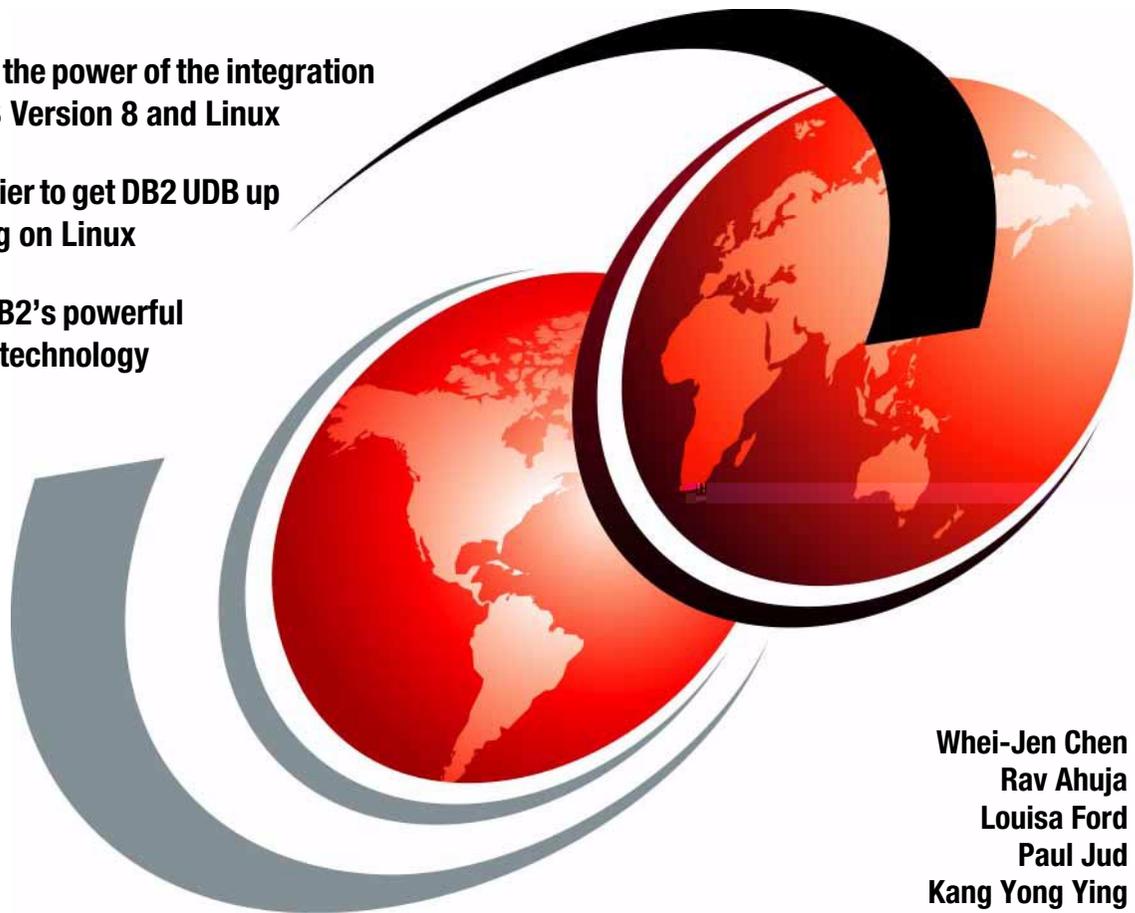


Up and Running with DB2 for Linux

Experience the power of the integration
of DB2 UDB Version 8 and Linux

Make it easier to get DB2 UDB up
and running on Linux

Leverage DB2's powerful
autonomic technology



Whei-Jen Chen
Rav Ahuja
Louisa Ford
Paul Jud
Kang Yong Ying



International Technical Support Organization

Up and Running with DB2 for Linux

March 2003

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

First Edition (March 2003)

This document created or updated on March 24, 2003.

This edition applies to DB2 Universal Database Version 8.1 for use with SuSE Linux Version 8.1 and Red Hat Linux Version 8.0 operating systems.

© Copyright International Business Machines Corporation 2003. 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	vii
Trademarks	viii
Preface	ix
The team that wrote this redbook	x
Acknowledgements	xi
Become a published author	xii
Comments welcome	xii
Chapter 1. Introduction	1
1.1 Overview	3
1.2 DB2 for Linux features and offerings	4
1.2.1 Features	4
1.2.2 Supported platforms	5
1.2.3 DB2 products and packages	6
1.3 DB2 environment	10
1.3.1 Deployment topologies	10
1.3.2 DB2 database objects	12
1.4 Parallelism with DB2	14
1.4.1 SMP environments	15
1.4.2 Database clusters	16
1.4.3 Partitioned database	18
Chapter 2. Installation	23
2.1 Basic requirements	24
2.1.1 Linux distributions supported by DB2	24
2.1.2 Required disk space	26
2.1.3 Memory requirements	26
2.1.4 Communication requirements	27
2.1.5 Required kernel levels and libraries	27
2.1.6 Additional software requirements	31
2.2 Installation considerations and planning	34
2.2.1 Installation methods	34
2.2.2 Storage planning	36
2.2.3 Network configuration for a multiple partitioned installation	42
2.2.4 User and group setup	48
2.3 Installing DB2	52
2.3.1 DB2 Setup	52
2.3.2 Multiple partition installation	61

2.3.3 db2_install utility	68
2.3.4 Adding an additional partition	76
2.3.5 Installing on NIS	76
Chapter 3. Post installation tasks	79
3.1 Control Center setup	80
3.2 Preparation for database creation	81
3.2.1 Linux-specific configuration	81
3.2.2 Multiple partitioned specific configuration	82
3.3 Creating a database	84
3.3.1 Create Database wizard	85
3.3.2 Command line	93
3.4 Add database partition	100
3.5 DB2 configuration	107
3.5.1 DBM configuration	108
3.5.2 DB configuration	113
3.5.3 DB2 registry and environment variables	119
3.6 Client configuration	122
3.7 Configuring licensing	127
3.7.1 DB2 License Center	128
3.7.2 db2licm and db2licd	131
Chapter 4. Migration	133
4.1 Migration planning	134
4.2 Installing the new DB2 V8 server	136
4.2.1 Installing DB2 V8 using DB2 Setup wizard	136
4.2.2 Installing DB2 V8 using db2_install	138
4.2.3 Installing DB2 V8 using a response file	139
4.2.4 Creating test instance	139
4.3 Migrating your instances and databases	139
4.3.1 Pre-migration tasks	139
4.3.2 Migrating instances and databases	140
4.3.3 Multiple partition migration	144
4.3.4 Migrating the DB2 Administration Server	144
4.3.5 Client and server compatibility	145
Chapter 5. Administering databases	147
5.1 DB2 database backup and recovery	149
5.1.1 Enable roll forward and log archiving	151
5.1.2 Enable incremental backup	155
5.1.3 Enable usage of TSM	155
5.1.4 Backup utility	158
5.1.5 Backup database with Control Center	158
5.1.6 Backup database in DB2 command line processor	161

5.1.7	DB2 database recovery utility	163
5.1.8	Backup and restore sample scenario	164
5.1.9	Recovery of a dropped table	166
5.2	Table/index reorganization and statistics collection	169
5.2.1	Table reorganization	169
5.2.2	Index reorganization	170
5.2.3	Statistics information collection	171
5.2.4	Packages rebinding.	171
5.2.5	Examples	172
5.3	Data movement via EXPORT, IMPORT, and LOAD	183
5.3.1	Export data to files from database tables	183
5.3.2	Import files into database tables or views	186
5.3.3	Load data into existing tables	189
5.3.4	Using db2move utility	210
5.4	Task Center, Scheduler, and DB2 Tools Catalog	211
5.4.1	DB2 Administration Server and Tools Catalog Database	211
5.4.2	Task Center and Scheduler	212
Chapter 6. Monitoring DB2		223
6.1	Health Monitor and Health Center	225
6.1.1	Health Indicator	225
6.1.2	Health Monitor	225
6.1.3	Health Center	226
6.1.4	Using Health Center and Health Monitor	226
6.2	Log files for troubleshooting	242
6.2.1	DB2 administration notification log	242
6.2.2	DB2 diagnostic log (db2diag.log)	244
6.2.3	Operating system log	245
6.3	Linux system monitoring tools	245
6.3.1	Top	246
6.3.2	Vmstat	248
6.3.3	lstat	250
6.3.4	Sar	251
6.3.5	Other system monitoring tools	254
Chapter 7. DB2 integrated tools and wizards		255
7.1	Design Advisor	258
7.2	Visual Explain	264
7.3	Memory Visualizer and Memory Tracker.	267
Chapter 8. Application development		271
8.1	Application configuration	272
8.2	DB2 application objects	273
8.2.1	Triggers	273

8.2.2	Use defined functions (UDFs)	275
8.2.3	Stored procedures	276
8.3	Programming languages	278
8.3.1	Perl	278
8.3.2	PHP	281
8.3.3	C++	282
8.3.4	Java	283
8.4	Application tools	284
8.4.1	Command Center	284
8.4.2	Development Center	286
8.4.3	Additional development tools	289
Appendix A. Issuing commands to multiple database partitions		291
	db2_all and rah	292
	db2_ps	294
	db2_call_stack	294
Appendix B. DB2 Tools Catalog creation		295
	DB2 Tools Catalog creation	296
Related publications		301
	IBM Redbooks	301
	Other publications	301
	Online resources	302
	How to get IBM Redbooks	304
Index		305

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 illustrates 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. 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 IBM's application programming interfaces.

Trademarks

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

AFS®	eServer™	pSeries™
AIX®	 eServer™	PowerPC®
AS/400®	IBM®	Redbooks™
CT™	Informix®	Redbooks (logo)™ 
DB2®	Intelligent Miner™	RETAIN®
DB2 Connect™	iSeries™	S/390®
DB2 Extenders™	Net.Data®	SP™
DB2 Universal Database™	Notes®	Tivoli®
DRDA®	OS/2®	WebSphere®
Everyplace™	OS/390®	xSeries™
e Strategy™	Perform™	z/OS™
	PowerPC®	zSeries™

The following terms are trademarks of other companies:

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

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

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

C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.

