

This tool facilitates the movement of large numbers of tables between DB2 databases located on workstations. It calls the DB2 export and import or load APIs, depending on the action requested by the user, and operates the files in PC/IXF format.

With the `-co COPY` option it can also be used to copy one or more schemas from a source database to a target database. In Example 6-60 we copy all tables from our `ITSODB` database into a new database called `ITSOREPL`. During the copy, the schema name is changed from `db2inst1` to `db2vs` and all tables are placed into the default table space. Note that you also can specify table space name mappings to use different table spaces.

Example 6-60 Change schema using d2move

```
db2inst1@mena:/db2backup/datamove> db2 create db itsorepl
DB20000I The CREATE DATABASE command completed successfully.
db2inst1@mena:/db2backup/datamove> db2move itsodb COPY -sn DB2INST1 -co
target_db itsorepl schema_map "((DB2INST1,DB2VS))" schema_map
"((DB2INST1,DB2VS))" owner db2vs -u db2inst1 -p mypassword
```

Application code page not determined, using ANSI codepage 1208

***** DB2MOVE *****

Action: COPY

Start time: Mon Feb 11 10:15:44 2008

All schema names matching: DB2INST1;

Connecting to database ITSODB ... successful! Server : DB2 Common Server
V9.5.0

Copy schema DB2INST1 to DB2VS on the target database ITSOREPL

Create DMT : "SYSTOOLS"."DMT_47b090d571d8d"

Start Load Phase :

db2move finished successfully

Files generated:

```
-----
COPYSHEMA.20080211101544.msg
LOADTABLE.20080211101544.MSG
```

Please delete these files when they are no longer needed.

End time: Mon Feb 11 10:18:35 2008

The files which were generated during the data move can be checked to see the details about the COPY and the LOAD. In our case, the following files were created:

```
COPYSCHEMA.20080211101544.msg
LOADTABLE.20080211101544.MSG
LOADTABLE.20080211101544.MSG.load.000
LOADTABLE.20080211101544.MSG.load.001
LOADTABLE.20080211101544.MSG.part.001
```

In Example 6-61 we confirm that the schema name was changed.

Example 6-61 Confirm the changed schema name during db2move

```
db2inst1@mena:/db2backup/datamove> db2 connect to itsodb
```

Database Connection Information

```
Database server      = DB2/LINUX8664 9.5.0
SQL authorization ID = DB2INST1
Local database alias = ITSODB
```

```
db2inst1@mena:/db2backup/datamove> db2 "select substr(tabschema,1,10),
substr(tabname,1,10), substr(tbspace,1,10) from syscat.tables where tabname =
'ITEM'"
```

1	2	3
-----	-----	-----
DB2INST1	ITEM	TSDATA2

1 record(s) selected.

```
db2inst1@mena:/db2backup/datamove> db2 connect to itsorepl
```

Database Connection Information

```
Database server      = DB2/LINUX8664 9.5.0
SQL authorization ID = DB2INST1
Local database alias = ITSOREPL
```

```
db2inst1@mena:/db2backup/datamove> db2 "select substr(tabschema,1,10),
substr(tabname,1,10), substr(tbspace,1,10) from syscat.tables where tabname =
'ITEM'"
```

1	2	3
-----	-----	-----
DB2VS	ITEM	USERSPACE1

1 record(s) selected.

For details regarding this utility, refer to *Command Reference*, SC23-5846-00.

The *ADMIN_COPY_SCHEMA* stored procedure is another method to copy a schema. This stored procedure can be used to copy a specific schema and all objects contained in it. The new target schema objects will be created using the same object names as the objects in the source schema, but with the target schema qualifier.

Note: The *ADMIN_COPY_SCHEMA* procedure can be used for copying schemas within the same database, while the *db2move* utility is used across databases.

In Example 6-62 we create a schema called *db2vs1*.

Example 6-62 Using ADMIN_COPY_SCHEMA procedure

```
db2inst1@mena:/db2backup/datamove> db2 "call
sysproc.admin_copy_schema('DB2VS', 'DB2VS1', 'COPY', NULL, 'USERSPACE1',
'USERSPACE1, SYS_ANY', 'DB2VS2', 'ERRORTAB')"
```

Value of output parameters

Parameter Name : ERRORTABSCHEMA

Parameter Value : -

Parameter Name : ERRORTABNAME

Parameter Value : -

Return Status = 0

```
db2inst1@mena:/db2backup/datamove> db2 "select substr(tabschema,1,10),
substr(tabname,1,10), substr(tbspace,1,10) from syscat.tables where tabname =
'ITEM'"
```

1	2	3
DB2VS	ITEM	USERSPACE1
DB2VS1	ITEM	USERSPACE1

2 record(s) selected.

```
db2inst1@mena:/db2backup/datamove> db2 "select count(*) from db2vs1.item"
```

```
1
-----
```

3000000

1 record(s) selected.

6.4 Task Center, Scheduler, and DB2 Tools Catalog

In this section, we discuss how to use the DB2 Task Center to create and schedule a task to run on Linux. The DB2 Administration Server and Tools Catalog database are the prerequisite for using the Task Center and most GUI tools. We discuss the DB2 Administration Server and Tools Catalog database first as the preparation work for enabling the Task Center and Scheduler.

6.4.1 DB2 Administration Server and Tools Catalog Database

The DB2 Administration Server (DAS) is a control point used only to assist with tasks on DB2 servers. You must have a running DAS if you want to use available tools such as the Task Center, the Control Center, or the Configuration Assistant. The DAS supports the Control Center and Configuration Assistant when working on the following administration tasks:

- ▶ Enabling remote administration of DB2 servers
- ▶ Providing the facility for job management, including the ability to schedule the running of both DB2 and operating system command scripts
- ▶ Defining the scheduling of jobs, viewing the results of completed jobs, and performing other administrative tasks against jobs located either remotely or locally to the DAS using the Task Center
- ▶ Providing a means for discovering information about the configuration of DB2 instances, databases, and other DB2 administration servers in conjunction with the DB2 Discovery utility

The DAS on Linux (plus UNIX and Windows) includes a scheduler to run tasks (such as DB2 and operating system command scripts) defined using the Task Center. Task information includes commands to be run, schedules, notifications, and completion actions associated with the task. The run results are stored in a DB2 database called the Tools Catalog database. The Tools Catalog database is created as part of the general setup. If you did not create the Tools Catalog database when installing DB2, it can be created and activated through the Control Center, or through the CLP using the `CREATE TOOLS CATALOG` command. For information regarding how to create Tools Catalog in DB2 Control Center or using DB2 CLP commands, refer to Appendix B, “DB2 Tools Catalog creation” on page 487.

The tools catalog database contains task information created by the Task Center and Control Center. These tasks are run by the scheduler on the DB2 Administration Server. Figure 6-45 shows the relationship between DB2 tools, DB2 instances, DAS, Scheduler, and the Tools Catalog database.

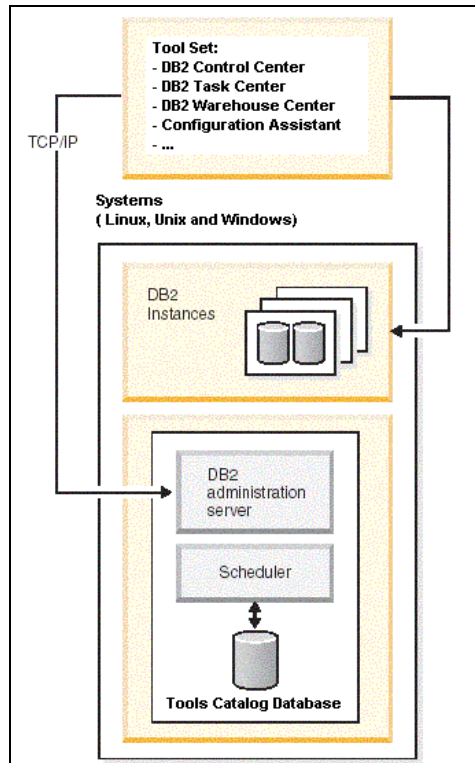


Figure 6-45 How DAS and Tools Catalog Database relate to other parts of DB2

6.4.2 Task Center and Scheduler

After the DAS has started and the Tools Catalog is ready, you can start the Task Center within the DB2 Control Center or other tools where the Tools menu is available. You can use the Task Center to run tasks, either immediately or according to a schedule, and to notify people about the status of the completed tasks. The Task Center includes functionality from the Script Center in previous versions of DB2, plus additional functionality.

A task is a script, together with the associated success conditions, schedules, and notifications. You can create a task within the Task Center, create a script within another tool and save it to the Task Center, import an existing script, or

save the options from a DB2 dialog or wizard such as the Load wizard. A script can contain DB2 commands, SQL, or operating system commands.

For each task, by using the Task Center, you can do the following:

- ▶ Schedule the task.
- ▶ Specify success and failure conditions.
- ▶ Specify actions that should be performed when this task completes successfully or when it fails.
- ▶ Specify e-mail addresses (including pagers) that should be notified when this task completes successfully or when it fails.

You can specify conditional coding by creating task actions. Each task action consists of a task and the action that should be performed by the task. For example, task one can have the following task actions:

- ▶ If task one is successful, task action A enables the schedule for task two.
- ▶ If task one fails, task action B runs task three.

The following is a sample with detailed steps to create a task within the Task Center. Task scheduling is also included.

1. Start the Task Center and create a new task (Figure 6-46 on page 337):

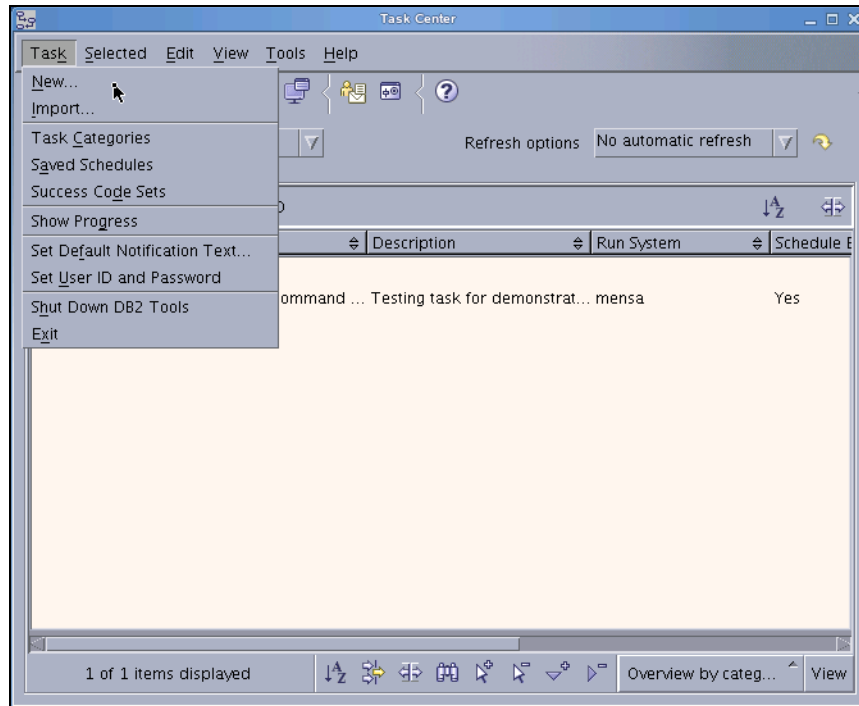


Figure 6-46 Task Center main panel

2. General information input for a new task is shown(Figure 6-47 on page 338).

You can input general info for a task in this panel. For example, the name and type for the task, the instance and database partitions the task will be running on, and so on. If you choose Grouping task for the type, then you can add several tasks into a group, and then specify scheduling, failure and success conditions, as well as related actions to the group.

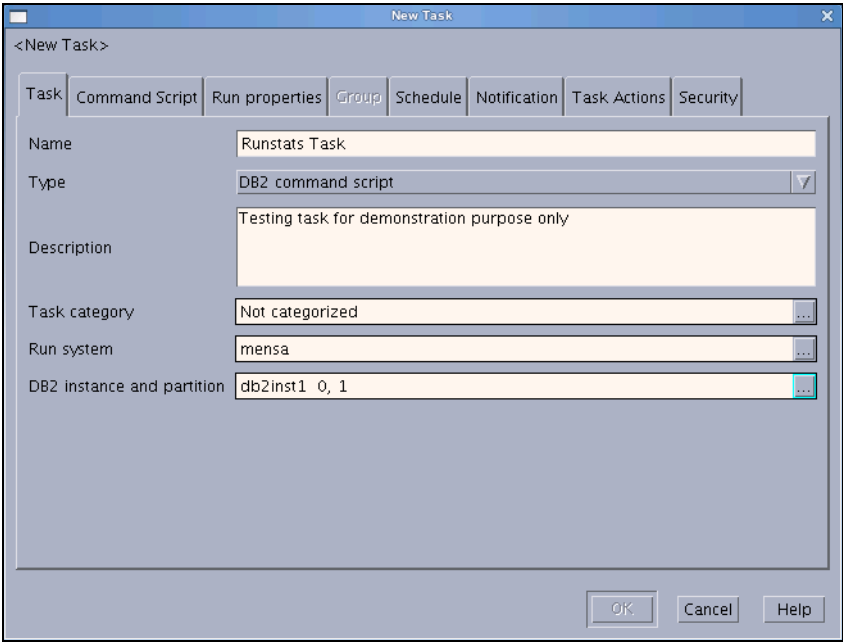


Figure 6-47 General info input for a new task

3. Command script for the new task.

Here you can input the script for the task. By default, the semicolon “;” is used as the DB2 statement termination character. If you want, you can change it. Our sample script is shown in Figure 6-48.

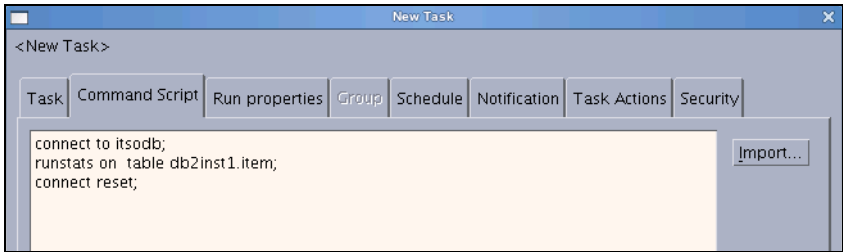


Figure 6-48 Command script input for the new task

4. Running properties setting.

You can define the specific success code for your task and choose if DB2 should stop execution when failure happens during the task execution. In our example, we choose **Stop execution at first return code that is failure** when failure happens (Figure 6-49).

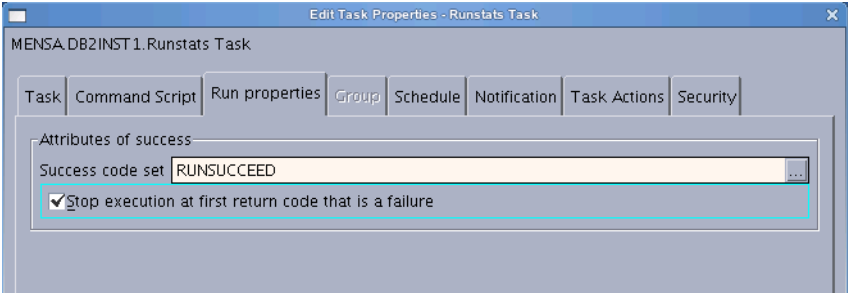


Figure 6-49 Run properties for the task

5. Schedule the task.

You can schedule the task to run at a designated time, only once or repetitively, depending on your real situation. Figure 6-50 provides a sample to specify running a weekly task and with an end date set.

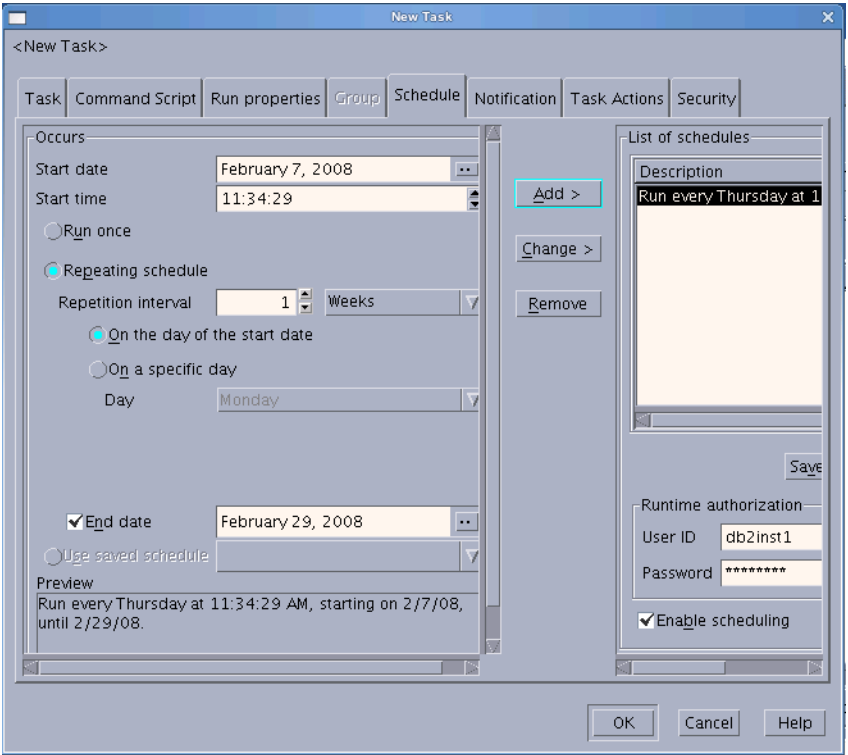


Figure 6-50 Scheduling tasks for repetitive running

Figure 6-51 is another example which is used to run a task once.

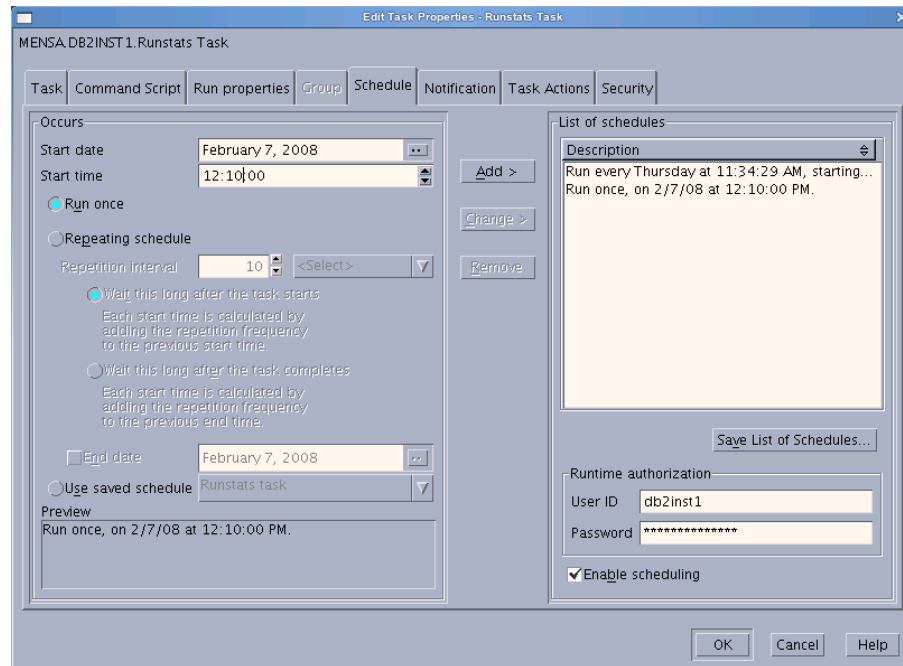


Figure 6-51 Scheduling tasks for running once

Furthermore, if you want to apply the scheduling pattern for other tasks, you can save the schedule list by clicking **Save List of Schedules ...** in the right pane of the panel. Once saved, the next time you schedule a task, you can choose **Use saved schedule** with the desired name to schedule a task.

6. Notification setting for different task running result.

You can specify under which condition a notification will be sent, and it can be sent to a contact or contact group, by e-mail or pager, or sent to DB2's Journal as a message entry. For conditions, it can be success, failure, or both. You also can specify multiple notifications for different conditions for only one task.

In Figure 6-52 on page 341, the notification will be sent to DB2's Journal as a message entry.

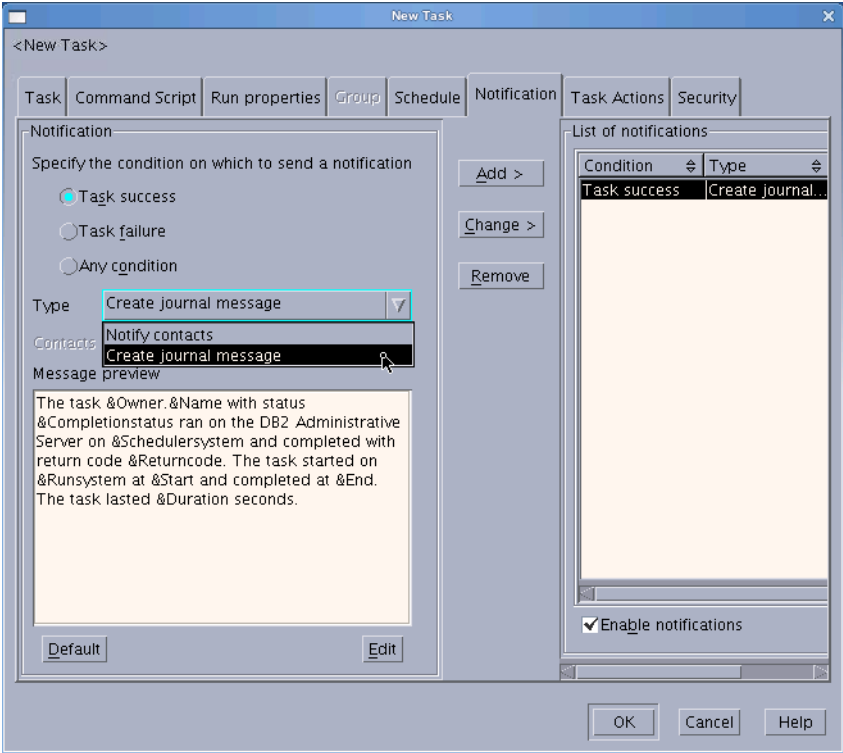


Figure 6-52 Notification setting for different task running result

7. Follow-up actions for different task running result (Figure 6-53 on page 342).

In addition to the notification setting for the results of task running, you can also specify follow-up actions. For example, if the task is failed at last, and for “Task Failure” condition, you can specify running another alternative task to remedy the condition, or to disable scheduling of another task to avoid unnecessary damage.

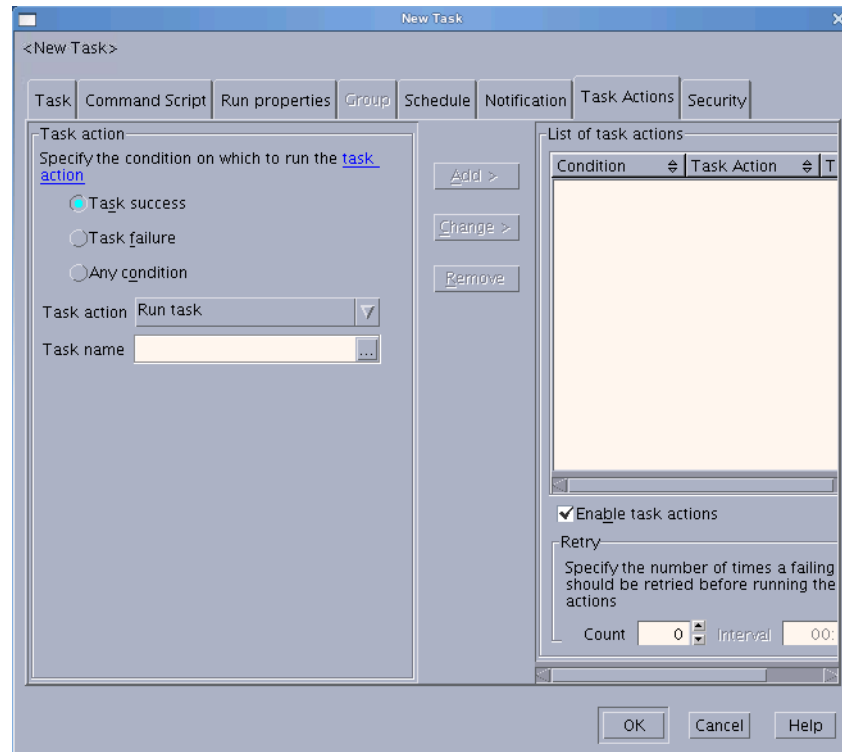


Figure 6-53 Follow-up actions setting for a task

You can also specify access privileges to the task, such as read, run, or execute for different users and groups under the Security tab.

Now you can submit the task to the Task Center by clicking **OK**.

If the task ran, you can get the result information through the context menu of the task as shown in Figure 6-54 on page 343.

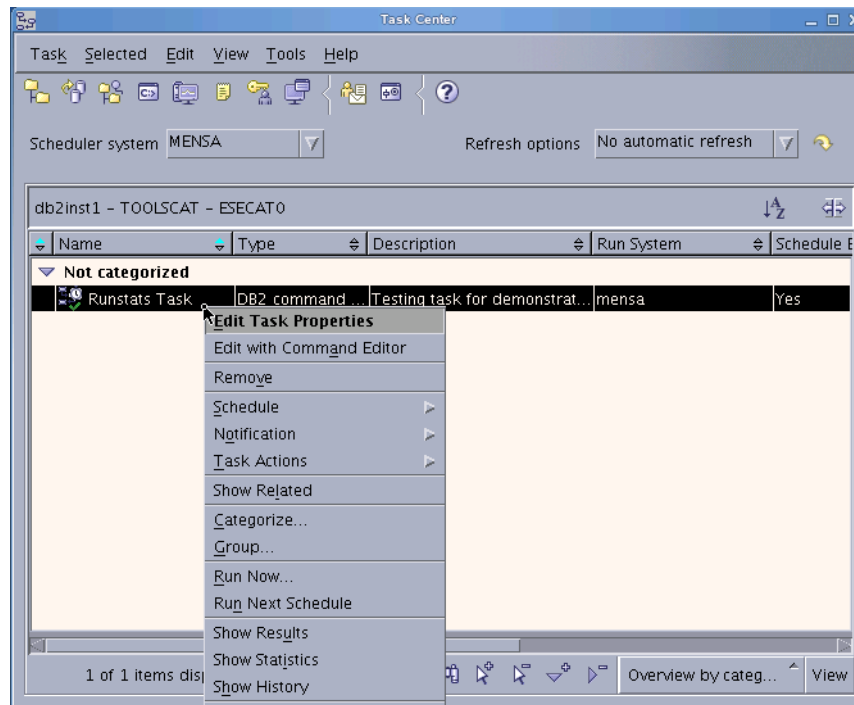


Figure 6-54 Context menu of a task

The result of our sample task is shown in Figure 6-55 on page 344.

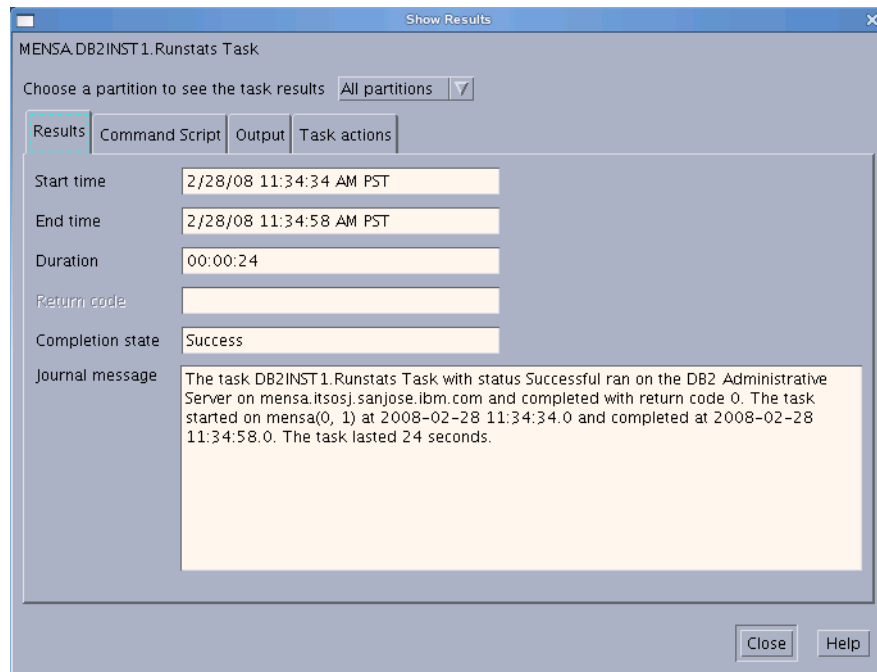


Figure 6-55 Show result panel

The task completed successful. The command script and running output of the commands specified in the task are shown in Figure 6-56 on page 345:

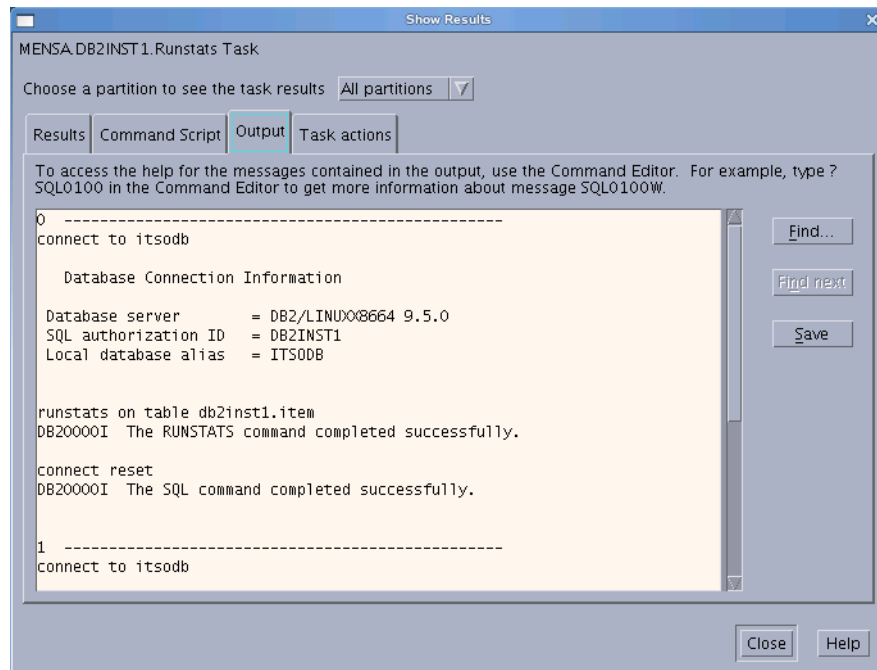


Figure 6-56 DB2 command output

Further modification can be done for the task within the Task Center. For example, you can change the schedule or disable the follow-up actions.

The scheduling function is also available for many other DB2 Wizards or GUI tools such as export, import, load, runstats, reorg, backup and restore. You can take advantage of these useful functions for task automation purposes in your environment.



Monitoring and troubleshooting DB2

This chapter discusses DB2's monitoring and troubleshooting features, the following topics are discussed:

- ▶ Health monitor and Health Center
- ▶ Memory Visualizer and Memory Tracker (db2mtrk)
- ▶ IBM single system monitor view for DB2 (db2top)
- ▶ Log files for troubleshooting
- ▶ DB2 PD/PSI tools
- ▶ Linux system monitoring tools

7.1 Health monitor and Health Center

Health monitor and Health Center are two DB2 features that can help you monitor the health of your DB2 systems. These tools add a *management-by-exception* capability to DB2 by alerting you to potential system health issues. This enables you to address health issues before they become real problems that affect your system.

7.1.1 Health indicator and health monitor

Health Indicator

Prior to using the health monitor, you need to understand the concept of a health indicator. A health indicator is a measurement that gauges the healthiness of some aspect of an object. The health monitor uses these indicators to evaluate specific aspects of database manager or database performance.

DB2 comes with default settings for all health indicators. Using the Health Center, DB2 commands, or APIs, you can customize the settings of the health indicators, and define who should be notified and what script or task should be run if an alert is issued. For example, you can customize the alarm and warning thresholds for the amount of space used in a table space.

Health monitor

The health monitor is a server-side tool that constantly monitors the health of the instance and all active database objects when DB2 is started. The health monitor is enabled by default when a instance is created; you can deactivate it using the database manager configuration parameter *HEALTH_MON*.

The health monitor gathers information about the health of the system using new interfaces that do not impose a performance penalty. It does not turn on any snapshot monitor switches to collect information. The health monitor automatically evaluates a set of health indicators, even without user interaction. If the health monitor finds that a defined threshold has been exceeded (for instance, the available log space is not sufficient), or if it detects an abnormal state for an object (for example, an instance is down), the health monitor raises an alert. There are three types of alerts: *attention*, *warning*, and *alarm*. Any time an alert is raised, the health monitor may take any of the following actions to report it:

- ▶ Record the alert information in the Journal (all alarm alerts are written to the Journal).
- ▶ Send alert notifications through e-mail or a pager address to the person responsible for the system.

- Carry out one or more predefined actions (for example, running a task).

You can use DB2 commands or APIs to retrieve health information from the health monitor, allowing you to integrate DB2 health monitoring with existing system-wide monitoring solutions.

7.1.2 Monitoring with the Health Center

The Health Center provides the graphical interface to the health monitor. You use it to configure the health monitor, to define the threshold values for desired health indicators and related activities when the threshold values are exceeded, for example, a notification to DBA, or a follow-up task will be executed. You can also use Health Center to view the rolled up alert state of your instances and database objects. Using the health monitor's drill-down capability, you can access details about current alerts and obtain a list of recommended actions that describe how to resolve the alert.

You can start the Health Center by selecting Health Center from the Tools menu within any DB2 GUI tools where the Tools menu is available, or by executing the command **db2hc** from the Command Line Processor.

Activating the health monitor

The health monitor is activated by default, and if it is stopped for any reason, you can reactivate it by using Health Center or using DB2 CLP commands.

When health monitor is stopped because the HEALTH_MON parameter in database manager configuration (DBM CFG) is OFF, the *Start Health Monitor* menu item will appear in the pop-up window, when you right-click on the instance name, as shown in Figure 7-1 on page 350. Similarly, the Stop Health Monitor menu item will appear if HEALTH_MON is set to ON and the health monitor is started.

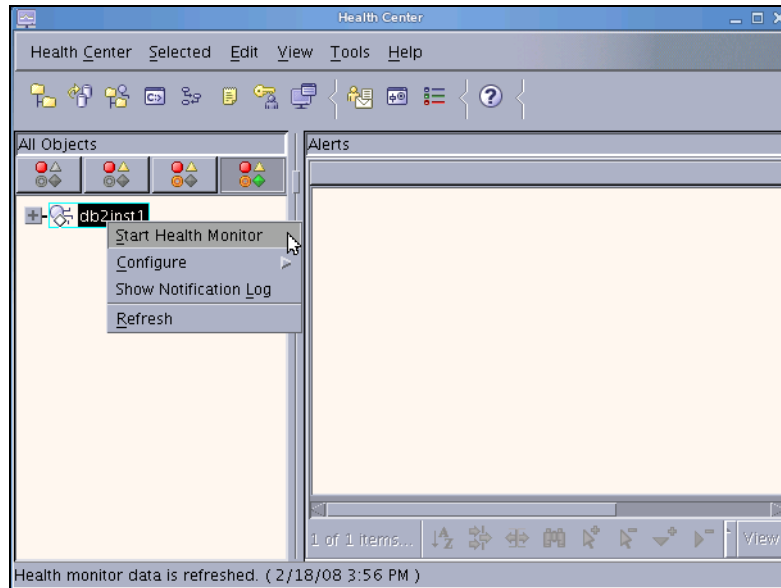


Figure 7-1 Start Health monitor for instance using Health Center


You can also use DB2 CLP commands to activate the health monitor by updating the HEALTH_MON parameter for DBM and view the current setting of HEALTH_MON parameter as shown in Example 6-1.

Example 7-1 Check if health monitor is enabled for instance

```
db2inst1@mensa:/db2home/db2inst1> db2 get dbm cfg | grep -i health
Monitor health of instance and databases (HEALTH_MON) = OFF
db2inst1@mensa:/db2home/db2inst1> db2 update dbm cfg using HEALTH_MON on
DB20000I The UPDATE DATABASE MANAGER CONFIGURATION command completed
successfully.
db2inst1@mensa:/db2home/db2inst1> db2 get dbm cfg | grep -i health
Monitor health of instance and databases (HEALTH_MON) = ON
```

The HEALTH_MON parameter can be dynamically changed. The update to the HEALTH_MON takes effect immediately. Once the HEALTH_MON is turned on (the default), an agent will collect information about the health of the objects that are active in your database environment. If an object is considered to be in an unhealthy condition, based on threshold or object state, the notifications can be sent and actions can be taken automatically.

Configuring Health Center

You can use the toggle buttons  to filter the alerts according to their severity (Figure 7-2 on page 351). The Health Center is opened with the

Object in Any Alert State toggle button (third toggle button) selected by default, which helps to identify those instances with current alerts that should be addressed. If the Health Center is opened for the first time, you can choose *All Objects* toggle button (the fourth toggle button) to display *All Objects* to view the navigation tree with all cataloged LUW instances on the left pane of the panel.

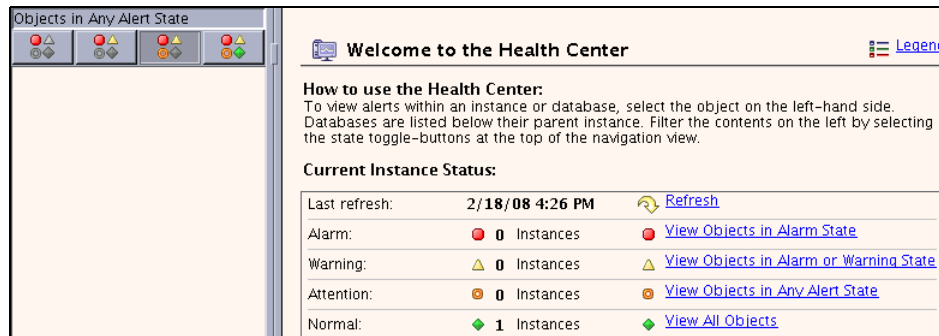


Figure 7-2 Using toggle buttons to filter the alerts

The Configuration option in the Health Center allows you to set up notification for the instance, and modify health indicator settings. When you choose the **Configure** menu item, a submenu with two items is displayed, see Figure 7-3. You can view or change the health indicator settings by clicking **Health Indicator Settings**, or you can use **Alert Notification** to set up the notification for the health monitor.

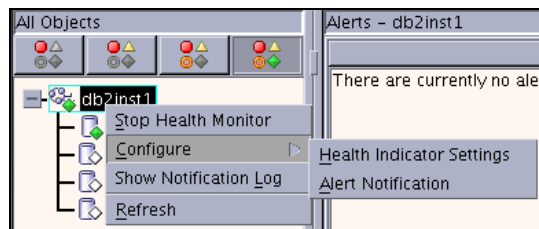


Figure 7-3 Configure menu in Health Center

If you want to notify people using e-mail or pager when an alert is generated, you need to define a contact list first. You can add a contact or contact group in the Contacts management panel or by using DB2 CLP commands. For example, to add a contact, *db2inst1@itsosj.sanjose.ibm.com* to the contact list, execute the following command:

```
db2 add contact db2inst1 type email address db2inst1@itsosj.sanjose.ibm.com
```

If you want to add the contact using the Contacts management GUI tool, select **Tools** → **Contacts** within the Health Center (Figure 7-4):



Figure 7-4 Starting Contacts management

Within the Contacts panel, you can add or change contacts or contact groups, or remove unwanted contacts or contact groups. In addition, you can also verify if the e-mail address specified for the contact is reachable by pressing the **Test** button, as shown in Figure 7-5 on page 352.