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

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

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

Preface

This IBM Redbook will help you install, configure, administer, and monitor DB2 UDB Enterprise Server Edition Version 8 on Linux systems, for both single and multi-partitioned database environments. This book shows you how to leverage the DB2's autonomic capabilities using the Self Managing and Resource Tuning (SMART) technologies.

Chapter 1 introduces DB2 Universal Database (UDB) Version 8 and its offerings for Linux, and provides an overview of the DB2 environment and architecture.

Chapter 2 covers DB2 installation for single and multi-partitioned database environments. It discusses how to prepare your Linux database environment for DB2 installation, and provides instructions on how to use the different installation programs provided by DB2.

Chapter 3 discusses post installation tasks that should be performed to have a functional DB2 environment. It covers Linux specific settings, and database creation and configuration.

Chapter 4 explains how to migrate your Version 6 or Version 7 database environment to DB2 UDB Version 8. This chapter covers tasks such as preparing your database environment for migration, installing the new database server, and performing post-installation tasks such as migrating instances and databases.

Chapter 5 contains information regarding some of the important database maintenance tasks for DB2 UDB on Linux platforms, for example, database backup and recovery, table and index reorganization and data movement via utilities like export, import and load, and so on.

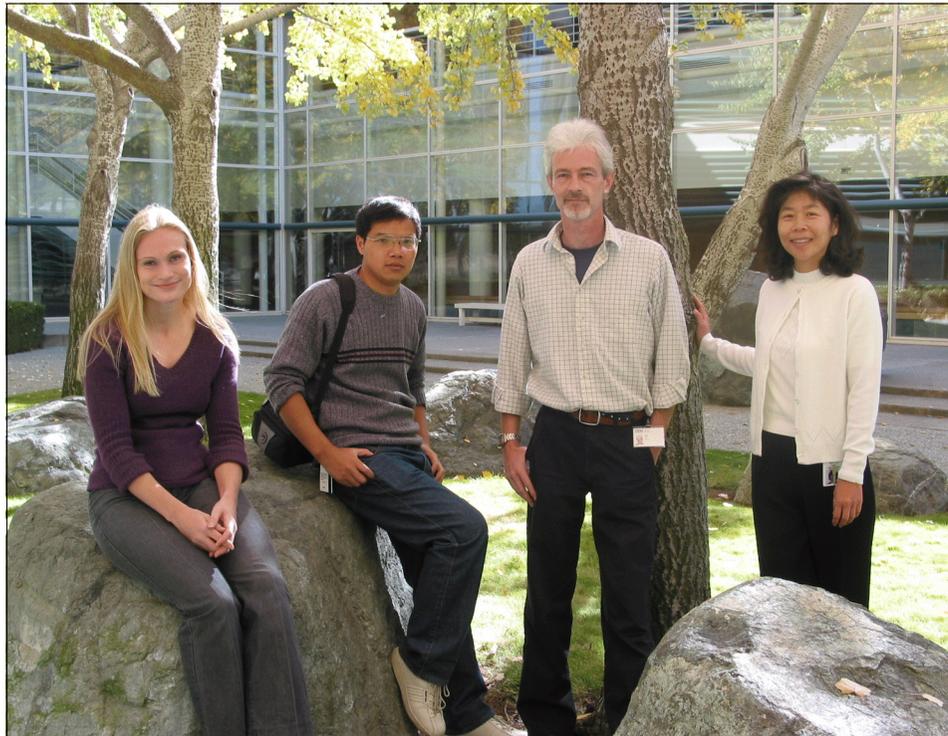
Chapter 6 discusses DB2's monitoring features provided by Health Center and Health Monitor which is part of the self-managing and resource tuning (SMART) database technology. In addition, DB2 notification and diagnostic log files as well as some commonly used Linux operating system monitoring tools are also introduced.

Chapter 7 discusses DB2 integrated tools. DB2 UDB provides a multitude of tools that makes it much easier and more efficient to install and maintain the database and develop applications. This chapter discusses most of these tools in a general way. Some of the integrated tools such as Design Advisor, Visual Explain, and Memory Visualizer are discussed in detail.

Chapter 8 concerns application development. DB2 provides accessibility from many standard interfaces and has various tools inside to help with developing applications. This chapter discusses some aspects of application configuration, DB2 application objects, such as stored procedures, programming languages, and application development tools supported by DB2.

The team that wrote this redbook

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



The UDB team (left to right): Louisa Ford, Kang Yong Ying, Paul Jud, Whei-Jen Chen

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

Rav Ahuja is the worldwide Product Manager for DB2 on Linux at the IBM Toronto Lab. He has been working with databases for over ten years and with DB2 on distributed platforms since Version 1.0. Having spent several years in the DB2 support team working with customers around the globe, Rav has gained expertise in database connectivity, application development, and Web access issues. He has contributed towards several DB2 technotes and the DB2 UDB Certification Guide. Rav is an IBM Certified DB2 Database Administrator and Application Developer.

Louisa Ford is a Software Engineer within DB2 User-Centered Design at the IBM Toronto Laboratory. She holds a degree in Industrial Engineering from the University of Toronto, with a specialization in human factors engineering. Before joining IBM, she was an Interface Designer and User-Centered Researcher working for Motorola. Her areas of expertise at IBM include usability of DB2 installation, packaging, and application development.

Paul Jud is a Advisory IT Specialist within IBM Global Services, Switzerland. He has over 16 years of experience working in the IT industry particularly in databases and data warehouses. For 11 years his primary focus was operation, optimization, and maintenance of large DB2 databases in the mainframe area. For the last five years he was primarily dedicated to DB2 UDB in a distributed environment. His areas of expertise include installation, administration, monitoring, and tuning of DB2 UDB EE and EEE, as well as Data Joiner and Data Replication. He is also an IBM Certified Database Administrator.

Kang Yong Ying is an Advisory IT Specialist from the Technical Support Center, IBM China. He has more than eight years of experience in database software for distributed platforms. He is also an IBM Certified Advanced Technical Expert — DB2 for Clusters. His areas of expertise include DB2 UDB administration and troubleshooting, as well as Business Intelligence solution implementation.

Acknowledgements

The authors express their deep gratitude for the help they received from **Darin McBride** from the IBM Toronto Laboratory. They would also like to thank the following people for their contributions to this project:

Jakob B. Carstensen
IBM Software Group, Linux Marketing

Yvonne Chan, Ryan Chase, Subho Chatterjee, Jessica Escott, Anson Kokkat,
Dan Scott, Dwaine Snow, Bill Wong
IBM Toronto Laboratory

Harold T. Morgan
World Wide Consulting - Data management group

Philipp Spaeti
IBM Switzerland

Emma Jacobs, Gabrielle Velez
International Technical Support Organization, San Jose Center

Become a published author

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

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

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

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

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

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

ibm.com/redbooks

- ▶ Send your comments in an Internet note to:

redbook@us.ibm.com

- ▶ Mail your comments to:

IBM Corporation, International Technical Support Organization
Dept. QXXE Building 80-E2
650 Harry Road
San Jose, California 95120-6099



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 can be downloaded from the Web site <http://ibm.com/db2/v8>, and visit <http://ibm.com/db2/linux> for further information.

This chapter introduces DB2 Universal Database (UDB) Version 8 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 offerings
 - Features
 - Supported platforms
 - Products and packages
- ▶ DB2 environment
 - Deployment topologies
 - DB2 database objects

- ▶ Parallelism with DB2
 - SMP environments
 - Database clusters
 - 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 Universal Database 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.

As a result IBM is the market leader on Linux for relational database systems¹. 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

¹ Based on: The RDBMS Top 10: License Sales Analysis and Market Forecast, 2001-2006 Analyst: Carl W. Olofson, IDC Research, Oct. 2002

- ▶ 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. Not only is DB2 known for lower initial cost, it also provides the lowest long-term total cost of ownership (TCO) when compared to other databases from Microsoft and Oracle². Running DB2 on the open Linux platform and Intel or AMD processor-based servers or blades delivers an even more compelling price/performance story. DB2 is renowned for critical self-tuning and self-healing features. The Self Managing and Resource Tuning (SMART) technologies make DB2 easy to use and maintain while minimizing administration costs. 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.

The following sections will 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 UDB Version 8 features, supported platforms, and product offerings.

1.2.1 Features

DB2 Universal Database 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

² For details refer to <http://www.ibm.com/software/data/highlights/db2tco.html>

- ▶ 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 Fail-over support
- ▶ J2EE certified and Java application interfaces for JDBC and SQLJ
- ▶ Support for open source interfaces like PHP, Python and Perl
- ▶ DB2 XML Extender for storage and retrieval of XML documents
- ▶ High-speed, in-memory search capability with DB2 Net Search Extender
- ▶ 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

DB2 has been commercially available on Linux since Version 6. However most deployments today run on Version 7.2 or later. In Nov. 2002, IBM released DB2 V8.1, marking the next stage in the evolution of relational databases. This version introduced over 400 new features³, including:

- ▶ Innovative self-managing technologies to reduce DBA intervention
- ▶ New tools and wizards, such as Health Center, Memory Visualizer
- ▶ Simplified management of large scale databases on clusters
- ▶ Federated Web Services for new levels of Information Integration
- ▶ Enhanced XML support and productivity tools
- ▶ Connection Concentrator for more 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
- ▶ Null and default compression

1.2.2 Supported platforms

IBM is committed to Linux and supports it on a variety of hardware platforms. Besides the Intel 32-bit hardware, DB2 for Linux today runs on IBM eServer zSeries. IBM has announced support for 64-bit Intel Itanium 2 based systems, and intends to support the PowerPC based IBM eServer iSeries and pSeries Linux platforms. IBM has also publicly demonstrated DB2 for Linux running on 64-bit AMD Opteron powered systems.

On the 32-bit x86 platform, DB2 requires systems equipped with Intel Pentium, Intel Xeon, AMD Athlon or later processors. DB2 V7.2 is supported on any Linux distribution with 2.4 kernel, glibc 2.1.2 and libstdc++ 2.9.0 or later. With DB2 V8, the Linux Validation Program has been introduced to ensure better inter-operability of DB2 on Linux platforms to enhance customer experience. IBM

³ A more complete overview about the feature set of DB2 Universal Database Version 8 can be found at <http://www.ibm.com/db2/v8>

is working closely with the Linux community, Linux Distribution Partners, and Independent Software and Hardware Vendors to test DB2 using various Linux kernels, distributions, third party products, hardware and other components that interact with DB2. As a part of the validation program, before a product is supported with DB2, it undergoes rigorous testing, providing users the confidence to run DB2 with other products right out of the box.

The Linux distributions that have been successfully validated for use with DB2 include Red Hat Linux Professional 7.2, 7.3, 8.0, Advanced Server 2.1, SuSE Professional 7.3, 8.0, 8.1, SuSE Linux Enterprise Server 7, 8, SCO Linux 4.0, Turbolinux 7 Server, Turbolinux 8 Server, Turbolinux Enterprise Server 8.0, and other distributions powered by United Linux 1.0 with kernel and library levels specified at the following validation Web site:

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

To run Java applications and the graphical tools that come with DB2 (for example, Control Center) IBM JDK 1.3.1 is required. For convenience it is shipped on DB2 V8.1 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.

1.2.3 DB2 products and packages

DB2 offerings on Linux span the spectrum, ranging all the way from products on handheld devices to large clusters and mainframes. See Figure 1-1. These products are discussed further under several categories:

- ▶ DB2 packages for the production environment
- ▶ Products for accessing legacy and host data
- ▶ Additional DB2 features
- ▶ DB2 packages for Application Developers
- ▶ DB2 for pervasive platforms

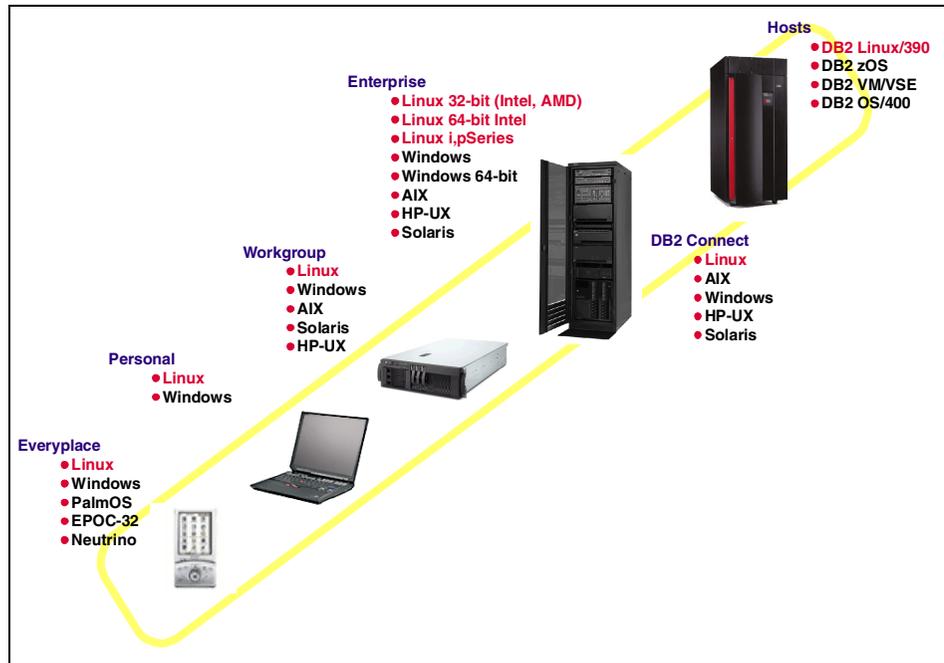


Figure 1-1 DB2 for Linux: From palmtop to teraflop

DB2 packages for the production environment

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

DB2 Personal Edition (PE): PE provides a single user database engine ideal for deployment on desktops of PC-based users. PE can be remotely managed and be used for managing other databases servers on the network, making it the perfect choice for deployment in occasionally connected or remote office implementations that don't require multi-user capability.

DB2 Workgroup Server Edition (WSE): WSE is the database server designed for deployment in a departmental or small business environment that involves a small number of internal users. WSE has a user based licensing model designed to provide an attractive price point for smaller installations while still providing a full function database server. WSE can be deployed on systems with up to four CPUs.

DB2 Workgroup Server Unlimited Edition (WSUE): WSUE offers a simplified per processor licensing model for deployment in a departmental or small business environment that has Internet users or number of users that makes per

processor licensing more attractive than the WSE licensing model. The WSUE is also for use on systems with up to four CPUs.

DB2 Enterprise Server Edition (ESE): ESE meets the database server needs of midsize to large businesses. ESE is the ideal foundation for building data warehouses, transaction processing, or Web-based solutions as well as a back-end for packaged solutions like ERP, CRM, and SCM. Additionally, ESE offers connectivity and integration for other enterprise DB2 and Informix data sources.

In versions of DB2 prior to V8.1, the product was available in two enterprise packages: DB2 Enterprise Edition (EE) and DB2 Enterprise-Extended Edition, with the latter providing the ability to create partitioned databases or run on a cluster. In V8.1, EE and EEE capabilities have been merged into a single ESE product, however running a partitioned database requires DPF licenses.

DB2 Database Partitioning Feature (DPF): The Database Partitioning Feature is a licensing option that allows ESE 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 application programming interface (API) drivers and connectivity infrastructure to enable direct connectivity from desktop applications to mainframe (zSeries, S/390, VM/VSE) and iSeries (AS/400) database servers. This product is specifically designed and is 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 addresses the needs of organizations that require robust connectivity from a variety of desktop systems to mainframe and iSeries database servers. DB2 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 (gateway) that accesses host data. The licensing model for this product is user based.

DB2 Connect Application Server Edition: DB2 Connect Application Server Edition product is identical to the DB2 Connect Enterprise Server in its

technology. However, its licensing terms and conditions are meant to address specific needs of multi-tier, client-server applications as well as applications that utilize Web technologies. DB2 Connect Application Server Edition license charges are based on the size of the number of processors available to the application servers where the application is running.

DB2 Connect Unlimited Edition: DB2 Connect Unlimited Edition product 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.

Additional DB2 features

In addition to the product offerings for accessing legacy and host data, DB2 also offers the following features.

DB2 Spatial Extender: This product enables users to leverage the power of the relational Database model for managing their location-based data just as easily as traditional business data, and use industry-standard SQL for spatial data analysis, which can add competitive advantages to existing and new applications. The DB2 Spatial Extender is available free of charge in the DB2 PE and DB2 WSE for up to five users. It is also available as a separate program for DB2 UDB WSUE and DB2 UDB ESE.

DB2 Net Search Extender: This product enables a solution to include a powerful in-memory search capability for text-based data. The integration of these advanced searching capabilities directly into the database saves the time and expense of integrating third-party, text-based solutions while providing the speed and flexibility in text searching demanded by e-commerce applications. The DB2 Net Search Extender is available free of charge in the DB2 PE and DB2 WSE for up to five users. It is also available as a separate program for DB2 UDB WSUE and DB2 UDB ESE.

DB2 Intelligent Miner Products: These products provide the analysis capabilities needed to make insightful business decisions. DB2 Intelligent Miner Scoring extends DB2 analysis capabilities and enables users to deploy mining in real-time applications with a simple SQL call. DB2 Intelligent Miner Modeling delivers DB2 Extenders (specifically stored procedures) for modeling operations like associations discovery, demographic clustering, and tree classification. DB2 Intelligent Miner Visualization provides Java visualizers, to interact and graphically present the results of modeling operations.

DB2 packages for Application Developers

DB2 provides two packages for application development.

DB2 Personal Developer's Edition (PDE): PDE enables a developer to design and build single user desktop applications. This offering comes with Linux and Windows versions of the DB2 PE products as well as the DB2 Extenders.

DB2 Universal Developers Edition (UDE): UDE offers a low-cost package for a single application developer to design, build, or prototype applications for deployment of any of the DB2 client or server platforms. It includes client and server DB2 editions for all Linux, UNIX and Windows supported platforms, DB2 Connect, DB2 Extenders, and Intelligent Miner. The software in this package cannot be used for production systems.

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.

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.

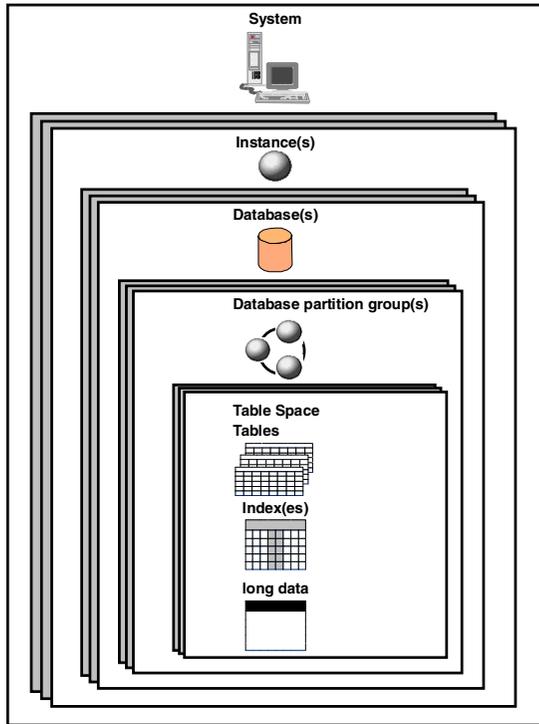


Figure 1-2 DB2 database objects

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 tablespace where the tables will be stored. If a partition group is not specified, there is a default group where tablespaces are allocated. In earlier versions of DB2, database partition groups were known as nodegroups. 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 tablespaces. A tablespace 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 tablespace can also be spread over one or more physical storage devices.

Tablespaces reside in database partition groups. Tablespace 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 over 99% 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. The unprecedented results of a recent SAP benchmark⁴ on an 8-way Linux system are a testament to the effectiveness of DB2 for Linux in a SMP environment. A large number of developers at the IBM Linux Technology Center are engaged with the Linux community to extend the capabilities of Linux on 16-way and higher SMP machines.