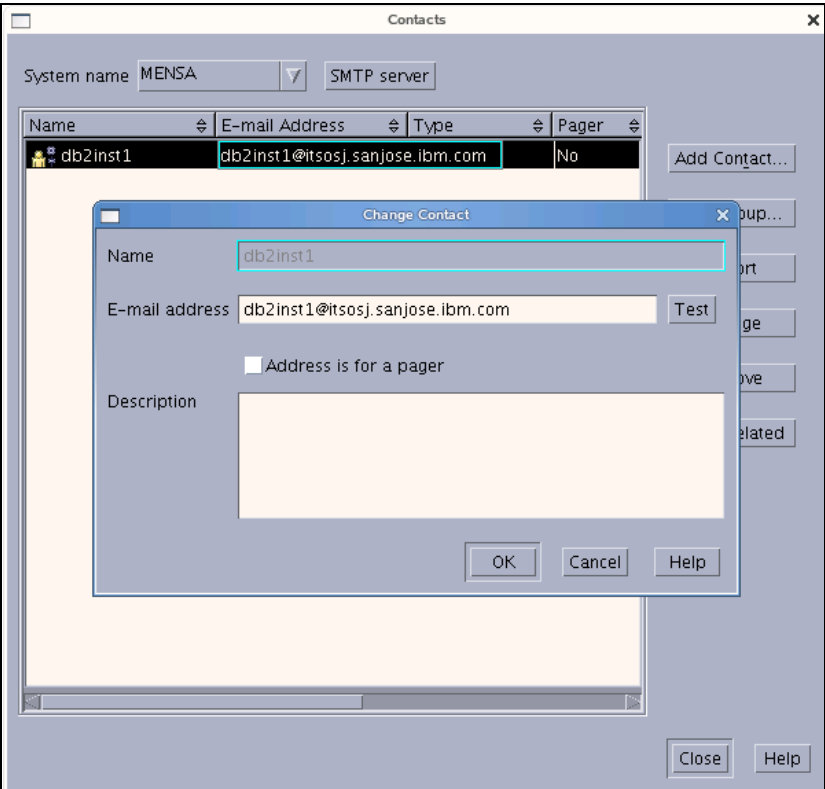


Figure 7-5 Contacts management

After pressing the Test button, if the DB2 message returned shows that the test message sent successfully, then an e-mail will be sent to the specified e-mail account (Figure 7-6).

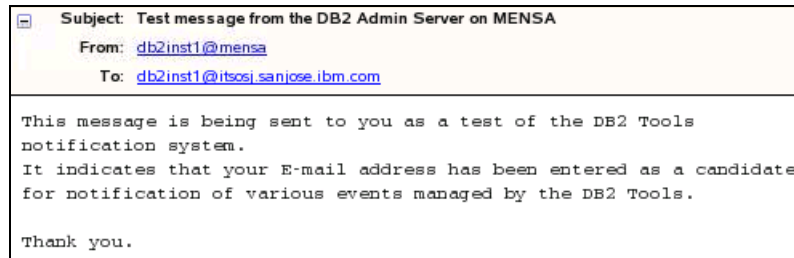


Figure 7-6 The testing e-mail sent by DB2 automatically

Note: Before using the e-mail notification function, you need to make sure that the e-mail system which is residing on the mail server (specified by SMTP_SERVER in administration configuration) is up and running normally

After your contact list is added, you can select **Configure** → **Alert Notification** from the pop-up panel (Figure 7-3 on page 351). Then, add the available contacts to the Health notification contact list, as shown in Figure 7-7.

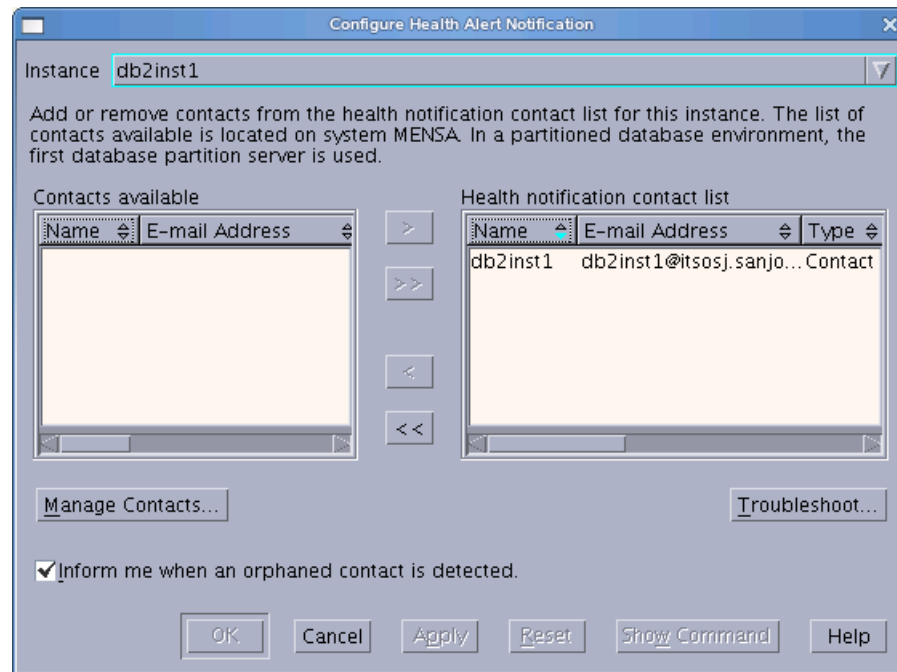


Figure 7-7 Configure health alert notification

The people in the notification list will be informed if any alert is generated with a severity covered by the current notify level setting. By default, the notification information will be written into DB2 instance's notification log which is located under directory \$HOME/sqllib/db2dump, file name is <instance name>.nfy. For details regarding the notification log as well as its setting, refer to 7.4.1, "DB2 administration notification log" on page 389.

Usage sample

In this section, we demonstrate how to configure a health indicator in the Health Center, or by DB2 CLP commands, to monitor DMS table space storage consumption. In this example, we expect when table space utilization, which is measured as the percentage of space consumed, exceeds the predefined threshold values for a warning or alarm, DB2 system will send notification to the administration notification log and the specified contacts by e-mail. You can get recommendations from the Health Center or by DB2 CLP commands for such a situation and take actions accordingly. Furthermore, we also show you how to specify follow-up actions for the occurrence of a warning or alarm for this specific health indicator. You can have the DB2 system take the corrective action automatically when the values for a health indicator setting are exceeded.

The example consists of the following steps:

1. Prepare DMS table space and table.
2. Review health indicator for the DMS table space
3. Populate a desired amount of data into the table space
4. Check the notification sent by DB2 system
5. Take corresponding action to resolve current health alerts
6. Set up additional follow-up actions

These steps are given in detail:

1. Prepare DMS table space and table.

In our example, a DMS table space with file container is used. The size of the table space is 10 MB on each partition at its first creation. After the table space is created, a table with a distribution key is created within (Example 7-2).

Example 7-2 Prepare table space and table for Health Center

```
db2inst1@mena:/db2home/db2inst1/script> db2 -tvf crt_sample.db2
CREATE LARGE TABLESPACE tbspdms_0 IN DATABASE PARTITION GROUP
IBMDEFAULTGROUP PAGESIZE 8192 MANAGED BY DATABASE USING (file
'/database/db2inst1/NODE000 $N /ITSODB/T0000008/C0000000.LRG' 10M)
AUTORESIZE NO1
DB20000I The SQL command completed successfully.
```

```
CREATE TABLE db2inst1.empinfo (ID INTEGER NOT NULL, LASTNAME VARCHAR(30),
HIREDATE DATE, SALARY INTEGER, PRIMARY KEY(ID)) in tbspdms_0 DISTRIBUTE BY
HASH(ID)
DB20000I The SQL command completed successfully.
```

2. Review health indicator settings for the DMS table space.

To configure the health indicator for DMS table spaces in the Health Center, select **Configure Health Indicator Settings** to start Health Indicator Configuration Launchpad panel as shown in Figure 7-8 on page 355.

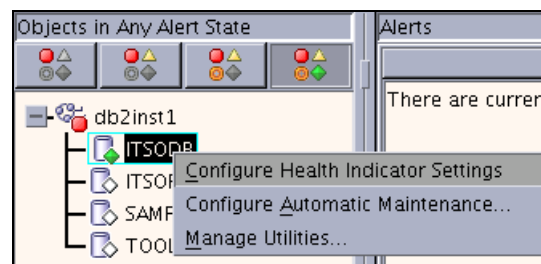


Figure 7-8 Configuring health indicator settings

¹ For test purpose, we change AUTORESIZE parameter for table space from default YES to NO.

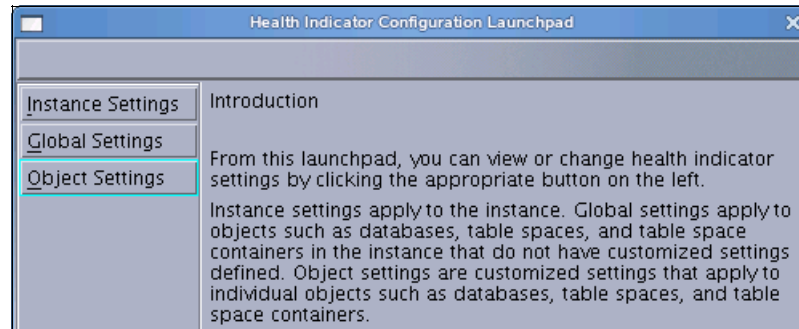



Figure 7-9 Health indicator configuration launchpad

From the *Health Indicator Configuration Launchpad* (Figure 7-9), you can view or change health indicator setting by clicking the appropriate button.

- a. *Instance Settings* apply to the instance.
- b. *Global Settings* apply to objects such as database, table spaces, and table space containers in the instance that do not have customized settings defined.
- c. *Object Settings* are customized settings that apply to individual objects such as databases, table spaces, and table space containers.

In our sample, we set health indicator for a specific table space, click **Object Settings**. On *Object Health Indicator Configuration* panel (Figure 7-10 on page 357), click button  on the *Object* field.

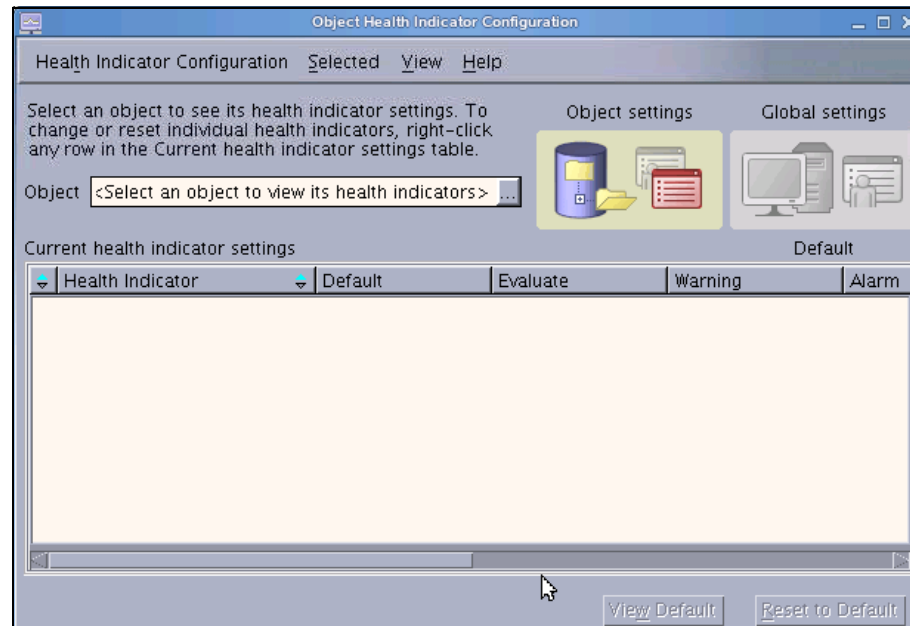


Figure 7-10 Object health indicator configuration

On the pop-up panel (Figure 7-11 on page 358), expanding the object tree until the desired object is shown, then select the object and click **OK**. Here we choose the DMS table space which is created beforehand in *Step 1* of this example.

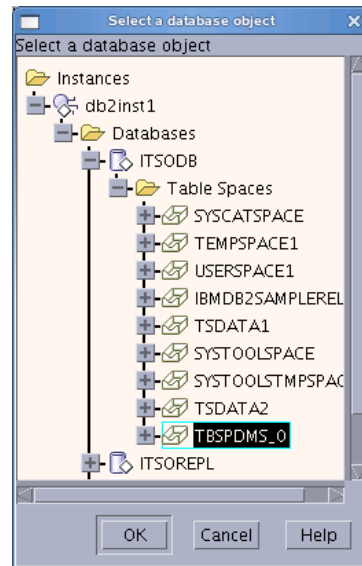


Figure 7-11 Select a database object

Once the object is chosen, you can modify the health indicators settings for the database objects. Figure 7-12 shows the default global setting from the table space level. In our example, a threshold value of 80 is set for a warning. Likewise, another threshold value, 90, is set for alarm. Both default values are expressed in percentage.

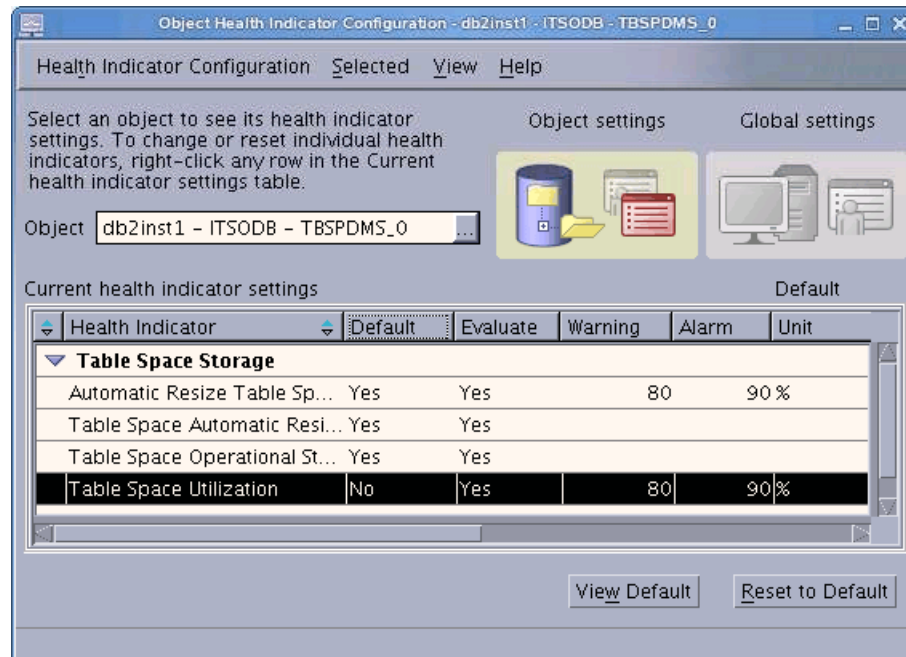


Figure 7-12 Health indicator settings in our sample

To configure a health indicator, double click the health indicator you want to change. For example, you can double click the health indicator for table space utilization to configure the settings for this health indicator. See Figure 7-13 on page 360.

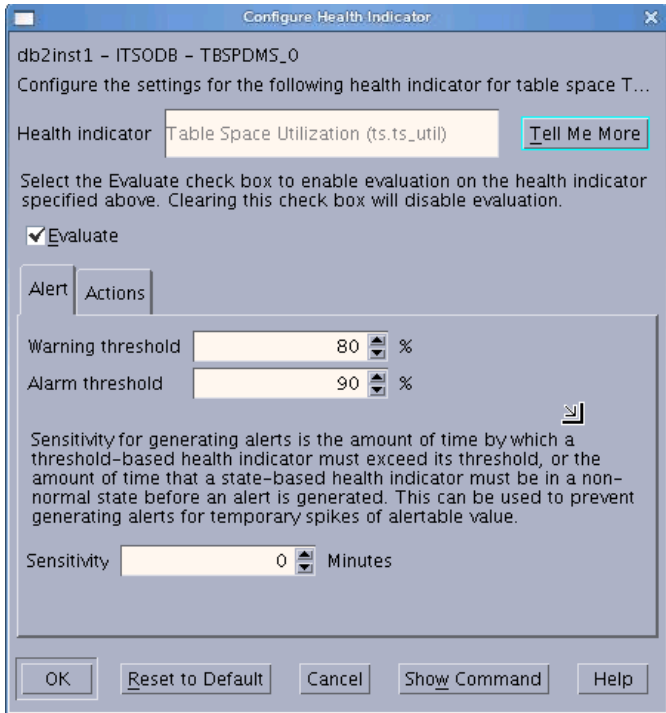


Figure 7-13 Configuring a health indicator for table space utilization

You can click **Show Command** to get the command based on options chosen. See Figure 7-14. You can run the command in DB2 CLP to change configurations for the health indicator, or click **OK** to submit your changes.

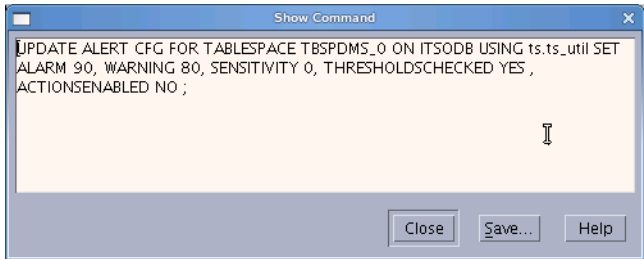


Figure 7-14 Set health indicator value command

You can use the DB2 CLP command **GET ALERT CFG** to verify the setting as shown in Example 7-3 on page 361.

Example 7-3 Using GET ALERT CFG to view health indicator status

```
db2inst1@mensa:/db2home/db2inst1> db2 "get alert cfg for tablespace
tbspdms_0 on itsodb"
```

Alert Configuration

```
Indicator Name           = ts.ts_util
Default                  = No
Type                     = Threshold-based
Warning                  = 80
Alarm                    = 90
Unit                     = %
Sensitivity               = 0
Formula                  =
((ts.ts_used_pages/ts.ts_usable_pages)*100);
Actions                  = Disabled
Threshold or State checking = Enabled

Indicator Name           = ts.ts_op_status
Default                  = Yes
Type                     = State-based
Sensitivity               = 0
Formula                  = ts.ts_status;
Actions                  = Disabled
Threshold or State checking = Enabled

Indicator Name           = ts.ts_auto_resize_status
Default                  = Yes
Type                     = State-based
Sensitivity               = 0
Formula                  = ts.ts_last_resize_fail;
Actions                  = Disabled
Threshold or State checking = Enabled

Indicator Name           = ts.ts_util_auto_resize
Default                  = Yes
Type                     = Threshold-based
Warning                  = 80
Alarm                    = 90
Unit                     = %
Sensitivity               = 0
Formula                  =
(((ts.ts_used_pages*ts.ts_page_size)/ts.ts_max_size)*100);
Actions                  = Disabled
Threshold or State checking = Enabled
```

3. Populate a desired amount of data into the table space.

We populate a certain amount of data into the table space to trigger the threshold. DB2 *common table expression* is used in our example to generate specified number of rows (Example 7-4). In our example 280,000 rows are inserted into the table and the threshold value of table space utilization will be exceeded.

Example 7-4 Populating data to trigger threshold for table space utilization

```
db2inst1@mensa:/db2home/db2inst1/script> more ins_empinfo.db2
INSERT INTO db2inst1.empinfo
WITH tmpinfo(ID) AS (VALUES(1) UNION ALL SELECT ID+1 FROM tmpinfo WHERE ID
< 250000 )
SELECT ID, TRANSLATE(CHAR(INTEGER(RAND()*1000000)),
CASE MOD(ID,4)
WHEN 0 THEN 'aeiou' || 'bcdfg'
WHEN 1 THEN 'aeiou' || 'hijklm'
WHEN 2 THEN 'aeiou' || 'npqrs'
ELSE 'aeiou' || 'twxyz' END,
'1234567890') AS LASTNAME,
CURRENT DATE - (RAND()*20000) DAYS AS HIREDATE,
INTEGER(10000+RAND()*200000) AS SALARY
FROM tmpinfo;
db2inst1@mensa:/db2home/db2inst1/script> db2 -tvf ins_empinfo.db2
INSERT INTO db2inst1.empinfo WITH tmpinfo(ID) AS (VALUES(1) UNION ALL
SELECT ID+1 FROM tmpinfo WHERE ID < 280000 ) SELECT ID,
TRANSLATE(CHAR(INTEGER(RAND()*1000000)), CASE MOD(ID,4) WHEN 0 THEN 'aeiou'
|| 'bcdfg' WHEN 1 THEN 'aeiou' || 'hijklm' WHEN 2 THEN 'aeiou' || 'npqrs'
ELSE 'aeiou' || 'twxyz' END, '1234567890') AS LASTNAME, CURRENT DATE
- (RAND()*20000) DAYS AS HIREDATE, INTEGER(10000+RAND()*200000) AS SALARY
FROM tmpinfo
DB20000I The SQL command completed successfully.
```

4. Check the notification sent by DB2 system.

Once DB2 health monitor detected that the threshold for warning or alarm has been exceeded, it will send warning or alarm messages to the DB2 administration notification log as well as specified contacts if you have configured notifications. In our example, the notification will be sent to notification log and to a contact point through e-mail.

We can use *DB2 Journal* to check the administration notification log. To start the DB2 Journal, select **Tools** → **Journal** in any DB2 Tools panel, where the Tools menu is available, or from Health Center, right-click on the instance name and select **Show Notification Log**. Then select the Notification Log tab, choose **Instance** to view the notification log. The instance in our example is db2inst1. The notification log for the instance is shown in the bottom pane of the panel. Use the *Notification Log Filter* to set up a filter to show health monitor notifications only. Figure 7-15 shows the notification log of our example. It shows that the value of current table space utilization is 95

percent and severity level is Alarm. The timestamp at that time the alarm condition was triggered is also shown.

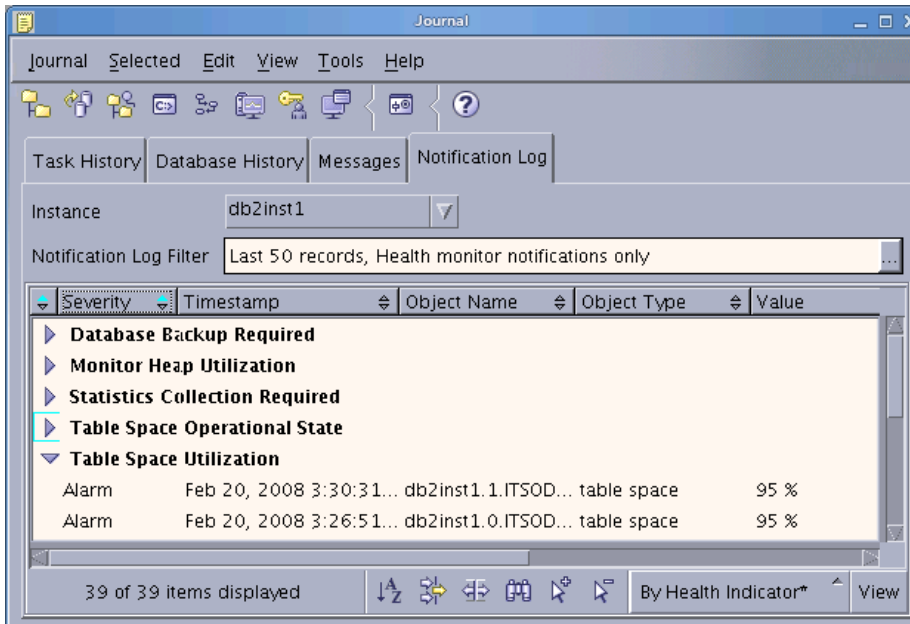


Figure 7-15 Administration notification log for health monitor indicator

An e-mail is sent to designated contact with content the one shown in Figure 7-16.

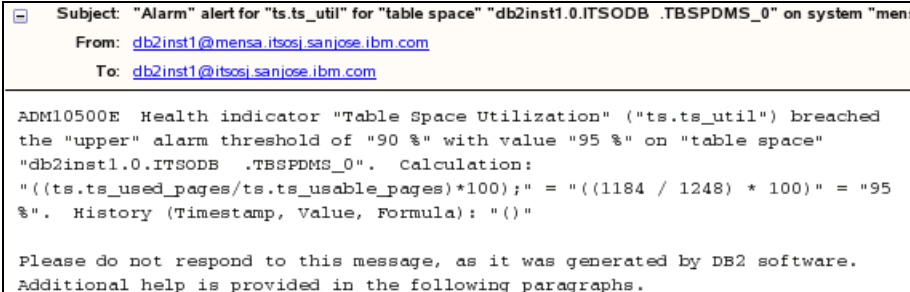


Figure 7-16 Notification for health indicator by e-mail

In addition to the notifications to the administration notification log and contact, there is more information regarding the alert available within Health Center. The Health Center GUI has red, yellow, orange, and green state indicator icons prefixed to the DB2 objects. The state icon provides you a quick view of system health condition. In Figure 7-17, you find that the red

circle icon prefixed to the instance name and database name in the left pane within the Health Center is displayed. It means alarm conditions happened within the instance or any of its databases or their table spaces and containers. On the right pane of the window, the Health Center lists all the alerts.

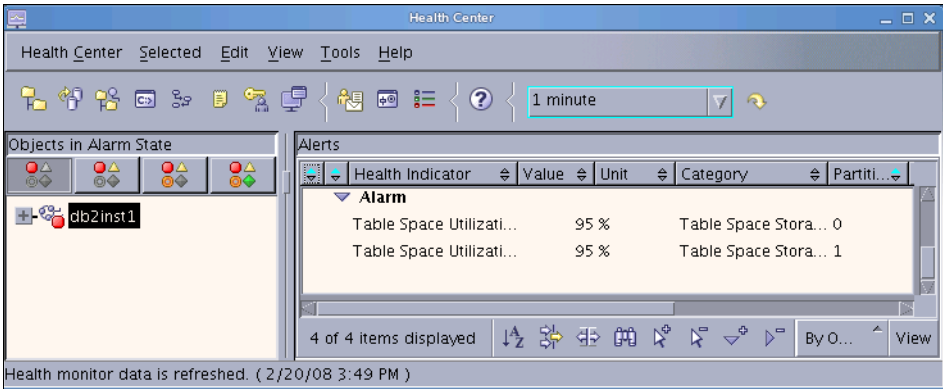


Figure 7-17 Monitor health indicators in Health Center

You can work on the alert directly from here. Select one of the alerts which corresponds to your desired health indicator, then right-click the alert entry. It provides you the Show details and Disable Evaluation options. To get detailed information, choose **Show Details**; to get the recommendation for this alert, choose **Recommendation Advisor**; to disable the health indicator choose **Disable Evaluation**. Figure 7-18 shows these options.

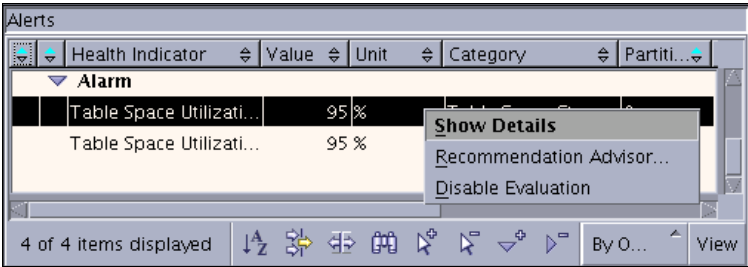


Figure 7-18 Choosing the options to handle an alert

If you chose *Show Details*, the following panel (Figure 7-19 on page 365) appears.

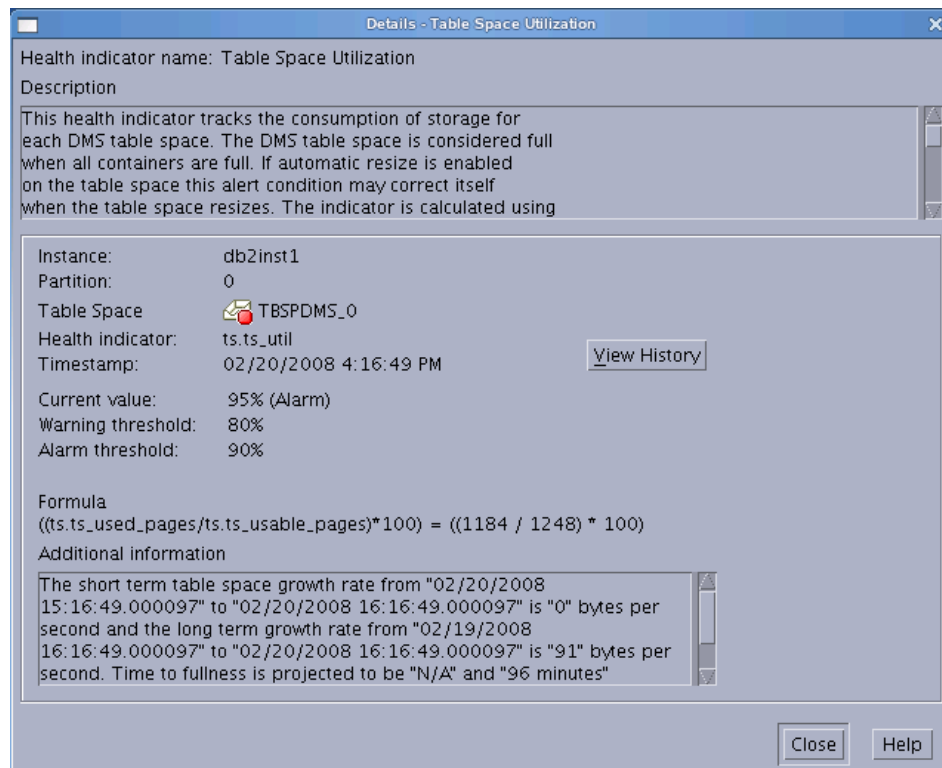


Figure 7-19 Detailed information for an alert within Health Center

5. Take corresponding action to resolve current health alerts.
 - a. Use Recommendation Advisor utility
 - i. If you choose Recommendation Advisor at previous panel to get the recommendation for this alert, the Recommendation Advisor utility will be launched. See Figure 7-20 on page 366. The first panel an introduction. You can review health alert details here. After reviewing health alert details, click **Next**.

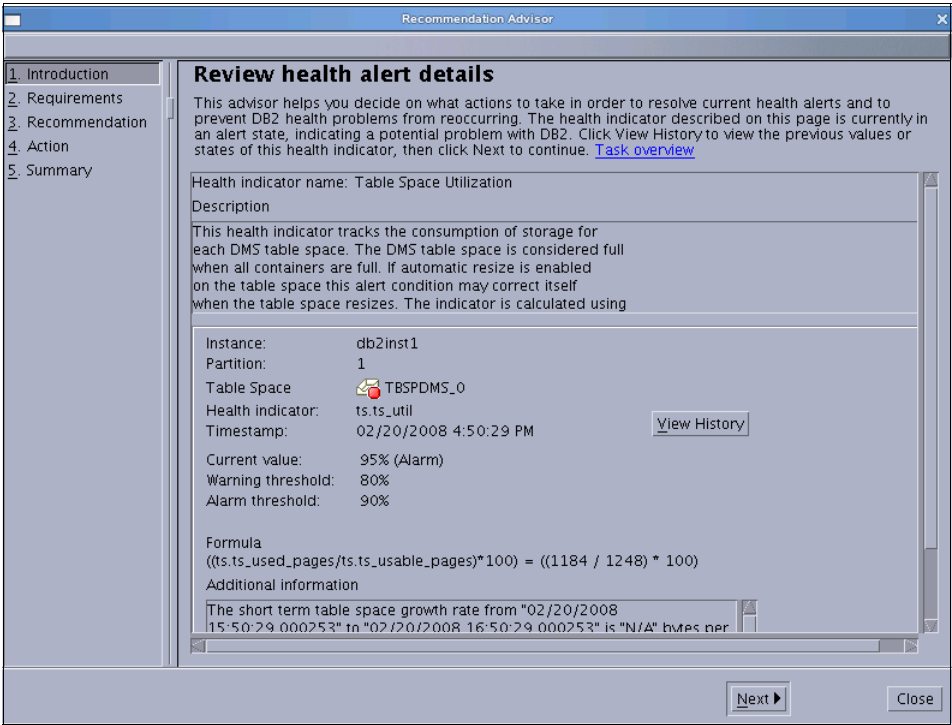


Figure 7-20 Recommendation Advisor introduction

- ii. On Specify your database requirements (Figure 7-21), specify your database requirement. In our sample, we choose **I would like an immediate solution to problem option**. Click **Next**.

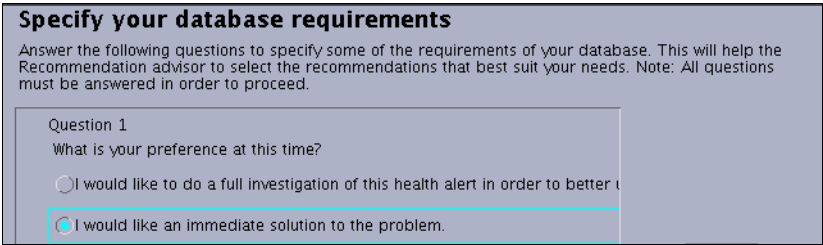


Figure 7-21 Specifying your database requirements

- iii. On Select a recommendation(Figure 7-22), select a recommendation from the Preferred recommendations list. We choose **Add new table space containers**. Click **Next**.

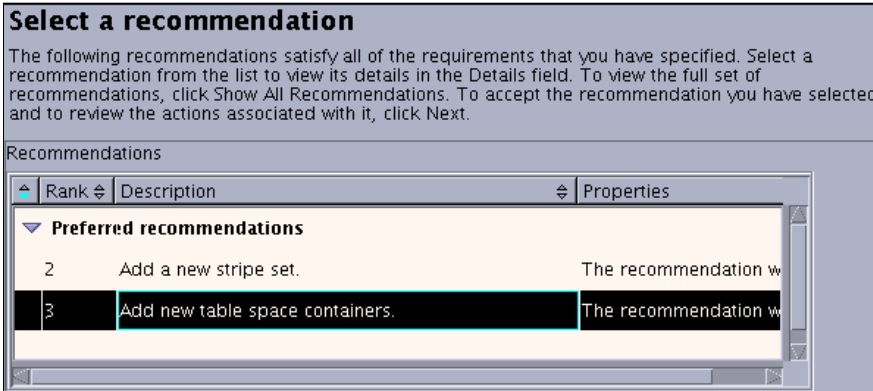


Figure 7-22 Selecting a recommendation

- iv. The Review the required actions panel shows (Figure 7-23). In our sample, we need to add containers in DB2 Control Center. Click **Launch** button and follow the *Usage instruction* to add containers in Control Center. After taking corresponding action. Click **Next**.

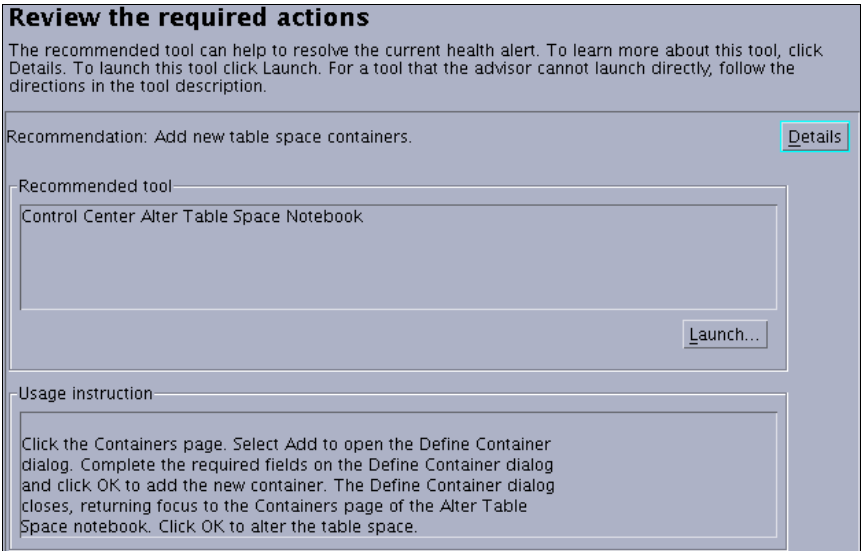


Figure 7-23 Reviewing the required actions and taking action

- v. On the Summary panel, click **Close** to quit Recommendation Advisor utility.

- b. Another way to get recommendations is using the DB2 CLP command **GET RECOMMENDATIONS** with specified health indicator. The following command can be used to get recommendation details for our example:

```
db2 get recommendations for health indicator ts.ts_util
```

Then the details of recommendations associated with the specified health indicator will be shown.

You can get the indicator name from the Health Center where you define properties for the health indicator. In general, the name, such as *ts.ts_util*, is listed in the Description field. For details about the indicators, refer to *System Monitor Guide and Reference*, SC23-5865-00.

- 6. Set up additional follow-up actions.

There are situations when you need a proactive action plan for the error condition. This can be accomplished by using a predefined script or task.

- a. To enable the predefined script or task, on the Object Health Indicator Configuration panel (Figure 7-12 on page 359), double click a specific health indicator. The Configure Health Indicator panel is shown (Figure 7-13 on page 360), select the **Actions** tab.
- b. In the Actions tab (Figure 7-24), select the *Enable actions* check box to enable the automatic running of the actions, then click **Add**.

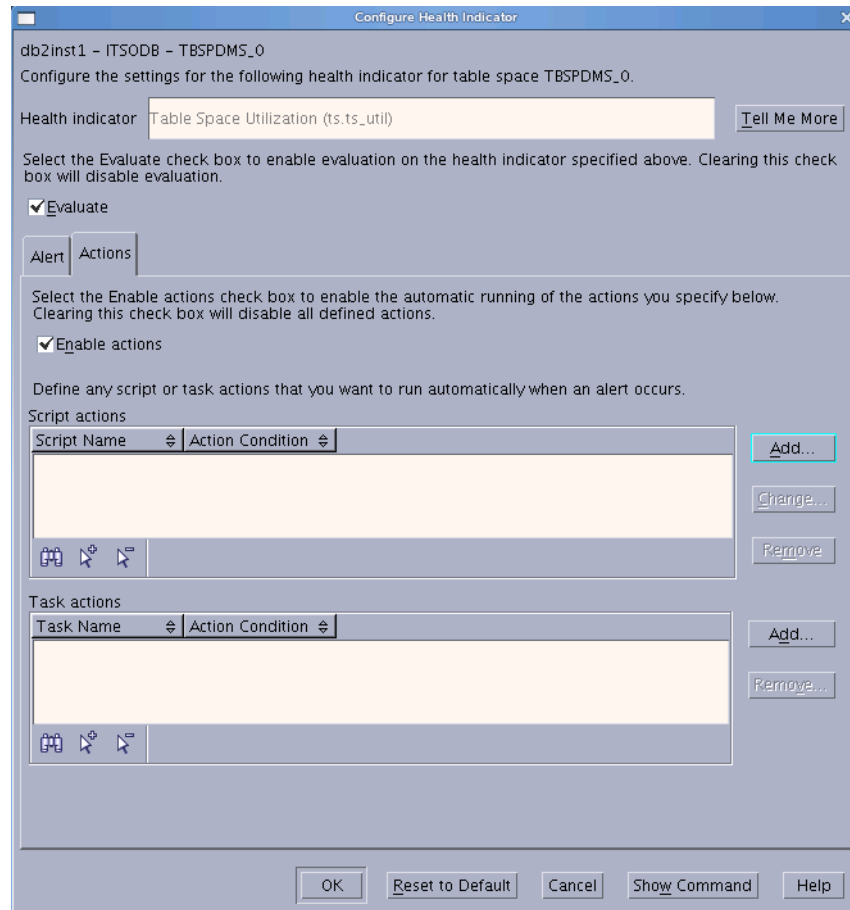


Figure 7-24 Enable actions in Configuring health indicator window

- c. On the pop-up panel as shown in Figure 7-25 on page 370, select the condition on which to run the action. In our sample, we choose *Alarm the threshold breach*. On the *Script* tab, choose system, a predefined script, and the type of script.

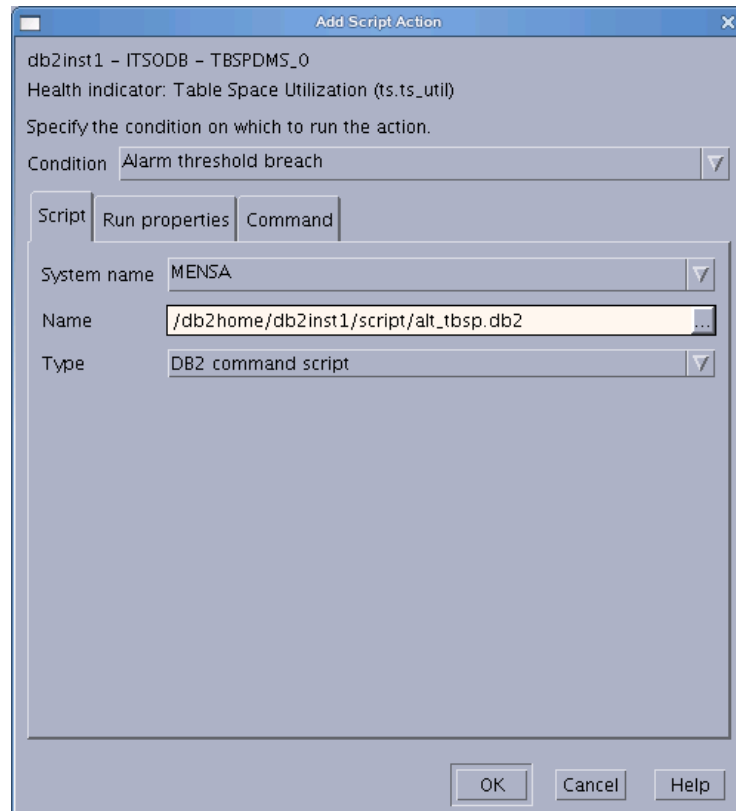


Figure 7-25 Add script action

- d. On the Run properties tab, as shown in Figure 7-26 on page 371, input the user ID and password for runtime authorization.

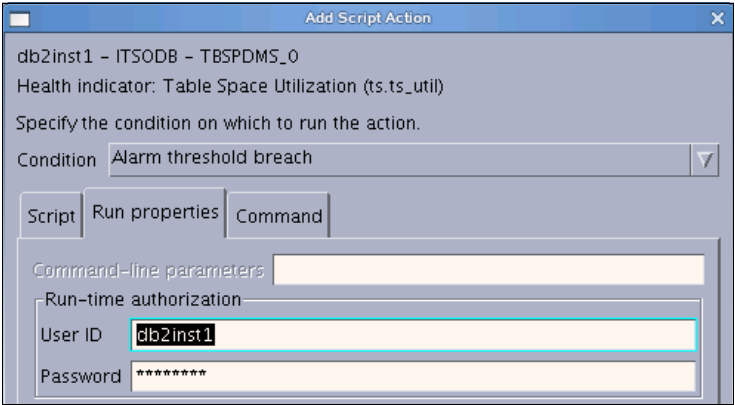


Figure 7-26 Runtime authorization

- e. On the Command tab, review the content of the predefined script, specify the script statement termination character and working directory. Click **OK**.

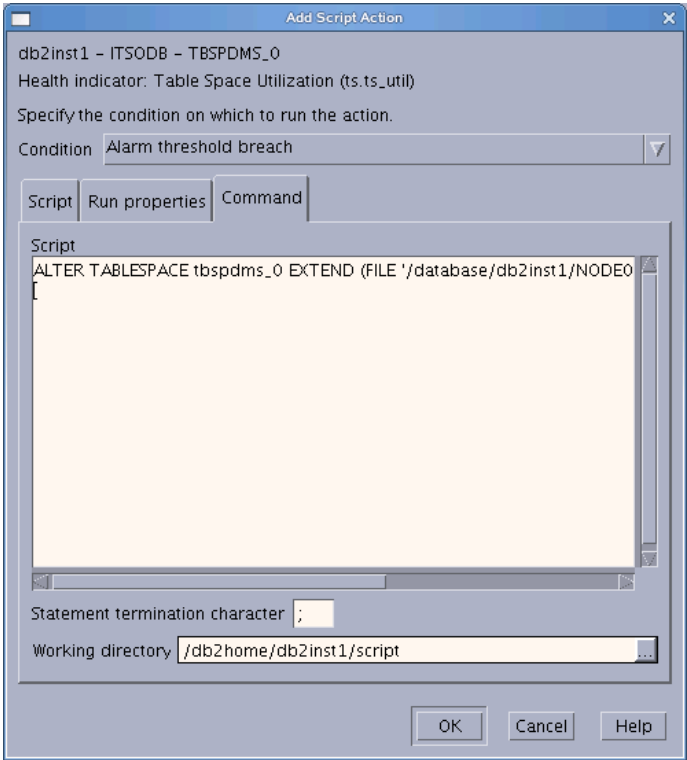


Figure 7-27 Review the content of the predefined script

Example 7-5 shows the content of the predefined script in our sample. The action is triggered by alarm condition of the health indicator *ts.ts_util* for table space *tbspdms_0*.

Example 7-5 Extend table space containers size

```
ALTER TABLESPACE tbspdms_0 EXTEND (FILE '/database/db2inst1/NODE000 $N
/ITSODB/T0000008/C0000000.LRG' 10M);
```

- f. On the Configuring health indicator panel (Figure 7-24 on page 369), click **OK** to finish the enabling actions.
- g. As shown in Example 7-6, you can use the DB2 CLP command **GET ALERT CFG** to verify the current setting.

Example 7-6 Using GET ALERT CFG to check current health indicator setting

```
db2inst1@mensa:/db2home/db2inst1> db2 "get alert cfg for tablespace
tbspdms_0 on itsodb"
```

Alert Configuration	
Indicator Name	= ts.ts_util
Default	= No
Type	= Threshold-based
Warning	= 80
Alarm	= 90
Unit	= %
Sensitivity	= 0
Formula	=
((ts.ts_used_pages/ts.ts_usable_pages)*100);	
Actions	= Enabled
Threshold or State checking	= Enabled
Script pathname	=
/db2home/db2inst1/script/alt_tbsp.db2	
Condition	= Alarm
Script type	= DB2
System	= MENSA
Working directory	= /db2home/db2inst1/script
Termination character	= ;
Userid	= db2inst1
.....	

- h. If we set up the action plan from the beginning, this script will be executed once we populated the data and the alarm for the table space is triggered, the follow-up corrective action is executed automatically without user intervention. As shown in Example 7-7, the output of LIST TABLESPACE SHOW DETAIL command shows that the total pages of the table space

TBSPDMS_0 has been extended to 20MB (8KB * 2560) on each partition after health monitor executing this predefined automatically.

Example 7-7 db2 list tablespaces show detail output

```
db2inst1@mensa:/db2home/db2inst1/script> db2_all "\"echo DB2NODE=##; db2
list tablespaces show detail"
```

DB2NODE=0

```

                Tablespaces for Current Database
.....
Tablespace ID          = 8
Name                   = TBSPDMS_0
Type                   = Database managed space
Contents               = All permanent data. Large table
space.
State                  = 0x0000
    Detailed explanation:
        Normal
Total pages             = 2560
Useable pages           = 2528
Used pages              = 1184
Free pages              = 1344
High water mark (pages) = 1184
Page size (bytes)       = 8192
Extent size (pages)     = 32
Prefetch size (pages)   = 32
Number of containers    = 1
Minimum recovery time   = 2008-02-21-22.46.03.000000
```

DB21011I In a partitioned database server environment, only the table spaces on the current node are listed.

mensa: db2 list tablespaces ... completed ok

DB2NODE=1

```

                Tablespaces for Current Database
.....
Tablespace ID          = 8
Name                   = TBSPDMS_0
Type                   = Database managed space
Contents               = All permanent data. Large table
space.
State                  = 0x0000
    Detailed explanation:
```

```
Normal
Total pages                = 2560
Useable pages              = 2528
Used pages                 = 1184
Free pages                 = 1344
High water mark (pages)   = 1184
Page size (bytes)         = 8192
Extent size (pages)       = 32
Prefetch size (pages)     = 32
Number of containers       = 1
Minimum recovery time      = 2008-02-21-22.46.03.000000

DB21011I  In a partitioned database server environment, only the table
spaces
on the current node are listed.

gemini: db2 list tablespaces ... completed ok
```

After corrective actions taking effect, the health indicator returns to the normal state and keep monitoring the system.

7.2 Memory Visualizer and Memory Tracker

In this section, we discuss two memory monitoring tools, Memory Visualizer and Memory Tracker.

Memory Visualizer

The *Memory Visualizer* can assist you in monitoring memory utilization for a DB2 instance. It uses visual displays and plotted graphs to help you understand memory components and their relationships to each other and helps you uncover and fix memory-related problems on a DB2 instance.

You can start Memory Visualizer in the Control Center by right-clicking an instance and selecting **View memory usage**. To start the Memory Visualizer from the command line, issue the **db2memvis** command. The Memory Visualizer panel appears as shown in Figure 7-28:

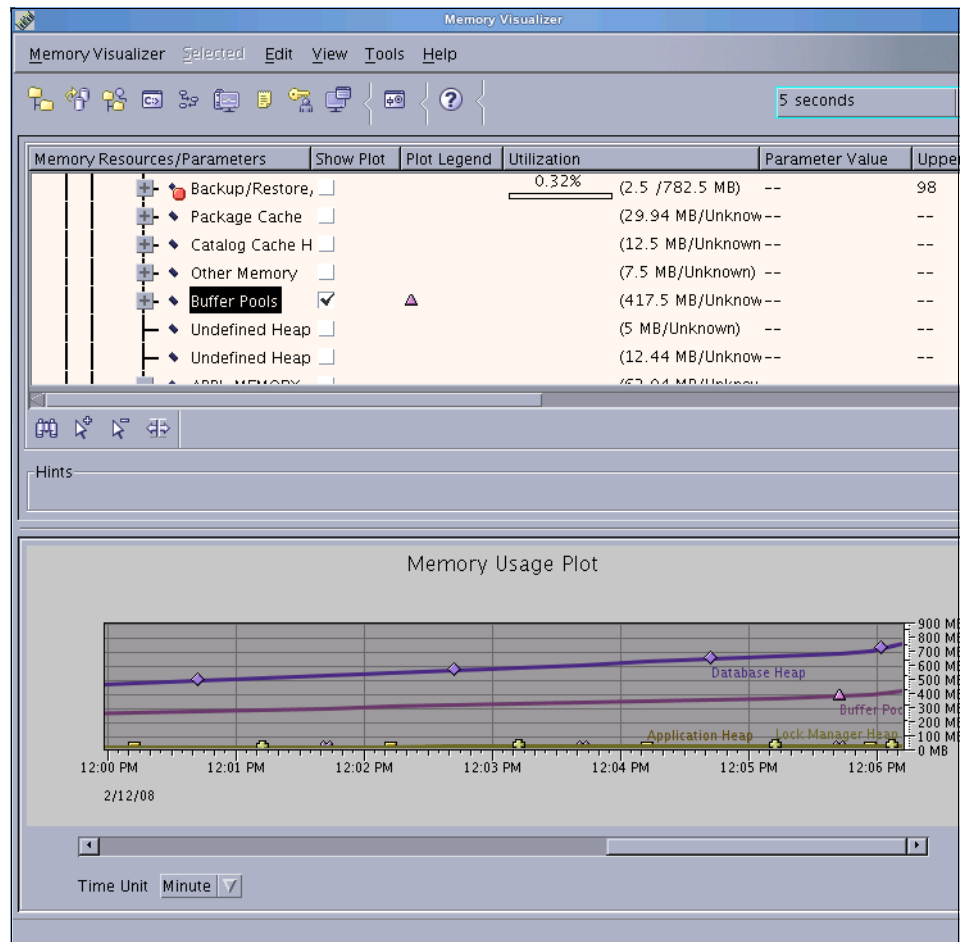


Figure 7-28 Using Memory Visualizer to monitor DB2 memory usage

Within this panel, you can:

- ▶ View memory usage for a specific memory heap or buffer, or view memory usage at the overall level.
- ▶ Specify which memory information to display and which information to hide for a DB2 instance and its databases in the Customize Columns panel, which can be displayed by choosing **View** → **Customize Columns**.
- ▶ Update the configuration parameters for an individual memory component to prevent it from using too much or too little memory — what you need to do is select the value in Parameter Value field, and then input a new one.
- ▶ Toggle which memory component will be displayed in the Memory Usage Plot graph, which is located at the bottom of Memory Visualizer main window.

- Save or load memory allocation data to or from a file for a Memory Visualizer window.

db2mtrk

DB2 also provides you a command line tool to obtain memory usage information for a DB2 instance. It is called Memory Tracker. The command name is **db2mtrk**. This command outputs the following memory pool allocation information:

- Current size
- Maximum size (hard limit)
- Largest size (high water mark)
- Type (identifier indicating function for which memory will be used)
- Agent who allocated pool (only if the pool is private)
- Application

You can obtain detailed help message for this tool by entering the following command in a Linux terminal:

```
db2mtrk -h
```

Example 7-8 shows you how to use db2mtrk to monitor memory usage by DB2 instances, databases and application.

Example 7-8 Using db2mtrk to monitor DB2 memory usage

```
db2inst1@mensa:/db2home/db2inst1> db2mtrk -i -d -a
Tracking Memory on: 2008/02/12 at 14:19:24
```

Memory for instance

other	monh	fcmbp
32.1M	192.0K	10.9M

No active databases

No active applications

```
db2inst1@mensa:/db2home/db2inst1> db2 connect to itsodb
```

Database Connection Information

Database server	= DB2/LINUX8664 9.5.0
SQL authorization ID	= DB2INST1
Local database alias	= ITSODB

```
db2inst1@mensa:/db2home/db2inst1> db2mtrk -i -d -a
Tracking Memory on: 2008/02/12 at 14:20:38
```

Memory for instance

other	monh	fcmbp
32.1M	320.0K	10.9M

Memory for database: ITS0DB

utilh	pckcacheh	other	catcacheh	bph (1)	bph (S32K)
64.0K	192.0K	192.0K	192.0K	8.2M	832.0K
bph (S16K)	bph (S8K)	bph (S4K)	shsorth	lockh	dbh
576.0K	448.0K	384.0K	0	640.0K	18.8M
apph (5231)	apph (5230)	apph (5229)	apph (5227)	appshrh	
64.0K	64.0K	64.0K	64.0K	128.0K	

Application Memory for database: ITS0DB

```
appshrh
128.0K
```

Memory for application 5230

apph	other
64.0K	192.0K

Memory for application 5229

apph	other
64.0K	192.0K

Memory for application 5227

apph	other
64.0K	576.0K

Memory for application 5231

apph	other
64.0K	320.0K

In the next example, as shown in Example 7-9 on page 378, the buffer pool size of IBMDEFAULTBP is changed from default 1,000 8K pages to 10,000 8K pages dynamically. You can monitor the change made by DB2 through db2mtrk very clearly. The first **db2mtrk -d** command can be used to obtain the buffer pool size for IBMDEFAULTBP before the change, which is highlighted in bold font in our example, the value of the "bph(1)" is 8.2 MB. Then you can use the ALTER BUFFERPOOL command to dynamically change the size of buffer pool IBMDEFAULTBP with the "immediate" parameter specified. After doing that, the

second **db2mtrk -d** command shown in the example can be used to obtain the current buffer pool size for IBMDEFAULTBP. It shows the size of the buffer pool is changed to 80.0 M (highlighted in bold) now.

Example 7-9 Using db2mtrk to monitor database memory usage

```
db2inst1@mensa:/db2home/db2inst1> db2mtrk -d
Tracking Memory on: 2008/02/12 at 14:28:38

Memory for database: ITSODB

      utilh      pckcacheh      other      catcacheh      bph (1)      bph (S32K)
      64.0K      384.0K      192.0K      192.0K      8.2M      832.0K

      bph (S16K)  bph (S8K)   bph (S4K)   shsorth      lockh      dbh
      576.0K      448.0K      384.0K      0            640.0K      18.8M

      apph (68339)apph (5231) apph (5230) apph (5229) apph (5227) appshrh
      64.0K      64.0K      64.0K      64.0K      64.0K      128.0K

db2inst1@mensa:/db2home/db2inst1> db2 "alter bufferpool ibmdefaultbp immediate size
10000"

db2inst1@mensa:/db2home/db2inst1> db2mtrk -d
Tracking Memory on: 2008/02/12 at 14:29:20

Memory for database: ITSODB

      utilh      pckcacheh      other      catcacheh      bph (1)      bph (S32K)
      64.0K      384.0K      192.0K      192.0K      80.0M      832.0K

      bph (S16K)  bph (S8K)   bph (S4K)   shsorth      lockh      dbh
      576.0K      448.0K      384.0K      0            640.0K      18.8M

      apph (68339)apph (5231) apph (5230) apph (5229) apph (5227) appshrh
      64.0K      64.0K      64.0K      64.0K      64.0K      128.0K
```

- Note:**
- 1. In a partitioned database environment, this command can be invoked from any database partition defined in the db2nodes.cfg file. It returns information only for current partition.
 - 2. You might have noticed that more than one “bph” is shown in the example for only one database; this is because, except the buffer pools created by users, there are four small hidden buffer pools created by DB2 to ensure that an appropriate buffer pool is available in all circumstances.
 - 3. With Version 9.5, the -p option (which lists private agent memory heaps) of the db2mtrk command is deprecated and has been replaced with the -a option (which lists all application memory consumption).

For a detailed explanation about command parameters and a field description of the output of db2mtrk, refer to *Command Reference*, SC23-5846-00.

7.3 db2top

The db2top monitoring utility, also known as IBM single system monitor view for DB2, monitors a DB2 environment on Linux platforms quickly and efficiently.

At the time of writing this book, the db2top version is 2.0 and the information about db2top is available on IBM AlphaWorks Web site:

<http://www.alphaworks.ibm.com/tech/db2top>

You can download the zipped file for db2top, *db2top-2.0-bin.zip*, from following Web site:

<http://www.alphaworks.ibm.com/tech/db2top/download>

7.3.1 db2top installation

Use these steps to install db2top:

1. Unzip db2top-2.0-bin.zip. The directory content of the unzipped file looks like:

```
2007-06-06 22:13          202,153 db2top User Manual.pdf
2007-06-07 00:00      2,995,973 db2top-2.0-bin.tar.gz
2007-07-19 14:43    <DIR>          license
2007-06-06 21:11    <DIR>          man
2007-06-06 21:13    <DIR>          samples
```

The *db2top User Manual.pdf* is the user manual and the *db2top-2.0-bin.tar.gz* contains executable files for all supported platforms.

2. Read the license agreement (HTML format) in *license* directory.

Note: This license agreement indicates that this program is still at *Early Release of Programs* stage at the time of writing this book. We recommend that you use db2top in development environment only.

3. If you accept this license agreement for Early Release of Programs, upload db2top-2.0-bin.tar.gz to your Linux system.
4. Enter the following commands to extract db2top executable files on Linux:

```
gunzip db2top-2.0-bin.tar.gz
tar xvf db2top-2.0-bin.tar
```

For DB2 9.5, 32-bit db2top executable file is placed in *linux_amd_32* directory and 64-bit db2top executable is placed in *linux_amd_64* directory.

7.3.2 Monitoring with db2top in interactive mode

db2top User Manual.pdf is the documentation for db2top usage. You also can obtain detailed online help by entering the following command in a Linux terminal:

```
db2top -h
```

A simple syntax of the db2top is as follows:

```
./db2top [-d dbname]
```

To start db2top, issue the following command in a Linux terminal:

```
./db2top -d ITSODB
```

Where, ITSODB is the database name in this book. The db2top starting window shows in Figure 7-29 on page 380.

You can press **h** to enter help panel. To get back to original page from help panel, press **Enter**.

We provide two samples in this section.

```
[~]14:13:34,refresh=2secs(0.005)
[d=Y,a=N,e=N,p=ALL]
Linux,part=[2/2],DB2INST1:ITS
[qp=0]

#####  #####  #####  #####  #####  #####  For help type h or ...
# # # # # # # # # # # # # # # # # # # # db2top -h: usage
# # # # # # # # # # # # # # # # # # # #
# # #####  #####  # # # # #####  Status: Active
# # # # # # # # # # # # # # # # # # # # Uptime: 04h:40m:35s
# # # # # # # # # # # # # # # # # # # # Last backup
#####  #####  #####  # # #####  # 2008/02/13 - 09:32:01

DB2 Interactive Snapshot Monitor V2.0
Use these keys to navigate:
d - Database          l - Sessions          a - Agent
t - Tablespaces       b - Bufferpools       T - Tables
D - Dynamic SQL       U - Locks            m - Memory
s - Statements        p - Partitions        u - Utilities
A - HADR              F - Federation        B - Bottlenecks
J - Skew monitor      q - Quit

Licensed Materials - Property of IBM
Copyright IBM Corp. 2005, 2006 All Rights Reserved.
db2top
```

Figure 7-29 db2top starting window

Monitoring buffer pool utilization

To monitor the buffer pool utilization, press **b**. By default this utility will refresh the display every two seconds. You can override the default refresh interval using the command option **-i** or press **i** after starting db2top. Figure 7-30 on page 381 shows a sample Buffer pool information screen.

By default, db2top issues global snapshots. If you want to monitor a specific partition, you can press **P** and then specify which partition you want to monitor. Note this applies all the monitoring, not just monitoring buffer pool utilization.

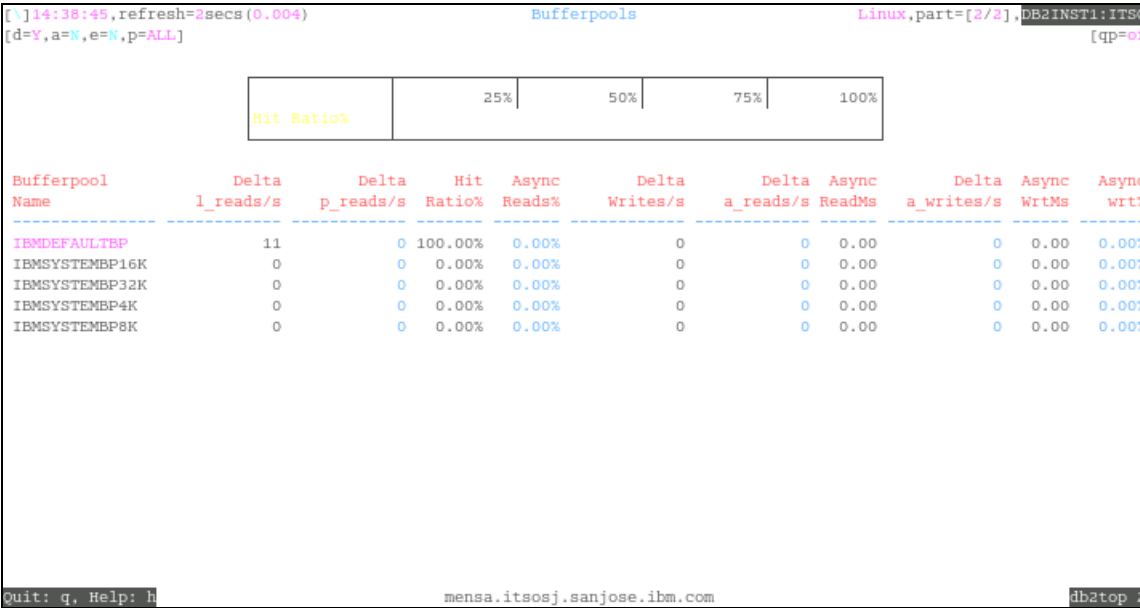


Figure 7-30 Monitoring buffer pool panel

Monitoring dynamic SQL

To monitor dynamic SQL, press **D**. In this panel (Figure 7-31 on page 382), you can view the number of times a SQL has been executed (Num Execution), average execution time, and average CPU time for a SQL statement.

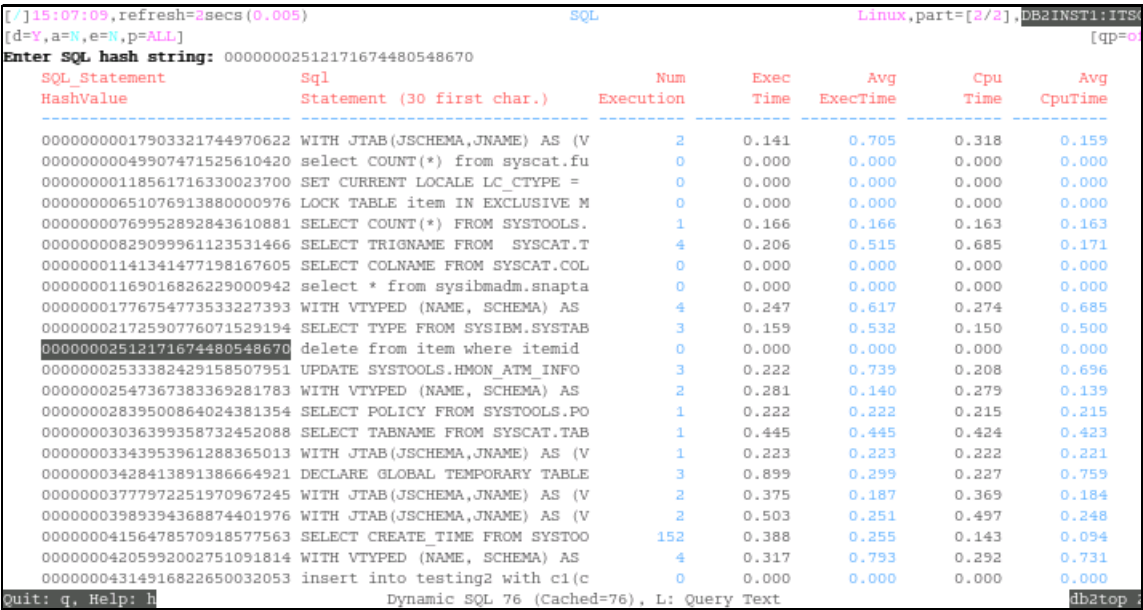


Figure 7-31 Monitoring dynamic SQL window

To check the SQL statement, in the Monitoring dynamic SQL panel, obtain the hash value of the query from the first column, press **L** and input the hash value. The SQL statement will be displayed as shown in Figure 7-32 on page 383.

You can press **e** to get db2expln utility output or press **x** to get db2exfmt utility output to analyze dynamic SQL execution plan.

In some cases, explain will fail if an object referenced by the query is not in the default schema. In this case, you can use the **-V** option in the command line to specify a default schema for explains.

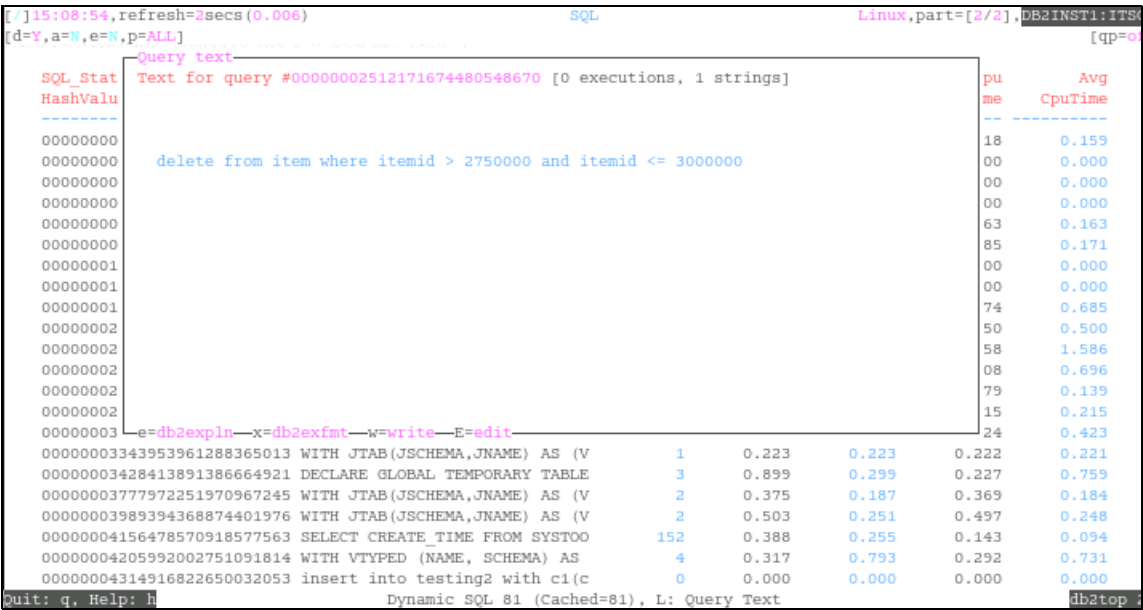


Figure 7-32 Query text window

7.3.3 Running db2top in the background mode

db2top background mode provides you the capability of automatic performance analysis or data collection for analysis using Microsoft Excel.

Analyzing the db2top data using Microsoft Excel

You can run db2top in background mode and analyze the data later using Microsoft Excel®. When using db2top with -b command option, db2top will display information in CSV format.

db2top provides an Excel report tool *db2top Reports* which is a macro for you to build graphical reports. db2top Reports macro is in the sample Excel file *db2top.xls* located in the *samples* directory after unzipping db2top-2.0-bin.zip.

To use db2top Reports, copy db2top.xls to a working directory on a Windows platform and double click it to open. Select *Enable Macros*. db2top Reports is presented for you to analyze db2top data, see Figure 7-33 on page 384.

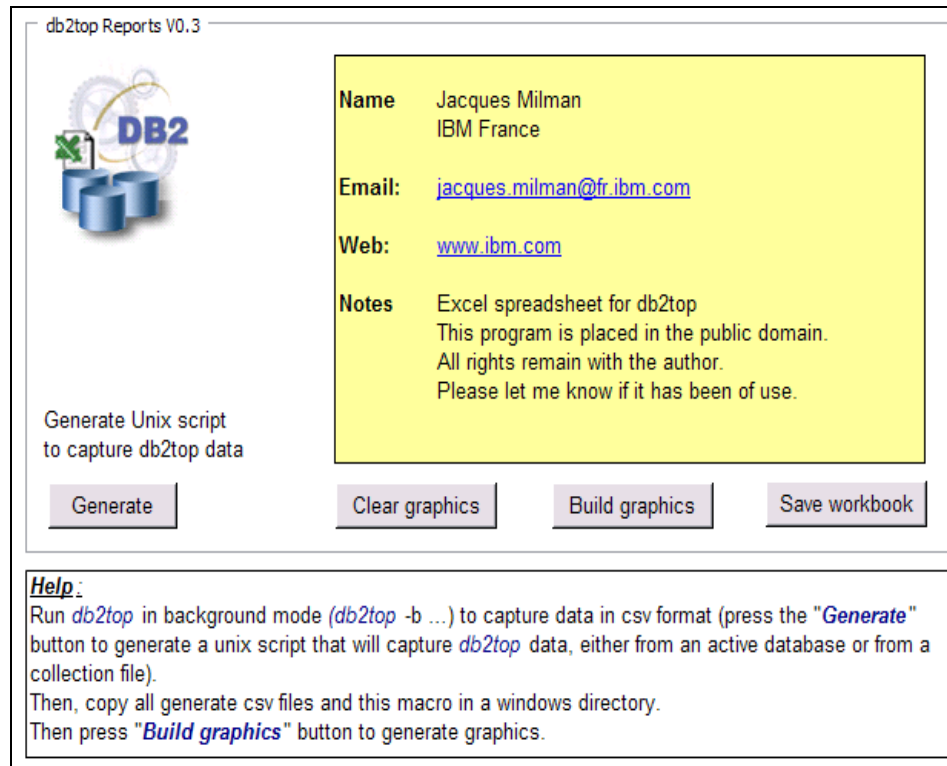


Figure 7-33 db2top Reports tool

Follow these steps to create graphical reports:

1. Generate an UNIX script to capture db2top data. This is an one time task. db2top Reports uses specific file format. You are required to use the script to produce the data files.

Click **Generate** on db2top Results tool panel. Example 7-10 shows the generated script.

Example 7-10 db2top.txt

```
#
# This script generates data for the db2top excel macro
# It will create a directory called /tmp/background
# Copy this directory under windows
# Copy the db2top excel macro in the directory
# Open the excel document and run full report
#
#
export WAITTIME=${1-5} # in minutes
```

```

export COLLECTFILE=${2} # in case there is a collection file
export DIR=/tmp/background
#
# Run db2top and exit after WAITTIME minutes
#
BACK_GROUND() {
db2top -m $WAITTIME -b $1 -o ${DIR}/db2top-${2}.csv &
# db2top -f $COLLECTFILE -m $WAITTIME -b $1 -o ${DIR}/db2top-${2}.csv &
}
#
# Start here
#
rm -fR $DIR
mkdir -p $DIR
echo "Running db2top for $WAITTIME minute(s)"
BACK_GROUND d DBM
BACK_GROUND l APPL
BACK_GROUND t TBS
BACK_GROUND b BP
BACK_GROUND T TABLES
BACK_GROUND m MEMORY
wait
echo "Done..."

```

Make the following modifications on db2top.txt:

- Combine BACK_GROUND definition to one line.
- Use **per1 -pi -e 's/;/,/g' \${DIR}/db2top*.csv &** to translate delimiter of CSV files from semicolon(;) to comma(,).

Note: This step is only required if your Excel setting uses semicolon(;) as the delimiter. Usually this Excel setting is determined by the Windows regional settings.

Example 7-11 shows an modified script.

Example 7-11 db2top.sh

```

#!/bin/sh
export WAITTIME=${1-5}
export DIR=/tmp/background
BACK_GROUND() { /software/db2top/linux_amd_64/db2top -m $WAITTIME -b $1 -o
${DIR}/db2top-${2}.csv & }
rm -fR $DIR
mkdir -p $DIR
echo "Running db2top for $WAITTIME minute(s)"
BACK_GROUND d DBM
BACK_GROUND l APPL

```

```

BACK_GROUND t TBS
BACK_GROUND b BP
BACK_GROUND T TABLES
BACK_GROUND m MEMORY
wait
perl -pi -e 's/;/,/g' ${DIR}/db2top*.csv &
echo "Done..."

```

2. Rename db2top.txt to db2top.sh and upload db2top.sh to a working directory on Linux. Execute db2top.sh as:

```

db2set DB2DBDFT=ITSODB
sh ./db2top.sh

```

Where, ITSODB represents the database which we monitor in our sample.

3. Use FTP to transmit all CSV format files generated in /tmp/background and to the working directory where db2top.xls exists in.
4. Click **Build graphics** to produce the graphical reports for analysis.
Figure 7-34 shows some sample reports.

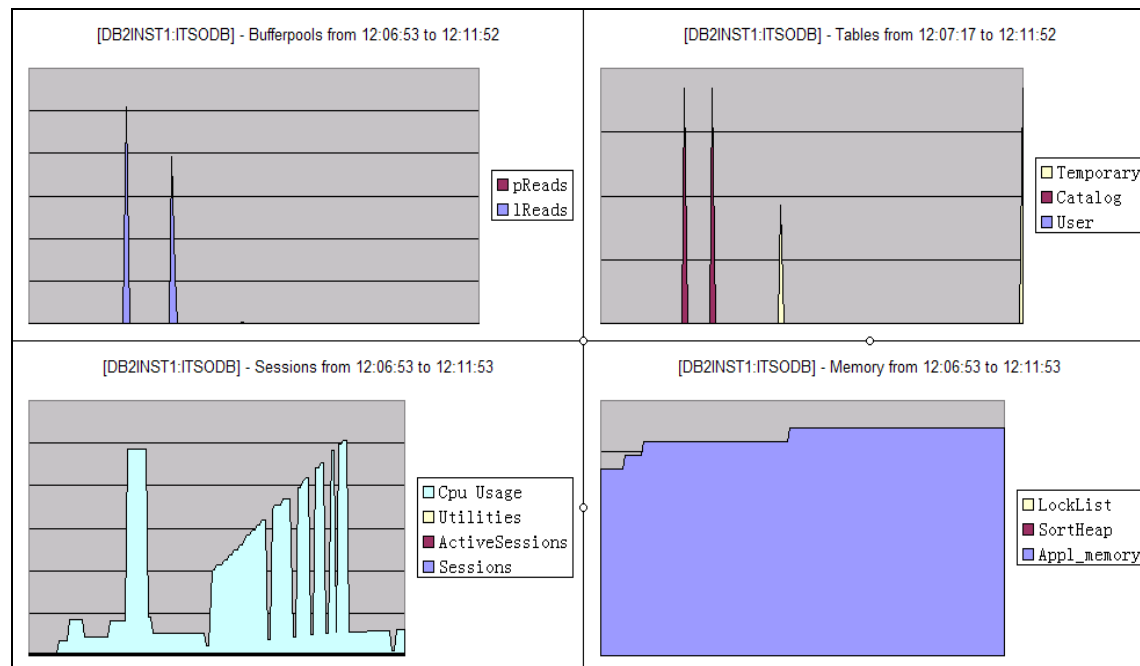


Figure 7-34 ALL_Reports in db2top.xlsb

Running db2top in snapshot collector mode

Another useful function of db2top is to run db2top in snapshot collector mode and have the system performance analyzed automatically.

You can use db2top with *-A* option to enable automatic performance analysis. db2top produces a report showing the top five performers. Automatic performance analysis is available for any functions supported in background mode. It is best to use this option in replay and background mode (*-b* option).

Valid sub-options for *-b* are:

- ▶ d : database
- ▶ l : sessions
- ▶ t : tablespaces
- ▶ b : bufferpools
- ▶ T : Tables
- ▶ D : Dynamic SQL
- ▶ s : Statements
- ▶ U : Locks
- ▶ u : Utilities
- ▶ F : Federation
- ▶ m : Memory pools

A typical usage of automatic performance analysis is as follows:

- ▶ Running db2top in collection mode for a period of time.

Example 7-12 shows running db2top in collector mode for 5 minutes in the first terminal session:

```
db2top -f collect.file -C -m 5
```

Example 7-12 Running db2top in collection mode for a period

```
db2inst1@mena:/software/db2top/linux_amd_64> db2set DB2DBDFT=ITSODB
db2inst1@mena:/software/db2top/linux_amd_64> ./db2top -f collect.filelect.file -C -m 5
[01:17:26] Starting DB2 snapshot data collector, collection every 2 second(s), max duration 5
minute(s), max file growth/hour 100.0M, hit <CTRL+C> to cancel...
[01:18:12] 1.0M written, time 45.865, 82.7M/hour
[01:18:54] 2.1M written, time 88.219, 89.7M/hour
[01:19:38] 3.4M written, time 132.572, 93.0M/hour
[01:20:27] 4.7M written, time 180.961, 95.1M/hour
[01:21:19] 6.2M written, time 233.386, 96.2M/hour
[01:22:18] 7.8M written, time 291.876, 97.4M/hour
[01:22:28] Max duration reached, 8.1M bytes, time was 301.956...
[01:22:28] Snapshot data collection stored in 'collect.file'
Exiting...
```

- ▶ Running db2top in replay mode with automatic performance analysis

To analyze the most active sessions, issue the following command in another terminal session while db2top is running in collection mode in the first terminal session. See As shown in Example 7-13.

```
db2top -f collect.file -b 1 -A
```

The analysis report will be generated when the job in the first terminal session is finished.

Example 7-13 Analyzing the most active sessions

```
db2inst1@mensa:/software/db2top/linux_amd_64> ./db2top -f collect.file -b 1 -A

Analyzing objects doing the most 'Cpu%_Total' in function 'Sessions'

*** End of input stream reached, size was 8514313...

--
-- Top twenty performance report for 'Sessions' between 01:17:40 and 01:22:26
-- Sort criteria 'Cpu%_Total'
--
Rank Application_Handle(Stat)      Percentage fromTime toTime      sum(Cpu%_Total)
-----
1 44930      35.4578% 01:17:40 01:22:26      673060
2 44947      64.5422% 01:17:40 01:22:26      1225140
3 65595      0.0000% 01:17:40 01:22:26      0
4 44934      0.0000% 01:17:40 01:22:26      0
5 83         0.0000% 01:17:40 01:22:26      0
6 82         0.0000% 01:17:40 01:22:26      0
7 81         0.0000% 01:17:40 01:22:26      0

--
-- Performance report, breakdown by 300 seconds
--
fromTime      sum(Cpu%_Total) Percentage      Top Five in 300 seconds interval
-----
01:17:40      1898200 100.0000%
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
Rank|Percentage|Application_Handle(Stat)
1|35.4578%|44930
2|64.5422%|44947
3|0.0000%|65595
4|0.0000%|44934
5|0.0000%|83

--
-- Performance report, breakdown by 0.5 hour
--
```


fromTime	sum(Cpu%_Total)	Percentage	Top Five in 0.5 hour interval		
01:17:40	1898200	100.0000%	Rank	Percentage	Application_Handle(Stat)
	-	-	1	35.4578%	44930
	-	-	2	64.5422%	44947
	-	-	3	0.0000%	65595
	-	-	4	0.0000%	44934
	-	-	5	0.0000%	83

Exiting...

7.4 Log files for troubleshooting

This section discusses DB2 and operating system log files which may provide more detailed information to help you find a problem or further determine the root cause of a problem. Afterwards you can take corrective actions based on the collected information.

7.4.1 DB2 administration notification log

The DB2 administration notification log contains user-friendly and national-language enabled messages that can help database administrators to obtain more information associated to specific issues, for example, information which is supplemental to an SQLCODE, information from health monitor for health indicators, information from task executions, and specific information from the applications written by application developers. This information can be very beneficial to the database administrators for resolving database or system related problems quickly and easily.

For the Linux platform, the notify log is a text file located in the directory specified by the database manager configuration parameter *DIAGPATH*². This also is the directory where db2diag.log resides. The name for notify log is <instance owner>.nfy. For example, /db2home/db2inst1/sqllib/db2dump/db2inst1.nfy, here the instance owner is db2inst1, and DIAGPATH is /db2home/db2inst1/sqllib/db2dump.

The Database manager parameter NOTIFYLEVEL can be utilized to specify the type of administration notification messages that are written into the administration notification log. Valid values for this parameter are:

² You may consider to change DIAGPATH to a directory on other file system than instance home's. Then the file system full problem of the diagnostic data directory will not impact your instance home.