⁴ Refer to <http://www.sap.com/benchmark/pdf/cert5702.pdf>

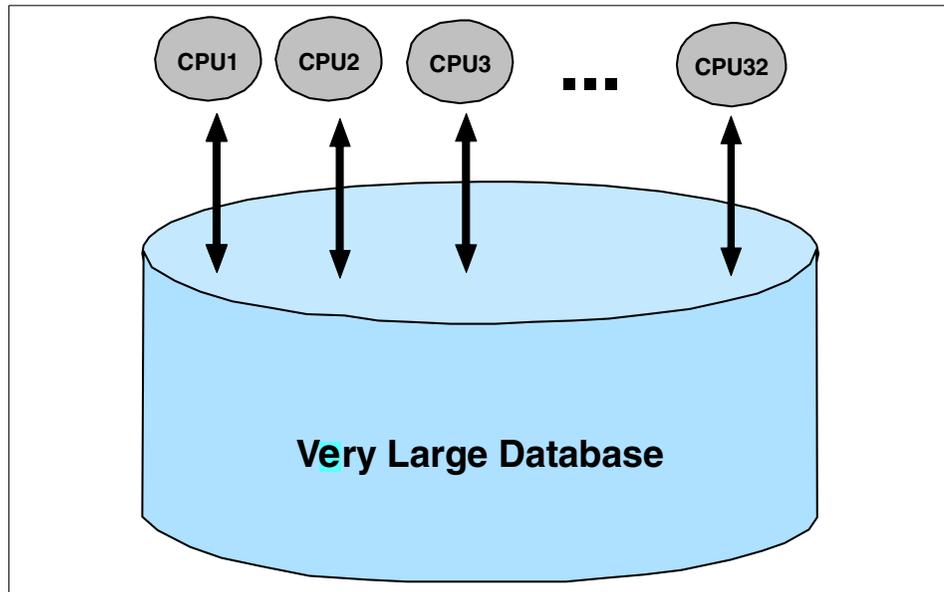


Figure 1-3 Database on a large SMP system

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.

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.

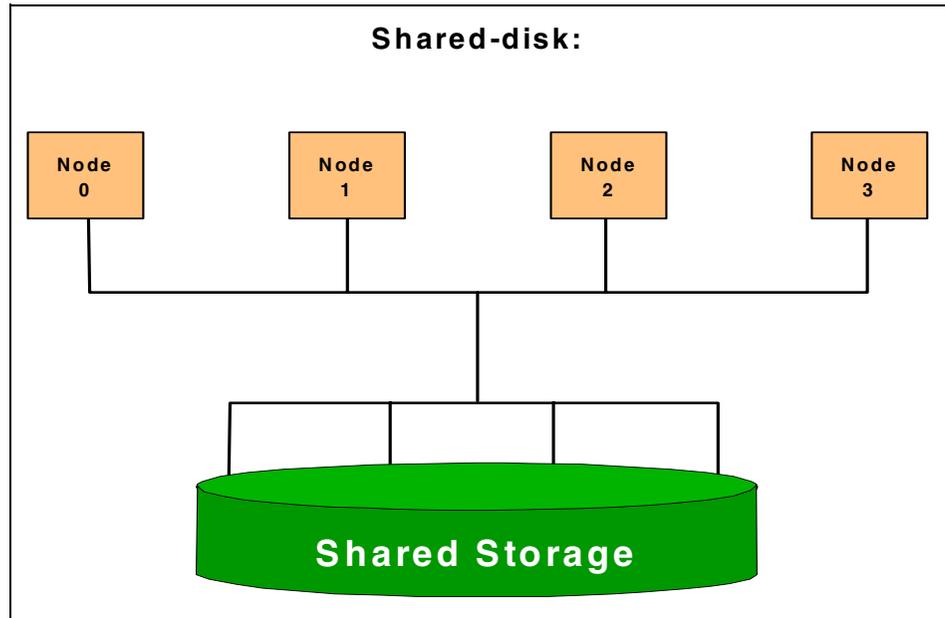


Figure 1-4 Shared disk architecture

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.

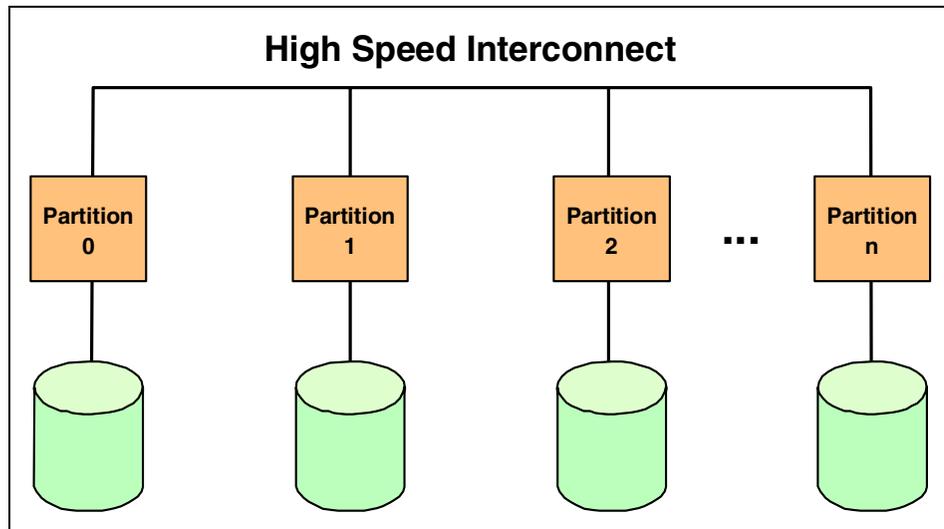


Figure 1-5 Shared-nothing architecture

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.

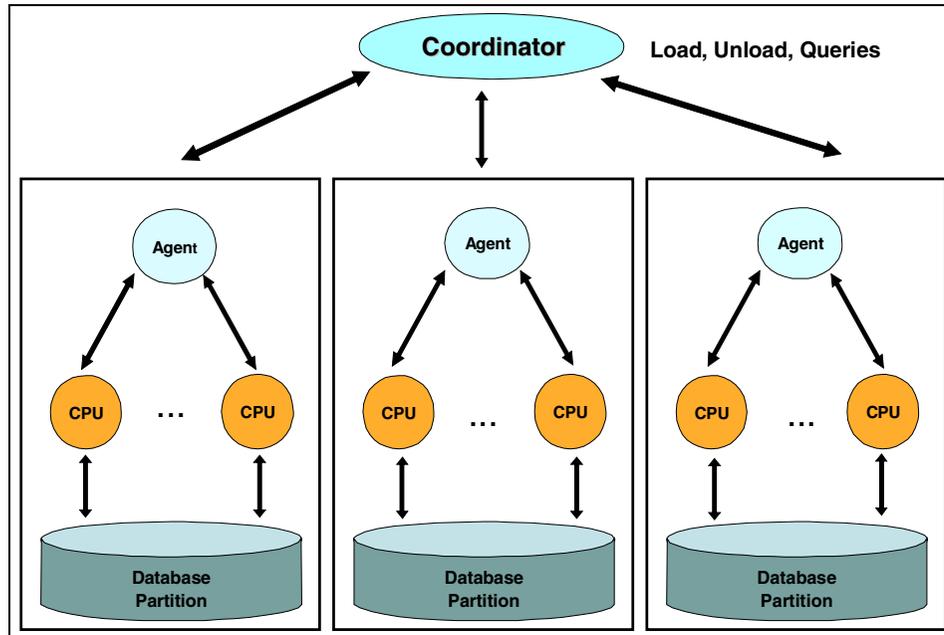


Figure 1-6 DB2 MPP environment

The data is distributed among database 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 1-7.

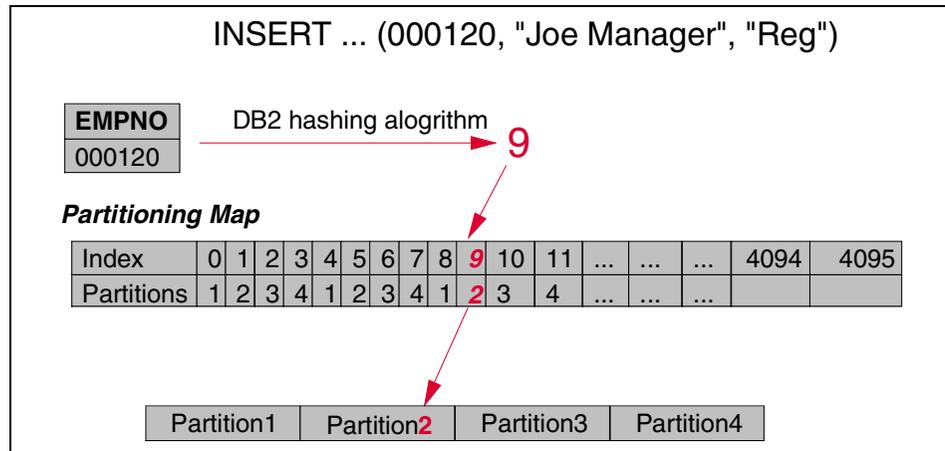


Figure 1-7 Intelligent data distribution across DB2 partitions

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 1 or 2 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 1-8.