- 0: No administration notification messages captured. (This setting is no recommended.)
- 1: Fatal or unrecoverable errors. Only fatal and unrecoverable errors are logged.
- 2: Immediate action required. Conditions are logged that require immediate attention from the system administrator or the database administrator. This level will capture health monitor alarms.
- 3: Important information, no immediate action required. Conditions are logged that are non-threatening and do not require immediate action but might indicate a non-optimal system. This level will capture health monitor alarms, health monitor warnings, and health monitor attentions.
- 4: Informational messages.

Messages written into notify log by setting a higher level include messages applicable to lower levels, for example, setting notify level to 3 will cause the administration notification log to include messages applicable to levels 1 and 2.

Example 7-14 is an excerpt the messages from notify log when lock escalation happens. It tells you when the lock escalation occurs, which instance, database and partition it takes place, the process ID and thread ID associated with the lock escalation, the application ID, component and function of DB2 used when the problem happens. In addition, detailed information specific to this occurrence of lock escalation is also displayed.

Example 7-14 DB2 administration notification log

```
2008-02-07-15.34.44.809250 Instance:db2inst1 Node:001
PID:10328(db2agntp (ITSODB) 1) TID:3506432320
Appid:*N0.db2inst1.080207233036
data management sqlEscalateLocks Probe:2 Database:ITSODB
```

```
ADM5500W DB2 is performing lock escalation. The total number of locks
currently held is "816", and the target number of locks to hold is "408".
^^
```

```
2008-02-07-15.34.44.859382 Instance:db2inst1 Node:001
PID:10328(db2agntp (ITSODB) 1) TID:3506432320
Appid:*N0.db2inst1.080207233036
data management sqlEscalateLocks Probe:3 Database:ITSODB
```

```
ADM5502W The escalation of "813" locks on table "DB2INST1.ITEM" to lock intent
"X" was successful.
```

7.4.2 DB2 diagnostic log (db2diag.log)

The Administration Notification log is the primary log file intended for use by database and system administrators. Administration notification log messages plus other details based on the setting are also logged to the db2diag.log using a standardized message format. The db2diag.log file is intended for use by DB2 Support for troubleshooting purposes and is the most often used file for DB2 problem investigation.

The file db2diag.log is located under the directory which is specified by the database manager configuration parameter *DIAGPATH* also. Another instance level parameter *DIAGLEVEL* is used to specify which type of diagnostic errors will be recorded in the db2diag.log. The following are valid values:

- 0: No diagnostic data captured
- 1: Severe errors only
- 2: All errors
- 3: All errors and warnings
- 4: All errors, warnings and informational messages

You may want to increase the value of this parameter to gather additional problem determination data to help resolve a problem. The default value of *DIAGLEVEL* is 3. If you changed it to 4 for some reason, remember to change it back to 3 or a lower value once the problem is resolved or the required information has been captured, because keeping *DIAGLEVEL* on a level of 4 will impact the system performance to a certain extent and the file system can fill up quickly.

As shown in Example 7-15, the system information is always recorded at the beginning of the db2diag.log file. Similar information is recorded while starting a instance.

Example 7-15 The beginning of the db2diag.log

```

2008-02-14-10.18.46.776339-480 I1E1283          LEVEL: Event
PID      : 32098          TID   : 47944161495904PROC : db2set
INSTANCE: db2inst1      NODE   : 000
FUNCTION: DB2 UDB, RAS/PD component, pdLogInternal, probe:120
START    : New Diagnostic Log file
DATA #1 : Build Level, 152 bytes
Instance "db2inst1" uses "64" bits and DB2 code release "SQL09050"
with level identifier "03010107".
Informational tokens are "DB2 v9.5.0.0", "s071001", "LINUXAMD6495", Fix Pack "0".
DATA #2 : System Info, 440 bytes
System: Linux mensa 6 2 x86_64
CPU: total:2 online:2 Cores per socket:2 Threading degree per core:1
Physical Memory(MB): total:3753 free:263
Virtual Memory(MB): total:11951 free:8455
Swap      Memory(MB): total:8198 free:8192

```

```

Kernel   Params: msgMaxMessageSize:65536 msgMsgMap:65536 msgMaxQueueIDs:1024
           msgNumberOfHeaders:65536 msgMaxQueueSize:65536
           msgMaxSegmentSize:16 shmMax:9223372036854775807 shmMin:1
           shmIDs:4096 shmSegments:4096 semMap:256000 semIDs:1024
           semNum:256000 semUndo:256000 semNumPerID:250 semOps:32
           semUndoSize:20 semMaxVal:32767 semAdjustOnExit:32767

```

Information in this record is only valid at the time when this file was created (see this record's time stamp)

Every time you use `db2set` to set or remove DB2 profile variables, DB2 records a message in `db2diag.log` when `DIAGLEVEL` is 1 or higher. As shown in Example 7-16, you will see a message after executing following command:

```
db2set DB2COMM=TCPIP
```

You may consider to use this method to check when you doubt that `db2diag.log` could not be written by DB2 engine.

Example 7-16 Message about setting DB2 profile variables

```

2008-02-14-10.18.46.710519-480 I1285E292          LEVEL: Event
PID       : 32098                TID  : 47944161495904PROC : db2set
INSTANCE: db2inst1              NODE  : 000
FUNCTION: DB2 UDB, oper system services, db2set_main, probe:40
CHANGE    : CFG DB2SET: DB2COMM: From: "TCPIP" To: "TCPIP"

```

Example 7-17 shows a log message extracted from `db2diag.log`. It is written into the log file when running the following DB2 command:

```
db2 update db cfg for ITS0DB using NEWLOGPATH /db2log
```

This message shows that `SQLCODE` is `5099` with tokens “`/db2log NEWLOGPATH 9`”, which are highlighted in bold in our example. Use `db2 ? sql5099` command to check the meaning of `SQL5099`. It tell us that reason code 9 means “*The path cannot be accessed*”. In our example, it is because the write access permission of `/db2log` is only assigned to the root user. The user `db2inst1` is not permitted to create a subdirectory under `/db2log`. After issuing the command `chown -R db2inst1.db2iadm1 /db2log` as root user, the problem is resolved.

Example 7-17 Message about setting NEWLOGPATH

```

2008-02-14-10.32.09.904898-480 I1578E825          LEVEL: Error
PID       : 25392                TID  : 47735688390976PROC : db2sysc 0
INSTANCE: db2inst1              NODE  : 000
APPHDL    : 0-7433              APPID: *N0.db2inst1.080214183209
AUTHID    : DB2INST1
EDUID     : 1129                EDUNAME: db2agent (instance) 0
FUNCTION: DB2 UDB, config/install, sqlfPdbUpdDbCfgReq, probe:923

```

```
DATA #1 : SQLCA, PD_DB2_TYPE_SQLCA, 136 bytes
sqlcaid : SQLCA      sqlcabc: 136      sqlcode: -5099      sqlerrml: 20
sqlerrmc: /db2log NEWLOGPATH 9
sqlerrp : SQL09050
sqlerrd : (1) 0x00000000      (2) 0x00000000      (3) 0x00000000
          (4) 0x00000000      (5) 0x00000000      (6) 0x00000000
sqlwarn : (1)      (2)      (3)      (4)      (5)      (6)
          (7)      (8)      (9)      (10)     (11)
sqlstate:
```

7.4.3 Operating system log

The Linux operating system also has some system utilities which provide support for system logging and kernel message trapping. Among those utilities, *syslogd* provides a kind of logging that many modern programs use. Every logged message by *syslogd* normally contains at least a time and a host name field plus a program name field, but that depends on how trusty the logging program is.

The *syslog.conf* file which is located in the */etc* directory is the main configuration file for the *syslogd*. It specifies rules for logging by *syslogd*. It is a text file and you can edit it with any favorable text editor to make it applicable to your environment, or simply use the default values which were set during Linux installation.

It is a good habit to check the system log file (the file could be */var/log/messages*, if you have not changed the default configuration) when you are facing problems that you are not sure if it is related to the system setting or not. For more information regarding how to set up *syslog.conf* and how to interpret the system log, refer to Linux-specific documentation.

Note: Instead of *syslogd* package, SUSE 10 installs *syslog-ng* package by default. The *syslog-ng.conf* file is the main configuration file and is located in */etc/syslog-ng* directory.

7.5 DB2 Tools for troubleshooting

DB2 product comes with several tools that can help you monitor and troubleshoot the DB2 system. We introduce some of these helpful tools and provide their usage examples.

7.5.1 db2diag

db2diag.log contains DB2 system running information and should be viewed periodically. You can use the *db2diag* analysis tool to filter and format the

db2diag.log file. As an alternative to using **db2diag**, you can use a text editor to view the diagnostic log file on the machine where you suspect a problem to have occurred. The most recent events recorded are the furthest down the file.

Note: The administration and diagnostic logs grow continuously. When they get too large, back them up and then erase the files. A new set of files is generated automatically the next time they are required by the system.

Here, we illustrate some examples on how to **db2diag**.

- Run **db2diag** command without any option:

```
db2inst1@mena:/> db2diag
```

If you use the **db2diag** command without any parameter, the tool process the db2diag.log file from the current directory and print it out to the screen. If the file is not found in the current directory, the directory set by the **DIAGPATH** variable is searched.

- Observe the db2diag.log file during runtime in a console mode:

To view the growth of the db2diag.log file, use

```
db2diag -f
```

This displays all records written to the db2diag.log up to now and the display continues until you press Ctrl-C to stop it.

- View logs in certain time interval:

To display all messages logged after timestamp 2008-02-14-10.00 inclusively:

```
db2diag -t 2008-02-14-10.00
```

To put all entries between 2008-02-14-12.00.30 and 2008-02-14 at 3 PM to an output file called db2diag_082014noon.log:

```
db2diag -t 2008-02-14-12.00.30:2008-02-14-15 -o db2diag_082014.log
```

The format for the date and time is YYYY-MM-DD-hh.mm.ss.nnnnnn, however, you do not need to use the full time stamp.

To display the entries of the last three days:

```
db2diag -H 3d
```

5. View logs for a specific database

If you have several databases in your DB2 instance and you want to see the db2diag.log entries only for one database, use:

```
db2diag -g db=ITS0DB
```

The -g option is a grep-like filtering capability for a comma separated list of field-pattern pairs. Another example is:

```
db2diag -g db!=SAMPLE,level=Event,pid=5971
```

It shows all events for process ID 5971 and not related to the SAMPLE database.

The field names are case-insensitive while search patterns (input fields) are case-sensitive. To get a list of all available field names, use **db2diag -h** command.

► **Partitioned database**

In a partitioned database environment, the `db2diag.log` file contains the entries for all database partitions. Every database partition writes their messages into the same file. You can filter the output as follows:

- To display only logged records for database partition 2:

```
db2diag -n 002
```

- To display all logged records containing only errors for database partitions 0 and 2:

```
db2diag -node "0, 2" -l "Error, warning"
```

► **Archive the `db2diag.log` file**

If you want to archive the current `db2diag.log` file and start a new one:

```
db2inst1@mena:/db2home/db2inst1> db2diag -A
db2diag: Moving "/db2home/db2inst1/sqllib/db2dump/db2diag.log"
to
"/db2home/db2inst1/sqllib/db2dump/db2diag.log_2008-02-14-16.36.57"
```

► **Format the `db2diag` output**

The following command formats the output of **db2diag** to only show the time stamp, database partition number, the message level, and the message text:

```
db2diag -fmt "Time: %{ts} Partition: %node Message Level: %{level} \n
Message: @{msg}\n"
```

► **DB2 internal error return codes**

To get the description about a DB2 internal error code 0x860F000A:

```
db2diag -rc 0x860F000A
```

To display a list of all DB2 ZRC return codes available, enter:

```
db2diag -rc zrc
```

The different options can be combined. There are a lot more options available. Use the command **db2diag -h** to see all options of **db2diag**. The output is shown in Example 7-18. To display more details and examples, use command **db2diag -h tutorial** for instance.

Example 7-18 Output of db2diag -help

```
db2inst1@mensa: /> db2diag -help
```

db2diag - The db2diag.log Analysis Tool

db2diag is a tool for filtering and formatting the db2diag.log file

Command syntax:

```

      .----- .-----
      v       |       v       |
>>--db2diag--+-----+-----+--><
              |         |         |
              --option-- --filename--

```

Command parameters:

filename	- one or more space-separated path names of diagnostic logs
-help , -h , ?	- help information. To get help on help, try "db2diag -h h"
-filter , -g	- case-sensitive search for a list of field-pattern pairs
-gi	- case-insensitive search for a list of field-pattern pairs
-gv	- case-sensitive invert matching
-gvi , -giv	- case-insensitive invert matching
-invert , -v	- invert the sense of matching for all filtering options
-exist	- record field must exist in order to be processed
-pid	- find all records for a list of process IDs
-tid	- find all records for a list of thread IDs
-eduid	- find all records for a list of EDU IDs
-node , -n	- find all records for a list of nodes
-error , -e	- find all records for a list of errors
-level , -l	- find all records for a list of severity levels
-history, -H	- display the history of logged records for a time interval
-time , -t	- display all the records within a particular time interval
-count , -c	- display a count of matching records
-verbose, -V	- display all record fields whether they contain data or not
-strict	- display records using one "field: value" pair per line
-cbe	- display records in the Common Base Event (CBE) format
-fmt	- format tool's output using a format string
-output , -o	- save output into a file
-follow , -f	- continuously display appended records as the file grows
-archive, -A	- archive a diagnostic log file
-readfile	- read from a file ignoring terminal input (used in scripts)
-rc	- display descriptions of DB2 error return codes, ZRC or ECF
-ecfid	- display function info extracted from the numeric ECF ID
-facility, -fac	- display messages from a particular facility

"db2diag -h <option1[,option2[,option3...]]>" - displays additional help and usage examples for one or more options specified in the options list


```

"db2diag -h brief"    - displays help for all options without examples

"db2diag -h examples" - displays a few typical examples to get started

"db2diag -h tutorial" - displays more advanced examples covering all features

"db2diag -h notes"    - displays usage notes and restrictions that apply

"db2diag -h all"      - displays help in the most complete form with detailed
                        information about all options and usage examples

```

More information can be find in *Troubleshooting Guide*, GI11-7857-00 and *Command Reference*, SC23-5846-00.

7.5.2 db2pd

The *db2pd* tool can be used for monitoring and troubleshooting DB2 problems. It collects information from the DB2 database system memory sets without acquiring any latches or using any engine resources, and returns the result very fast.

If you use the **db2pd** command without any option, **db2pd** starts in interactive mode and you can enter options. You also can run **db2pd** with option directly. Both are shown in Example 7-19.

Example 7-19 db2pd command prompt

```

db2inst1@mensa:/db2home/db2inst1> db2pd
db2pd> You are running db2pd in interactive mode.
db2pd> If you want command line mode, rerun db2pd with valid options.
db2pd> Type -h or -help for help.
db2pd> Type q to quit.
db2pd> -memsets

```

```
Database Partition 0 -- Active -- Up 0 days 23:19:41
```

Memory Sets:

Name	Address	Id	Size(Kb)	Key	DBP	Type
Unrsv(Kb)	Used(Kb)	Cmt(Kb)	Uncmt(Kb)			
DBMS	0x0000000200000000	412352519	29568	0xF197AE61	0	0
1280	7360	8000	21568			
FMP	0x0000000210000000	412385288	22592	0x0	0	2
0	320	22592	0			
Trace	0x0000000000000000	412319750	39227	0xF197AE74	0	-1
0	39227	39227	0			
FCM	0x0000000220000000	412418057	238592	0xF197AE62	0	11
204032	34560	34560	204032			

```

db2pd> q

```

```
db2inst1@mensa:/db2home/db2inst1> db2pd -osinfo
```

Operating System Information:

```
OSName:   Linux
NodeName: mensa
Version:  2
Release:  6
Machine:  x86_64
```

CPU Information:

TotalCPU	OnlineCPU	ConfigCPU	Speed(MHz)	HMTDegree
2	2	2	600	1 2

Physical Memory and Swap (Megabytes):

TotalMem	FreeMem	AvailMem	TotalSwap	FreeSwap
3753	209	n/a	8198	8192

Virtual Memory (Megabytes):

Total	Reserved	Available	Free
11951	n/a	n/a	8401

Message Queue Information:

MsgSeg	MsgMax	MsgMap	MsgMni	MsgTql	MsgMnb	MsgSsz
n/a	65536	65536	1024	65536	65536	16

Shared Memory Information:

ShmMax	ShmMin	ShmIds	ShmSeg
9223372036854775807	1	4096	4096

Semaphore Information:

SemMap	SemMni	SemMns	SemMnu	SemMsl	SemOpm	SemUme
SemUsz	SemVmx	SemAem				
256000	1024	256000	256000	250	32	n/a
20	32767	32767				

CPU Load Information:

Short	Medium	Long
0.160000	0.140000	0.080000

CPU Usage Information:

Total	Usr	Sys	Wait	Idle
0.080000	n/a	n/a	0.920000	n/a

You can set a default option for **db2pd** using the **DB2PDOPT** environment variable. In Example 7-20 we set the default to **-agents** to get information about running agents.

Example 7-20 Setting default option for db2pd

```
db2inst1@mena:/db2home/db2inst1> db2pd
db2pd> You are running db2pd in interactive mode.
db2pd> If you want command line mode, rerun db2pd with valid options.
db2pd> Type -h or -help for help.
db2pd> Type q to quit.
db2pd> q
```

```
db2inst1@mena:/db2home/db2inst1> export DB2PDOPT="-agents"
db2inst1@mena:/db2home/db2inst1> db2pd
```

Database Partition 0 -- Active -- Up 0 days 23:49:40

Agents:

```
Current agents:      6
Idle agents:         0
Active coord agents: 4
Active agents total: 4
Pooled coord agents: 2
Pooled agents total: 2
```

Address	AppHandl	[nod-index]	AgentEUID	Priority	Type	State
ClientPid	Userid	ClientNm	Rowsread	Rowswrtn	LkTmOt	DBName
0x000000020023D600	1054	[000-01054]	527	0	Coord	
Inst-Active 9045		db2inst1 db2bp	131	0	NotSet	ITSODB
0x000000020023C140	1056	[000-01056]	554	0	Coord	
Inst-Active 9045		db2inst1 db2taskd 2		0	NotSet	ITSODB
0x00000002007C0080	1057	[000-01057]	555	0	Coord	
Inst-Active 9045		db2inst1 db2wlm	0	0	NotSet	ITSODB
0x000000020023AC80	1058	[000-01058]	556	0	Coord	
Inst-Active 9045		db2inst1 db2evmg_	0	0	3	
ITSODB						
0x00000002007C1540	0	[000-00000]	557	0	Coord	Pooled
n/a	n/a	n/a	0	0	NotSet	ITSODB
0x00000002007C2A00	0	[000-00000]	558	0	Coord	Pooled
n/a	n/a	n/a	0	0	NotSet	ITSODB

We have shown **db2pd** with options, **-memsets**, **-osinfo**, and **-agents**. In the following, we provide some more examples of using the tool and what information can be retrieved:

- All database and instance-scope information

```
db2pd -inst -alldbs
```

- Active transactions

```
db2pd -db itsodb -transactions
```

- Locks

```
db2pd -db itsodb -locks
```

- A specific application: 1072 is a application handle.

```
db2pd -db itsodb -applications 1072
```

- Database logs

```
db2pd -db itsodb -logs
```

- Fenced vendor process

```
db2pd -db itsodb -fvp lam1
```

This applies to backup, restore, prune history, load, load copy (roll forward) and Log Manager, where a vendor media device is being used.

- Automatic storage paths defined for the database

```
db2pd -db itsodb -storagepaths
```

- Table and data partition reorganization

```
db2pd -db itsodb -reorgs
```

- Tables and indexes statistics

```
db2pd -db itsodb -tcbstats 7 4
```

In the example, the command returns the information for table ID 4 in table space ID 7.

The total number of updates on the table, the UPDATE, INSERT, DELETE (UDI), and real-time statistics UDI counters (RTSUDI) are returned as well. UDI is a counter to measure how much data has been changed since the last **RUNSTATS**. The value is used by DB2 to determine if the statistics for a table should be updated.

- List all EDUs in the instance

```
db2pd -edus
```

In DB2 Version 9.5, the implementation of the DB2 engine has evolved from a process based model to a thread based on Linux and UNIX systems. When invoking the **ps** command with the **-fu instancename** option, the output lists only two DB2 processes, db2sysc and db2acd. To display the individual threads associated with the db2sysc process, you must use the applicable thread options on the **ps** command. On Linux, you can use the **-lfp** option or use the **db2pd** command with the new **-edus** option.

As you can see from the our examples, the db2pd command is a very powerful tool for problem determination. To list all options, use **db2pd -h**.

Many more examples and information are in *Troubleshooting Guide*, GI11-7857-00 and *Command Reference*, SC23-5846-00.

7.5.3 DB2 snapshot command

Like the **db2pd** command, you can use the snapshot monitor to capture information about the database and any connected applications at a specific time. If you take it at regular intervals, they are also useful for observing trends and foreseeing potential problems. A number of different snapshot request types are available, each returning a specific type of monitoring data.

When using the command in a partitioned database environment, the snapshots can be taken at any partition of the instance, or globally. A global snapshot aggregates the data collected at each partition and returns a single set of values. Example 7-21 has two samples showing the difference.

Example 7-21 global snapshot

```
db2inst1@mena:/> db2_all 'db2 get snapshot for all on itsodb | grep "Locks
held currently"'

Locks held currently                                = 1
mena: db2 get snapshot for ... completed ok

Locks held currently                                = 8
gemini: db2 get snapshot for ... completed ok

db2inst1@mena:/s> db2 get snapshot for all on itsodb global | grep "Locks held
currently"
Locks held currently                                = 9

db2inst1@mena:/> db2_all 'db2 get snapshot for all on itsodb | grep "Lock
list"'

Lock list memory in use (Bytes)                      = 5888
mena: db2 get snapshot for ... completed ok

Lock list memory in use (Bytes)                      = 3392
gemini: db2 get snapshot for ... completed ok

db2inst1@mena:/> db2 get snapshot for all on itsodb global | grep "Lock list"
Lock list memory in use (Bytes)                      = 9280
```

In 6.2.2, “Manual table maintenance” on page 275, we used the **GET SNAPSHOT** command from DB2 command line (CLP) to monitor the inplace table reorganization. You can also capture a snapshot from administrative views, SQL table functions, or by using the snapshot monitor APIs in a C or C++ application.

There are several *snapshot administrative views* available for you to query the snapshot information. Example 7-22 shows the list.

Example 7-22 Snapshot administrative views

```
db2inst1@mena:/ db2 "select substr(tabschema,1,10) tabschema,
substr(tabname,1,30) tabname from syscat.tables where tabname like 'SNAP%'"
```

TABSCHEMA	TABNAME
-----	-----
SYSIBMADM	SNAPAGENT
SYSIBMADM	SNAPAGENT_MEMORY_POOL
SYSIBMADM	SNAPAPPL
SYSIBMADM	SNAPAPPL_INFO
SYSIBMADM	SNAPBP
SYSIBMADM	SNAPBP_PART
SYSIBMADM	SNAPCONTAINER
SYSIBMADM	SNAPDB
SYSIBMADM	SNAPDBM
SYSIBMADM	SNAPDBM_MEMORY_POOL
SYSIBMADM	SNAPDB_MEMORY_POOL
SYSIBMADM	SNAPDETAILLOG
SYSIBMADM	SNAPDYN_SQL
SYSIBMADM	SNAPFCM
SYSIBMADM	SNAPFCM_PART
SYSIBMADM	SNAPHADR
SYSIBMADM	SNAPLOCK
SYSIBMADM	SNAPLOCKWAIT
SYSIBMADM	SNAPSTMT
SYSIBMADM	SNAPSTORAGE_PATHS
SYSIBMADM	SNAPSUBSECTION
SYSIBMADM	SNAPSWITCHES
SYSIBMADM	SNAPTAB
SYSIBMADM	SNAPTAB_REORG
SYSIBMADM	SNAPTbsp
SYSIBMADM	SNAPTbsp_PART
SYSIBMADM	SNAPTbsp_QUIESCER
SYSIBMADM	SNAPTbsp_RANGE
SYSIBMADM	SNAPUTIL
SYSIBMADM	SNAPUTIL_PROGRESS

30 record(s) selected.

Each snapshot view returns a table with one row per monitored object per database partition with each column representing a monitor element. In Example 7-23, we capture a snapshot of database information for the ITS0DB database using the SNAPDB administrative view.

Example 7-23 Database snapshot using SNAPDB administrative view

```
db2inst1@mena:/> db2 "select snapshot_timestamp, db_status,
substr(db_name,1,10) dbname, dbpartitionnum from sysibmadm.SNAPDB"
```

```
dbpartitionnum from sysibmadm.SNAPDB" <
```

SNAPSHOT_TIMESTAMP	DB_STATUS	DBNAME	DBPARTITIONNUM
2008-02-15-17.42.32.773025	ACTIVE	ITSODB	0
2008-02-15-17.42.29.024832	ACTIVE	ITSODB	1

2 record(s) selected.

You can capture the same information using table functions. Each table function returns a table with one row per monitored object for the specified database partition. The column names of the returned table correlate with the monitor element names.

In Example 7-24 we capture the same snapshot of database information as in Example 7-23 using the SNAPDB administrative view, but this time we use the table function SNAP_GET_DB_V95.

Example 7-24 Getting snapshot using table function

```
db2inst1@mensa: /> db2 "SELECT SNAPSHOT_TIMESTAMP, DB_STATUS,
substr(DB_NAME,1,10) DBNAME, DBPARTITIONNUM FROM
TABLE(SNAP_GET_DB_V95('ITSODB'))"
```

SNAPSHOT_TIMESTAMP	DB_STATUS	DBNAME	DBPARTITIONNUM
2008-02-15-17.58.26.933775	ACTIVE	ITSODB	0
2008-02-15-17.58.25.915850	ACTIVE	ITSODB	1

2 record(s) selected.

To list all available snapshot table functions, use the following command:

```
SELECT SUBSTR(FUNCSHEMA,1,10) FUNCSHEMA, SUBSTR(FUNCNAME,1,30) FUNCNAME FROM
SYSCAT.FUNCTIONS WHERE FUNCNAME LIKE 'SNAP%'
```

The SQL table functions have two input parameters:

- ▶ Database name:
If you enter NULL, the name of the currently connected database is used.
- ▶ Database partition number:
To capture a snapshot for the currently connected database partition, enter a value of -1. To capture a global aggregate snapshot, enter a value of -2. To capture a snapshot from all database partitions as seen in Example 7-24, do not specify a value for this parameter.

Note: For a detailed overview of snapshot administrative views and table functions, refer to *System Monitor Guide and Reference*, SC23-5865-00.

7.5.4 db2ls

With the ability to install multiple copies of DB2 products on your system and the flexibility to install DB2 products and features in the path of your choice, you need a tool to help you keep track of what is installed and where it is installed. The *db2ls* command lists the DB2® products and features installed on your Linux and UNIX systems, including the DB2 Version 9 HTML documentation. You cannot query DB2 products using the Linux native **rpm** command.

The **db2ls** command can be used to list:

- ▶ Where DB2 products are installed on your system and list the DB2 product level.
- ▶ All or specific DB2 products and features in a particular installation path.

Example 7-25 shows you the output of the **db2ls** command from our environment.

Example 7-25 Output of db2ls command

```
db2inst1@mensa:/db2home/db2inst1> db2ls
```

Install Path	Level	Fix Pack	Special	Install Number	Install Date	Installer UID

-						
/opt/ibm/db2/V9.5	9.5.0.0	0			Fri Jan 25 15:52:16 2008 PST	0
/opt/ibm/db2/V9.1	9.1.0.3	3			Tue Jan 29 09:35:07 2008 PST	0

It lists the path where DB2 products are installed on our system and the DB2 product level. To get more information for each DB2 product installed on the system, you can use the commands as shown in Example 7-26.

Example 7-26 product information

```
db2inst1@mensa:/db2home/db2inst1> db2ls -q -p -b /opt/ibm/db2/V9.5
```

Install Path: /opt/ibm/db2/V9.5

Product Response File ID	Level	Fix Pack	Product Description

ENTERPRISE_SERVER_EDITION	9.5.0.0	0	DB2 Enterprise Server Edition

```
db2inst1@mensa:/db2home/db2inst1> db2ls -q -b /opt/ibm/db2/V9.5
```

Install Path: /opt/ibm/db2/V9.5

Feature Response File ID	Level	Fix Pack	Feature Description
BASE_CLIENT	9.5.0.0	0	Base client support
JAVA_SUPPORT	9.5.0.0	0	Java support
SQL_PROCEDURES	9.5.0.0	0	SQL procedures
BASE_DB2_ENGINE	9.5.0.0	0	Base server support
JDK	9.5.0.0	0	IBM Software Development Kit (SDK) for Java(TM)
CONNECT_SUPPORT	9.5.0.0	0	Connect support
COMMUNICATION_SUPPORT_TCPIP	9.5.0.0	0	Communication support - TCP/IP
REPL_CLIENT	9.5.0.0	0	Replication tools
CONTROL_CENTER	9.5.0.0	0	Control Center
DB2_DATA_SOURCE_SUPPORT	9.5.0.0	0	DB2 data source support
LDAP_EXPLOITATION	9.5.0.0	0	DB2 LDAP support
INSTANCE_SETUP_SUPPORT	9.5.0.0	0	DB2 Instance Setup wizard
XML_EXTENDER	9.5.0.0	0	XML Extender
APPLICATION_DEVELOPMENT_TOOLS	9.5.0.0	0	Base application development tools
FIRST_STEPS	9.5.0.0	0	First Steps
DB2_SAMPLE_DATABASE	9.5.0.0	0	Sample database source

7.5.5 db2support

The **db2support** utility is designed to automatically collect all DB2 and system diagnostic data. This program generates information about a DB2 server, including information about its configuration and system environment. The output of this program is stored in one compressed file named `db2support.zip`, located in the directory specified as part of the command invoked under command line. In one simple step, the tool can gather database manager snapshots, configuration files, and operating system parameters, which should make the problem determination quicker.

You can use the command **db2support -h** to display the complete list of command options. The following basic invocation is usually sufficient for collecting most of the information required to diagnosis a problem:

```
db2support outputpath -d dbname -c
```

For example, execute the following command to collect information about database ITSODB:

```
db2support . -d ITSODB -c
```

The zipped output file `db2support.zip` is in the current directory. You can send this file to your DB2 technical service representative for analysis.

Note: If the `-c` option is used, the utility will establish a connection to the database.

7.6 Linux system monitoring tools

There are many Linux system monitoring tools available to assist you in identifying where the performance or system setting issue occurs. These tools can be used to monitor system resource usage such as disk I/O, memory consumption, CPU activities and network status. Through the utilization of comprehensive information provided by these operating system tools, combining database system monitoring tools from DB2, you can understand your system more clearly; for example, which table space containers are under high disk I/O pressure, whether or not the excessive sort heap allocation will lead to a mass of paging space activities, if the network bandwidth is the bottleneck for poor response time of client applications, and so on. Then you can make pertinent adjustments to your applications, database system, or operating system to improve the system performance.

In this section, we discuss some of the most commonly used performance monitoring tools for the Linux platform: `top`, `iostat`, `vmstat`, and `sar`.

7.6.1 `top`

The **`top`** utility is a very useful tool that gives you a lot of information on one screen. `Top` can show you which process is taking the longest processor time, the largest amount of memory (both in percentages), plus how long the system has been up, and the amount of free memory. By default the utility will refresh the display every 3.0 seconds. You can override the default refresh interval via the command option `-d`. The users should be aware of that the `top` command does not report total memory used for DB2 correctly. `Top` will overstate memory usage by counting shared memory segment multiple times.

Note: In Linux, DB2 will not free shared memory automatically. To free the shared memory, you need to recycle DB2.

Some frequently used command line options for `top` are listed in Table 7-1.

Table 7-1 Frequently used command options for `top`

Command option	Description
d	Specifies the delay between screen updates.

Command option	Description
p	Monitor only processes with given process id. This flag can be given up to twenty times.
i	Start top ignoring any idle or zombie processes.
c	Displays command line instead of the command name only.
b	Batch mode. Useful for sending output from top to other programs or to a file. In this mode, top will not accept command line input. It runs until it produces the number of iterations requested with the n option or until killed. Output is plain text suitable for display on a dumb terminal.

If you are not using batch mode to run top, then when the output screen of top displays, you can use the interactive command to control the display. For example, you can toggle off or on the display of CPU states, or sort the processes list by the memory occupancy percentage, or kill a process in the list. Table 7-2 shows some commonly used commands.

Table 7-2 Frequently used interactive commands for top

Commands	Description
space	Immediately updates the display.
h or ?	Displays a help screen giving a brief summary of commands, and the status of secure and cumulative modes.
n or #	Changes the number of processes to display. You will be prompted to enter the number.
q	Quit.
f or F	Adds fields to display or remove fields from the display.
c	Toggles display of command name or full command line.
A	Sorts tasks by age (newest first).
P	Sorts tasks by CPU usage (default).
M	sort tasks by resident memory usage.
W	Write current setup to ~/.toprc. This is the recommended way to write a top configuration file.

Example 7-27 shows a interactive screen of top where some interactive commands are used. For example, press “M” to re-order the processes list by resident memory usage (by default, it is ordered by process ID), press “c” to

toggle full command line display, and using “n” to control the number of tasks that will be displayed.

Example 7-27 Using top to monitor system resource usage

```
top - 16:58:23 up 9 days, 16:12, 6 users, load average: 0.05, 0.04, 0.01
Tasks: 112 total, 1 running, 111 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.0%us, 0.2%sy, 0.0%ni, 99.2%id, 0.7%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 3842648k total, 3558428k used, 284220k free, 274856k buffers
Swap: 8393920k total, 6184k used, 8387736k free, 2884176k cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1785	db2inst1	25	0	1255m	152m	118m	S	0	4.1	0:07.46	db2sysc 0
2928	db2fenc1	15	0	522m	56m	33m	S	0	1.5	0:00.40	db2fmp (,0,0,0,0,0
2921	db2fenc1	15	0	532m	55m	33m	S	0	1.5	0:00.30	db2fmp (,1,0,0,0,0
1783	root	16	0	980m	44m	25m	S	0	1.2	0:00.09	db2wdog 0
1803	db2inst1	16	0	555m	36m	20m	S	0	1.0	0:03.33	db2acd 0 ,0,0,0,1,0
29932	db2inst1	16	0	739m	35m	13m	S	0	0.9	0:01.33	/db2home/db2inst1/s
1786	root	16	0	724m	26m	8276	S	0	0.7	0:00.00	db2ckpwd 0
1787	root	16	0	724m	26m	8204	S	0	0.7	0:00.00	db2ckpwd 0
1788	root	16	0	724m	26m	8204	S	0	0.7	0:00.00	db2ckpwd 0
5420	gdm	15	0	94424	18m	10m	S	0	0.5	0:00.42	/opt/gnome/lib64/gd
31914	db2inst1	16	0	123m	14m	9012	S	0	0.4	0:00.04	/db2home/db2inst1/s
4462	db2inst1	16	0	121m	13m	8036	S	0	0.3	0:00.02	/db2home/db2inst1/s
14531	root	16	0	110m	11m	7516	S	0	0.3	0:00.01	db2licd
4262	root	15	0	305m	8884	4336	S	0	0.2	0:00.66	/usr/X11R6/bin/X :0
26450	dasusr1	20	0	136m	7644	5300	S	0	0.2	0:00.22	/home/dasusr1/das/a

Regarding the field descriptions for the output screen of the top command, refer to the man help pages for top or other Linux documentation.

Here is another example for using top to monitor specific processes:

```
top d 1 p `pgrep db2sampl` p 1785 p 2928
```

You can specify up to 20 processes for top to monitor by using the *-p* command option. You also can utilize the Linux shell function to obtain the process ID directly in the top command. In our example, we use Linux command **pgrep db2sampl** with top to obtain the process ID for *db2sampl* process and add it into the top monitor process list.

In addition to *-p* command option, the *-d* command option is also specified in the command. It means the output screen of top will be refreshed every second. The output shows in Example 7-28.

Example 7-28 Using top to monitor specified processes

```
top - 17:11:13 up 9 days, 16:25, 7 users, load average: 0.49, 0.22, 0.07
Tasks: 3 total, 0 running, 3 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.0%us, 0.5%sy, 0.0%ni, 50.0%id, 49.5%wa, 0.0%hi, 0.0%si, 0.0%st
```

```
Mem: 3842648k total, 3577476k used, 265172k free, 276224k buffers
Swap: 8393920k total, 6184k used, 8387736k free, 2890004k cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1785	db2inst1	25	0	1252m	154m	122m	S	1	4.1	0:21.29	db2sysc
2928	db2fenc1	15	0	528m	58m	34m	S	0	1.6	0:00.50	db2fmp
5092	db2inst1	16	0	119m	12m	8308	S	0	0.3	0:00.03	db2saml

7.6.2 vmstat

vmstat command can be used to report virtual memory statistics as well as information about processes, paging, block IO and CPU activities. It is contained in the procs package for the Linux platform.

The first report produced gives averages since the last reboot. Additional reports give information on a sampling period of length delay. The process and memory reports are instantaneous in either case.

The following are field descriptions for the output report generated by vmstat. To acquire more information regarding this command, refer to man help pages for vmstat or other Linux documentation.

- ▶ Procs
 - r: The number of processes waiting for run time.
 - b: The number of processes in uninterruptable sleep.
 - w: The number of processes swapped out but otherwise runnable.
- ▶ Memory
 - swpd: The amount of virtual memory used (kB).
 - free: The amount of idle memory (kB).
 - buff: The amount of memory used as buffers (kB).
- ▶ Swap
 - si: Amount of memory swapped in from disk (kB/s).
 - so: Amount of memory swapped to disk (kB/s).
- ▶ IO
 - bi: Blocks sent to a block device (blocks/s).
 - bo: Blocks received from a block device (blocks/s).
- ▶ System
 - in: The number of interrupts per second, including the clock.
 - cs: The number of context switches per second.
- ▶ CPU
 - us: User time

- sy: System time
- id: Idle time

A sample report generated by `vmstat` is shown in Example 7-29. The “`vmstat 2 10`” command reports virtual memory statistics 10 times and interval is 2 seconds.

Example 7-29 Using `vmstat` to observe virtual memory statistics

```
pdb2inst1@mensa:/db2home/db2inst1> vmstat 2 10
```

procs		-----memory-----				---swap--		-----io----		-system--		-----cpu-----				
r	b	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	wa	st
0	0	6184	285668	274036	2883968	0	0	6	24	3	14	0	0	99	1	0
0	0	6184	285668	274036	2883968	0	0	0	38	628	310	0	0	99	1	0
0	0	6184	285668	274036	2883968	0	0	0	12	612	291	0	0	100	0	0
0	0	6184	285668	274036	2883968	0	0	0	36	616	301	0	0	100	0	0
0	0	6184	285684	274036	2883968	0	0	0	50	792	529	0	0	98	2	0
0	0	6184	285684	274036	2883968	0	0	0	24	613	293	0	0	100	0	0
0	0	6184	285700	274036	2883968	0	0	0	24	624	310	0	0	99	1	0
0	0	6184	285700	274036	2883968	0	0	0	36	614	301	0	0	100	0	0
0	0	6184	285700	274036	2883968	0	0	0	24	614	296	0	0	100	0	0
0	0	6184	285700	274036	2883968	0	0	0	32	786	521	0	0	99	1	0

7.6.3 iostat

The `iostat` command is used for reporting Central Processing Unit (CPU) statistics and monitoring system input/output device statistics for devices and partition. The report generated by the `iostat` command can be used to assist in changing the system configuration to better balance the input/output load between physical disks.

Note: The `iostat` command is contained in the `sysstat` package (the `sar` command is also included in this package), before using this command, you need to install this package.

The first report generated by the `iostat` command provides statistics concerning the time since the system was booted. Each subsequent report covers the time since the previous report.

There are some useful command options that can be used for the `iostat` command. For example, using `-k` to display statistics in kilobytes instead of blocks, using `-x` to obtain extended statistics information for devices, using `-t` to print the time for each report displayed, and so on. For more details, please refer to the man help pages for `iostat` or other Linux documentation.

Example 7-30 shows using `iostat` command to monitor CPU and device I/O statistics. The `iostat 2 5` command displays the current CPU load average and disk I/O information 5 times and interval is 2 seconds.

Example 7-30 Using iostat to get CPU and I/O statistics

```
db2inst1@mensa:/db2home/db2inst1> iostat 2 5
Linux 2.6.16.46-0.12-smp (mensa)      02/14/2008
```

avg-cpu:	%user	%nice	%system	%iowait	%steal	%idle
	0.23	0.00	0.19	0.67	0.00	98.91

Device:	tps	Blk_read/s	Blk_wrtn/s	Blk_read	Blk_wrtn
sda	2.83	23.70	87.41	19896080	73373644
sdb	0.72	0.05	15.74	37783	13215608
sdc	0.00	0.00	0.00	1816	0
sdd	0.00	0.00	0.00	1816	0
sde	0.00	0.00	0.00	1816	0
sdf	0.00	0.00	0.00	1816	0

avg-cpu:	%user	%nice	%system	%iowait	%steal	%idle
	0.00	0.00	0.25	0.50	0.00	99.00

Device:	tps	Blk_read/s	Blk_wrtn/s	Blk_read	Blk_wrtn
sda	1.49	0.00	43.78	0	88
sdb	0.00	0.00	0.00	0	0
sdc	0.00	0.00	0.00	0	0
sdd	0.00	0.00	0.00	0	0
sde	0.00	0.00	0.00	0	0
sdf	0.00	0.00	0.00	0	0

avg-cpu:	%user	%nice	%system	%iowait	%steal	%idle
	0.00	0.00	0.50	0.00	0.00	99.50

Device:	tps	Blk_read/s	Blk_wrtn/s	Blk_read	Blk_wrtn
sda	17.50	0.00	240.00	0	480
sdb	0.00	0.00	0.00	0	0
sdc	0.00	0.00	0.00	0	0
sdd	0.00	0.00	0.00	0	0
sde	0.00	0.00	0.00	0	0
sdf	0.00	0.00	0.00	0	0

avg-cpu:	%user	%nice	%system	%iowait	%steal	%idle
	0.00	0.00	0.00	0.00	0.00	100.00

Device:	tps	Blk_read/s	Blk_wrtn/s	Blk_read	Blk_wrtn
sda	0.00	0.00	0.00	0	0
sdb	0.00	0.00	0.00	0	0
sdc	0.00	0.00	0.00	0	0
sdd	0.00	0.00	0.00	0	0
sde	0.00	0.00	0.00	0	0

sdf	0.00	0.00	0.00	0	0
-----	------	------	------	---	---

7.6.4 sar

The **sar** command is an integrated and powerful System Activity Report (sar) utility. You can use it to collect and report CPU utilization, memory and swap space utilization statistics, I/O and transfer rate statistics, as well as network statistics, and so forth. You can save the output of the sar command into a designated file and retrieve information from that file at a later time.

The general syntax to run the sar command is:

sar [options] [interval [count]]

Table 7-3 gives some frequently used command options for sar that are extracted from sar’s man pages.

Table 7-3 Frequently used command options for sar

Options	Description
-A	flag is equivalent to specifying -bBcdqrUvwWy -I SUM -I XALL -n FULL -P ALL.
-b	Reports I/O and transfer rate statistics.
-c	Reports process creation activity.
-d	Reports activity for each block device.
-e	Sets the ending time of the report.
-f	Extracts records from filename (created by the -o filename flag). The default value of the filename parameter is the current daily data file, the /var/log/sa/sadd file. The -f option is exclusive of the -o option.
-n	Reports network statistics.
-o	Saves the readings in the file in binary form.
-q	Reports queue length and load averages.
-r	Reports memory and swap space utilization statistics.
-s	Sets the starting time of the data, causing the sar command to extract records time-tagged at, or following, the time specified.
-u	Reports CPU utilization.
-v	Reports status of inode, file and other kernel tables.
-w	Reports system switching activity.

Options	Description
-x	Reports statistics for a given process.

Some samples of using sar command are provided below. In Example 7-31, at first, sar is started with *-A* command option and the output is saved in *sartest.bin* file:

```
sar -A -o sartest.bin 5 60> /dev/null &
```

Where, *sartest.bin* is the output file name, the interval is five seconds and repeat 1000 times.

To see the CPU utilization information since a specified starting time, use the following command:

```
sar -u -s 14:34:00 -f sartest.bin|more
```

If you are loading data into DB2 database in a multiple partition environment, use you can use this command to see the CPU consumption of the DB2 data load

The following command can be used to get process creation/fork and system context switch information:

```
sar -w -c -s 14:34:45 -e 14:35:30 -f sartest.bin|more
```

The following command can be used to obtain network statistics for a specific network interface:

```
sar -n DEV -s 14:36:00 -f sartest.bin|egrep "eth0|IFACE"
```

Example 7-31 shows these command and their output.

Example 7-31 Using sar to monitor system activities

```
mensa:/tmp # sar -A -o sartest.bin 5 60 > /dev/null &
[1] 11457
mena:/tmp # sar -u -s 14:34:00 -f sartest.bin|more
Linux 2.6.16.46-0.12-smp (mena)      02/15/08

14:34:01      CPU      %user      %nice      %system      %iowait      %idle
14:34:06      all       1.00       0.00       1.10       19.58       78.32
14:34:11      all       2.30       0.00       1.40       17.78       78.52
14:34:16      all       0.20       0.00       0.80       18.92       80.08
14:34:21      all       2.30       0.00       0.90       23.42       73.37
14:34:26      all       0.50       0.00       0.70       17.68       81.12
14:34:31      all       1.80       0.00       0.90       23.90       73.40
14:34:36      all       1.20       0.00       0.80       14.70       83.30
14:34:41      all       1.00       0.00       0.90       19.48       78.62
14:34:46      all       2.10       0.00       0.90       15.30       81.70
```

```

Average:          all      1.38      0.00      0.93      18.97      78.72
mena:/tmp # sar -w -c -s 14:34:45 -e 14:35:30 -f sarestest.bin|more
Linux 2.6.16.46-0.12-smp (mena)      02/15/08

```

```

14:34:46      proc/s
14:34:51      0.60
14:34:56      1.20
14:35:01      0.40
14:35:06      1.00
14:35:11      0.40
14:35:16      1.40
14:35:21      0.40
14:35:26      1.00
Average:      0.80

```

```

14:34:46      cswch/s
14:34:51      787.03
14:34:56      1479.36
14:35:01      617.80
14:35:06      1562.20
14:35:11      600.60
14:35:16      1559.92
14:35:21      627.29
14:35:26      1510.82
Average:      1092.50

```

```

mena:/tmp # sar -n DEV -s 14:36:00 -f sarestest.bin|egrep "eth0|IFACE"

```

	IFACE	rxpck/s	txpck/s	rxbyt/s	txbyt/s	rxcmp/s	txcmp/s	rxmcst/s
14:36:01	eth0	883.40	1207.60	662308.40	1471505.20	0.00	0.00	0.00
14:36:06	eth0	19.80	18.60	3153.20	17137.40	0.00	0.00	0.00
14:36:11	eth0	832.06	1154.51	654760.92	1455199.00	0.00	0.00	0.00
14:36:16	eth0	115.20	152.60	23889.60	165605.60	0.00	0.00	0.00
14:36:21	eth0	769.26	1063.47	638018.56	1315718.56	0.00	0.00	0.00
14:36:26	eth0	523.92	719.32	396419.44	884977.36	0.00	0.00	0.00
Average:	eth0							

For the details about using the sar command, refer to the man help pages for sar or other Linux documentation.

Note: As mentioned in 7.6.3, “iostat” on page 410, the sar utility is in the sysstat package. Before using this command, you need to make sure that the sysstat package is installed in your system.

7.6.5 Other system monitoring tools

There are also a multitude of Linux system monitoring tools available for monitoring Linux system resource usage, such as the **ps** command which is known to almost everybody is used to report processes status, **ps tree** can be

used to display a tree of processes, **netstat** for printing network related information, **nfsstat** for NFS specific statistics, and so forth. In addition, **pgrep** and **kill** commands are very helpful when you want to look up or signal processes based on name and other attributes of processes. And if you like monitoring tools with graphical user interface, utilities like *gnome-system-monitor* and *ksysguard* also can be used. For more information, refer to Linux specific documentation.

8

Application development

It has become increasingly important for a database to effectively support a wide range of open standards and open source products for application development. DB2 provides accessibility for many standard interfaces and comes with a full suite of development tools.

This chapter discusses the following topics:

- ▶ Application configuration
- ▶ DB2 application objects
- ▶ Programming languages
- ▶ Application development tools

In this chapter, we demonstrate the examples with the SAMPLE database DB2 provides. You can create a sample database in your instance by running the command **db2samp1** from the command line. Alternatively, you can create the SAMPLE database from the First Steps (**db2fs**) at the end or after the installation. In our examples we usually refer to the DB2 instance home directory as *db2inst1*. You can change that if you use a different instance name.

You can find startup information in *Getting Started with Database Application Development*, GC23-5856-00 and on this Web site:

<http://www.ibm.com/software/data/db2/ad/>

DB2 provides many programming examples in the directory
<instance home>/**sql1ib/samples**

8.1 Application configuration

DB2 supports local and remote application development that provides the flexibility in application design. Figure 8-1 illustrates three different database connection configurations.

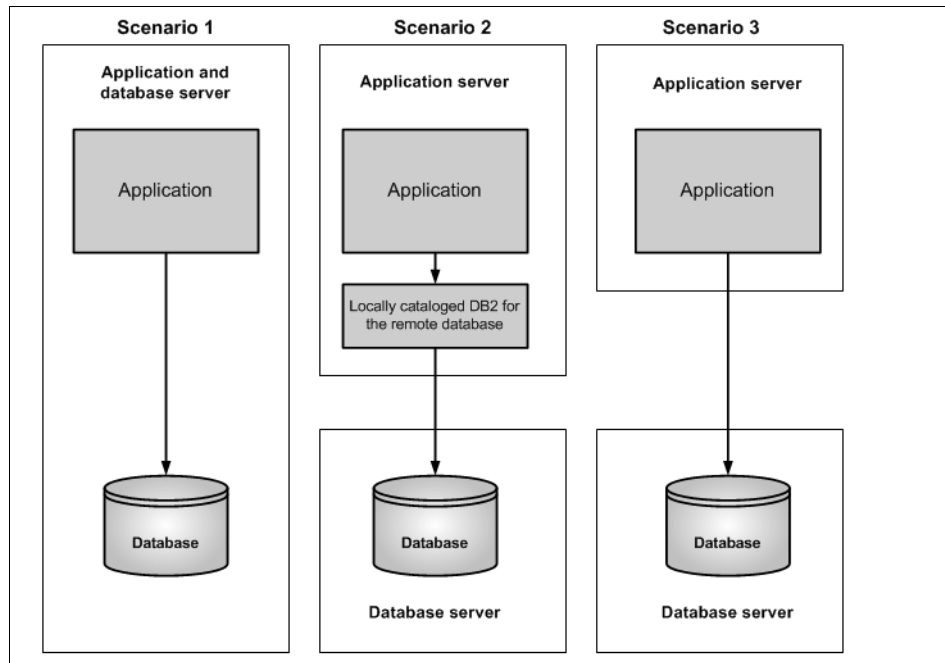


Figure 8-1 Connection scenarios for applications

- ▶ **Scenario 1:** A local client connects to a local database.
 This configuration requires a locally installed database server. In this scenario the application and the database run on the same server. This is useful for small systems or for test and development systems.
- ▶ **Scenario 2:** A local client connects to a remote database by using a locally cataloged node.
 This configuration requires only a locally installed DB2 client. The database is hosted on another system that has a DB2 data server installed. After cataloging the database node, the remote database looks like a local database from the application point of view.
- ▶ **Scenario 3:** A local client connects to a remote database directly.
 This configuration does not require any DB2 software installed on the local system. Instead you use a driver that supports the direct connection to the

remote database. For instance a Type 4 JDBC driver. The database server is installed on the other machine.

Separating the application and the database servers as in scenario 2 and 3 is usually a preferred architecture for the larger installations. It provides more flexibility in tuning the servers separately, applying maintenance tasks, and implementing failover.

DB2 supports multi-version installation on one system. Different versions or the same version with different fix pack levels can co-exist in the same machine. For instance, you can run a DB2 version 8.2 and a DB2 version 9.5 instance on same machine or V8.1 FixPak 1 and V8.1 FixPak 2 in one system.

An instance is related to a Linux user ID and can be accessed through the network by a TCP/IP port. If you want to access a database remotely, the database has to be configured to listen on a particular port. The port can be configured by using:

```
db2 UPDATE DBM CONFIG USING SVCENAME 50000
```

In addition, the TCP/IP protocol must be enabled by using:

```
db2set DB2COMM=tcpip
```

The database manager needs to be restarted afterwards. Figure 8-2 shows the concept of the instance setup.

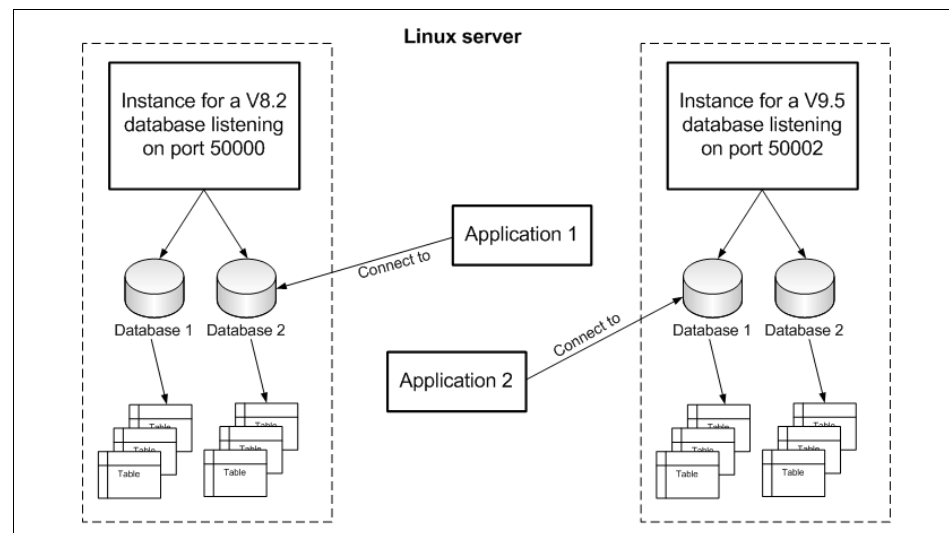


Figure 8-2 Instance setup for application connections

8.2 DB2 application objects

DB2 supports stored procedures (SP), user defined functions (UDFs), and triggers, all of which are used to develop applications. You can develop the applications from the command line or you can use the Data Studio.

8.2.1 Triggers

A *trigger* is a defined set of SQL statements stored as a DB2 object in a DB2 database. This set of SQL statements execute when a certain event occurs against a DB2 table. Types of events that might invoke triggers are Insert, Update, or Delete to a given DB2 table. You can set up triggers to execute *before*, *after*, or *instead of* an insert, update, or delete event:

- ▶ **[No cascade] before:** The defined action will be executed before the triggering action is performed.
- ▶ **After:** All triggered actions will be applied when the triggering action is done.
- ▶ **Instead of:** The original action will be replaced by the action defined in the trigger.

Triggers are defined by using the **create trigger** DDL statement.

Note: The CREATE TRIGGER statement is explained in more detail in the Information Center in **Database fundamentals** → **SQL** → **Statements** → **CREATE TRIGGER** and in *SQL Reference, Volume 2*, SC10-4250.

Triggers are useful to implement business logic in a central place. A trigger becomes part of the transaction which fires it.

Example 8-1 demonstrates how to create a trigger to audit changes on either the SALARY or BONUS column in the table EMPLOYEE of our SAMPLE database. The changes are captured into the table EMPLOYEE_HIST.

Example 8-1 Trigger sample to store history data

```
CONNECT TO sample;

CREATE TABLE employee_hist(
    empno          CHAR(6),
    salary_old     DECIMAL(9,2),
    salary_new     DECIMAL(9,2),
    bonus_old      DECIMAL(9,2),
    bonus_new      DECIMAL(9,2),
    chg_date       TIMESTAMP);
```



```

CREATE TRIGGER employee_audit
  AFTER UPDATE OF salary, bonus ON employee
  REFERENCING OLD AS old_row NEW AS new_row
  FOR EACH ROW MODE db2sql
  INSERT INTO employee_hist
    Values(old_row.empno,
      old_row.salary,
      new_row.salary,
      old_row.bonus,
      new_row.bonus,
      current timestamp);

```

In Example 8-2 we modify some salary amounts on the employee table to fire the trigger and populate the EMPLOYEE_HIST table. By selecting data from EMPLOYEE_HIST, you can see how our trigger works.

Example 8-2 Each update forces one insert into employee_hist

```

db2 "update employee set salary = 55000 where empno = '000070'"
db2 "update employee set salary = 45000 where empno = '000110'"
db2 "update employee set salary = 55000 where empno = '000170'"
db2 "update employee set salary = 65000 where empno = '000070'"

```

```

db2inst1@puget:~> db2 "select * from employee_hist order by empno"

```

EMPNO	SALARY_OLD	SALARY_NEW	BONUS_OLD	BONUS_NEW	CHG_DATE
000070	96170.00	55000.00	700.00	700.00	2008-01-29-03.06.17.126586
000070	55000.00	65000.00	700.00	700.00	2008-01-29-03.06.19.649943
000110	66500.00	45000.00	900.00	900.00	2008-01-29-03.06.17.287657
000170	44680.00	55000.00	500.00	500.00	2008-01-29-03.06.17.353814

4 record(s) selected.

8.2.2 Use defined functions

DB2 provides many functions called built-in functions. There are two types of built-in functions: scalar functions, such as *date*, *length*, *month*, and *substr*; and column functions, such as *count*, *sum*, *max*, *avg*, and *min*.

User defined function (UDFs) are functions that can be added to the database. They can include scalar-function, row-function, column-function, or table functions. UDFs can be written in languages such as C, Java, or in pure DB2 SQL. This section discusses UDFs written in SQL. Developing UDF using C and Java is provided in “UDF in C” on page 438 and “Stored procedures in Java” on page 444 separately.

UDFs are very useful if you want to share business logic between different applications. The applications can run on different systems and on different architectures.

Use the CREATE FUNCTION DDL statement to register a function in the database.

Note: The CREATE FUNCTION statement is explained in more detail in the Information Center in **Database fundamentals** → **SQL** → **Statements** → **CREATE FUNCTION** and in *SQL Reference, Volume 2*, SC10-4250.

Example 8-3 creates an SQL UDF called salary_diff which receives an employee ID as input and calculates the difference between the old and new salary values from the EMPLOYEE_HIST which is populated through our trigger (see Example 8-1 on page 420). To get the detailed information from the EMPLOYEE table with history data stored in EMPLOYEE_HIST, we can join EMPLOYEE and EMPLOYEE_HIST to calculate the column salary_diff, or we can use an UDF to perform the function.

Example 8-3 UDF to calculate salary difference from audit table

```
CONNECT TO sample;

CREATE FUNCTION salary_diff (emp CHAR(6))
  RETURNS DECIMAL (9,2)
  LANGUAGE SQL
  READS SQL DATA
  NO EXTERNAL ACTION
  DETERMINISTIC
  RETURN
  SELECT (salary_new - salary_old) AS salary_diff
  FROM employee_hist h,
       employee      o
  WHERE h.empno = emp
       AND h.empno = o.empno
       AND h.chg_date = (SELECT MAX(chg_date)
                        FROM employee_hist
                        WHERE empno = h.empno)
```

Note: The statement SELECT MAX(chg_date) is required, because for each empno there can exist more than one row in EMPLOYEE_HIST table, but a UDF can return only one row.

In Example 8-4 on page 423 we can use the newly created function salary_diff in our query.

Example 8-4 Use the UDF in a select statement

```
db2inst1@puget:~> db2 "select empno, firstnme, lastname, salary_diff(empno) as
salary_diff from employee where empno in ('000110','000070')"
```

EMPNO	FIRSTNME	LASTNAME	SALARY_DIFF
000110	VINCENZO	LUCCHESI	-25000.00
000070	EVA	PULASKI	10000.00

2 record(s) selected.

Note: More information about UDFs can be found in *Developing User-defined Routines (SQL and External)*, SC23-5855-00.

8.2.3 Stored procedures

A *stored procedure* helps to reduce unnecessary data transfer over the network between client and server. The SQL statements and definitions are stored on the database server. The client sends only the necessary data over the network to call the stored procedure and gets the results sent back. This helps to improve the overall performance of client server applications.

Like UDFs, stored procedures are very useful to put common business logic into a central place and to make it accessible to any client.

The CREATE PROCEDURE DDL statement defines a procedure within an application server. There are two types of procedures: external procedures, which are written in a programming language; and SQL procedures, which are written in SQL. Stored procedures written in other languages than SQL are handled very similar like UDFs written in other languages. Chapter 8.3.3, “C/C++” on page 432 and Chapter 8.3.4, “Java” on page 440 discuss UDFs written in C and Java.

Example 8-5 on page 424 shows an SQL procedure that calculates the new salary and bonus for an employee. The input is an employee number and rating, and the result is to update the table EMPLOYEE.

The file *basecase.db2* is part of the DB2 samples and can be found in *<instance home>/sqlib/samples/sqlproc*.

Note: The CREATE PROCEDURE statement is explained in more detail in the Information Center in **Database fundamentals** → **SQL** → **Statements** → **CREATE PROCEDURE** and in *SQL Reference, Volume 2*, SC10-4250.

Example 8-5 A simple example of an SQL procedure

```
db2inst1@puget:~/sqllib/samples/sqlproc> db2 connect to sample
```

Database Connection Information

```
Database server      = DB2/LINUX 9.5.0
SQL authorization ID = DB2INST1
Local database alias = SAMPLE
```

```
db2inst1@puget:~/sqllib/samples/sqlproc> db2 -td@ -vf basecase.db2
```

```
CREATE PROCEDURE update_salary
(IN employee_number CHAR(6), IN rating INT)
LANGUAGE SQL
BEGIN
  DECLARE SQLSTATE CHAR(5);
  DECLARE not_found CONDITION FOR SQLSTATE '02000';
  DECLARE EXIT HANDLER FOR not_found
    SIGNAL SQLSTATE '02444';
```

```
  CASE rating
  WHEN 1 THEN
    UPDATE employee
    SET salary = salary * 1.10, bonus = 1000
    WHERE empno = employee_number;
  WHEN 2 THEN
    UPDATE employee
    SET salary = salary * 1.05, bonus = 500
    WHERE empno = employee_number;
  ELSE
    UPDATE employee
    SET salary = salary * 1.03, bonus = 0
    WHERE empno = employee_number;
  END CASE;
```

```
END
```

```
DB20000I The SQL command completed successfully.
```

Note: In the file basecase.db2 we use the “@” character as a command terminator because the semicolon “;” is already used to terminate the procedure statements.

Example 8-6 shows how to call the SQL procedure from the command line.

Example 8-6 Procedure call on command line

```
db2inst1@puget:~> db2 "CALL update_salary ('000100', 1)"
```

Return Status = 0

A stored procedure usually runs in a separate process under the fenced user. The fenced user is defined in the CREATE INSTANCE command: `./db2icrt -u <fenced id> <instance id>`. During development of SPs and UDFs you should change the database manager configuration parameter KEEPFENCED (in previous version KEEPDARI) to NO, so that every time the procedure is invoked the actual version is loaded:

```
db2 update dbm cfg using KEEPFENCED NO
```

Note: More information about stored procedures can be found in *Developing User-defined Routines (SQL and External)*, SC23-5855-00.

8.3 Programming languages

In this section, we introduce some of the programming languages you can use on Linux and explain how to set up the environment for each one. The languages covered in this section are:

- ▶ C/C++
- ▶ Java
- ▶ PHP
- ▶ Perl

In general, the following steps are required to use these languages with DB2:

1. Set up the application development environment for the language you are using. Build, install, and test the required programs, modules and drivers, and make any necessary changes to permissions and paths.
2. Identify and include DB2 specific drivers in your code. In order for your program to access DB2, you must copy or reference the required code into your program.
3. Connect to a specific DB2 database. Usually an identifier is generated from the connect command, which is used with the SQL requests in the program.
4. Use SQL statements to manipulate and retrieve data in the database.

For performing software development in most cases you need install at least the IBM Data Server Client.

In addition to the programming languages described in this section, DB2 also support application development for the following:

- ▶ ADO.NET support
Developing ADO.NET and OLE DB Applications, SC23-5867-00

- ▶ ODBC
Call Level Interface Guide and Reference, Volume 1, SC23-5844-00
Call Level Interface Guide and Reference, Volume 2, SC23-5845-00
- ▶ OLD DB
Developing ADO.NET and OLE DB Applications, SC23-5867-00
- ▶ COBOL
Developing Embedded SQL Applications, SC23-5852-00
- ▶ Fortran
Developing Embedded SQL Applications, SC23-5852-00
- ▶ REXX™
Administrative API Reference, SC23-5842-00
- ▶ Ruby/Ruby on Rails
Information Center: **Database application development** → **Database applications** → **Ruby on Rails**

8.3.1 Perl

Here we introduce Perl programming on DB2 and describe how to set up a Perl application development environment on DB2 for Linux.

Perl overview

Perl is a popular, general-purpose programming language that is freely available on Linux. It was originally developed for text manipulation, but is now used for a wide range of tasks including system administration, Web development, network programming, and GUI development. It has also become the premier scripting language of the Web. According to Larry Wall, its creator, he included in Perl “all the cool features found in other languages and left out those features that weren't so cool”.

Perl is typically provided by the Linux distributions, but you can also download it from:

<http://www.cpan.org/>

Perl and DB2

You can create DB2 applications using Perl by using the DBD::DB2 driver with the Perl Database Interface (DBI) Module. Both of these components are freely available on the Internet. Currently, DB2 supports the following versions:

- ▶ Perl 5.8 or later
- ▶ DBI 1.41 or later

Perl is an ideal language to use with DB2:

- ▶ It is an interpreted language and has a DBI Module that uses dynamic SQL. This is ideal for quickly creating and revising prototypes of DB2 applications.
- ▶ The Perl DBI module uses an interface that is quite similar to the CLI and JDBC interfaces, which makes it easy for you to port your Perl prototypes to CLI and JDBC.
- ▶ Perl offers very powerful functions for formatting text and handling strings. That allows you to generate reports very quickly.

Setting up a Perl application development environment

In order to set up a Perl application development environment for DB2 on Linux, the following components must be installed and configured on your workstation:

- ▶ DB2 client (or server)
- ▶ Perl 5.8 or later
- ▶ DBI 1.41 or later
- ▶ DBD::DB2 driver

Here is the installation procedure:

1. Install DB2 on your workstation. Refer to Chapter 2, “Installation” on page 25 for instructions on how to install DB2 on Linux.
2. Install Perl 5. Perl is typically provided by the Linux distributions, but you can download it from:

<http://www.cpan.org/>

3. Build, test, and install the DBI module. DBI is an open standard API that provides database access for client applications written in Perl.

Usually the Perl DBI module is shipped with your Linux distribution. If you want to install it you can download it from the CPAN Web site at:

http://www.cpan.org/modules/by-category/07_Database_Interfaces/DBI/

Refer to the Readme file on this Web site for installation instructions.

4. Build, test and install the DBD::DB2 driver.

The DBD::DB2 driver works with DBI and a DB2 client to access databases. The latest DB2 Perl DBI driver can be downloaded from:

<http://www.ibm.com/software/data/db2/perl/>

At the time of writing, the latest available driver was DBD-DB2-1.1.

- a. Ensure that the C compiler is installed. It is needed for compiling the Perl module. In order to compile the module you need the DB2 client installed on your system.
- b. Before installing DBD::DB2 driver, enable the DB2 environment if it hasn't already been done for your user in .profile.

```
. ~db2inst1/sql/lib/db2profile
```

- c. Untar DBD-DB2-1.1 by entering the following command.

```
tar xvzf DBD-DB2-1.1.tar.gz
```

- d. In the DBD-DB2-1.1 directory run the following commands to build and install the DBD::DB2 driver:

```
perl Makefile.PL
make
make test
```

- The **make** command builds the software
- The **make test** command executes self tests

- e. Now login as the root user and run the following command from the same directory:

```
make install
```

- The **make install** command installs the DBI.

5. Enable the DBI module.

To enable Perl to load the DBI module, you must include the following line in your perl program:

```
use DBI;
```

The DBI module automatically loads the DBD::DB2 driver when you create a database handle using the `DBI->connect` statement using the following syntax:

```
my $dbhandle = DBI->connect('dbi:DB2:dbalias', $userID, $password);
```

Where:

- **\$dbhandle** represents the database handle returned by the connect statement.
- **\$dbalias** represents a DB2 alias cataloged in your DB2 database directory.
- **\$userID** represents the user ID used to connect to the database.
- **\$password** represents the password for the user ID to connect to the database.

Example 8-7 shows a short Perl program which uses DBI. It uses the SAMPLE database. Before running the example you have to replace the `<password>`.

Example 8-7 Perl script using DBI - perltest.pl

```
#!/usr/bin/perl
```

```
use DBI;
```



```

my $dbhhandle = DBI->connect('dbi:DB2:SAMPLE', 'db2inst1', '<password>');

my $sth = $dbhhandle->prepare('
select
    empno,
    lastname,
    firstnme,
    salary
from
    employee
where
    salary < ?
order by lastname
') or die "Couldn't prepare statement: " . $dbh->errstr;

$sth->execute(36000);

while (@data = $sth->fetchrow_array()) {
    print "$data[0],$data[1],$data[2],$data[3]\n";
}

exit 0;

```

Example 8-8 shows the output of Example 8-7 on page 428.

Example 8-8 Output of the DBI perl script

```

db2inst1@puget:~/redbook/perl> ./perltest.pl
200340,ALONZO,ROY,31840.00
000290,PARKER,JOHN,35340.00
000310,SETRIGHT,MAUDE,35900.00
200310,SPRINGER,MICHELLE,35900.00
200330,WONG,HELENA,35370.00

```

Note: RHEL 5.1 and SLES 10.1 include Perl and Perl DBI RPM packages (Red Hat: perl-5.8.8-10 and perl-DBI-1.52-1.fc6; SUSE: perl-5.8.8-14.2 and perl-DBI-1.50-13.2). To create a DB2 application with Perl, The Red Hat and SUSE users only have to install and compile the DBD::DB2 package.

More examples can be found under *<instance home>/sqllib/samples/perl*.

Here are some good references for Perl application development on DB2:

- Everything you need to know about mod_perl technology:

<http://perl.apache.org/docs/1.0/guide/>

- ▶ Useful information about DBI:
<http://dbi.perl.org/>
- ▶ The IBM DB2 Perl database interface Web site:
<http://www.ibm.com/software/data/db2/perl/>

Note: More information about Perl development can be found in *Developing Perl and PHP Applications*, SC23-5854-00.

8.3.2 PHP

If your requirements call for a Web application, consider using DB2 for Linux with PHP (PHP: Hypertext Preprocessor). PHP excels when chosen to develop applications that need to be up and running very quickly without a large commitment of staff or budget resources. You might use PHP to provide department access to DB2 data over the LAN through a Web browser, or you can build an e-commerce store which demands high transactional integrity on the Internet.

PHP is very flexible and can leverage your investment in DB2 in many innovative ways.

PHP overview

PHP is a scripting language most often embedded in HTML documents and executed on the server before output is sent to the Web browser. It can also be used as a command-line tool outside of the Web server environment.

Learning PHP is easy, but developing and managing applications properly takes some care. Fortunately, there is a large community of users and a myriad of libraries to make your job easier. In particular, the Zend Framework and CakePHP provide Web application foundations and database abstraction facilities which ease migration to DB2 from another data server, such as MySQL. The PEAR project also provides many high quality components written in PHP intended for reuse.

Tip: You can learn more about PHP at IBM developerWorks. Refer to the Recommended PHP reading list:

<http://www.ibm.com/developerworks/opensource/library/os-php-read/>

PHP and DB2

There are four extensions that you can use to write PHP applications with DB2. All of the PHP interfaces ultimately communicate with DB2 using the Call Level

Interface (CLI). The interfaces are PHP extensions, which are written in C and compiled with the DB2 libraries when you install PHP:

- ▶ `ibm_db2` - provides the `db2_*` functions.
- ▶ `PDO_IBM` - provides the PDO object for CLI.
- ▶ `PDO_ODBC` - provides the PDO object for ODBC.
- ▶ `Unified ODBC` - provides the `odbc_*` functions.

IBM recommends using `ibm_db2` or `PDO_IBM` to get the best performance and features out of DB2. Which you choose is dependent on the type of application you are writing and the version of PHP (4 or 5).

Refer to chapter 4 of *Developing PHP Applications for IBM Data Servers*, SG24-7218 to learn more about choosing and installing a PHP extension for DB2.

Tip: The Zend Core for IBM provides all of the DB2 interfaces for PHP in an easily installed and supported package:

<http://www.zend.com/en/products/core/for-ibm>

PHP and the Web server

PHP is most often built as an Apache module, though it can also be used through CGI. Many other Web servers support PHP too, including the Apache-based IBM HTTP Server.

If you intend to use PHP with DB2 using a Web server front-end, you should consult chapter 1 of *Developing PHP Applications for IBM Data Servers*, SG24-7218 to guide your decision on an HTTP server and learn how to install it with PHP and DB2.

Note: If you intend to share an HTTP server with WebSphere Application Server, the IBM WebSphere Developer Technical Journal article *Pair J2EE with PHP to implement a common Web application infrastructure* provides instructions to help them coexist:

http://www.ibm.com/developerworks/websphere/techjournal/0505_krook/0505_krook.html

Here are some further resources for maximizing the benefits of using PHP with DB2:

- ▶ Official PHP documentation
<http://www.php.net/docs.php>

- ▶ Zend Core for IBM is a seamless out-of-the-box, easy to install and supported PHP development and production environment; featuring tight integration with DB2 drivers.

<http://www.zend.com/en/products/core/for-ibm>

- ▶ Zend Framework is based on simplicity, object-oriented best practices, corporate friendly licensing, and a rigorously tested agile codebase.

<http://framework.zend.com/>

- ▶ CakePHP enables PHP users at all levels to rapidly develop robust Web applications.

<http://cakephp.org/>

- ▶ PEAR is a framework and distribution system for reusable PHP components.

<http://pear.php.net/>

- ▶ The PDT project provides a PHP Development Tools framework for the Eclipse platform.

<http://www.eclipse.org/pdt/>

8.3.3 C/C++

C and C++ are very popular and powerful programming language under Linux. Currently, DB2 for Linux supports the C/C++ programming languages and compilers shown in Table 8-1:

Table 8-1 Supported C/C++ compilers

Linux version	Compiler version
Linux on x86 (32-bit DB2 instances and compilers only)	GNU/Linux gcc versions 3.3 and 3.4 Intel C Compiler Version 9.0
Linux on AMD64/EM64T (x86-64, x64)	GNU/Linux gcc versions 3.3 and 3.4 Intel C Compiler Version 9.0
Linux on IPF (IA64) (64-bit DB2 instances and compilers only)	GNU/Linux gcc versions 3.3 and 3.4 Intel C Compiler Version 9.0
Linux on POWER (PowerPC®, iSeries®, pSeries)	GNU/Linux gcc versions 3.3 and 3.4 IBM XL C/C++ Advanced Edition Version 7.0 for Linux
Linux on zSeries (s/390x)	GNU/Linux gcc versions 3.3 and 3.4

In order to develop C/C++ code you need the DB2 client installed on your system.

The CLI interface

The *Call Level Interface* (CLI) is a C/C++ API to access DB2. It uses function calls to pass dynamic SQL statements. It is an alternative to *Embedded SQL* (ESQL). CLI does not require host variables or a precompiler.

Example 8-9 shows a CLI program which selects from the EMPLOYEE table in the SAMPLE database. Before running the example you have to replace the *<password>*.

Example 8-9 CLI program - cli_test.c

```
#include <stdio.h>
#include <sqlcli1.h>

main() {
    SQLRETURN cliRC = SQL_SUCCESS;
    SQLHANDLE henv, hdbc, hstmt;

    /* Variable which return the row data */
    struct { SQLINTEGER ind; SQLCHAR val[7]; } empno;

    /* Allocate environment handle */
    if((cliRC = SQLAllocHandle(SQL_HANDLE_ENV, SQL_NULL_HANDLE, &henv)) !=
SQL_SUCCESS) {
        fprintf(stderr, "CLI error %d in SQLAllocHandle\n", cliRC); exit(cliRC);
    }

    /* Allocate database handle */
    if((cliRC = SQLAllocHandle(SQL_HANDLE_DBC, henv, &hdbc)) != SQL_SUCCESS) {
        fprintf(stderr, "CLI error %d in SQLAllocHandle\n", cliRC); exit(cliRC);
    }

    /* connect to the database */
    if((cliRC = SQLConnect(hdbc,
                          (SQLCHAR *)"SAMPLE", SQL_NTS,
                          (SQLCHAR *)"db2inst1", SQL_NTS,
                          (SQLCHAR *)"<password>", SQL_NTS)) != SQL_SUCCESS) {
        fprintf(stderr, "CLI error %d in SQLConnect\n", cliRC); exit(cliRC);
    };

    /* allocate a statement handle */
    if((cliRC = SQLAllocHandle(SQL_HANDLE_STMT, hdbc, &hstmt)) != SQL_SUCCESS) {
        fprintf(stderr, "CLI error %d in SQLAllocHandle\n", cliRC); exit(cliRC);
    }

    /* directly execute the statement */
    cliRC = SQLExecDirect(hstmt, (SQLCHAR *)"select empno from employee where
salary < 36000", SQL_NTS);
    if(cliRC != SQL_SUCCESS) {
```

```

    fprintf(stderr,"CLI error %d in SQLExecDirect\n", cliRC); exit(cliRC);
}

/* bind columns to variables */
if((cliRC = SQLBindCol(hstmt, 1, SQL_C_CHAR, empno.val, sizeof(empno.val),
&empno.ind)) != SQL_SUCCESS) {
    fprintf(stderr,"CLI error %d in SQLBindCol 1\n", cliRC); exit(cliRC);
}

/* fetch each row, and display */
if((cliRC = SQLFetch(hstmt)) != SQL_SUCCESS) {
    fprintf(stderr,"CLI error %d in SQLFetch\n", cliRC); exit(cliRC);
}

while (cliRC != SQL_NO_DATA_FOUND) {
    /* bind columns to variables */
    if((cliRC = SQLGetData(hstmt, 1, SQL_C_CHAR, empno.val, sizeof(empno.val),
&empno.ind)) != SQL_SUCCESS) {
        fprintf(stderr,"CLI error %d in SQLGetData 1\n", cliRC); exit(cliRC);
    }

    printf("%-6.6s\n", empno.val);

    if((cliRC = SQLFetch(hstmt)) != SQL_SUCCESS) {
        if(cliRC != SQL_NO_DATA_FOUND) {
            fprintf(stderr,"CLI error %d in SQLFetch\n", cliRC); exit(cliRC);
        }
    }
}

/* Free statement handle */
if((cliRC = SQLFreeHandle(SQL_HANDLE_STMT, hstmt)) != SQL_SUCCESS) {
    fprintf(stderr,"CLI error %d in SQLFreeHandle\n", cliRC); exit(cliRC);
}

/* disconnect from the database */
if((cliRC = SQLDisconnect(hdbc)) != SQL_SUCCESS) {
    fprintf(stderr,"CLI error %d in SQLConnect\n", cliRC); exit(cliRC);
};

/* Free database handle */
if((cliRC = SQLFreeHandle(SQL_HANDLE_DBC, hdbc)) != SQL_SUCCESS) {
    fprintf(stderr,"CLI error %d in SQLFreeHandle\n", cliRC); exit(cliRC);
}

/* Free environment handle */
if((cliRC = SQLFreeHandle(SQL_HANDLE_ENV, henv)) != SQL_SUCCESS) {
    fprintf(stderr,"CLI error %d in SQLFreeHandle\n", cliRC); exit(cliRC);
}

```

```
}

```

Example 8-10 shows the commands to build and run the program.

Example 8-10 Running the CLI program

```
db2inst1@puget:~> export LD_LIBRARY_PATH=/home/db2inst1/sqllib/lib
db2inst1@puget:~> cc -c -I/home/db2inst1/sqllib/include cli_test.c
db2inst1@puget:~> cc -o cli_test -L/home/db2inst1/sqllib/lib -ldb2 cli_test.o
db2inst1@puget:~> ./cli_test
000290
000310
200310
200330
200340
```

Note: You can find a detailed introduction of the CLI functions in the Information Center in **Database application development** → **Database Applications** → **CLI**. The detailed reference can be found in **Reference** → **APIs** → **CLI**. The available manuals are *Call Level Interface Guide and Reference, Volume 1*, SC23-5844-00 and *Call Level Interface Guide and Reference, Volume 2*, SC23-5845-00.

Embedded SQL

Embedded SQL allows you to put SQL statements directly into the C/C++ code. You can *embed* them into the code. This requires a *precompilation* step of the program before it can be compiled with the native C/C++ compiler.

The precompile step produces a bind file (*package*) and a program source file. The package has the suffix *.bnd*. The program file can then further be compiled by the C/C++ compiler. The package contains the *access plan* for the embedded SQL statements. This package is dedicated to this particularly compiled file. If you compile it again you'll get a new bind file. Before you can run a program, you need to bind the related packages to the database.

The advantage of using the package is that the access plans of the SQL statements have already been generated during the precompilation step. They don't need to be generated at the runtime anymore. For complex SQL statements that may save a considerable amount of execution time.

Packages also allow you to grant permissions to the users or applications on package level without the need to grant access to the individual tables.