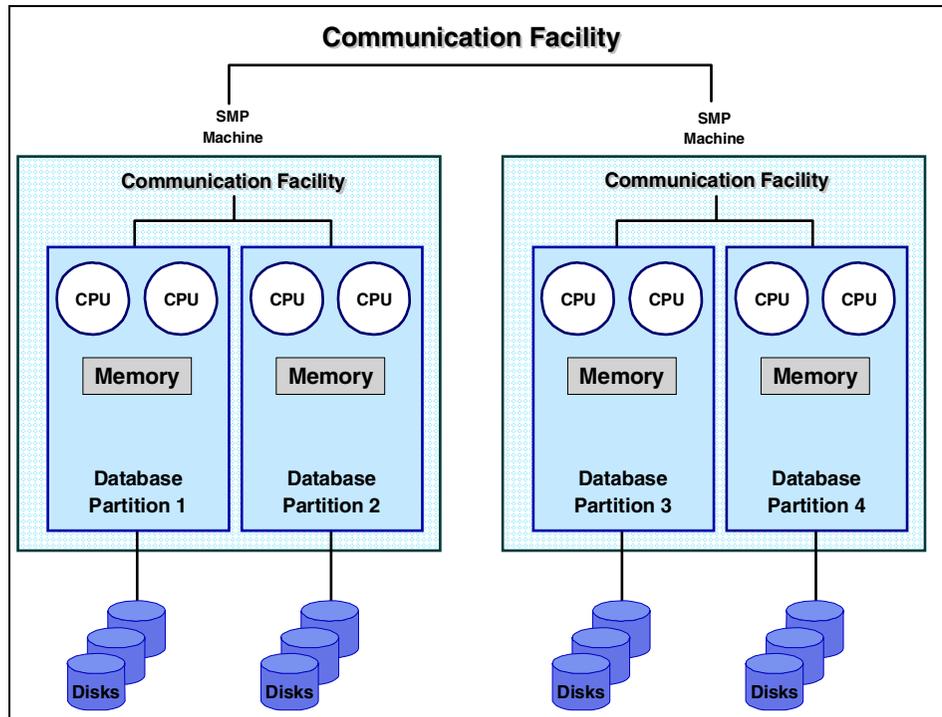


Figure 1-8 A partitioned database across a cluster of SMP machines

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.



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 all three installation methods: DB2 Setup, db2_install, and response file install.

The following topics are discussed:

- ▶ Basic requirements
- ▶ Installation considerations and planning
- ▶ Multiple-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:

- ▶ Linux distributions supported by DB2
- ▶ Required disk space
- ▶ Memory requirements
- ▶ Communication requirements
- ▶ Required kernel levels and libraries
- ▶ Additional software requirements

2.1.1 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-3.ibm.com/software/data/db2/linux/validate>

At the time of writing this book, the following Linux distributions have been successfully validated for use with DB2 (Table 2-1):

Table 2-1 Currently supported Linux distributions, kernels, and libraries

Platform	Distribution	Kernel	Libraries	Comments
IA 32	Red Hat 7.2	2.4.9-34	glibc 2.2.4	
	Red Hat 7.3	2.4.18	glibc 2.2.5	
	Red Hat AS 2.1	2.4.9	glibc 2.2.4	
	Red Hat 8.0	2.4.18-14	glibc 2.2.93-5	
	SCO Linux 4.0	2.4.19	glibc 2.2.5	
	SuSE 7.3	2.4.10	glibc 2.2.4	
	SuSE Pro 8.0	2.4.18	glibc 2.2.5	
	SuSE Pro 8.1	2.4.19	glibc 2.2.5	
	SuSE SLES-7	2.4.7	glibc 2.2.2	
	SuSE SLES-8	2.4.19	glibc 2.2.5	
	TurboLinux Server 7	2.4.9	glibc 2.2.4	
	TurboLinux Server 8	2.4.18-5	glibc 2.2.5	
	United Linux 1.0	2.4.19	glibc 2.2.5	
IA 64	SuSE SLES 8	2.4.19	glibc 2.2.5	
	United Linux 1.0	2.4.19	glibc 2.2.5	
zSeries (32 bit)	Red Hat 7.2	2.4.9-38	2.2.4	Run up2date to get the required kernel fixes from Red Hat
	SuSE SLES 7	2.4.7-58	2.2.4	SuSE refresh is required: SLES-7-PatchCD-1-s390-20020522.iso -- compat.rpm contains libstdc++ 6.1
	SuSE SLES 8	2.4.19	2.2.5	

2.1.2 Required disk space

You must take into account the following disk requirements while configuring your partitions (Table 2-2):

Table 2-2 Required disk space

Install Type	Description	Required disk space
Typical	DB2 is installed with most features and functionality, including graphical tools such as the Control Center and Configuration Assistant.	450 to 500 MB
Compact	Installs only basic DB2 features and functions and does not include graphical tools.	350 to 400 MB
Custom	This option allows you to select the features you want to install.	350 to 700 MB

You should also allocate disk space for required software, communication products, DB2 documentation, and databases. In DB2 Version 8, HTML and PDF documentation are now provided on separate CD-ROMs. Installing core DB2 HTML documentation in *English only* requires 25 MB of disk space; installing all HTML documentation in *English only* requires 82 MB of disk space. Additional space is required for additional languages.

2.1.3 Memory requirements

We recommend that you allocate a minimum of 256 MB of RAM for DB2 ESE, but 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.
- ▶ Additional memory should be allocated for non-DB2 software that may be running on your system.
- ▶ Additional memory is required to support database clients and database activity.
- ▶ Memory requirements will be affected by the size and complexity of your database system, as well as specific performance requirements.

2.1.4 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 the following should be collected from the network administrators (Table 2-3):

Table 2-3 Required network information

Name	Example number
Host IP address	191.72.1.3
Subnet mask	255.255.255.0
Gateway	191.72.1.1
Domain name server	191.72.3.1

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 2.1.6, "Additional software requirements" on page 31.

2.1.5 Required kernel levels and libraries

The minimum supported kernel levels and libraries are provided in Table 2-1 on page 25. Your Linux distribution should also include RPM 3 or later.

Kernel parameter values

You may be required to update some of the default kernel parameter settings for DB2 to run successfully on Linux. For example, the 2.4.x series kernel message queue parameter *msgmni* has a default value that only allows a limited number of simultaneous connections to DB2. The semaphore array and shared memory limit values are also insufficient by default.

In general, these are the recommended values for DB2 to run optimally:

```
kernel.shmmax=268435456 for 32-bit; 1073741824 for 64-bit
kernel.msgmni=1024
fs.file-max=8192
kernel.sem="250 32000 32 1024"
```

Fortunately, DB2 Version 8 has a new feature that checks values of the *kernel.semmni*, *kernel.msgmni*, and *kernel.shmmax* parameters at `db2start` time, and changes them for you if the current values are not optimal. `DB2start` 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)

For example, after issuing `db2start` for the first time, we got the following messages in `db2diag.log`:

Example 2-1 db2diag.log entries after starting instance

```
ADM0506I DB2 has automatically updated the "semmni"
kernel parameter from
"128" to the recommended value "1024".
```

```
2002-10-31-16.38.59.074791 Instance:db2inst1
Node:000
PID:15996(db2sysc) TID:8192 Appid:none
base sys utilities sqlesysc_main Probe:9
```

```
ADM0506I DB2 has automatically updated the "msgmni"
kernel parameter from "16"
to the recommended value "1024".
```

```
2002-10-31-16.38.59.076916 Instance:db2inst1
Node:000
PID:15996(db2sysc) TID:8192 Appid:none
base sys utilities sqlesysc_main Probe:9
```

```
ADM0506I DB2 has automatically updated the "shmmax"
kernel parameter from
"33554432" to the recommended value "268435456".
```

```
2002-10-31-16.39.01.262594 Instance:db2inst1
Node:000
PID:15994(db2star2) TID:8192 Appid:none
base sys utilities startdbm Probe:911
```

```
ADM7513W Database manager has started.
```

You may have noticed that `db2start` did not update the `fs.file-max` parameter — this is because any kernel at 2.4.18 and higher does this automatically for you.

Because of this new feature, it is no longer necessary for you to manually update the *kernel.shmmax*, *kernel.msgmni*, and *kernel.sem* parameters prior to install.

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, enter the **ipcs -l** command. Your output should look something like this:

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

----- Semaphore Limits -----
max number of arrays = 1024
max semaphores per array = 250
max semaphores system wide = 32000
max ops per semop call = 32
semaphore max value = 32767

----- Messages: Limits -----
max queues system wide = 1024
max size of message (bytes) = 8192
default max size of queue (bytes) = 16384
```

Modifying 2.4.x series kernel parameters on Red Hat

In this example, we will explain how to update the *kernel.shmmax*, *kernel.sem*, and *kernel.msgmni* parameters on Red Hat and set them after each reboot.

1. Login as a user with root authority
2. Using a text editor, add the following entries to */etc/sysctl.conf*:

```
kernel.shmmax=268435456
kernel.msgmni=1024
kernel.sem="250 32000 32 1024"
```

3. Load these entries into sysctl by entering the following command: **sysctl -p**

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

Modifying 2.4.x series kernel parameters on SuSE

Modifying kernel parameters on SuSE is a little different than Red Hat. These instructions will explain how to update the *kernel.shmmax*, *kernel.sem*, and *kernel.msgmni* parameters and set them for reboot.

1. Log in as a user with root authority.
2. Some SuSE distributions do not have a */etc/sysctl.conf* file. If that is the case, you need to create one manually using a text editor.
3. In the */etc/sysctl.conf* file, add the following entries:

```
kernel.shmmax=268435456
kernel.msgmni=1024
fs.file-max=8129
kernel.sem="250 32000 32 1024"
```

4. Run **sysctl -p** to load in sysctl settings from the default file */etc/sysctl.conf*.

```
sysctl -p
```
5. Next, you need to add `sysctl -p` to a system initialization file to set kernel parameters after each reboot. To do this, write a script and configure it to run automatically at runlevel 5. Specifically, you need to create an executable file in */etc/init.d*, then add pointers to this script in */etc/init.d/rc5.d*. Follow our example:

In */etc/init.d*, we created an executable file named *kerneldb2*. This file contains the following script:

```
#!/bin/sh
#
#
# /etc/init.d/kerneldb2
#
### END INIT INFO
touch /var/lock/subsys/kerneldb2
/sbin/sysctl -p >> /var/lock/subsys/kerneldb2
```

Then in */etc/init.d/rc5.d*, we added pointers to the *kerneldb2* script by entering the following commands:

```
bash# cd /etc/init.d/rc5.d
bash# ln -s ../kerneldb2 S99kerneldb2
bash# ln -s ../kerneldb2 K99kerneldb2
```

Note: We chose the number 99 because the number between letter S and the name should be greater than the number on all other links in the rc5.d directory for this script to be executed last.

Like Red Hat, you can change kernel parameters at run-time (for testing purposes) using the **sysctl -w** command. For example, to change the semaphore array parameter, enter:

```
sysctl -w kernel.sem="250 32000 32 1024"
```

Remember, these settings will only be saved at the next reboot if they are in the `/etc/sysctl.conf` file.

2.1.6 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.

- ▶ The IBM Java Developer's Kit (JDK) Version 1.3.1, Service Release 1. This 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, it will be installed for you during the installation process if you use DB2 Setup or a response file. The `db2_install` utility does *not* install the JDK for you. Refer to the end of this section for instructions about how to install the JDK...

Note: If you have another version of the JDK installed on your machine, we recommend that you remove it before installing DB2. The Control Center may not launch if two different versions of the JDK co-exist on your machine. If your application requires the existing JDK or JRE, you may be required to disable it by unsetting environment information such as `PATH` or `CLASSPATH` before installation.

- ▶ Web browser, to view DB2 HTML documentation.
- ▶ X Window System software, capable of rendering a graphical user interface. You need this if you want to use the DB2 Setup wizard to install DB2 (it's a graphical installer) or any DB2 graphical tools.
- ▶ Additional software, provided in the following tables (Table 2-4 and Table 2-5). The `pdsh` package is required for all DB2 systems, and the `rsh-server` and `nfs-utils` packages are mostly required for partitioned database systems.

Table 2-4 Package requirements for SuSE

Package name	RPM name	Description
pdksh	pdksh-5.2.14-293.i386.rpm (for V7.2) pdksh-5.2.14-515.src.rpm (for V8)	Korn Shell. This is required for multi-partitioned database environments.
rsh-server	rsh-0.17-214.src.rpm (for V7.2) rsh-0.17-285.src.rpm (for V8)	Contains a set of server programs which allow users to run commands on remote machines, login in to other machines, and copy files between machines (rsh, rexec, rlogin, and rcp)
nfs-utils	nfs-utils-0.3.3-109.src.rpm (for V7.2) nfs-utils-1.0.1-26.src.rpm (for V8)	Network File System support package. It allows access for local files to remote machines.

Table 2-5 Package requirements for Red Hat

Directory	Package name	RPM name	Description
/System Environment/ Shell	pdksh	pdksh-5.2.14-13.i386.rpm (for V7.2) pdksh-5.2.14-19.i386.rpm (for V8)	Korn Shell. This is required for multi-partitioned database environments.
/System Environment/ Daemons	rsh-server	rsh-0.17-5.i386.rpm (for V7.2) rsh-0.17-10.src.rpm (for V8)	Contains a set of programs which allow users to run commands on a remote machine. Required for multiple-partition environments.
/System Environment/ Daemons	nfs-utils	nfs-utils-0.3.3-5.src.rpm (for V7.2) nfs-utils-1.0.1-2.src.rpm (for V8)	Network File System support package. It allows access for local files to remote machines.

To check whether you have these packages installed, use the `rpm -q` command:

```
rpm -qa | grep pdksh
rpm -qa | grep rsh
rpm -qa | grep nfs
```

To install these packages on Red Hat, use the Red Hat CDs and `rpm` command. For example, to install `pdksh` on Red Hat V8, mount the Red Hat CD #3, and enter the following command:

```
rpm -ivh /mnt/cdrom/RedHat/RPMS/pdksh-5.2.14-19.i386.rpm
```

You can also use `rpm` on SuSE. If your SuSE distribution has X Window System software, it is easiest to install these packages using YaST. For example, to install `pdksh` on SuSE V8, do the following:

1. Log on as root
2. From the YaST2 Control Center, select **Software -> Install or Remove Software**, then choose the appropriate packages.

Notes:

- ▶ The RPM name may change depending on the Linux version you are using. Check the latest Linux information for the RPM name that you require.
- ▶ The `nfs-utils` package provides a daemon for the kernel NFS server and related tools, which provides a much higher level of performance than the traditional Linux NFS server used by most users.
- ▶ Although the packages `compat-libs` and `compat-egcs-C++` were required for DB2 version 7 C/C++ development, they are no longer required for DB2 Version 8 development. In V8 we have switched to the Intel compiler.

Installing the IBM Developer Kit for Java, Version 1.3.1

The `db2_install` utility will not install the IBM Developer Kit for Java (JDK), which is required for DB2 GUI tools, such as the Control Center and Task Center, and for application development. Therefore, if you plan to use `db2_install` and also want to use the GUI tools or application development tools, you must install the JDK manually by following these steps.

1. Acquire the IBM Developer Kit for Java, Version 1.3.1, from your DB2 ESE CD-ROM in the `/ese/db2/linux/Java-1.3` directory.
2. Install the IBM Developer Kit for Java by entering the following command as root:

```
rpm -ivh IBMJava2-SDK-1.3.1-2.0-i386.rpm
```

3. If you plan to develop Java applications, set up the Java environment for all users by adding the following lines to your `/etc/profile` file:

```
export PATH=$PATH:/opt/IBMJava2-131/jre/bin
```

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:

- ▶ Installation methods - `db2setup` versus `db2_install` versus response file
- ▶ Storage planning
 - Linux filesystems versus raw devices
 - Raw device configuration
 - Filesystem configuration
 - DB2 log space
 - DB2 temporary table space
- ▶ Network configuration for multiple-partitioned installation
 - Enabling `rsh`
 - Setting up NFS
- ▶ User and group setup
 - User and group requirements
 - Creating users - single partition
 - Creating users - multiple partition
 - DAS user considerations for a multiple partition environment

2.2.1 Installation methods

There are four methods in which you can install DB2 on Linux: (1) the DB2 Setup wizard, (2) the `db2_install` utility, (3) using a response file, and (4) Linux RPM command. 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 filesets, 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

The `db2_install` script installs all DB2 ESE packages on your Linux system using the RPM installation utility. This method is reliable and commonly used by expert users for installing DB2 on larger, more complex multiple-partition systems. Tasks such as userids/groups set up, instance creation, tools catalog database creation, and notification set up, need to be performed manually after the installation.

Some people prefer this method because it bypasses the configuration performed by DB2 Setup and allows you to configure DB2 your preferred way in the first place. For example, it gives you more control over database management. By default, the DB2 Setup wizard creates the tools catalog database in the instance owner's home directory if you choose not to put it in an existing database, and also allocates the home directory for storage of other databases. Particularly for larger, multiple-partition systems, we recommend that the databases be stored in a separate filesystem (for example, `/database`). Performing manual configuration allows you to allocate your preferred default database directory in the first place.

A limitation to `db2_install` is that it only installs support for English. This means that help, messages, and tool interfaces are in English. The DB2 Setup wizard, on the other hand, can install support for one or more different languages. Other limitations are that the installation may take longer (considering the higher number of manual configuration tasks), it requires a higher level of skill, and it cannot create response files.

Response file install

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.

RPM command

RPM is a Linux software installation command. The various DB2 installation methods use the RPM command to lay down the appropriate filesets on the system. Using the RPM command to install DB2 allows you to select specific DB2 files. However, the RPM installation method will only install the DB2 code. You will not be able to create instances, user IDs, or response files during DB2 installation. Please note that this installation method is not supported or recommended.

2.2.2 Storage planning

In this section we discuss the following storage considerations:

- ▶ File systems versus raw devices
- ▶ Raw device configuration
- ▶ File system configuration
- ▶ Log storage
- ▶ Temporary table space storage
- ▶ Location of the DB2 tools catalog

File systems versus raw devices

In this section, we describe and compare file systems and raw devices as storage mechanisms for DB2 on Linux.

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). SMS tablespaces are commonly used because they provide good performance with very little administration cost. 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.

Raw devices

Raw devices can only be used for DMS storage. DMS tablespaces require more administration, but provide superior performance. Raw I/O itself improves performance because it bypasses the buffering performed by the Linux kernel and allows processes to interact with the physical device directly. In comparison, the filesystem uses system-based I/O, which means that data is buffered twice —

first at the database manager level and then at the kernel (or filesystem) level. At this level, the kernel intercepts the calls and transfers the data to its own buffer before passing it on to the physical device or process. The additional buffering impedes performance.

DB2 supports raw I/O on the Linux distributions listed in Table 2-6. This means that a raw device can be attached to any UDB Universal Database system running on these systems.

The naming of default raw device nodes differs between Linux distributions, as shown in the following table. Raw devices are not supported by DB2 on Linux/390.

Table 2-6 Linux distributions currently supporting raw I/O

Distribution	Raw device nodes	Raw device controller
Red Hat or TurboLinux	/dev/raw/raw1 to 255	/dev/rawctl
SuSE	/dev/raw1 to 63	/dev/raw

Raw device configuration

For a detailed procedure of how to set up raw I/O on your Linux machine, refer to *IBM DB2 UDB Administration Guide: Implementation*, SC09-4820. In this section, we explain how to bind the raw device to the Linux kernel.

To perform raw I/O on a block device, you must first bind the Linux kernel's pool of raw device nodes to that block device. This is done by running the **raw** command as root. The raw utility is normally supplied by the Linux distributors nowadays.

For example, if your disk partition is */dev/hda12* and you want to bind it to */dev/raw/raw1*, enter the following command:

```
bash# raw /dev/raw/raw1 /dev/hda12
```

Next, configure the */etc/rc.d/boot.local* file so that DB2 can access the raw device at boot time. In Example 2-2 we added the following entries to */etc/rc.d/boot.local*:

Example 2-2 Bind raw device with block device

```
raw /dev/raw/raw1 /dev/hda12  
raw /dev/raw/raw2 /dev/hda13
```

To show all current raw devices available, issue following command:

```
raw -qa
```

Example 2-3 was our output:

Example 2-3 Display available raw devices