In order to run embedded SQL programs you need at least a DB2 client installed on the local system.

Example 8-11 shows a simple C program using embedded SQL. It reads records from the EMPLOYEE table and uses a cursor to cycle through the result set. Before running the example you have to replace the *<password>*.

Example 8-11 Embedded SQL program - test_emb.sqc

```
EXEC SQL INCLUDE SQLCA;

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main(int argc, char *argv[])
{
    EXEC SQL BEGIN DECLARE SECTION;
    char empno[7];
    char database[20];
    char user[20];
    char password[20];
    EXEC SQL END DECLARE SECTION;

    strcpy(database,"SAMPLE");
    strcpy(user,"db2inst1");
    strcpy(password,"<password>");

    EXEC SQL CONNECT TO :database USER :user USING :password;
    if(sqlca.sqlcode != 0) {
        fprintf(stderr,"Error connect %d\n", sqlca.sqlcode); exit(sqlca.sqlcode);
    }

    EXEC SQL DECLARE EMP_CUR CURSOR FOR
    SELECT EMPNO FROM EMPLOYEE WHERE SALARY < 36000;

    EXEC SQL OPEN EMP_CUR;
    if(sqlca.sqlcode != 0) {
        fprintf(stderr,"Error open cursor %d\n", sqlca.sqlcode);
        exit(sqlca.sqlcode);
    }

    EXEC SQL FETCH EMP_CUR INTO :empno;
    if(sqlca.sqlcode != 0) {
        fprintf(stderr,"Error fetch cursor %d\n", sqlca.sqlcode);
        exit(sqlca.sqlcode);
    }

    while (sqlca.sqlcode != 100) {
```



```

    printf("%s\n", empno);

    EXEC SQL FETCH EMP_CUR INTO :empno;
    if(sqlca.sqlcode != 0) {
        if(sqlca.sqlcode != 100) {
            fprintf(stderr, "Error fetch cursor %d\n", sqlca.sqlcode);
        }
    }
    exit(sqlca.sqlcode);

    EXEC SQL CLOSE EMP_CUR;
    if(sqlca.sqlcode != 0) {
        fprintf(stderr, "Error close cursor %d\n", sqlca.sqlcode);
    }
    exit(sqlca.sqlcode);
}

```

In Example 8-12 we compile and run the program. The precompile step generates the .c and the .bnd file. The first **cc** step compiles the C program and the second **cc** step links it.

Example 8-12 Run the embedded SQL program

```
db2inst1@puget:~> db2 connect to SAMPLE
```

Database Connection Information

```

Database server      = DB2/LINUX 9.5.0
SQL authorization ID = DB2INST1
Local database alias = SAMPLE

```

```
db2inst1@puget:~> db2 precompile test_emb.sqc bindfile package
```

```
LINE    MESSAGES FOR test_emb.sqc
```

```

-----
      SQL0060W  The "C" precompiler is in progress.
      SQL0091W  Precompilation or binding was ended with "0"
                  errors and "0" warnings.
db2inst1@puget:~> cc -c -I/home/db2inst1/sqllib/include test_emb.c
db2inst1@puget:~> cc -o test_emb -L/home/db2inst1/sqllib/lib -ldb2 test_emb.o
db2inst1@puget:~> ./test_emb
000290
000310
200310
200330
200340

```

If you want to run this program on another database, you have to bind the package first by issuing the following command.

```
db2 bind test_emb.bnd
```

Otherwise, you'll receive a -805 SQL error.

Note: More documentation can be found in *Developing Embedded SQL Applications*, SC23-5852-00

Administrative API

The *administrative API* offers an application interface for different programming languages. It allows you to perform a variety of administrative commands on the database from your application. For instance, you can start the database or you can create table spaces.

The API is described in the Information Center in **Database administration** → **Administrative interfaces** → **Administrative APIs** and in *Administrative API Reference*, SC23-5842-00.

UDF in C

One of the supported programming languages for User Defined Functions (UDFs) is C. UDFs are a good choice if you want to access the same business logic from different applications and if it is difficult to share a common API. After the program code has been compiled the command **CREATE FUNCTION** is used to register a function into the database manager.

Example 8-13 shows a UDF written in C. It converts a string to lowercase.

Example 8-13 UDF example written in C - tolower.c

```
#include <stdio.h>
#include <string.h>
#include <sqludf.h>

void SQL_API_FN
to_lower(SQLUDF_VARCHAR *value,
          SQLUDF_VARCHAR *result,
          SQLUDF_NULLIND *value_ind,
          SQLUDF_NULLIND *result_ind,
          SQLUDF_TRAIL_ARGS)
{
    int i, len;

    if(*value_ind == -1) {
        *result_ind = -1;
        return;
    }
}
```

```

    }

    len = strlen(value);

    for(i = 0; i < len; i++) {
        result[i] = tolower(value[i]);
    }

    result[i] = '\0';
}

```

In Example 8-14 on page 439 we compile this UDF. The first **cc** step compiles the program and the second **cc** step creates a dynamic object. On 64-bit machines, use the options **-m64** and **lib64** instead.

Example 8-14 Compiling a UDF

```

db2inst1@puget:~> SQLL=/home/db2inst1/sqllib
db2inst1@puget:~> cc -m32 -fpic -I$SQLL/include -c tolower.c -D_REENTRANT
db2inst1@puget:~> cc -m32 -fpic -shared -o tolower tolower.o
-Wl,-rpath,$SQLL/lib32 -L$SQLL/lib32 -ldb2 -lpthread
db2inst1@puget:~> cp tolower $SQLL/function

```

The resulting executable `tolower` was copied to the directory `<instance home>/sqllib/function`. From there the database manager will access it once the function is registered by the `CREATE FUNCTION` command as shown in Example 8-15.

Example 8-15 Activate the UDF

```

CREATE FUNCTION to_lower(VARCHAR(40)) RETURNS VARCHAR(40)
FENCED
EXTERNAL NAME 'tolower!to_lower'
NOT VARIANT NO SQL PARAMETER STYLE DB2SQL LANGUAGE C
NO EXTERNAL ACTION;

```

After that it can be used as demonstrated in Example 8-16.

Example 8-16 Execute the UDF

```

db2inst1@puget:~> db2 "select to_lower('AAA') from sysibm.sysdummy1"

1
-----
aaa

1 record(s) selected.

```

If you want to drop the function, you can run the command:

```
db2 "DROP FUNCTION to_lower(VARCHAR(40))"
```

UDFs can be run in *fenced* or *not fenced* mode. Fenced routines run as a separate process. They have some communication overhead involved. Not fenced routines run as part of the database server. They are a little bit more efficient than fenced routines. On the other side if a not fenced routine crashes it will bring down the database server too.

Therefore it is suggested to develop routines in fenced mode. Well tested routines on production systems can be considered to be run as not fenced if performance is important.

Fenced routines can also be run under a different user ID. That allows to secure them from the other instance files.

While developing the UDFs you should set the database manager parameter KEEPFENCED to NO. So each call to the UDF will reload the function.

Note: More information about UDFs can be found in *Developing User-defined Routines (SQL and External)*, SC23-5855-00.

8.3.4 Java

Java is an ideal language for writing programs that will run on multiple platforms. Unlike languages, such as C or C++, Java is completely specified and each Java-enabled platform supports a known core of libraries. One such library is JDBC, which you can think of as a Java version of ODBC. Using Java in conjunction with JDBC allows you to write portable database applications.

Note: You can find information about programming Java for DB2 in *Developing Java Applications*, SC23-5853-00.

JDBC

Because of their portability Java applications can be developed on any system. If you want to develop Java programs under Linux you can use different tools like Data Studio, Eclipse, Rational Developer, Netbeans, or just a plain JDK from Sun™.

Example 8-17 demonstrates the development of a small JDBC based Java program using the plain SDK and Type 4 driver. It selects data from the EMPLOYEE table in the SAMPLE database.

First the driver is loaded by the driver manager. Then a connection is created based on the *driver URL*:

```
jdbc:db2://<hostname>:<port>/<database>
```

The application then performs a SELECT on the table and loops through the result set.

Before running the example, the *<port>* must be replaced by the port number of your instance and *<password>* by the actual password.

Example 8-17 Sample JDBC Java program - test_java.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.ResultSet;
import java.sql.Statement;

public class test_java {
    public static void main(String[] args) {
        try {

            Class.forName("com.ibm.db2.jcc.DB2Driver");

            Connection con = DriverManager.getConnection(
                "jdbc:db2://localhost:<port>/SAMPLE",
                "db2inst1",
                "<password>");

            Statement stmt = con.createStatement();
            ResultSet rs = stmt.executeQuery("select empno, salary from employee"+
                " where salary < 36000 order by lastname");

            while(rs.next()) {
                System.out.println(rs.getString("empno") + "," + rs.getDouble("salary"));
            }

            con.close();
        } catch(SQLException e) {
            System.out.println(e.getMessage() + "\nSQLCode=" + e.getErrorCode()
                + ", SQLState=" + e.getSQLState());
        } catch(ClassNotFoundException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Example 8-18 shows the steps to compile and run this program.

Example 8-18 Compile and run the Java application - test_java.java

```
db2inst1@puget:~> DB2JAVA=/home/db2inst1/sql1lib/java
db2inst1@puget:~> $DB2JAVA/jdk32/bin/javac test_java.java
db2inst1@puget:~> $DB2JAVA/jdk32/bin/java -classpath $DB2JAVA/db2jcc.jar
test_java.class
200340,31840.0
000290,35340.0
000310,35900.0
200310,35900.0
200330,35370.0
```

Note that no DB2 client is needed for running this Java program. Only db2jcc.jar is required to be in the CLASSPATH. On 64-bit systems, you have to replace the jdk32 path with the jdk64 path.

Example 8-17 on page 441 uses the driver db2jcc.jar which is a Type 4 driver and is the suggested driver type. Type 2 driver is also available but is deprecated. The name of the class file for the Type 2 driver is db2java.zip.

DB2 9.5 supports JDBC 3.0 and JDBC 4.0. If you want to use JDBC 4.0 you have to use the driver db2jcc4.jar.

SQLJ

SQLJ allows you to embed SQL statements into the Java code. Example 8-19 shows an SQLJ program. You have to replace *<port>* and *<password>* by the actual values.

Example 8-19 SQLJ example program - test_sqlj.sqlj

```
import sqlj.runtime.*;
import java.sql.*;

#sql context EzSqljCtx;
#sql iterator EzSqljNameIter (String empno, Double salary);

public class test_sqlj {
    public static void main(String[] args) throws ClassNotFoundException {
        try {

            Class.forName("com.ibm.db2.jcc.DB2Driver");
            Connection con =
                DriverManager.getConnection("jdbc:db2://localhost:<port>/SAMPLE",
                                           "db2inst1",
                                           "<password>");

            EzSqljCtx ctx = new EzSqljCtx(con);
```

```

        EzSqljNameIter iter;

        #sql [ctx] iter = {select empno, salary from employee where salary <
36000 order by lastname};

        while(iter.next()) {
            System.out.println(iter.empno() + "," + iter.salary());
        }

        iter.close();
        ctx.close();

    } catch(SQLException e) {
        System.out.println(e.getMessage() + "\nSQLCode=" + e.getErrorCode()
            + ", SQLState=" + e.getSQLState());
    } catch(ClassNotFoundException e) {
        System.out.println(e.getMessage());
    }
}
}

```

Example 8-20 shows how to compile and run the SQLJ program.

Example 8-20 Compile and run an SQLJ program

```

db2inst1@puget:~> DB2JAVA=/home/db2inst1/sqllib/java
db2inst1@puget:~> export PATH=$DB2JAVA/jdk32/bin:$PATH
db2inst1@puget:~> sqlj test_sqlj.sqlj
db2inst1@puget:~> db2sqljcustomize -url jdbc:db2://localhost:<port>/SAMPLE
-user db2inst1 -password <password> test_sqlj_SJProfile0.ser
[jcc][sqlj]
[jcc][sqlj] Begin Customization
[jcc][sqlj] Loading profile: test_sqlj_SJProfile0
[jcc][sqlj] Customization complete for profile test_sqlj_SJProfile0.ser
[jcc][sqlj] Begin Bind
[jcc][sqlj] Loading profile: test_sqlj_SJProfile0
[jcc][sqlj] Driver defaults(user may override): BLOCKING ALL VALIDATE BIND
STATICREADONLY YES
[jcc][sqlj] Fixed driver options: DATETIME ISO DYNAMICRULES BIND
[jcc][sqlj] Binding package TEST_S01 at isolation level UR
[jcc][sqlj] Binding package TEST_S02 at isolation level CS
[jcc][sqlj] Binding package TEST_S03 at isolation level RS
[jcc][sqlj] Binding package TEST_S04 at isolation level RR
[jcc][sqlj] Bind complete for test_sqlj_SJProfile0
db2inst1@puget:~> $DB2JAVA/jdk32/bin/java -classpath $DB2JAVA/db2jcc.jar:.
test_sqlj
200340,31840.0
000290,35340.0
000310,35900.0

```

200310,35900.0
200330,35370.0

On 64-bit systems the path jdk32 needs to be replaced by the path jdk64.

If you want to run the java code on another database, you have to bind the package there first. For that you need the file with the suffix .ser and run the following command:

```
db2sqljbind -url jdbc:db2://localhost:<port>/SAMPLE -user db2inst2 -password
<password> test_sqlj_SJProfile0.ser
```

Like embedded SQL programs in C, SQLJ also use a precompiled access path at runtime. Therefore, no compilation of the SQL statements happens at runtime. You can also grant access to the users on package level instead on table level.

Stored procedures in Java

Only IBM JDKs are supported to create and run Java UDFs and stored procedures. Non-IBM versions of the SDK for Java are supported only for building and running standalone Java applications.

In the database manager configuration, the JDK_PATH parameter must point to the directory where your SDK is installed. You can view it by running:

```
db2 get dbm config | grep JDK_PATH
```

It should show:

```
Java Development Kit installation path      (JDK_PATH) =
/home/db2inst2/sqllib/java/jdk32
```

The path may differ on your local system depending on your instance name and whether you run on a 32-bit or a 64-bit system.

Now we can start writing a small UDF in Java. Example 8-21 shows a simple Java UDF that receives one string as input and sends back a concatenated string to the output.

Example 8-21 Java UDF sample - hello.java

```
public class hello {
    public static String hello(String param1) {
        return "Hello to " + param1;
    }
}
```

Example 8-22 shows how to compile and install the program hello.java. <path> has to be replaced by the path name of the jar file.

Example 8-22 Compile and install hello.java as UDF

```
db2inst1@puget:~> JRE=/home/db2inst1/sqllib/java/jdk32
db2inst1@puget:~> $JRE/bin/javac hello.java
db2inst1@puget:~> $JRE/bin/jar cvf hello.jar hello.class
added manifest
adding: hello.class(in = 443) (out= 274)(deflated 38%)
db2inst1@puget:~> db2 "CALL sqlj.install_jar('file:<path>/hello.jar',
'HELLOJAR')"
DB20000I The CALL command completed successfully.
```

On 64-bit systems you have to use jdk64 instead of jdk32. Next, we create a UDF in DB2 use this Java routine as shown in Example 8-23.

Example 8-23 Create the UDF in DB2

```
CREATE FUNCTION hello(VARCHAR(40)) RETURNS VARCHAR(60)
FENCED
EXTERNAL NAME 'HELLOJAR:hello.hello'
NOT VARIANT NO SQL PARAMETER STYLE java LANGUAGE java
NO EXTERNAL ACTION
```

The external name references the jar file HELLOJAR which was registered by the call **sqlj.install_jar**. The first **hello** refers to the class name and the second **hello** refers to the method name.

Now we can run the UDF as shown in Example 8-24.

Example 8-24 Running the java UDF

```
db2inst1@puget:~> db2 "select hello('Java Guru') from sysibm.sysdummy1"

1
-----
Hello to Java Guru

1 record(s) selected.
```

When developing UDFs, we recommend that you set the database manager parameter **KEEPFENCED** to **NO** so the UDF will be reload in each call.

For troubleshooting UDF, check db2diag.log for a Java stack trace.

8.4 pureXML

XML becomes more and more important for Web, Java, and J2EE development. It is used to store configuration information or to exchange data in a system independent way. DB2 has now implemented native XML support. This section provides an overview over the XML capabilities.

8.4.1 Storage model

Prior to DB2 9.1, XML could be stored in DB2 in the follow ways:

- ▶ Store as CLOB in the database:

This method preserves the original XML structure but you cannot search by attribute. If you want to extract a particular attribute you have to parse the whole XML from the CLOB. In order to implement search and extract more efficiently, you could create separate columns for the attributes. But that would increase the needed space and would add maintenance overhead.

- ▶ Store all attributes in the relational data model:

Using this method, you parse the XML document to insert into the database. To have the original XML document structure, recreating is required. This will work out for simpler XML structures. For more complex and more nested XML structures, you have to create a much more complex relational data model.

DB2 pureXML supports the native storage of XML data in the DB2 database. It allows you to work on the XML attributes using SQL statements or XQuery directly. You can create indexes on particular XML attributes based on XQuery expressions.

Example 8-25 shows a simple XML document.

Example 8-25 Simple XML document - products.xml

```
<products>
  <product xmlns="http://posample.org" pid="10">
    <description>
      <name>Fleece jacket</name>
      <price>19.99</price>
    </description>
  </product>
  <product xmlns="http://posample.org" pid="11">
    <description>
      <name>Nylon pants</name>
      <price>9.99</price>
    </description>
  </product>
```

</products>

DB2 pureXML store the XML data is stored in database in a tree structure. It does not store the XML document in text format but parses the document and stores the attribute values in a tree. The attribute names are not stored in the tree but in a lookup table. This storage model provides efficient data retrieval. The DB2 components such as indexes, explain, the programming languages, and the optimizer support this XML data model too.

Figure 8-3 on page 447 shows the storage layout for the XML document in Example 8-25.

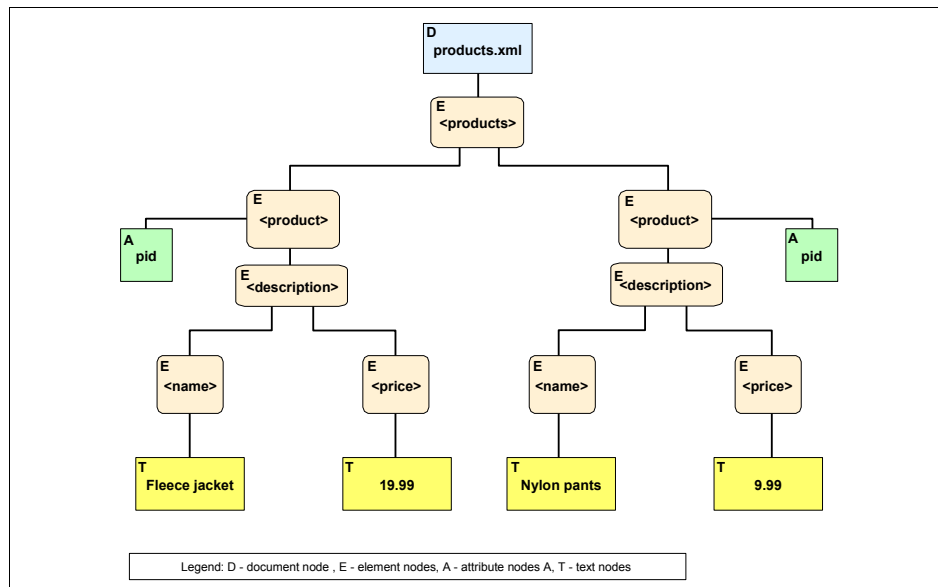


Figure 8-3 Storage layout for products.xml

8.4.2 Using XML data

In this section, we demonstrate, by examples, how XML data can be used in DB2. Example 8-26 shows the DDL to create a simple XML table PRODUCTS for storing document ID, timestamp, and XML document.

Example 8-26 Create a table with an XML column

```
CREATE TABLE product (
    entityid  INTEGER NOT NULL,
    lastchange  TIMESTAMP NOT NULL,
    xdata      XML,
    PRIMARY KEY (entityid)
```

);

We insert a few records into the PRODUCT table as shown in Example 8-27.

Example 8-27 insert.dml - insert XML data into the product table

```
INSERT INTO product
(entityid,lastchange,xdata)
VALUES
(0,CURRENT_TIMESTAMP,'<description><name>Fleece
jacket</name><price>19.99</price></description>'),
(1,CURRENT_TIMESTAMP,'<description><name>Nylon
pants</name><price>9.99</price></description>'),
(2,CURRENT_TIMESTAMP,'<description><name>Silk
pants</name><price>9.99</price></description>'),
(3,CURRENT
TIMESTAMP,'<description><name>Jean</name><price>7.99</price></description>'),
(4,CURRENT
TIMESTAMP,'<description><name>Shirt</name><price>3.99</price></description>')
;
```

Now we can start querying the data. Example 8-28 retrieves the product names for all products where the price is less than 9 in XML format. The query is specified in XQuery syntax.

Example 8-28 Retrieve columns in XML format by using XQuery - xquery_example.sql

```
xquery
for $i in db2-fn:xmlcolumn('PRODUCT.XDATA')/description/name[../price < 9]
return $i ;
```

Example 8-29 shows the query execution and the result in XML format. Note that for better readability, we have stripped the blanks in the output using the **cut** command.

Example 8-29 Result of the XQuery

```
db2inst1@puget:~> db2 -x -tf xquery_example.sql | cut -c1-30
<name>Jean</name>
<name>Shirt</name>
```

You can also use SQL statement to retrieve data with the XQuery expression in the FROM clause as shown in Example 8-30. The function **XMLEXISTS** matches all records from the XQuery expression.

Example 8-30 Match XML records in the FROM clause - select_xquery.sql -

```
SELECT entityid
```

```
FROM product
WHERE
    XMLEXISTS(
        '$i/description[price < 9]'
        PASSING xdata AS "i"
    ) ;
```

The output is shown in Example 8-31 on page 449.

Example 8-31 Result of the script select_xquery.sql

```
db2inst1@puget:~> db2 -tf select_xquery.sql
```

```
ENTITYID
-----
      3
      4

      2 record(s) selected.
```

In Example 8-32 we retrieve the result of the XQuery as a table and join it up to other relational tables directly.

Example 8-32 Retrieve XML data as a table - select_table.sql

```
SELECT
    p.entityid,
    x.*
FROM
    product p,
    XMLTABLE(
        '$i/description[price < 9]' PASSING p.xdata AS "i"
        COLUMNS "name" CHAR(30) PATH 'name',
                "price" DECIMAL(9,2) PATH 'price'
    ) AS x
;
```

The function **XMLTABLE** converts the XML document into a table. You can see the result in Example 8-33. The column ENTITYID comes from the relational table and the columns name and price come from the XML data.

Example 8-33 Retrieve the XML data as table

```
db2inst1@puget:~> db2 -tf select_table.sql
```

```
ENTITYID    name                                price
-----
      3 Jean                                7.99
```

4 Shirt

3.99

2 record(s) selected.

Example 8-34 shows an enhanced XQuery example which retrieves an HTML list.

Example 8-34 XQuery example with a loop - select_xq.sql

```
VALUES
  XMLSERIALIZE(
    XMLQUERY(
      '<ul>
      {
        for $x in
          db2-fn:xmlcolumn("PRODUCT.XDATA")/description/name[../price < 9]
        return <li>{data($x)}</li>
      }
      </ul>'
    )
  AS CLOB(70)
);
```

We use the function **XMLSERIALIZE** to convert the data into a CLOB. The function **XMLQUERY** retrieves the result of the XQuery expression. The result is shown in Example 8-35.

Example 8-35 Result of XQuery example

```
db2inst1@puget:~> db2 -tf select_xq.sql
```

```
1
-----
<ul><li>Jean</li><li>Shirt</li></ul>
```

1 record(s) selected.

As you can see, XML is seamless integrated into DB2. You can convert between the relational and the XML structure. The query in Example 8-34 on page 450 can be embedded into PHP very easily.

In Example 8-36 we retrieve relational data into an XML structure.

Example 8-36 Extract XML structure from a table - extract_xml.sql

```
SELECT
  XMLELEMENT(NAME "employee",
    XMLELEMENT(NAME "empno", empno),
```

```

        XMLELEMENT(NAME "lastname",lastname),
        XMLELEMENT(NAME "firstname",firstnme),
        XMLELEMENT(NAME "salary",salary)
    )
FROM
    employee
WHERE
    salary < 37000;

```

The function **XMLEMENT** is used to construct the XML data structure. You can see the output in Example 8-37.

Example 8-37 Example for extracted XML

```

db2inst1@puget:~> db2 -x -tf extract_xml.sql | cut -c1-130
<employee><empno>000280</empno><lastname>SCHNEIDER</lastname><firstname>ETHEL</f
firstname><salary>0036250.00</salary></employee>
<employee><empno>000290</empno><lastname>PARKER</lastname><firstname>JOHN</firs
tname><salary>0035340.00</salary></employee>
<employee><empno>000310</empno><lastname>SETRIGHT</lastname><firstname>MAUDE</f
irstname><salary>0035900.00</salary></employee>
<employee><empno>200310</empno><lastname>SPRINGER</lastname><firstname>MICHELLE
</firstname><salary>0035900.00</salary></employee>
<employee><empno>200330</empno><lastname>WONG</lastname><firstname>HELENA</firs
tname><salary>0035370.00</salary></employee>
<employee><empno>200340</empno><lastname>ALONZO</lastname><firstname>ROY</first
name><salary>0031840.00</salary></employee>

```

As a last example we demonstrate the usage of XSLT. In Example 8-38 we create an auxiliary table for storing the XSLT document. The record containing the XSLT document is used later for the transformation.

Example 8-38 Store the XSLT document- create_xslt.ddl

```

CREATE TABLE xslt (
    doc_id INTEGER NOT NULL,
    xslt XML NOT NULL,
    PRIMARY KEY(doc_id)
);

INSERT INTO xslt VALUES (
1, '
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="products">
    <html>
    <body>
        <h1>List of items</h1>
        <table border="1">

```

```

        <tr bgcolor="#cccccc">
            <th align="left">Product</th>
        </tr>
    </xsl:apply-templates/>
</table>
</body>
</html>
</xsl:template>
<xsl:template match="name">
    <tr><td><xsl:value-of select="."/></td></tr>
</xsl:template>
</xsl:stylesheet>'
);

```

In Example 8-39 we use query for applying the transformation.

Example 8-39 Apply XSLT - query_html.sql

```

VALUES
    SUBSTR(
        XSLTRANSFORM(
            XMLELEMENT(NAME "products",
                XMLQUERY(
                    'db2-fn:xmlcolumn("PRODUCT.XDATA")/description/name[../price < 9]'
                )
            ) USING (SELECT xslt FROM xslt WHERE doc_id = 1)
        ),1,200) ;

```

The function **XMLQUERY** retrieves the result set as XML documents. The function **XMLELEMENT** places the **<product>** attribute around the individual XML documents. Then **XSLTRANSFORM** is used to transform the document into the HTML output by using the XSLT.

Example 8-40 shows the output which is an HTML and can be used directly by a browser.

Example 8-40 Result of the XML transformation as text

```

db2inst1@puget:~> db2 -x -tf select6.sql
<html>
<body>
<h1>List of items</h1>
<table border="1">
<tr bgcolor="#cccccc">
<th align="left">Product</th>
</tr>
<tr>
<td>Jean</td>

```



```

</tr>
<tr>
<td>Shirt</td>
</tr>
</table>
</body>
</html>

```

Figure 8-4 on page 453 shows the HTML in a browser.

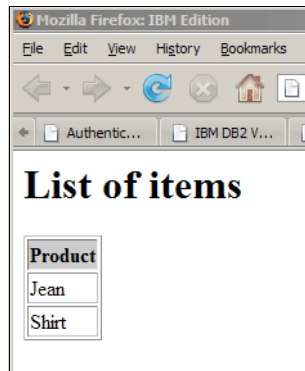


Figure 8-4 Result of the XSLT transformation in a Browser

The source for learning the details about pureXML are

- ▶ *pureXML Guide*, SC23-5871-00
- ▶ Information Center in **Database Fundamentals** → **pureXML**.
- ▶ *PureXML: DB2 9: pureXML Overview and Fast Start*, SG24-7298
- ▶ *DB2 9 pureXML Guide*, SG24-7315

Note: UDB 9.5 doesn't support pureXML in a partitioned database environment!

8.5 Application development tools

DB2 provides many robust tools for developing DB2 applications. In this section we cover the *explain* facility, *design advisor*, and application development using *IBM Data Studio*.

In addition to the development tools described in this section, the following tools are also supported:

- ▶ Rational Application Developer
- ▶ Microsoft Visual Studio®

- ▶ Eclipse 3.2
- ▶ Zend Framework (PHP development)
- ▶ Alphablox Blox Builder
- ▶ Ruby on Rails

For demonstrating the explain facility and the design advisor, we add a column EXPLTEST to the EMPMDC table in the SAMPLE database as shown in Example 8-41 on page 454.

Example 8-41 Add a column to EMPMDC for testing EXPLAIN

```
db2inst1@puget:~> db2 "alter table EMPMDC add column expltest character(20)"
DB20000I The SQL command completed successfully.
db2inst1@puget:~> db2 "update empmdc set expltest = 'TEST' where div = 1"
DB20000I The SQL command completed successfully.
db2inst1@puget:~> db2 runstats on table db2inst2.empmdc and indexes all
DB20000I The RUNSTATS command completed successfully.
```

This column is created without an index. We populate it with sample data and perform a runstats so that the index statistics are up to date. The new column is used to demonstrate the usage of explain and the design advisor.

As a second preparation step we create the explain tables running the script:

```
db2 -tvf <instance_dir>/sqllib/misc/EXPLAIN.DDL
```

8.5.1 Explain facility

Before DB2 executes an SQL statement, it creates an *access plan* that describes how DB2 accesses the data in order to generate the result set. For static SQL statements the access plan is stored in the related package. For dynamic SQL statements the access plan is generated at runtime.

If you want to tune a query, you can use explain facility to view the access plan to understand how DB2 accesses the tables to form the result set. You can then decide if the query performance can be improved by adding indexes or rearrange the query.

There are two tools for creating a visual access path, *Visual explain* and the *explain* command.

Note: The explain facility is described in detail in *Tuning Database Performance*, SC23-5867-00, in *Visual Explain Tutorial*, SC23-5868-00, and in the Information Center in **Database fundamentals** → **Performance tuning** → **Tuning database application performance** → **Explain facility**.

Explain command

Use the `explain` command from command line to get access plan. The following statement explains a `SELECT` query on the non indexed column on the `EMPMDC` table:

```
db2 "EXPLAIN ALL WITH SNAPSHOT FOR select count(*) from empmdc where expltest = 'TEST'"
```

We then use the `explain` command to format the output as follows:

```
db2exfmt -d sample -e db2inst1 -g TIC -o explain.out -w -1 -n '%' -s '%' -# 0
```

The output file `explain.out` contains the access path for the statement as well as other detailed information about the query processing. Example 8-42 shows an excerpt from the file with the access plan.

Example 8-42 Access path for an SQL statement

Access Plan:

```
-----
          Total Cost:          322.112
          Query Degree:        1

          Rows
          RETURN
          ( 1)
          Cost
          I/O
          |
          1
          GRPBY
          ( 2)
          322.112
          320
          |
          1000
          TBSCAN
          ( 3)
          322.029
          320
          |
          10000
          TABLE: DB2INST2
          EMPMDC
```

You can see that DB2 performs a table scan which takes longer time. Adding an index on the column `EXPLTEST` will allow DB2 to scan on index that can improve the query performance.

Visual explain

Visual explain can be started from the DB2 Control Center. It offers a graphical view for the access path. To start visual explain from the Control Center, in the explorer view, select the database with the right mouse button and select **Explain Query...** as shown in Figure 8-5 on page 456.

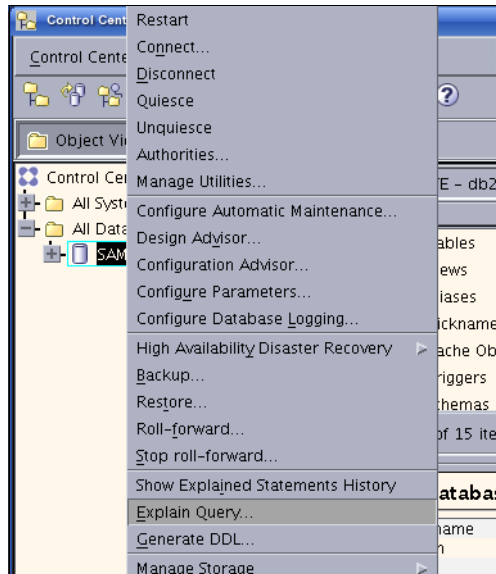


Figure 8-5 Select Visual explain in the Control Center

In the explain panel you can enter the query as shown in Figure 8-6.

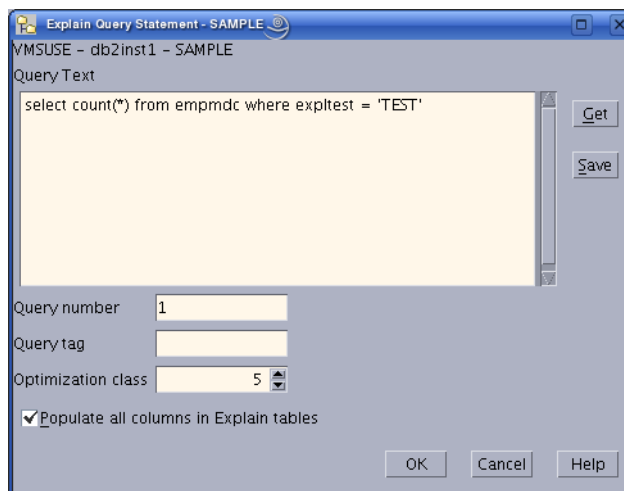


Figure 8-6 Explain panel

Figure 8-7 on page 457 shows the access path in graphical view. You can right click or double click on the graphical items to get more details.

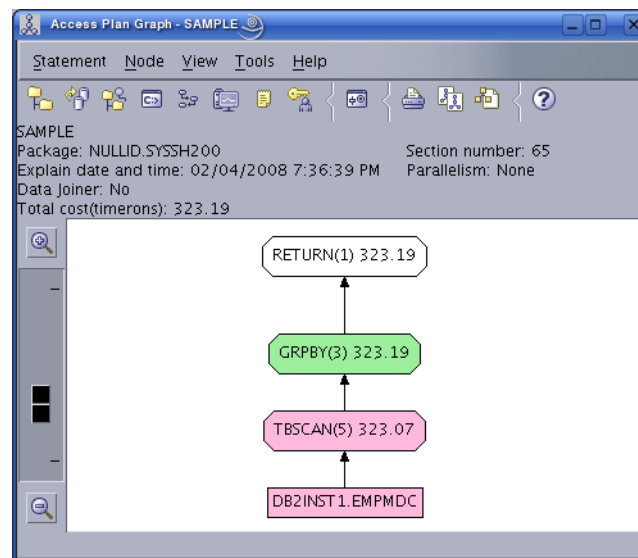


Figure 8-7 Access path in visual explain

8.5.2 Design advisor

The Design Advisor can help you significantly improve the performance of your queries. It takes a set of SQL statements called *workload* and suggests ways to improve the performance. The suggestions may include:

- ▶ New indexes
- ▶ New materialized query tables (MQTs)
- ▶ Clustering indexes
- ▶ Conversion to multidimensional clustering (MDC) tables
- ▶ Redistribution of tables
- ▶ Deletion of unused indexes and MQTs

There are two ways to call the Design Advisor, from the command line by using the command **db2adv** and from the Control Center.

Note: The Design Advisor is described in detail in *Tuning Database Performance*, SC23-5867-00 and in the Information Center in **Database fundamentals** → **Performance tuning** → **Tuning database performance** → **Design Advisor**.

db2advis

db2advis reads the workload from a file containing the SQL statements or from the command line directly. The following command calls the advisor for our sample query:

```
db2advis -d sample -s "select count(*) from empmdc where expltest = 'TEST'" -t
5
```

Example 8-43 shows an excerpt of the output of the db2advis command. In this example, the tool suggests an index on the column EXPLTEST and that would improve the performance by 97 percent.

Example 8-43 Output of the db2advis command

```
Recommending indexes...
total disk space needed for initial set [ 0.032] MB
total disk space constrained to [ 21.722] MB
Trying variations of the solution set.
Optimization finished.
  1 indexes in current solution
[322.0000] timerons (without recommendations)
[ 8.0000] timerons (with current solution)
[97.52%] improvement

--
--
-- LIST OF RECOMMENDED INDEXES
-- =====
-- index[1], 0.032MB
CREATE INDEX "DB2INST2"."IDX802051509060000" ON "DB2INST2"."EMPMDC"
("EXPLTEST" ASC) ALLOW REVERSE SCANS ;
COMMIT WORK ;
RUNSTATS ON TABLE "DB2INST2"."EMPMDC" FOR INDEX
"DB2INST2"."IDX802051509060000" ;
COMMIT WORK ;
```

Design Advisor

The graphical version of the Design Advisor can be started from the Control Center. To start it from the Control Center, in the explorer view, select the database with the right mouse button and select **Design Advisor...** as shown in Figure 8-8 on page 459.

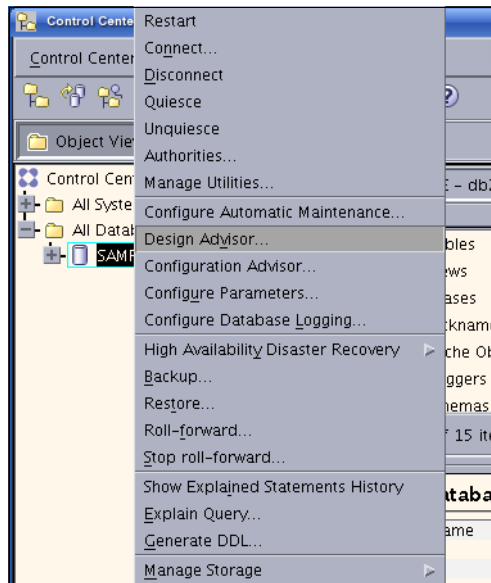


Figure 8-8 Select Design Advisor from the Command Center

The advisor leads you through the process panel by panel. At the Workload panel you can enter our test query:

```
select count(*) from empmdc where expltest = 'TEST'
```

After pressing Next at the Recommendations panel, the tool creates the suggestions for the provided workload. Figure 8-9 on page 460 shows that creating an index is recommended.

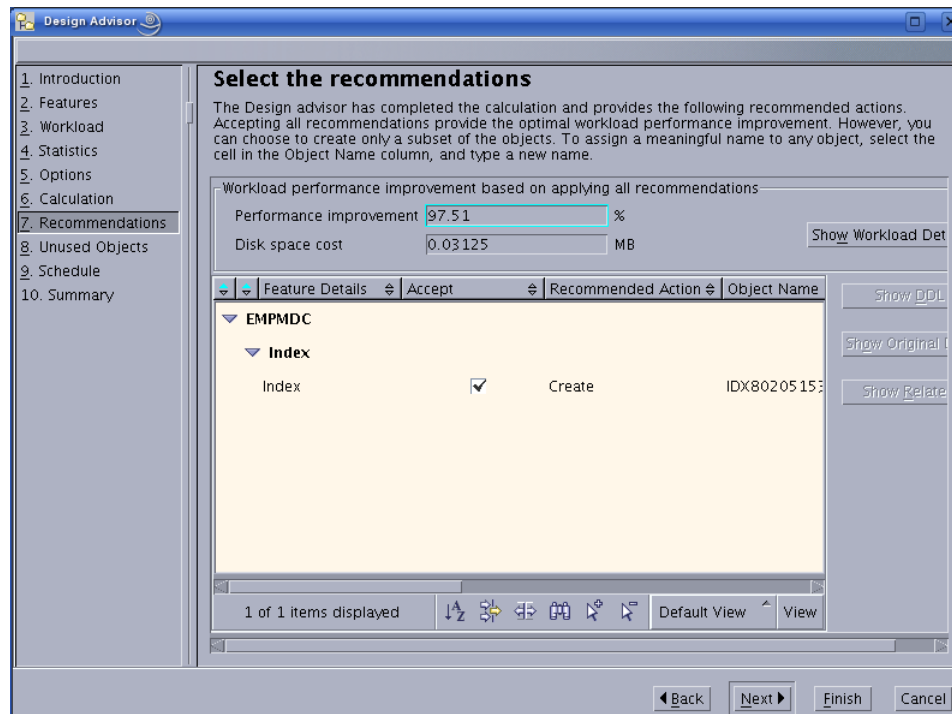


Figure 8-9 Recommendations in Visual Explain

The page *Unused Objects* shows indexes which are not needed for the evaluated workload.