```
udb1nx04:~ # raw -qa
/dev/raw/raw1: bound to major 3, minor 12
/dev/raw/raw2: bound to major 3, minor 13
```

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 filesystems, 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 pathname 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 filesystem formats, such as Ext3, JFS, XFS and ReiserFS. In our example, for database-related file systems, we use the Ext3 format to create a journal filesystem. Please note that use of Ext3 file system is not recommended for use with Kernel levels lower than 2.4.18 without appropriate fixes.

To show all defined file systems, issue the **df** command, as shown in Example 2-4.

Example 2-4 df command

```
udb1nx04:~ # df
Filesystem      1K-blocks    Used Available Use% Mounted on
/dev/hda1        4200828    1588840   2611988   38% /
/dev/hda5         54416      4682     46925   10% /boot
/dev/hda8       3099260    32828   2909000    2% /database
/dev/hda9       1035660    32828    950224    4% /db2log1
/dev/hda10      2071384    32828   1933332    2% /db2temp
shmfs           387124      0     387124    0% /dev/shm
/dev/hda7       1035660    32828    950224    4% /db2home_Loc1
```

/dev/hda11	3099260	20	2941808	1%	/software_Local
/dev/hda19	1035660	32828	950224	4%	/db2log2
udblnx02.almaden.ibm.com:/software	3099260	1044136	1897692	36%	/software

To show all NFS mounted filesystems, issue following command on the host where the filesystem physically exists (Example 2-5).

showmount -a

Example 2-5 showmount command

```
udblnx02:~ # showmount -a
All mount points on udblnx02:
udblnx03.local:/software
udblnx04.local:/software
udblnx05.local:/software
```

Filesystem 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*. Avoid creating databases on this file system.

For *partitioned 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 */db2home* with a mount point */db2home*. Refer to “Setting up NFS” on page 45 for instructions about how to set up this file system on NFS.

- ▶ We also recommend that you create a separate filesystem 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 tablespaces (this is discussed in more detail later in this section). In addition, you may want to create a separate filesystem for user tablespaces 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 2-7). We will be using this system setup throughout the rest of this book.

Table 2-7 Sample partition setup

Partition name	Description
/	Linux
/boot	Linux
/swap	Linux Swap space
/db2home	For storing the home directories for DB2 users (for example, db2inst1, db2fenc1, dasusr1)
/database	For storing the database
/db2log1	Used to storing primary copy of logs
/db2log2	Used to store DB2 mirrored logs
/db2temp	For storing DB2 temporary tablespaces
/software	Used for storing software. For example, we downloaded the DB2 install image into this directory.
/tablespaces	Holds all user tablespaces

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.

Log space

By default, DB2 sets the log path to the default database path during database creation. For example, Our default database path is /db2home and our log path is:

```
/db2home/db2inst2/db2inst2/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.

In Example 2-6, we change the primary log path from the default to /db2log1 and set the mirror log path to /db2log2.

Example 2-6 db2 logs configuration

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

The procedure of changing the log path is discussed in “Change log path” on page 97.

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

Temp space

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

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 2-7 shows how to create a system temporary tablespace on multiple nodes in the /db2temp filesystem:

Example 2-7 Create temp tablespace in a different filesystem

```
connect to db_name;

create temporary tablespace TEMPSPACE01 in IBMTEMPGROUP
managed by SYSTEM
using ('/db2temp/$INSTANCE/db_name/n001tmp/tempspace01') on dbpartitionnum (1)
using ('/db2temp/$INSTANCE/db_name/n002tmp/tempspace01') on dbpartitionnum (2)
using ('/db2temp/$INSTANCE/db_name/n003tmp/tempspace01') on dbpartitionnum (3)
using ('/db2temp/$INSTANCE/db_name/n004tmp/tempspace01') on dbpartitionnum (4)
extentsize 32
```

```
prefetchsize 128
bufferpool IBMDEFAULTBP
overhead      24.1
transferrate  0.9;

drop tablespace tempSPACE1;
```

For detailed steps of creating temp tablespaces in a different filesystem, refer to “Reallocate temp and user space1” on page 95.

2.2.3 Network configuration for a multiple partitioned 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 rsh and NFS.

Enabling rsh

To run DB2 ESE successfully, the rsh server has to be available on all machines in the partitioned database system. This server spawns a shell on the remote host and allows the user to execute commands. The rsh service is managed by inetd. Prior to enabling rsh, please ensure the rsh server rpm is installed, because many distributions do not install it by default.

Enabling rsh on SuSE

To enable the rsh server and other related services on SuSE, perform the following steps:

1. Log on as root and manually edit the `/etc/inetd.conf` file. Remove # at the beginning of the line to enable a service, and disable what services you do not need by commenting them out (by adding a # at the beginning of the line). Only uncommented services are available. Example 2-8 provides an excerpt of the `/etc/inetd.conf` file on SuSE v8. In this example, we have enabled the `in.ftpd`, `in.telnetd`, `in.rshd`, and `in.rlogind` services, which enables ftp, telnet, rsh, and rlogin on your machine.

Example 2-8 /etc/inetd.conf file

```
# /etc/inetd.conf
# These are standard services.
#
ftpd stream tcp nowait root /usr/sbin/tcpd in.ftpd
# ftp stream tcp nowait root /usr/sbin/tcpd vsftpd
#
# If you want telnetd not to "keep-alives" (e.g. if it
# runs over a ISDN uplink), add "-n". See 'man telnetd' for more details.
telnet stream tcp nowait root /usr/sbin/tcpd in.telnetd
```

```

# nntp stream tcp nowait news /usr/sbin/tcpd /usr/sbin/leafnode
# smtp stream tcp nowait root /usr/sbin/sendmail sendmail -L sendmail -Am
#-bs
# Shell, login, exec and talk are BSD protocols.
# The option "-h" permits ``.rhosts'' files for the superuser. Please look
# at man-page of rlogind and rshd to see more configuration possibilities
#about .rhosts files.
shell stream tcp nowait root /usr/sbin/tcpd in.rshd -L
# shell stream tcp nowait root /usr/sbin/tcpd in.rshd -aL
#
# If you want rlogind not to "keep-alives" (e.g. if it runs over a ISDN
# uplink), add "-n". See 'man rlogind' for more details.
login stream tcp nowait root /usr/sbin/tcpd in.rlogind
# login stream tcp nowait root /usr/sbin/tcpd in.rlogind -a
# exec stream tcp nowait root /usr/sbin/tcpd in.rexecd
# talk dgram udp wait root /usr/sbin/tcpd in.talkd
# ntalk dgram udp wait root /usr/sbin/tcpd in.talkd
# End

```

-
2. Re-start inetd super-server by logging in as root and entering the following command:

```
/etc/init.d/inetd restart
```

You can also use the GUI based YaST2 network configuration tool to enable inetd services. To do so:

1. From the YaST2 menu, select **YAST2 modules** → **Network/Basic** → **Start/stop services (inetd)**.
2. On the window that appears, choose **ON WITH CUSTOM CONFIGURATION** and click **Next**. On the enable/disable network services window, use your mouse to highlight the lines corresponding to the rshd service and press **Activate or Deactivate** for each one. When you are finished, press **Finish**.

Enabling rsh on Red Hat

1. Log in as a user with root authority.
2. Edit the /etc/xinetd.d/rsh file by changing disable flag to **no**.
3. Restart xinetd server by entering the following command:

```
/etc/init.d/xinetd restart
```

Your /etc/xinetd.d/rsh file should look like Example 2-9:

Example 2-9 Red Hat rsh configuration file

```

# default: on
# description: The rshd server is the server for the rcmd(3) routine and, \

```

```

#      consequently, for the rsh(1) program. The server provides \
#      remote execution facilities with authentication based on \
#      privileged port numbers from trusted hosts.
service shell
{
    socket_type          = stream
    wait                 = no
    user                  = root
    log_on_success        += USERID
    log_on_failure        += USERID
    server                = /usr/sbin/in.rshd
    disable               = no
}

```

Alternatively, you can use the **ntsysv** command to launch Red Hat's *ntsysv* tool. This application allows you to configure which services are started at boot time for each runlevel. Please note that the changes that you make will not take effect immediately and that *ntsysv* cannot be used to stop, start or re-start services.

Network troubleshooting on Red Hat

If you are having problems getting network communications set up on Red Hat, you should look at the following network configuration files.

- ▶ */etc/hosts* - This is the hostname resolution file (resolves a hostname to an IP address). This file *must* contain the **localhost** entry **127.0.0.1**. Our */etc/hosts* file looks like this:

```

# /etc/hosts file on host "udb1nx03"
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1      localhost
9.1.38.63     udb1nx01
9.1.38.64     udb1nx02
9.1.38.65     udb1nx03 udb1nx03.local
9.1.38.66     udb1nx04
9.1.38.67     udb1nx05

```

- ▶ */etc/sysconfig/network* - This file specifies routing and host information for all network interfaces. It should look something like this:

```

# our /etc/sysconfig/network file on host "udb1nx03"
NETWORKING=yes
HOSTNAME=udb1nx03

```

- ▶ */etc/sysconfig/network-scripts/ifcfg-eth0* - This contains an interface configuration script that controls the first network interface card in the system. For example, our *ifcfg-eth0* file contains the following data:

```

DEVICE=eth0
BOOTPROTO=none

```

```
BROADCAST=9.1.39.255
IPADDR=9.1.38.65
NETMASK=255.255.254.0
NETWORK=9.1.38.0
ONBOOT=yes
USERCTL=no
PEERDNS=no
TYPE=Ethernet
GATEWAY=9.1.38.1
```

- ▶ */etc/resolv.conf* - This configuration file specifies the IP addresses of the DNS server and the search domain. Unless configured to do otherwise, the network initialization scripts populate this file. Our */etc/resolv.conf* file contains the IP addresses of two nameservers:

```
nameserver 9.1.8.254
nameserver 9.1.24.254
```

There is plenty of information about these files in the Red Hat **man** pages and on the Red Hat Web site.

Another thing to check is your default firewall settings. If the firewall is set to the default (*high security*) your connections may not go through. We recommend that you change this firewall setting and customize it to your environment.

Setting up NFS

In order to get NFS up and running, you need to configure the */etc/exports* file 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 will lead you through the following 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 your 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:

```
showmount: ServerA: RPC: Program not registered
```

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.