8.5.3 Application development with the Data Studio

Data Studio, an *Eclipse* based tool, offers the features provided by Eclipse. For instance:

- ▶ Context dependent help
- ▶ Context assist
- ▶ Software Configuration Management with all compatible systems (CVS, CMVC, SVN and so on)
- ▶ Debugger
- ▶ For general information about Eclipse, use the following Web site
<http://www.eclipse.org/>

The Data Studio is a replacement of the Development Center and offers functions and features beyond application development. In regards to application development it offers the following additional features:

- ▶ Integrated Query Editor for SQL and XQuery
- ▶ SQLJ Editor
- ▶ SQL Builder:
The Query Builder allows you to create your queries by using Drag & Drop. You can test the queries directly by running them and viewing the results.
- ▶ SQL Routine Debugger
Allows you to debug SQL stored procedures.
- ▶ Java Routine Debugger
You can create your Java procedures, promote them to the server, and even debug them remotely on the database server as you are used to do with standalone Java applications.
- ▶ XML Editor
It allows you to edit your XML document in a tree structure. You do not need to deal with the XML syntax.
- ▶ XML Schema Editor
It allows you to edit your XML schemas in a graphical way. You do not need to deal with the XML syntax.
- ▶ Visual Explain:
It shows the access plan of your query so that you can troubleshoot performance problems.
- ▶ ER Diagramming:
You can create your ER diagrams directly from the database schema or vice versa. The connection between the diagrams and the database remains so that changes of the one are reflected in the other.
- ▶ Data Web Services:
With a few mouse clicks you can create a Web service based on a query, deploy it to an application server, or create a WAR file from it.
- ▶ pureQuery for Java
pureQuery offers an easy access level to the database. In relationship to JDBC, it simplifies many tasks. The editor GUI supports all the features you know from Java program editing. You can create, edit, context assist, and run queries directly by selecting them within the program code.

In this section, we implement and debug a simple Java stored procedure to demonstrate some of the Data Studio's powerful features. The general introduction of the Data Studio is provided in Chapter 5, "IBM Data Studio" on page 193.

Setting up a project

The Data Studio can be started with the command:

```
/opt/IBM/SDP70/eclipse
```

Alternatively you can start it by selecting the **Start menu** → **Development** → **IBM Data Studio Developer** or **Start menu** → **IBM Software Development Platform** as shown in Figure 8-10.

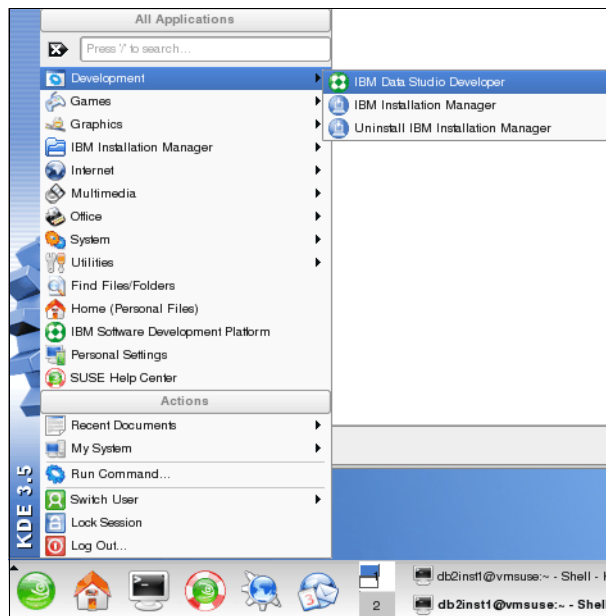


Figure 8-10 Start Data Studio from the Start menu

Take the default or choose another location for the workspace as shown in Figure 8-11 on page 463. If you open a workspace the first time, the Data Studio shows a help screen. Close the help will lead you to the Select a workspace panel.

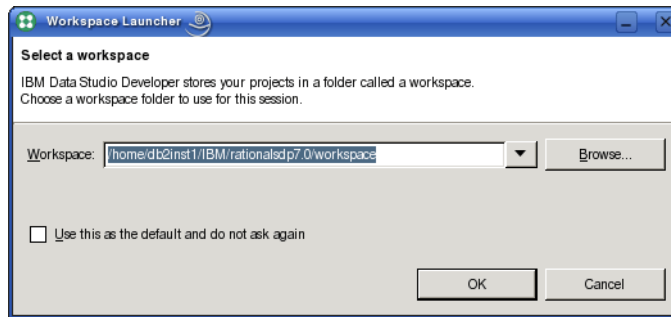


Figure 8-11 Workspace selection in Data Studio

To develop an application, start from setting up a project. Switch to the *Data Perspective* by selecting **Window** → **Open Perspective** → **Data** if you are not already there. We create a *Data Development Project* by selecting **File** → **New** → **Data Development Project**. We name it *DataProject* and take all the default values as shown in Figure 8-12. Press **Next**.

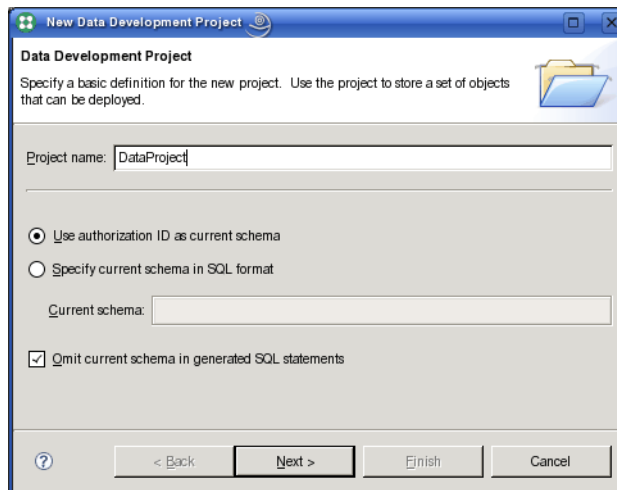


Figure 8-12 Create Project

Next we get to the connection panel. You can either select an existing connection or create a new one (Figure 8-13 on page 464).

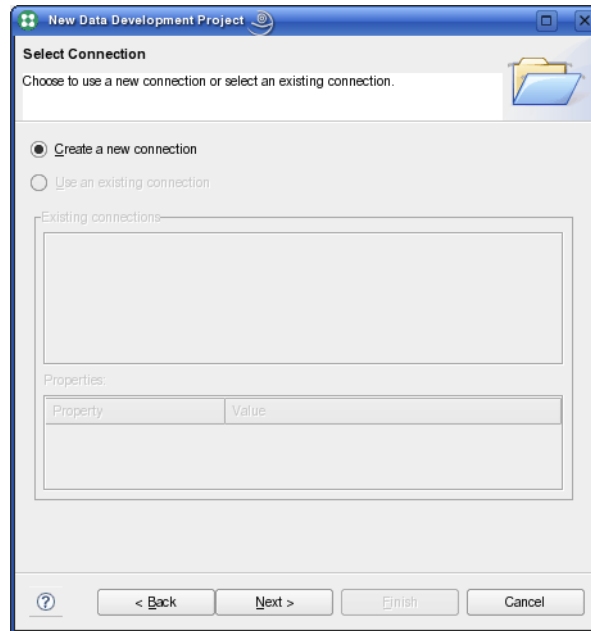


Figure 8-13 Select a new connection

We're going to select a new connection for the SAMPLE database. The we get to Figure 8-14 on page 465.

If you haven't had a database connection established yet, you'll get to the Connection Parameters panel automatically. In the Connection Parameters (Figure 8-14 on page 465), fill the connection information. We use the SAMPLE database for the demonstration. **Test Connection** allows you to test the connection before creating it. If database connection test successes, click **Finish**.

The new database connection shows up in the Database Explorer. In the explorer tree, you can view the database structure or perform database maintenance tasks.

If you have already had a database connection setup, select **Use an existing connection** select the desired connection and click **Next**. Then you'll get your project created and end up at Figure 8-16 on page 466. In our example we continue with Figure 8-14 on page 465.

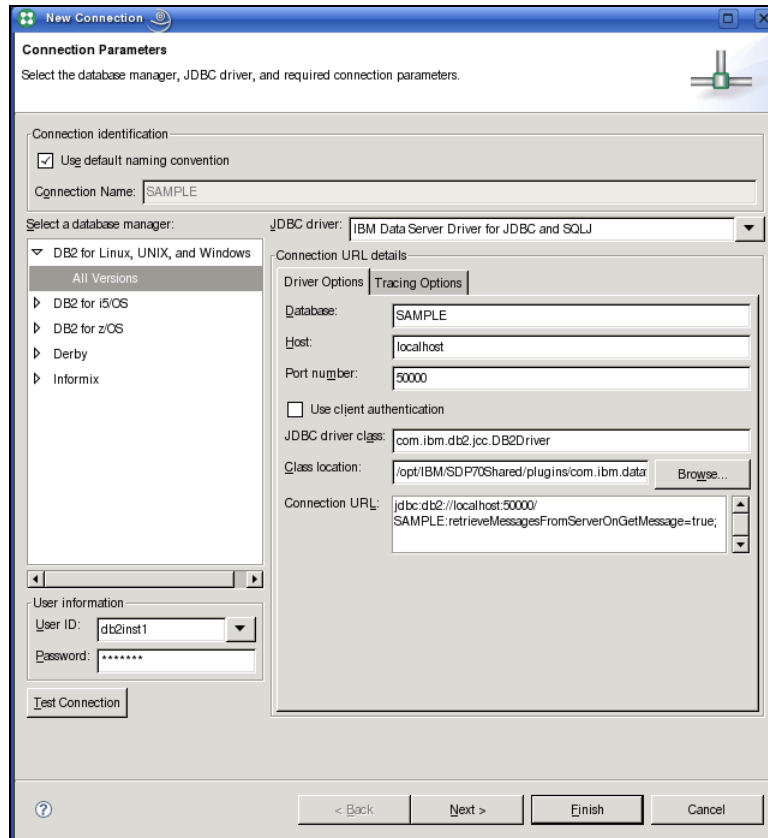


Figure 8-14 Create connection panel in Data Studio

Then press **Next**. At the schema selection panel. At this panel you can specify what schemas are displayed in the Database Explorer. We take the default (Figure 8-15 on page 466).

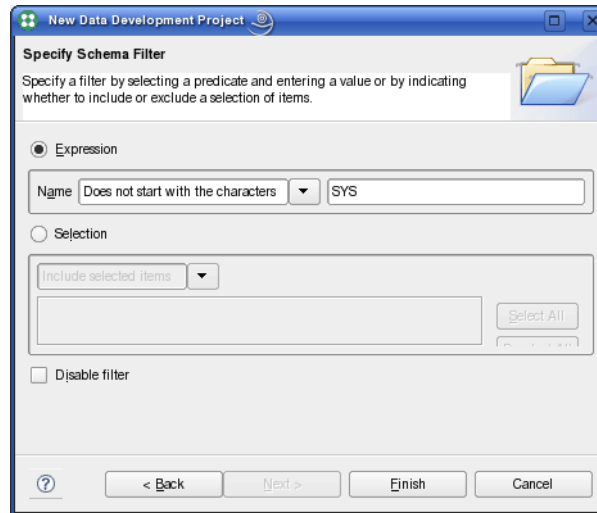


Figure 8-15 Schema selection panel

Now press **Finish** to create the connection and the project.

The newly created project DataProject is presented in the *Data Project Explorer*. See Figure 8-16.

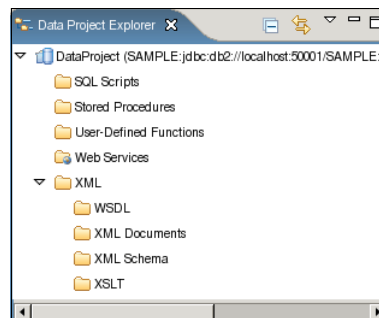


Figure 8-16 Data Explorer

Creating an SQL statement

With the SQL builder in Data Studio, you can easily create complicate SQL queries and XQuery scripts with out knowing the syntax details.

To create a query, follow these steps:

1. In Database Explorer view, expand the Connections, right mouse click on SAMPLE database, select **New SQL or XQuery Script...**

2. In the popup panel, enter a name for the query and select **SQL builder** as shown in Figure 8-17. We name the query EmpSalary. Press **OK**,

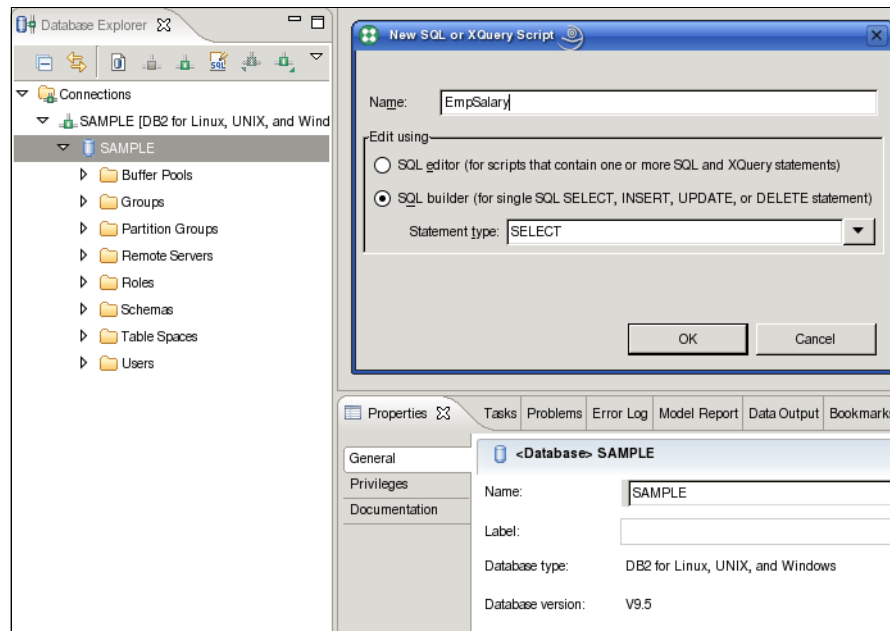


Figure 8-17 Create a query in Data Studio

3. The *Query Builder* view shows. From Database Explorer, expand the tree **Connections** → **SAMPLE** → **SAMPLE** → **Schemas** → **DB2INST1** → **Tables** and drag and drop the EMPLOYEE and DEPARTMENT tables into the middle pane of the Query Builder. The tables show up as boxes with the column names. You can move them around or resize them.
4. Create the join. Right-click on the EMPLOYEE table in the Query Builder and select **Create Join...**. A window pops up where you can select the source and the target of the join. The source table is EMPLOYEE, the source column is WORKDEPT, the target table is DEPARTMENT, and the target column is DEPTNO. Figure 8-18 on page 468 shows the Create Join panel.
After pressing **OK**, you can see a line between the tables in the middle pane.

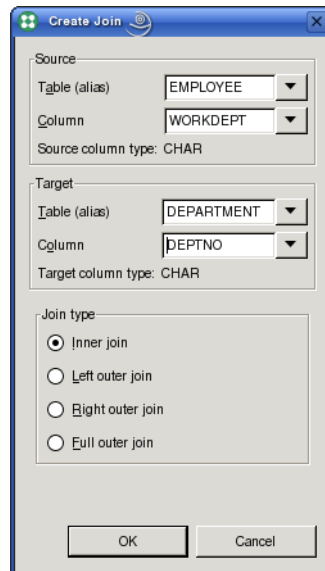


Figure 8-18 Create a join in the Query Builder

5. Select the columns for the query. Just check EMPNO, LASTNAME in the EMPLOYEE box and DEPTNO and DEPTNAME in the DEPARTMENT box in the middle pane.

At each of these actions, you can see how the query is constructing accordingly in the upper pane.

6. As a last step, we add a condition on the SALARY column. Select the **Conditions** tab on the bottom pane. Double click left mouse button on a field “opens” the field for you to enter a value or use the dropdown menu to choose one. We select EMPLOYEE.SALARY in the Column field, choose “<” for operator, and enter the value 37000 in the value column.

Now we have finished the steps. Figure 8-19 on page 469 shows the complete Query Builder window after our changes.

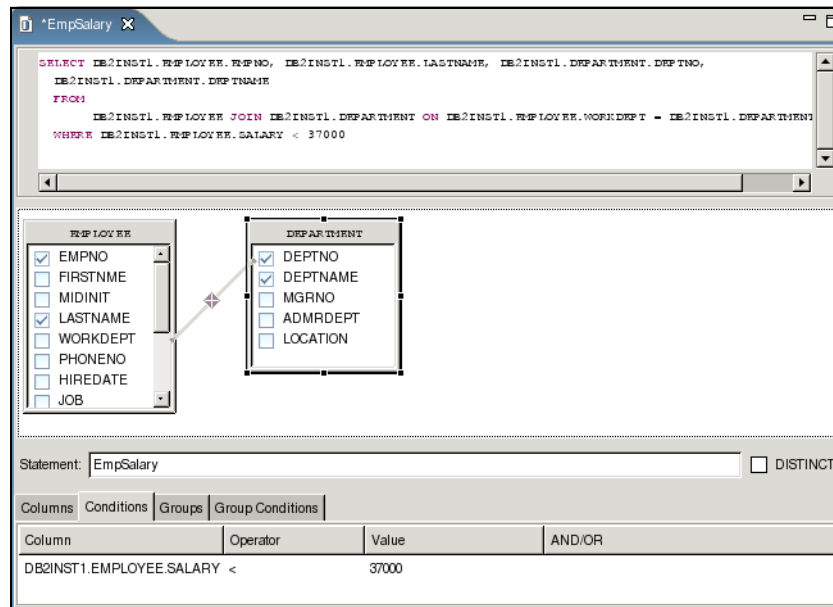


Figure 8-19 Query Builder

We can save our query. Select **File** → **Save**. In the panel, select our DataProject project as shown in Figure 8-20.

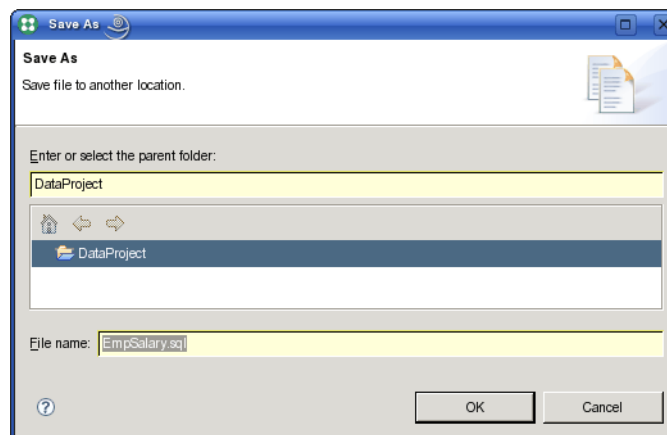


Figure 8-20 Save query in data project

The query shows up in the Data Project Explorer under the SQL Scripts folder. The query can be tested by right clicking on **EmpSalary.sql** and select **Run SQL**. The result is presented in the *Data Output* view as shown in Figure 8-21 on page 470.

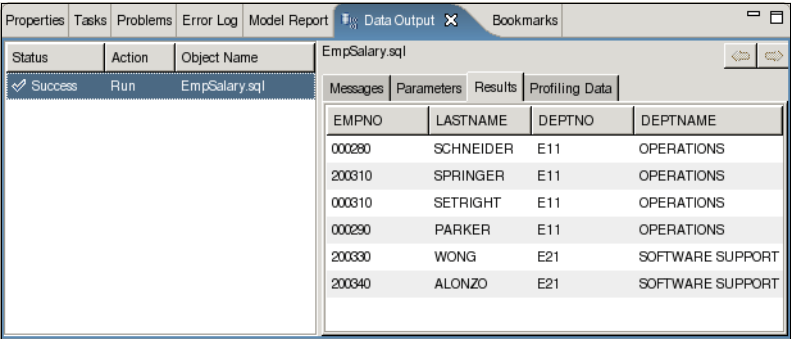


Figure 8-21 Data Output view

The Data Studio allows you to easily create and run queries. However, you are not limited to the graphical way to create queries. You can also enter the SQL statement as text directly. When creating your query manually, you can get help using *Content Assist* (Figure 8-22) which can be invoked by pressing the shortcut **Ctrl + Space** or right clicking on the query and selecting Content Assist.

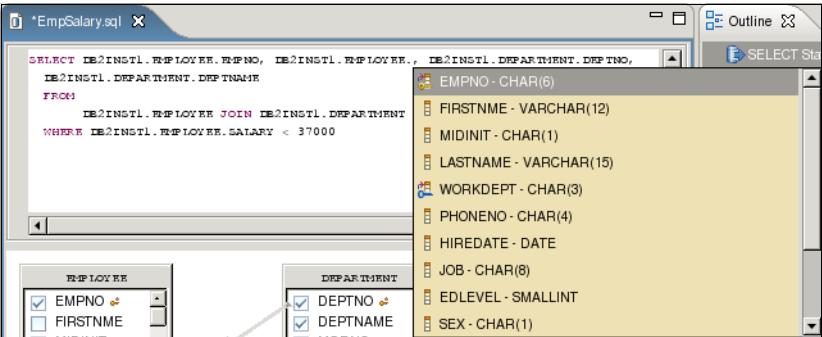


Figure 8-22 Content assist in Query Builder

Creating a stored procedure in Java

In “Stored procedures in Java” on page 444, we discussed how to create stored procedures from the command line. In this section we show how easy this task can be accomplished by using the Data Studio. The graphical aids provided by the tool makes this task much less error-prone.

To start the creation, in the Data Project Explorer, expand **DataProject** → **SQL Script-EmpSalary.sql**. Right click on it and select **Generate Stored Procedure...** We name the procedure **EMPSALARY** as shown in Figure 8-23 on page 471. We take the reset of default values to create a JDBC program. Press **Next**.

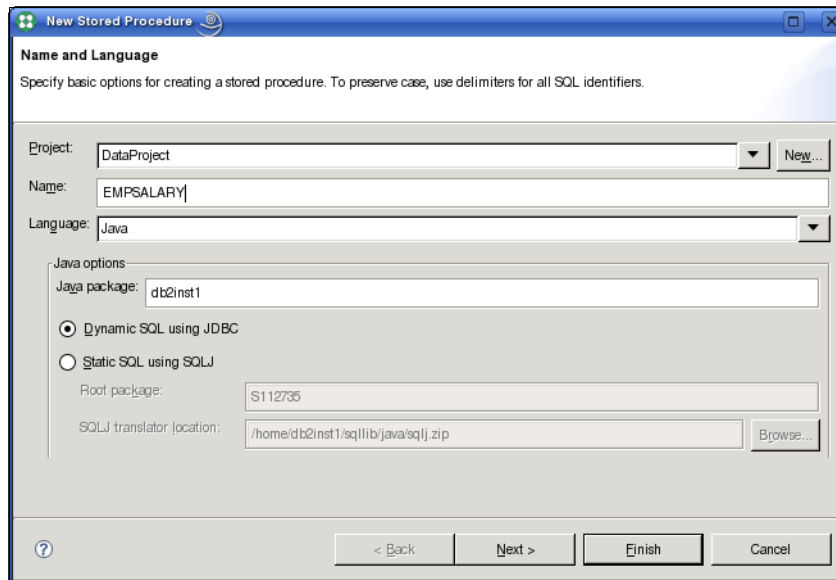


Figure 8-23 Create a stored procedure - Name and Language

In SQL Statement panel, we use all the default value as shown in Figure 8-24. Click **Next**.

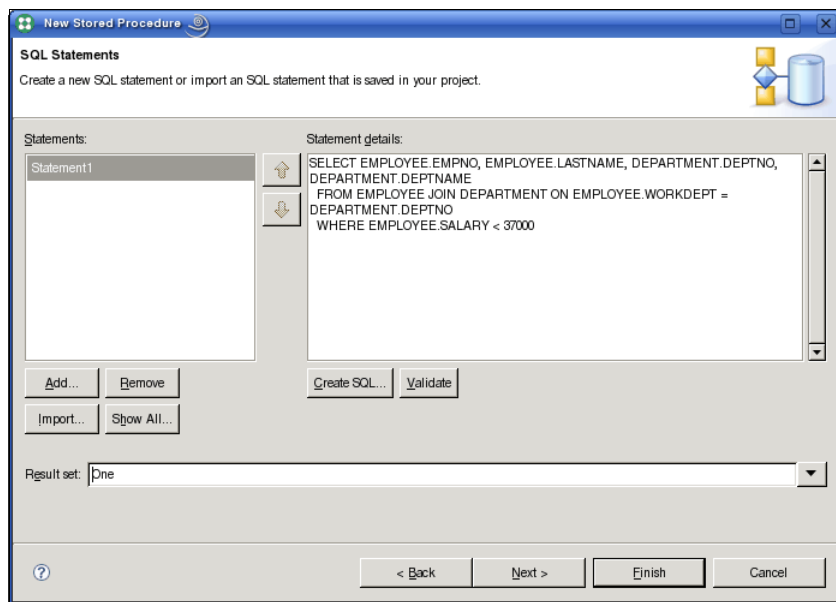


Figure 8-24 Create a stored procedure - SQL Statement

In Add Parameters panel (Figure 8-25), we add an integer parameter with the name SALARY and leave the rest of values on the panel unchanged. Press **OK**.

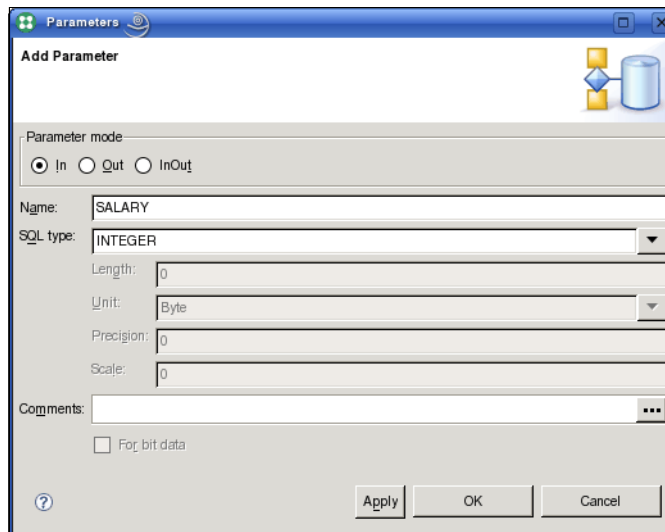


Figure 8-25 Add a parameter

We are done now with the Parameter panel in Figure 8-26. Press **Next**.

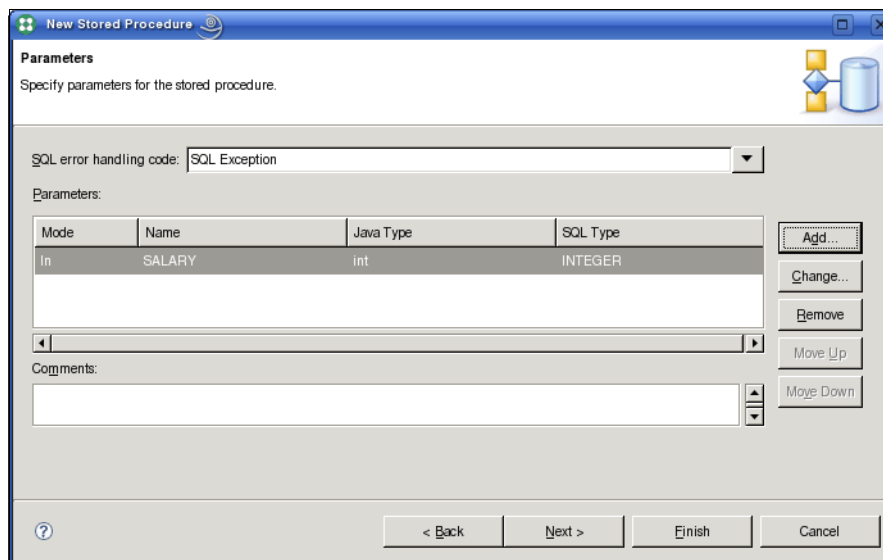


Figure 8-26 parameter panel

In the Deploy panel (Figure 8-27), enter EMPLOYEE in the Specific name field, check **Enable debugging**. Click **Next**.

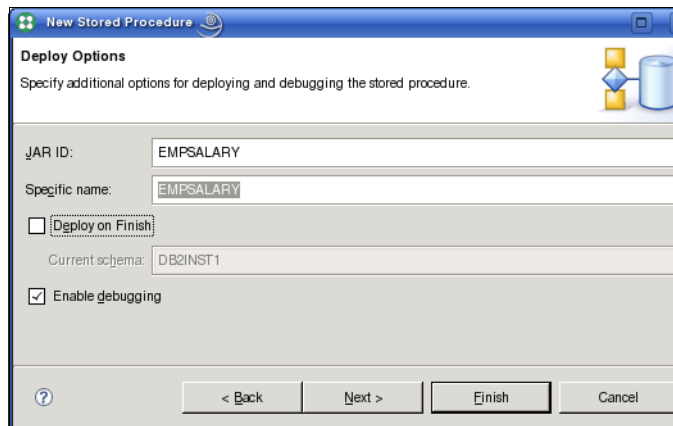


Figure 8-27 Deploy options for stored procedure

In the Code Fragments panel (Figure 8-28), you can enter additional files with code fragments which are to be included into the procedure. In our example, we do not have additional file to add. Click **Next**.

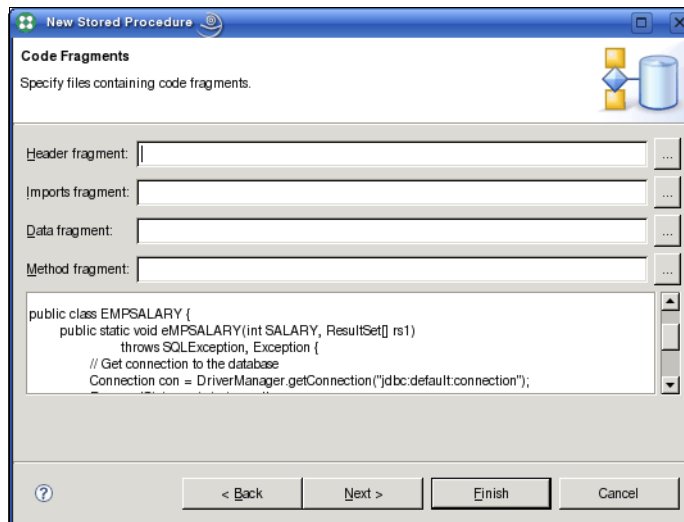


Figure 8-28 Code Fragment panel

In the summary panel (Figure 8-29 on page 474), press **Finish**.

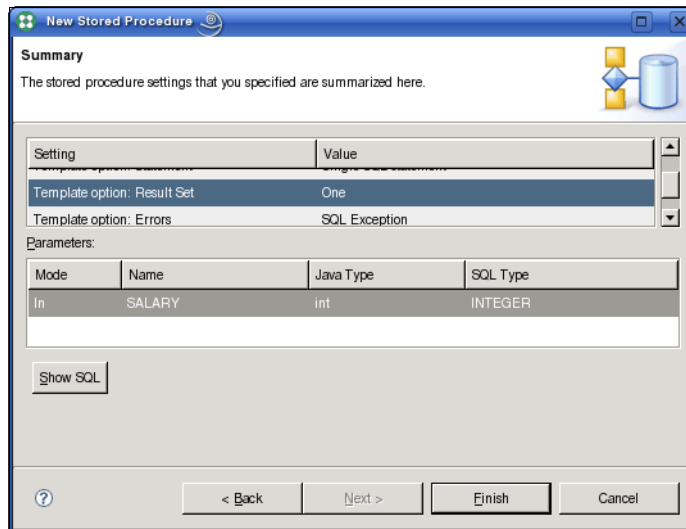


Figure 8-29 Summary page

The procedure will be created. It shows up in the Data Project Explorer under Stored Procedures and the code are presented. See Figure 8-30.

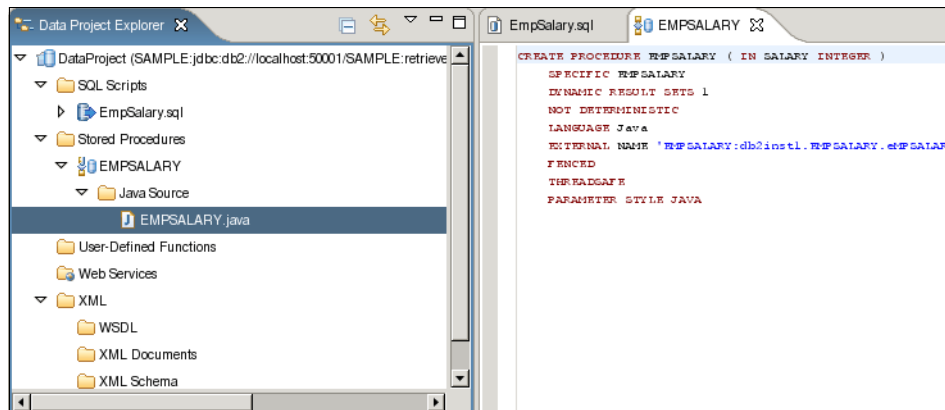


Figure 8-30 New procedure created

If you double-click onto the Java program **EMP_SALARY.java** in the Explorer the Data Studio opens the Java program. In the Java code, change this line

```
+ " WHERE EMPLOYEE.SALARY < 37000";
```

to

```
+ " WHERE EMPLOYEE.SALARY < " + SALARY;
```

Then save the file. You can see it in the editor window as shown in Figure 8-31.

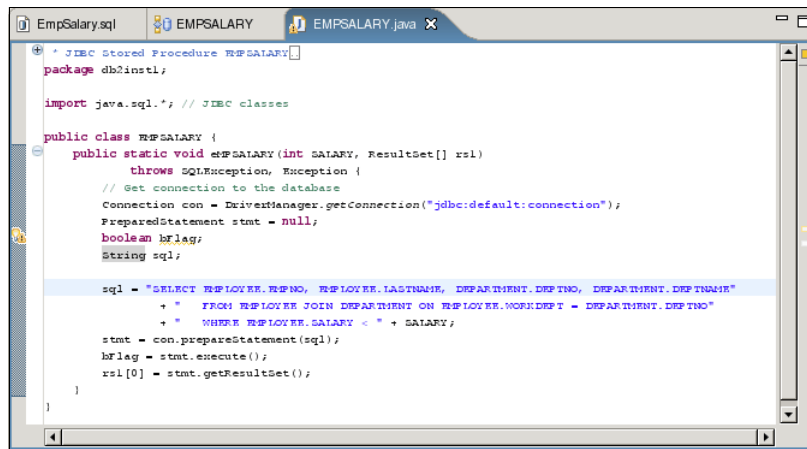


Figure 8-31 Java code of the stored procedure

Now we are ready to deploy the code to the database server. Expand the project **DataProject** → **Stored Procedures** → **EMPSALARY**, right-click on **EMPSALARY** and select **Deploy...** Keep the Deploy options as shown in Figure 8-32 on page 475. Press **Next**.

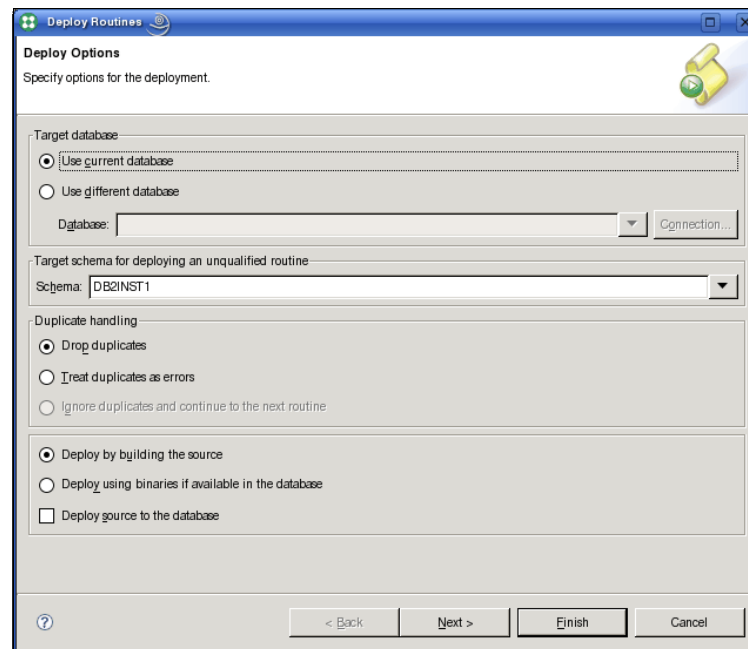


Figure 8-32 Deploy Options

In Routine Option, keep the defaults. See Figure 8-33. Press **Next**.

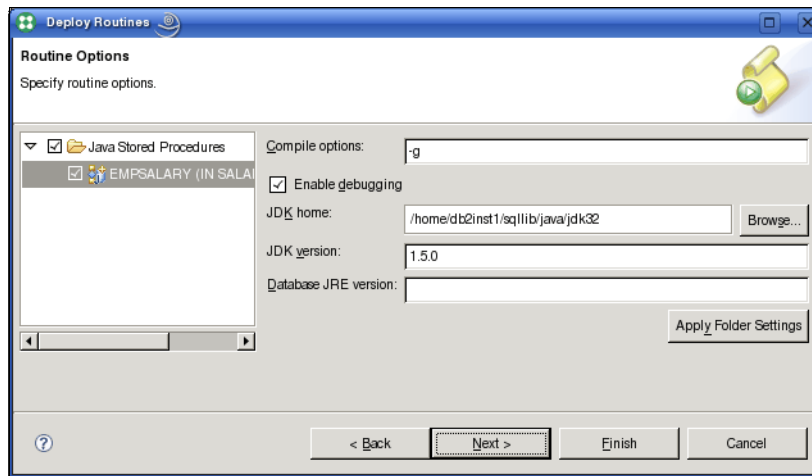


Figure 8-33 Routine options

At the Summary panel (Figure 8-34 on page 476), press **Finish**.

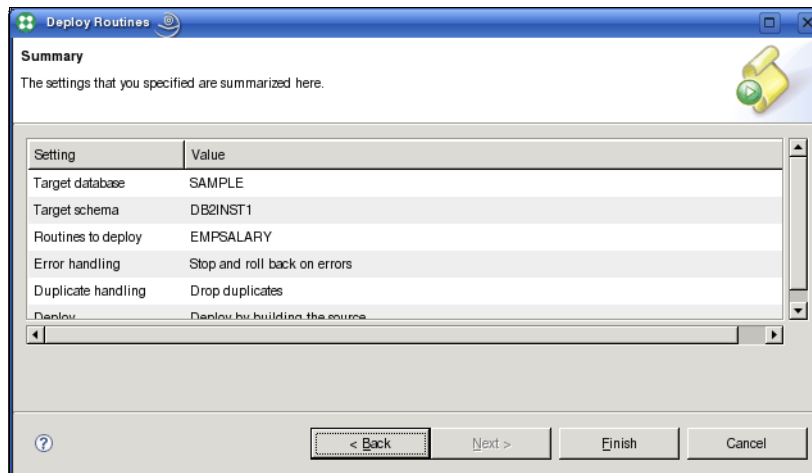


Figure 8-34 Summary panel

In the Data Output panel (Figure 8-35), you can see the result of the deployment.

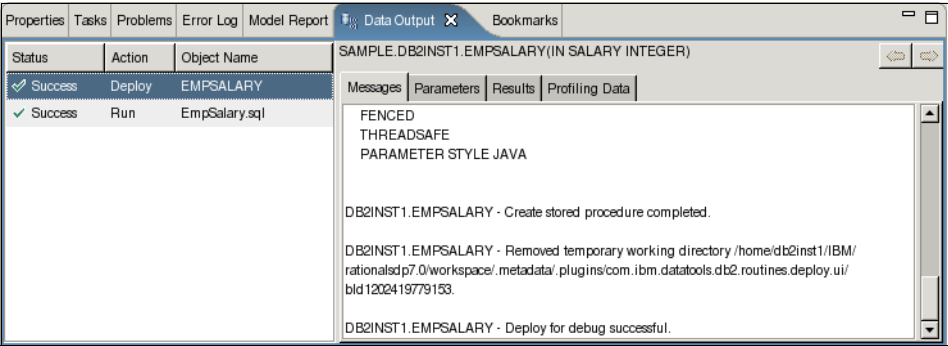


Figure 8-35 Output of the deployment action

Now the code is deployed on the server. The procedure is ready to be run. In the Data Project Explorer right-click EMPLOYEE from **DataProject** → **Stored Procedures** → **EMPLOYEE** and select **Run** as shown in Figure 8-36 on page 477.

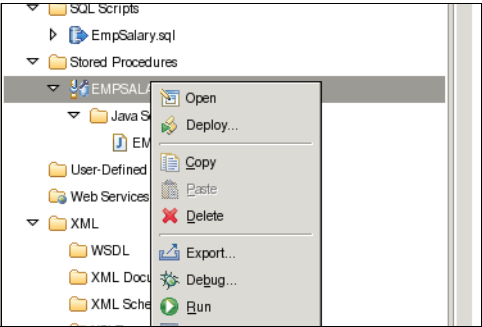


Figure 8-36 Run the procedure

You will be asked for the parameter as shown in Figure 8-37. Enter the value 37000.

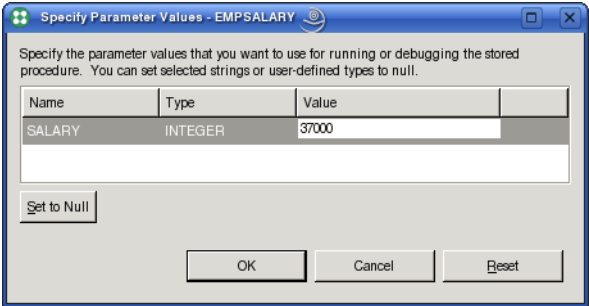


Figure 8-37 Enter run parameter

The output can be seen in the Data Output view. See Figure 8-38.

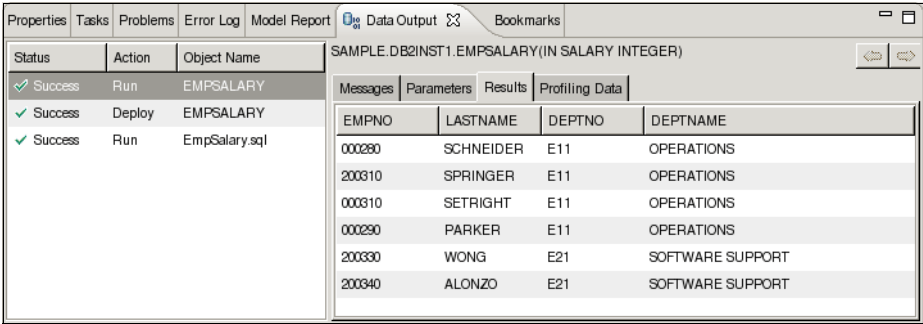


Figure 8-38 Run the stored procedure

Alternatively, you can run the stored procedure from the command line as following:

```
db2 "call EMPSALARY(37000)"
```

If you want to debug the procedure, set a breakpoint in the procedure, select **Debug** instead of the **Run** when you right-click on **EMPSALARY**. The Run Configuration window is presented. See Figure 8-39.

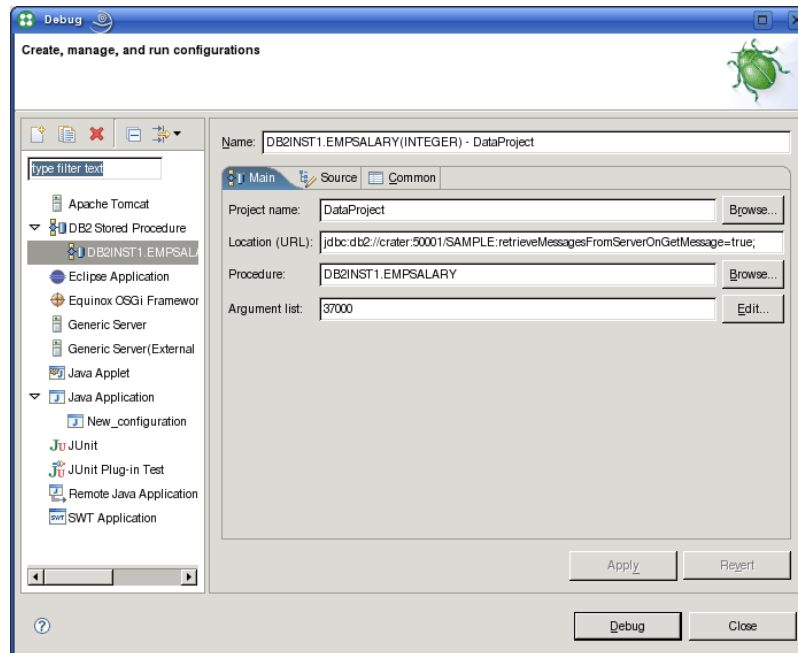


Figure 8-39 Run configuration

Here you can change the parameter (37000) and press **Debug**. The Data Studio will switch to the *Debug Perspective*. Dependent on your settings the Data Studio asks you to confirm the switch. In the Debug Perspective (Figure 8-40 on page 479), you can step through the code, view variables, change values, and do everything you need to debug your code.

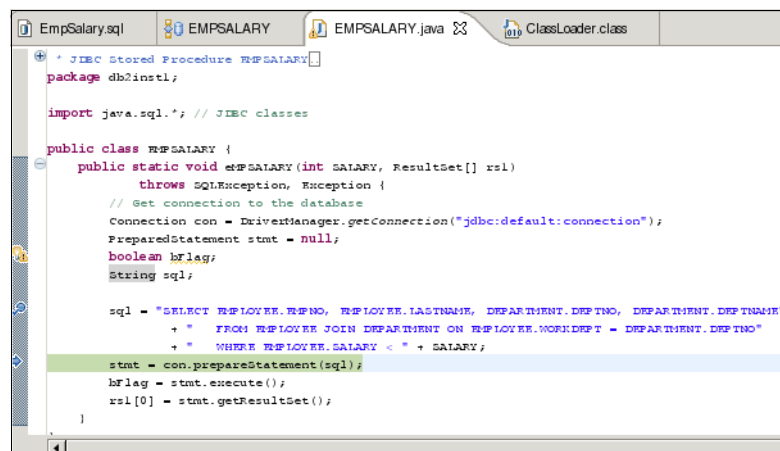
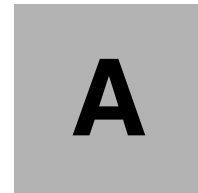


Figure 8-40 Debug stored procedure

The Data Studio is a very powerful tool to perform application development with DB2. It takes many error-prone tasks away from you. It supports you in developing application code, makes entering table and column names easy, and provides intelligent defaults for the various tasks. You can concentrate on implementing and testing your business logic. The time you need for development will be dramatically reduced.



Issuing commands to multiple database partitions

This appendix discusses some commonly used DB2 commands in a multiple partition environment (MPE). These commands are very useful when you need to collect information from multiple database partitions or machines, or make specified commands running against multiple partitions or machines. The commands that are discussed in this appendix include `db2_ps`, `db2_all`, `rah`, `db2_call_stack`, and so on.

db2_all and rah

In a multiple partitions environment, you may want to issue commands to be run on more than one machine or partition in the instance, and you don't want to repeat the same commands on every machine or partition involved — for this purpose, you can use the rah command or the db2_all command.

The rah command allows you to issue commands you want to run on all machines where the instance is spread across. If you want the commands to run at database partition level in the instance, you run the db2_all command.

To run a command on all machines with rah, the general command syntax is:

rah [commands]

To run the commands on all database partitions that you specified, use the db2_all command. The general command syntax is:

db2_all [commands]

If the command to be run doesn't follow rah or db2_all, you will be prompted to enter one by rah and db2_all. In general, if you specify the commands as the parameter on the command line, you must enclose it in double quotation marks if it contains any of the following special characters:

| & ; < > () { } [] unsubstituted \$

To obtain help information about db2_all or rah syntax, type:

db2_all "?"

Or,

rah "?"

There are some command prefix sequences (here a prefix sequence is one or more special characters) for db2_all and rah that can be used to control the command running mode, for example, running commands in sequence or parallel on all involved machines or partitions, running in foreground or background, whether or not to execute environment setting script before executing specified commands, and so on. Table A-1 shows some commonly used command prefix sequences.

Table A-1 Commonly used command prefix sequences for db2_all and rah

Sequence	Description
	Runs the commands in sequence in the background.

Sequence	Description
&	Runs the commands in sequence in the background and terminates the command after all remote commands have completed.
	Runs the commands in parallel in the background.
& or ;	Runs the commands in parallel in the background and terminates the command after all remote commands have completed.
<	Commands sent to all the machines for running except this one.
<<-nnn<	Commands sent to all other database partitions but not the database partition nnn for running.
<<+nnn<	Commands sent to only database partition nnn for running
>	Substitutes occurrences of <> with the machine name.
“	Substitutes occurrences of () by the machine index, and substitutes occurrences of ## by the node number.

The Example A-1 shows how to connect and list all the table spaces on all database partitions.

Example: A-1 Using db2_all on a multiple partitions environment

```
db2inst1@gemini:~> db2_all "db2 CONNECT TO itsodb ;db2 LIST TABLESPACES"

Database Connection Information

Database server          = DB2/LINUX8664 9.5.0
SQL authorization ID    = DB2INST1
Local database alias    = ITSODB


Tablespaces for Current Database

Tablespace ID           = 1
Name                    = TEMPSPACE1
Type                    = System managed space
Contents                = System Temporary data
State                   = 0x0000
Detailed explanation:
    Normal

Tablespace ID           = 2
Name                    = USERSPACE1
Type                    = Database managed space
Contents                = All permanent data. Large table space.
State                   = 0x0000
```

Detailed explanation:
Normal

Tablespace ID	= 3
Name	= IBMDB2SAMPLEREL
Type	= Database managed space
Contents	= All permanent data. Large table space.
State	= 0x0000

Detailed explanation:
Normal

DB21011I In a partitioned database server environment, only the table spaces on the current node are listed.

Mensa: db2 CONNECT TO itsodb completed ok

Database Connection Information

Database server	= DB2/LINUX8664 9.5.0
SQL authorization ID	= DB2INST1
Local database alias	= ITSODB

Tablespaces for Current Database

Tablespace ID	= 0
Name	= SYSCATSPACE
Type	= Database managed space
Contents	= All permanent data. Regular table space.
State	= 0x0000

Detailed explanation:
Normal

Tablespace ID	= 1
Name	= TEMPSPACE1
Type	= System managed space
Contents	= System Temporary data
State	= 0x0000

Detailed explanation:
Normal

Tablespace ID	= 2
Name	= USERSPACE1
Type	= Database managed space
Contents	= All permanent data. Large table space.
State	= 0x0000

Detailed explanation:


```
Normal

Tablespace ID          = 3
Name                   = IBMDB2SAMPLEREL
Type                   = Database managed space
Contents                = All permanent data. Large table space.
State                  = 0x0000
    Detailed explanation:
        Normal
```

DB21011I In a partitioned database server environment, only the table spaces on the current node are listed.

gemini: db2 CONNECT TO itsodb completed ok

Note: With DB2 Version 9.5 you do not need use db2_all to execute database back up or update database configurations anymore.

You can now back up and restore multiple database partitions at once using the new single system view (SSV) backup. You can get further informations about back up at Chapter 6, “Administering databases” on page 223

To update dabase configuration just use the **db2 update dababase command**, this command updates all database partitions by default, except when DBPARTITIONNUM is specified to update only one database partition.

db2_ps

You can use db2_ps command to list all the db2 engine processes on all logical partitions.

db2_call_stack

This command, on UNIX-based platforms, causes all processes running on all database partition servers to write call trace back to the syslog. In general, you don't need to run this command unless it is required by DB2 support staff.



DB2 Tools Catalog creation

This appendix provides information related to the Tool Catalog creation and a sample to create the DB2 Tools Catalog Database manually.

DB2 tools catalog creation

If you did not create a tools catalog when you installed DB2, you can create it by using DB2 Control Center or by the Command Line Processor (CLP). If using DB2 Control Center to do that, after invoking DB2 Control Center you can select **Tools** → **Tools Settings** → **Scheduler Settings** (Figure B-1).

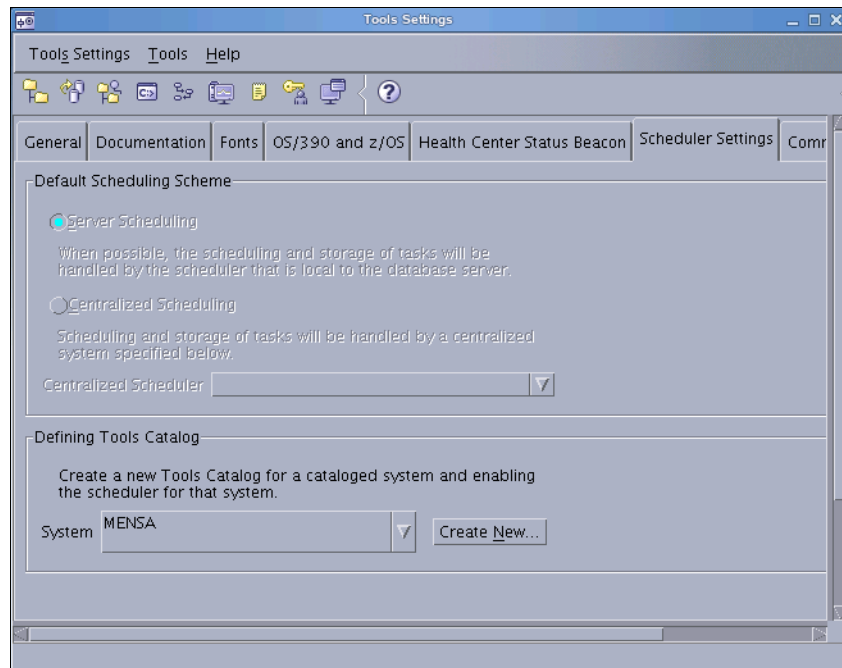


Figure B-1 Defining Tools Catalog

You can choose the system and then create a new tools catalog database as shown in Figure B-2 on page 489.

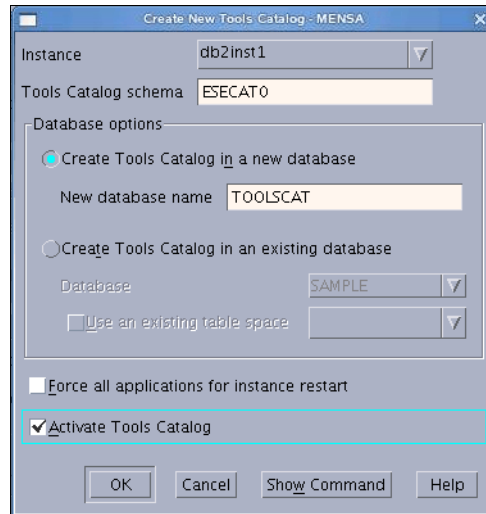


Figure B-2 Create new Tools Catalog

If you want to use the DB2 Command Line Processor, here we provide a sample to create the Tools Catalog by using the DB2 CLP manually, also included are the steps to check the DB2 administration server (DAS) configuration variations before and after creating Tools Catalog, and the new table spaces and buffer pool created by **CREATE TOOLS CATALOG** command automatically.

Before we create the tools catalog, we use **GET ADMIN CFG** to get the current configurations for DAS as shown in Example B-1. You can see that some parameters associated with the tools catalog are empty currently, including **TOOLSCAT_DB**, **TOOLSCAT_INST**, **TOOLSCAT_SCHEMA**. We will use this information to compare the DAS configuration changes made by the **CREATE TOOLS CATALOG** command at a later time.

Example: B-1 Checking current DAS configuration

```
db2inst1@Mensa:/> db2 get admin cfg
```

Admin Server Configuration

Authentication Type DAS (AUTHENTICATION) = SERVER_ENCRYPT

DAS Administration Authority Group Name (DASADM_GROUP) = dasadm1

DAS Discovery Mode (DISCOVER) = SEARCH

Name of the DB2 Server System (DB2SYSTEM) = MENSA

```
Java Development Kit Installation Path DAS    (JDK_PATH) = AUTOMATIC
(/db2home/dasusr1/das/java/jdk)
Java Development Kit Installation Path DAS    (JDK_64_PATH) = AUTOMATIC
(/db2home/dasusr1/das/java/jdk)

DAS Code Page                                (DAS_CODEPAGE) = 0
DAS Territory                                (DAS_TERRITORY) = 0

Location of Contact List                      (CONTACT_HOST) =
Execute Expired Tasks                        (EXEC_EXP_TASK) = NO
Scheduler Mode                               (SCHED_ENABLE) = OFF
SMTP Server                                  (SMTP_SERVER) =
Tools Catalog Database                     (TOOLSCAT_DB) =
Tools Catalog Database Instance           (TOOLSCAT_INST) =
Tools Catalog Database Schema             (TOOLSCAT_SCHEMA) =
Scheduler User ID                            =

Diagnostic error capture level                (DIAGLEVEL) = 2
```

Now let's create the Tools Catalog by CLP as in Example B-2.

Example: B-2 Creating Tools Catalog by using CLP

```
db2inst1@Mensa:/> db2 create database toolscat
DB20000I The CREATE DATABASE command completed successfully.
db2inst1@Mensa:/> db2 create tools catalog esecat0 use existing database
toolscat
DB20000I The CREATE TOOLS CATALOG command completed successfully.
```

Note: In the preceding example, if you want, you can create the new database (here it is TOOLSCAT) using the **CREATE TOOLS CATALOG** command with the **CREATE NEW DATABASE** clause directly. For details regarding the command syntax, refer to *Command Reference*, SC23-5846-00.

Then you can check the DAS configuration parameters again and you may find that the parameters associated with the Tools Catalog were changed automatically. See Example B-3.

Example: B-3 DAS Tools Catalog parameters changed automatically

```
db2inst1@Mensa:/> db2 get admin cfg

Admin Server Configuration

Authentication Type DAS                      (AUTHENTICATION) = SERVER_ENCRYPT

DAS Administration Authority Group Name      (DASADM_GROUP) = dasadm1
```

DAS Discovery Mode	(DISCOVER) = SEARCH
Name of the DB2 Server System	(DB2SYSTEM) = MENSA
Java Development Kit Installation Path DAS (/db2home/db2inst1/sqllib/java/jdk64)	(JDK_PATH) = AUTOMATIC
Java Development Kit Installation Path DAS (/db2home/db2inst1/sqllib/java/jdk64)	(JDK_64_PATH) = AUTOMATIC
DAS Code Page	(DAS_CODEPAGE) = 0
DAS Territory	(DAS_TERRITORY) = 0
Location of Contact List	(CONTACT_HOST) =
Execute Expired Tasks	(EXEC_EXP_TASK) = NO
Scheduler Mode	(SCHED_ENABLE) = ON
SMTP Server	(SMTP_SERVER) =
Tools Catalog Database	(TOOLSCAT_DB) = TOOLSCAT
Tools Catalog Database Instance	(TOOLSCAT_INST) = db2inst1
Tools Catalog Database Schema	(TOOLSCAT_SCHEMA) = ESECAT0
Scheduler User ID	=
Diagnostic error capture level	(DIAGLEVEL) = 2

In addition to the automatic DAS parameters change, within the Tools Catalog Database, a series of new tables are created with the schema name specified by **TOOLSCAT_SCHEMA**, here it is ESECAT0.

Also two new table spaces and a related buffer pool with page size of 32K are created within the database. You can use the commands and SQL statements shown in Example B-4 to check that.

Example: B-4 New table spaces and buffer pool created automatically

db2inst1@Mensa: /> db2 list tablespaces

Tablespaces for Current Database

Tablespace ID	= 0
Name	= SYSCATSPACE
Type	= Database managed space
Contents	= All permanent data. Regular table
space.	
State	= 0x0000
Detailed explanation:	
Normal	
Tablespace ID	= 1
Name	= TEMPSPACE1

```

Type                = System managed space
Contents            = System Temporary data
State              = 0x0000
  Detailed explanation:
    Normal

Tablespace ID       = 2
Name                = USERSPACE1
Type                = Database managed space
Contents            = All permanent data. Large table space.
State              = 0x0000
  Detailed explanation:
    Normal

Tablespace ID       = 3
Name                = TBSP32K0000
Type                = System managed space
Contents            = All permanent data. Regular table
space.
State              = 0x0000
  Detailed explanation:
    Normal

Tablespace ID       = 4
Name                = TBSP32KTMP0000
Type                = System managed space
Contents            = System Temporary data
State              = 0x0000
  Detailed explanation:
    Normal

```

DB21011I In a partitioned database server environment, only the table spaces on the current node are listed.

```
db2inst1@Mensa:/> db2 "select substr(tbspace,1,15) TBSPACE, tbspaceid,
substr(dbpgname,1,15) DBPGNAME, bufferpoolid from syscat.tablespace"
```

TBSPACE	TBSPACEID	DBPGNAME	BUFFERPOOLID
SYSCATSPACE	0	IBMCATGROUP	1
TEMPSPACE1	1	IBMTMPGROUP	1
USERSPACE1	2	IBMDEFAULTGROUP	1
TBSP32K0000	3	IBMCATGROUP	2
TBSP32KTMP0000	4	IBMTMPGROUP	2

5 record(s) selected.

```
db2inst1@Mensa:/> db2 "select substr(bpname,1,15) BPNAME, bufferpoolid,
substr(dbpgname,1,15) DBPG, npages, pagesize from syscat.bufferpools"
```


BPNAME	BUFFERPOOLID	DBPG	NPAGES	PAGESIZE
IBMDEFAULTBP	1	-	1000	4096
BP32K0000	2	-	250	32768

2 record(s) selected.

In the preceding example, the table spaces TBSP32K0000 with table space ID 3 and TBSP32KTMP0000 with table space ID 4 were created automatically by the **CREATE TOOLS CATALOG** command, each with a page size of 32K. The first one is a regular table space that is used to store the tasks related tables, and the second one is a temporary table space. In addition, a new buffer pool named BP32K0000 is created with a page size of 32K.

You need to recycle the database manager to have all the new changes take effect. Before you stop the instance, if you use **LIST APPLICATIONS** command, you may find that some applications are connected to the Tools Catalog Database with the application name db2dasstm as shown in Example B-5.

Example: B-5 Applications connected to Tools Catalog Database

db2inst1@Mensa:/> db2 list applications

Auth Id	Application Name	Appl. Handle	Application Id	DB Name	# of Agents
DB2INST1	db2dasstm	271	*NO.db2inst1.080128234908	TOOLSCAT	1

To restart the database manager, you may use **DB2STOP** with **FORCE** option, see the following steps (as the instance owner, here db2inst1):

1. **db2set -a11 | grep ADMINSERVER**
to get the name of the DAS user, here dasusr1
2. **/db2home/dasusr1/das/bin/db2admin stop**
3. **db2stop [force]**
force may be required if other applications are still connected
4. **db2start**
5. **/db2home/dasusr1/das/bin/db2admin start**

At this point in time, the Tools Catalog is ready to use. You can now start defining schedules using the Task Center.

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 on ordering these publications, see “How to get IBM Redbooks” on page 498.

- ▶ *DB2 UDB Exploitation of the Windows Environment*, SG24-6893
- ▶ *DB2 9: pureXML Overview and Fast Start*, SG24-7298
- ▶ *DB2 9 pureXML Guide*, SG24-7315
- ▶ *DB2 Security and Compliance Solutions for Linux, UNIX, and Windows*, SG24-7555
- ▶ *Database Partitioning, Table Partitioning, and MDC for DB2 9*, SG24-7467
- ▶ *Developing PHP Applications for IBM Data Servers*, SG24-7218

Other publications

These publications are also relevant as further information sources:

IBM - DB2 9.5

- ▶ *What's New*, SC23-5869-00
- ▶ *Administrative API Reference*, SC23-5842-00
- ▶ *Administrative Routines and Views*, SC23-5843-00
- ▶ *Call Level Interface Guide and Reference, Volume 1*, SC23-5844-00
- ▶ *Call Level Interface Guide and Reference, Volume 2*, SC23-5845-00
- ▶ *Command Reference*, SC23-5846-00
- ▶ *Data Movement Utilities Guide and Reference*, SC23-5847-00
- ▶ *Data Recovery and High Availability Guide and Reference*, SC23-5848-00
- ▶ *Data Servers, Databases, and Database Objects Guide*, SC23-5849-00
- ▶ *Database Security Guide*, SC23-5850-00

- ▶ *Developing ADO.NET and OLE DB Applications*, SC23-5851-00
- ▶ *Developing Embedded SQL Applications*, SC23-5852-00
- ▶ *Developing Java Applications*, SC23-5853-00
- ▶ *Developing Perl and PHP Applications*, SC23-5854-00
- ▶ *Developing User-defined Routines (SQL and External)*, SC23-5855-00
- ▶ *Getting Started with Database Application Development*, GC23-5856-00
- ▶ *Getting Started with DB2 installation and administration on Linux and Windows*, GC23-5857-00
- ▶ *Internationalization Guide*, SC23-5858-00
- ▶ *Message Reference, Volume 1*, GI11-7855-00
- ▶ *Message Reference, Volume 2*, GI11-7856-00
- ▶ *Migration Guide*, GC23-5859-00
- ▶ *Net Search Extender Administration and User's Guide*, SC23-8509-00
- ▶ *Partitioning and Clustering Guide*, SC23-5860-00
- ▶ *Query Patroller Administration and User's Guide*, SC23-8507-00
- ▶ *Quick Beginnings for IBM Data Server Clients*, GC23-5863-00
- ▶ *Quick Beginnings for DB2 Servers*, GC23-5864-00
- ▶ *Spatial Extender and Geodetic Data Management Feature User's Guide and Reference*, SC23-8508-00
- ▶ *SQL Reference, Volume 1*, SC23-5861-00
- ▶ *SQL Reference, Volume 2*, SC23-5862-00
- ▶ *System Monitor Guide and Reference*, SC23-5865-00
- ▶ *Troubleshooting Guide*, GI11-7857-00
- ▶ *Tuning Database Performance*, SC23-5867-00
- ▶ *Visual Explain Tutorial*, SC23-5868-00
- ▶ *Workload Manager Guide and Reference*, SC23-5870-00
- ▶ *pureXML Guide*, SC23-5871-00
- ▶ *XQuery Reference*, SC23-5872-00

Online resources

These Web sites and URLs are also relevant as further information sources:

DB2

- ▶ DB2 Information Center
<http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/>
- ▶ Database and Information Management home page
<http://www.ibm.com/software/data/>
- ▶ DB2 Technical Support
http://www.ibm.com/software/data/db2/support/db2_9/
- ▶ DB2 Product Family Library
<http://www.ibm.com/software/data/db2/library/>
- ▶ DB2 developerWorks
<http://www.ibm.com/developerworks/db2/>
- ▶ DB2 for Linux
<http://www.ibm.com/software/data/db2/linux/>
<http://www.ibm.com/software/data/db2/linux/validate/>
- ▶ DB2 Universal Database V9 Application Development
<http://www.ibm.com/software/data/db2/ad/>
- ▶ Planet DB2
<http://www.planetdb2.com/>

Linux

- ▶ IBM Software for Linux
<http://www.ibm.com/software/os/linux/software/>

Other

- ▶ SAP Standard Application Benchmarks
<http://www.sap.com/solutions/benchmark/index.epx>
- ▶ DBI.perl.org
<http://dbi.perl.org>
- ▶ DB2 Perl Database Interface
<http://www.ibm.com/software/data/db2/perl>
- ▶ Comprehensive Perl Archive Network
<http://www.cpan.org>
http://www.cpan.org/modules/by-category/07_Database_Interfaces/DBI
- ▶ Apache HTTP Server Project
<http://httpd.apache.org>

- ▶ Perl.apache.org
<http://perl.apache.org/docs/1.0/guide/>
- ▶ PHP scripting language
<http://www.php.net/>

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

Index

Symbols

.nfy 389
 .Xauthority 91
 /etc 393

Numerics

2-tier 10
 64-bit 2

A

access path 177
 access plan 435
 active log file 230
 admin_cmd stored procedure 298
 administration notification log 183, 363
 administration server 36
 ADO.NET 425
 AMD 2
 analysis tool 393
 APIs 348
 application client 11
 archive logging 119, 224
 archive storage pool 234
 AS/400 11
 automatic backup 245
 automatic statistics collection 178
 automatic storage 167
 automatic storage paths 260
 availability 17
 availability monitoring 196
 availability requirement 226

B

backup 42
 backup image 4
 backup image file 250
 backup pending state 173
 backup storage pool 234
 backup strategy 226
 benchmarks 2
 bio-informatics 2
 block device 97

block IO 409
 bookmarks 211
 buffer pool 16, 95
 buffer pool size 377
 built-in function 421
 business intelligence 2

C

cache 18
 catalog table space 95
 catalog tables 95
 cataloged system 92
 central processing unit (CPU) 410
 centralized database 11
 chown 392
 circular logging 119, 224
 CLI 3
 CLP 298
 cluster 3
 COBOL 426
 code page 93
 column function 421
 columns 15
 command
 df 42
 iostat 406
 ipcs 31
 rpm 34
 sar 406
 showmount 47
 sysctl 31
 vmstat 406
 Command Line Processor 298, 488
 command option 406, 412
 command-line tool 430
 commands
 ar 412
 db2top 379
 ssh 91
 communication 46
 requirements 29
 compact 27–28, 60, 65, 78
 configuration file 136

configuration parameters 136

archretrydelay 230

blk_log_dsk_ful 231

catalogcache_sz 137

dbheap 137

dft_degree 138

dft_monswitches 132

dftdbpath 132

diaglevel 119, 132, 176, 183, 391

diagpath 175, 389, 391

failarchpath 230

health_mon 350

instance_memory 133

intra_parallel 133

java_heap_sz 133

keepdari 425

keepfenced 425

logarchmeth1 224, 230

logarchmeth2 224, 230

logbufsz 137

logfilsiz 166, 176, 183

logprimary 176, 183

logsecond 166, 176, 183

max_querydegree 133

maxfilop 137

mirrorlogpath 231

mon_heap_sz 132

notifylevel 389

numarchretry 230

sheapthres 132

sorheap 137

configuration task 93

console 211

contact group 352

contact management 351

content assist 220

content management 2

context assist 460

control center 488

Control Center. 92

CPU utilization 413

crash recovery 250

create function 422

create procedure 423

create trigger 420

cumulative backup 225

custom 27–28, 60, 65, 78

D

dasupdt 5

Data 195

data compression 4

data distribution viewer 195

data management 195

data management solution 194

data population 4

data server 5

data warehouse 13

database administration server (das) 39, 49

database code page 167

database collating sequence 167

database level privileges 147

database managed space (DMS) 15

database managed storage (DMS) 40

database managed table spaces 96

database object health indicator setting 355

database partition group 15

database territory 167

DB2 Administration Server (DAS) 75

DB2 Database Partitioning Feature (DPF) 9

DB2 Enterprise Server Edition (ESE) 8

DB2 Everyplace 10

DB2 object 420

DB2 setup wizard 38, 50

DB2 tools catalog 487

DB2 Workgroup Server Edition (WSE) 8

DB2 Workgroup Server Unlimited Edition (WSUE)
8

db2_all 110, 237, 482

db2_call_stack 481

db2_install 38

db2_ps 485

db2admin start 90

db2ca 143

db2ckmig 173

db2diag 393

db2dump directory 120

db2exmig 177

db2expln 382

db2fs 417

db2imigr 174

db2iupdt 5

db2jcc4.jar 442

db2ln 76

db2logmgr 230

db2look 94

db2move 94

db2mtrk 376–377
db2pd 397, 399
db2rbind 177
db2relocatedb 172
db2rmln 76
db2sampl 417
db2start 116
db2stop 116
db2undgp 171
DBI 426
debugger 460
delta backup 226
delta database backup 236
deploy 194
Design Advisor 458
Developer Workbench 206
diagnostic log file 394
digital signature algorithm 53
disable evaluation options 364
disk capacity 42
disk I/O 16, 406
disk requirements 27
distribution key 355
DMS 355
DMS table space 355
document ID 447
domain name server 29
DRDA 3
driver 425
driver type 442
DSA 53
DSS 12
dynamic size 27
dynamic SQL 381

E

Eclipse 194
e-commerce 430
editor tool bar 211
embedded SQL 433
engine processes 485
enterprise applications 2
ER diagram 195, 216
error log 211
ESQL 433
evaluation option 364
EXECUTE privilege 171
explain 177

ext2 41
ext3 41–42
extension 430
external procedure 423

F

failover 4
Fast Communication Manager 103
fast communication manager 6
FCM 103
FCP 41
fdisk command 97
fenced use 425
file system 47
file system caching 95
flexibility 2
Fortran 426
framework 220
front-end 431
fstab file 46
full backup 225
full offline database backup 235

G

gateway 2, 11, 29
global snapshot 381
gnome-system-monitor 415

H

health indicator 348, 354
health issues 348
health monitor 348
health monitor alarm 390
health_mon 348–349
heterogeneous environment 11
high availability 4, 13
high performance computing applications 2
high-SMP systems 23
host IP address 29
host trust relationship 57
host-based authentication 53
HTML 85
HTML document 430

I

I/O operations 178
ibm_db2 431

IBMDEFAULTBP 377
 inconsistent state 173
 incremental backup 226
 index 15
 index reorganization 6
 Information Integration application 2
 inplace table reorganization 276
 ins 194
 instance 14
 instance-owning machine 52
 instant level authority
 SYSADM 146
 SYSCTRL 146
 SYSMAINT 146
 SYSMON 146
 Integrated Query Editor 195
 interactive mode 397
 intra-parallelism 16
 iostat 410
 ipcs 30
 ISO8859-1 94

J

Java 425
 Java Routine Debugger 195
 JDBC 3
 jdk32 444
 jdk64 444
 journal filesystem 42

K

kernel 26
 kernel parameter 26
 kernel parameters
 sem 30
 keyboard shortcut 220
 kilobytes 410
 ksysguard 415

L

large object (LOB) 15
 layout 220
 libaio 32
 license status 156
 life cycle 194
 life sciences 2
 Linux distribution 26

load balancer 12
 load copy image 4
 local workspace 213
 Locking memory 129
 LOCKTIMEOUT 137
 log 44
 log directory 230
 log file manager 226
 log manager 230
 log mirroring 44
 log path 44, 116
 log space 35
 logical database partition 22
 logical partition 21, 485
 logical_port 103
 logprimary 166

M

memory 26
 memory component 374–375
 memory consumption 406
 Memory Tracker 376
 memory usage 375
 Memory Visualizer 374
 message queue limit 30
 migrate database 177
 migration 163, 165
 migration roadmap 164
 migration strategy 164
 minimum recovery time 255
 mirror 44
 mirrored logs 43
 mobile devices 10
 module 425
 monitoring memory 374
 multiple partition environment (MPE) 481
 multithreaded architecture 5
 multi-tier 10
 multi-tier application 5

N

Navigator 210
 netstat 415
 network 406
 Network File System (NFS) 47–48
 NFS 46, 415
 NFS server machine 43
 nfsstat 415

No cascade before 420
 no_root_squash 48
 nodegroups 15
 non-root 36
 normal state 173
 notification 38–39, 349, 351, 353–354, 362
 NUM_POOLAGENTS 132

O

object setting 356
 ODBC 3, 426
 offline backup 225
 offline reorganization 276
 OLAP 3
 OLD DB 426
 OLTP 12
 online backup 236
 online table reorganization 276
 Oracle RAC 18
 OS/390 11
 outline view 210

P

package cache 129
 package group 198
 paging 409
 parallel I/O 96
 parallel processing 16
 parallelism 3
 parameter 29
 partial declustering 22
 partition group 15, 21
 partitioned database environment 45
 PDO_IBM 431
 PDO_ODBC 431
 performance 2, 42
 Perl 4, 425–426
 Perl Database Interface 426
 personal digital assistants (PDAs) 10
 perspective 208
 pgrep 415
 PHP 4, 425
 physical partition 21
 physical storage devices 15
 pkill 415
 point in time 226
 pop-up window 353
 processor time 406

processors 18
 procps 409
 programming language 425
 project explorer 212
 Project Management 195
 protocol 64, 76, 194
 proxy component 12
 pstree 414
 public key-based authentication 53
 Python 4

Q

query 13

R

rah 482
 raw device 35, 39, 97
 real-time statistics 178
 rebind package 177
 reboot 409
 recommendation 166
 recoverable database 225
 recovery history file 231
 recovery log 15
 recovery time 4
 Redbooks Web site 498
 Contact us vi
 redirected restore 260
 redistribute command 124
 registry 128
 registry variable 176
 registry variables
 db2rshcmd 84
 ReiserFS 41
 relational database 15
 relations 15
 response file 39, 50
 restore pending state 173
 revoke statements 171
 REXX 426
 Rivest-Shamir-Adleman algorithm 53
 rollforward pending state 173
 root_squash 48
 row-function 421
 rows 15
 RSA 53
 runtimes 220

S

sartest.bin 413
 scalability 2, 17
 scalar function 421
 scaling out 16
 scaling up 16
 Schema Management 195
 scp 91
 SCSI 41
 secondary logs 44
 security access control 195
 security control 95
 Self-Managing and Resource Tuning (SMART) 3
 self-tuning memory 6
 semaphore array 30
 semicolon 424
 several ways to install DB2 77
 shared 18
 shared memory segment 30
 shared resources directory 198
 shared-disk 18
 shared-nothing 18
 single processor 16
 single system view 237
 single-tier 10
 Small Computer System Interface 41
 SMP architectures 16
 SMTP_SERVER 353
 sort memory 129
 SQL 3
 SQL Builder 195
 SQL procedure 423
 SQL Routine Debugger 195
 SQL statement 425
 ssh_known_hosts 54
 sshd_config 53
 SSV 237
 SSV backup 237
 standby cluster 12
 star schemas 13
 stored procedure 4, 420, 423, 478
 Structured Query Language (SQL) 15
 subnet mask 29
 SVCENAME 133
 swap space 412
 symmetric multi-processor 16
 syntax 482
 syscatspace 166
 sysctl 31

syslog 485
 syslog.conf 393
 syslogd 393
 sysstat package 410
 system activity report (sar) 412
 system configuration 35
 system managed space (SMS) 15
 system managed storage (SMS) 40
 system managed table spaces 95

T

table 15
 table function 421
 table space 15, 35, 355
 tasks view 210
 TCP/IP 76
 team function 212
 temporary table space 45, 95
 TEMPSPACE1 166
 threshold 349
 timestamp 447
 tools catalog 39, 180
 Top 406
 traceback 485
 TRACKMOD 225
 transactional enterprise system 2
 transactional integrity 430
 transfer rate 412
 trigger 420
 triggering action 420
 tutorial session 221
 Type 4 driver 442
 Typical 60
 typical 27–28, 65, 78

U

UDFs 4
 UDTs 173
 Unified ODBC 431
 UNIX 485
 user defined functions (UDFs) 420
 user table space 95
 user-defined types 173
 UTF-8 94

V

version recovery 250

view variable 479
virtual memory 409–410
Visual Explain 195
VM/VSE 11

W

Web 2.0 194
Web application 430
Web browser 205
Web services 194
Web-browser 11
workload management 5
workspace 207
workspace location 207

X

X server 91
XML 213
XML attributes 446
XML document 446
XML Editor 195
XML Schema Editor 195
XML structure 446
XSD 213
XSD document 213

To determine the spine width of a book, you divide the paper PPI into the number of pages in the book. An example is a 250 page book using Plainfield opaque 50# smooth which has a PPI of 526. Divided 250 by 526 which equals a spine width of .4752". In this case, you would use the .5" spine. Now select the Spine width for the book and hide the others: **Special>Conditional Text>Show/Hide>SpineSize(-->Hide:)>Set** . Move the changed Conditional text settings to all files in your book by opening the book file with the spine.frm still open and **File>Import>Formats** the Conditional Text Settings (ONLY!) to the book files.

Draft Document for Review April 17, 2008 2:10 pm

6899spine.frm 507



Redbooks

Up and Running with DB2 on Linux

(1.0" spine)
0.875"<->1.498"
460 <-> 788 pages

To determine the spine width of a book, you divide the paper PPI into the number of pages in the book. An example is a 250 page book using Plainfield opaque 50# smooth which has a PPI of 526. Divided 250 by 526 which equals a spine width of .4752". In this case, you would use the .5" spine. Now select the Spine width for the book and hide the others: **Special>Conditional Text>Show/Hide>SpineSize(--->Hide:)>Set** . Move the changed Conditional text settings to all files in your book by opening the book file with the spine.fm still open and **File>Import>Formats** the Conditional Text Settings (ONLY!) to the book files.

Draft Document for Review April 17, 2008 2:10 pm



Draft Document for Review April 17, 2008 2:10 pm

Up and Running with DB2 on Linux



Experience the power of the integration of DB2 9.5 and Linux

Make it easier to get DB2 for LUW up and running on Linux

Leverage DB2's powerful autonomic technology

Linux is one of the fastest growing server operating platforms within the past few years. DB2 has long been known for its technology leadership. This IBM Redbooks publication is an informative guide that describes how to effectively integrate DB2 for Linux, UNIX, and Windows (LUW) with SUSE and Red Hat Linux operating systems. This book provides both introductory and detailed information on installing, configuring, managing, and monitoring DB2 in a Linux environment.

We describe the DB2 product family and features for Linux, and provide step-by-step instructions for a single as well as for multiple partitions DB2 system installation and configuration. We cover how to migrate single and multiple partition DB2 to DB2 Version 9.5, and discuss, in detail, DB2 database administration in a Linux environment, procedures and tools for database backup and recovery, online maintenance, and system monitoring. We cover DB2 integrated tools and their features and use.

We discuss aspects of DB2 application development in the Linux environment, and provide general tips about building and running DB2 applications on Linux, and the use of DB2 application development tools.

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