If NFS is not running:

- ▶ Make sure you have the `nfs-utils` or `nfs-server` rpm package installed on each machine.

- ▶ Start the server manually by running the following command as root:

```
[root@udblnx02 /root]# /etc/init.d/nfs restart
```

(For Red Hat, it is “nfs”; for SuSE, it is “nfsserver”)

- ▶ You may check the status of NFS by running the following command:

```
[root@udblnx02 /root]# /etc/init.d/nfs status
```

Create and configure the DB2 home file system to NFS

As mentioned earlier in 2.2.2, “Storage planning” on page 36, 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 call this file system `/db2home`. NFS is used to share this file system.

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

```
mount /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 `/db2home`. In our setup, the entry is:

```
/dev/hda7 /db2home ext3 defaults 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:

```
/db2home/ *(rw,no_root_squash,sync)
/software/ *(rw,no_root_squash,sync)
```

We also wanted to share our `/software` file system, because this is where the DB2 install image and response files are stored.

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.

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` which are deleted from `/etc/exports`.

6. On each of the remaining machines in the cluster, 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:

```
udblnx05:/db2home /db2home nfs rw,timeo=300,retrans=5,
hard,intr,bg,suid,rw
```

Where, `udblnx05` represents the NFS server machine name.

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 udb1nx05:/db2home /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:

On SuSE:

```
/etc/init.d/nfsserver restart
```

On Red Hat:

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

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

```
showmount -e udb1nx05
```

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. For example, if your server is exporting its `/usr/local` hierarchy, you should place the directory in `/export`, thereby creating `/export/usr/local`. Because the server itself will need access to the `/export/usr/local` directory, you should create a symbolic link from `/usr/local` that points to the real location, `/export/usr/local`.
- ▶ 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.

2.2.4 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) user. 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 IDs and group names, which are shown in Table 2-8.

Table 2-8 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 JDFs and stored procedures.
DAS user	dasusr1	db2asgrp	Administers the DB2 Administration Server

Creating users

You must have root authority to create users and groups. There are three ways in which you can create a DB2 user ID: (1) using the DB2 Setup wizard, (2) a response file or (3) manually using the command line.

1. DB2 Setup Wizard

The DB2 Setup Wizard creates all of the required users and groups for you during installation. The default users and groups that get created are displayed in Table 2-8. DB2 Setup also gives you an option to specify your own user and group names.

2. Response file

Users can also be created during a response file installation if you specify user and group information in the response file. For example, the following entries in our response file create the three required users and groups for DB2:

Example 2-10 Using a response file to create users and groups

```
* DAS user
DAS_USERNAME = dasusr1
DAS_GROUP_NAME = dasadm1
DAS_HOME_DIRECTORY = /home/dasusr1
DAS_PASSWORD = 235262333285355231346
ENCRYPTED = DAS_PASSWORD

* Instance-owning user
inst1.NAME = db2inst1
inst1.GROUP_NAME = db2grp1
inst1.HOME_DIRECTORY = /db2home/db2inst1
inst1.PASSWORD = 235262333285355231346
```

```
ENCRYPTED = inst1.PASSWORD
inst1.AUTOSTART = YES
inst1.AUTHENTICATION = SERVER
inst1.SVCENAME = db2c_db2inst1
inst1.PORT_NUMBER = 50001
inst1.FCM_PORT_NUMBER = 60000
inst1.MAX_LOGICAL_NODES = 4
* Fenced user
nst1.FENCED_USERNAME = db2fenc1
inst1.FENCED_GROUP_NAME = db2fgrp1
inst1.FENCED_HOME_DIRECTORY = /db2home/db2fenc1
inst1.FENCED_PASSWORD = 235262333285355231346
ENCRYPTED = inst1.FENCED_PASSWORD
```

3. Manually using command line. To use this method, follow these steps:
 - a. Log on to your machine as root.
 - b. 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 db2asgrp
```

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.
 - c. Create a user that belongs to each group and specify the home directory. We chose to place all home directories in /db2home by entering the following commands:

```
useradd -u 1004 -g db2iadm1 -m -d /db2home/db2inst1 db2inst1 -p password1
useradd -u 1003 -g db2fadm1 -m -d /db2home/db2fenc1 db2fenc1 -p password2
useradd -u 1002 -g db2asgrp -m -d /db2home/dasusr1 dasusr1 -p password3
```
 - d. 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.

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. The DB2 Administration Server (DAS) has changed significantly in DB2 Version 8; therefore, you should take note of the following DAS user considerations.

DAS user considerations for a partitioned database

- ▶ 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 V8 DAS on each machine (although a V7 and V8 DAS can co-exist on one machine).
- ▶ 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. (The latter is the case for our environment — we have a separate user ID `dasusr1` on each machine, each with its own local home directory `dasusr1`. It works just fine).
- ▶ If the same user ID is to be used on each machine, then that user ID's home directory cannot be shared with the other machines.
- ▶ If a different user ID is used for each DAS, then the home directories of the user IDs that are used can be 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.

2.3 Installing DB2

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

- ▶ DB2 Setup — single and multiple-partition installation
- ▶ db2_install utility
- ▶ Adding a partitioned system
- ▶ Installing on NIS

2.3.1 DB2 Setup

There are two ways in which to use DB2 Setup: (1) install a single partition database system using either the DB2 Setup wizard or a response file, or (2) install a partitioned database system using a response file. Both methods are discussed in this section.

Single partition installation

If you want to install DB2 on a single machine, single partition, use DB2 Setup in the following way:

1. Logon as root.
2. Mount the CD-ROM by entering the following command:

```
mount /mnt/cdrom or mont /cdrom
```

If your Linux distribution disables execute privileges on CD-ROM devices by default, issue the following command as root to mount with execute permission:

```
mount -o exec /mnt/cdrom
```

If your CD-ROM was not automatically mounted, enter:

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

Where, /mnt/cdrom represents the mount point of the CD-ROM.

3. As root, run **db2setup**, and select **Install Products** on the first window.
4. Select **DB2 UDB Enterprise Server Edition** and click **Next**.
5. Click **Next** again. On License Agreement window, read the license agreement and then select **Accept**.
6. Select the installation type (Figure 2-1). You may choose a Typical, Compact, or Custom installation.

- *Typical* installs most DB2 components except for Application development and Business Intelligence 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 Application Development tools with DB2 UDB. With the Typical option, the Application Development tools requires a separate installation.

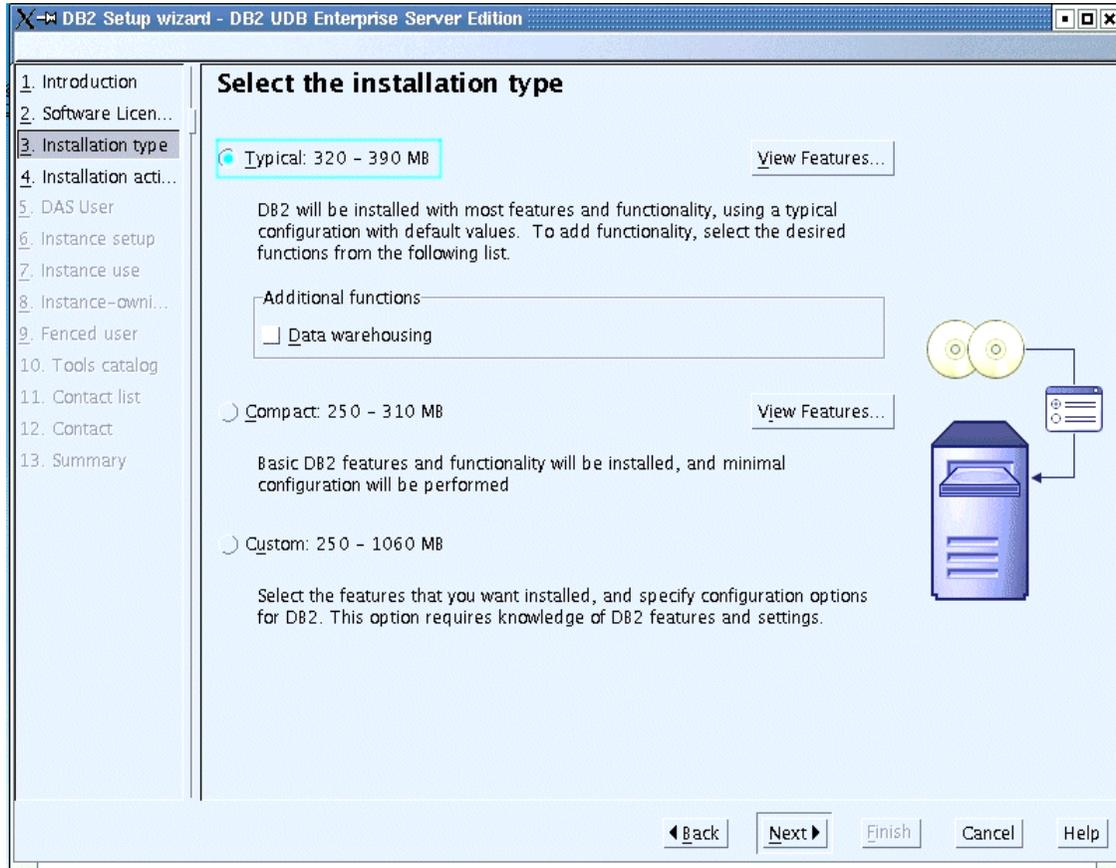


Figure 2-1 Select installation type

7. On the Select the Installation Action window, you may choose to save installation settings in a response file. If you want to do this, check off the Save your settings in a response file checkbox (Figure 2-2).

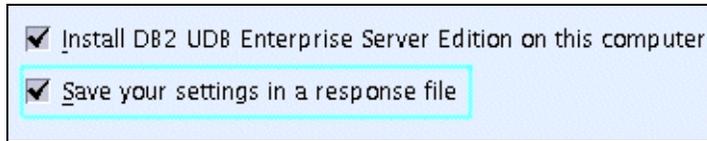


Figure 2-2 Response file option

- Set user information for the Database Administration Server (Figure 2-3). This user administers the Database Administration Server.

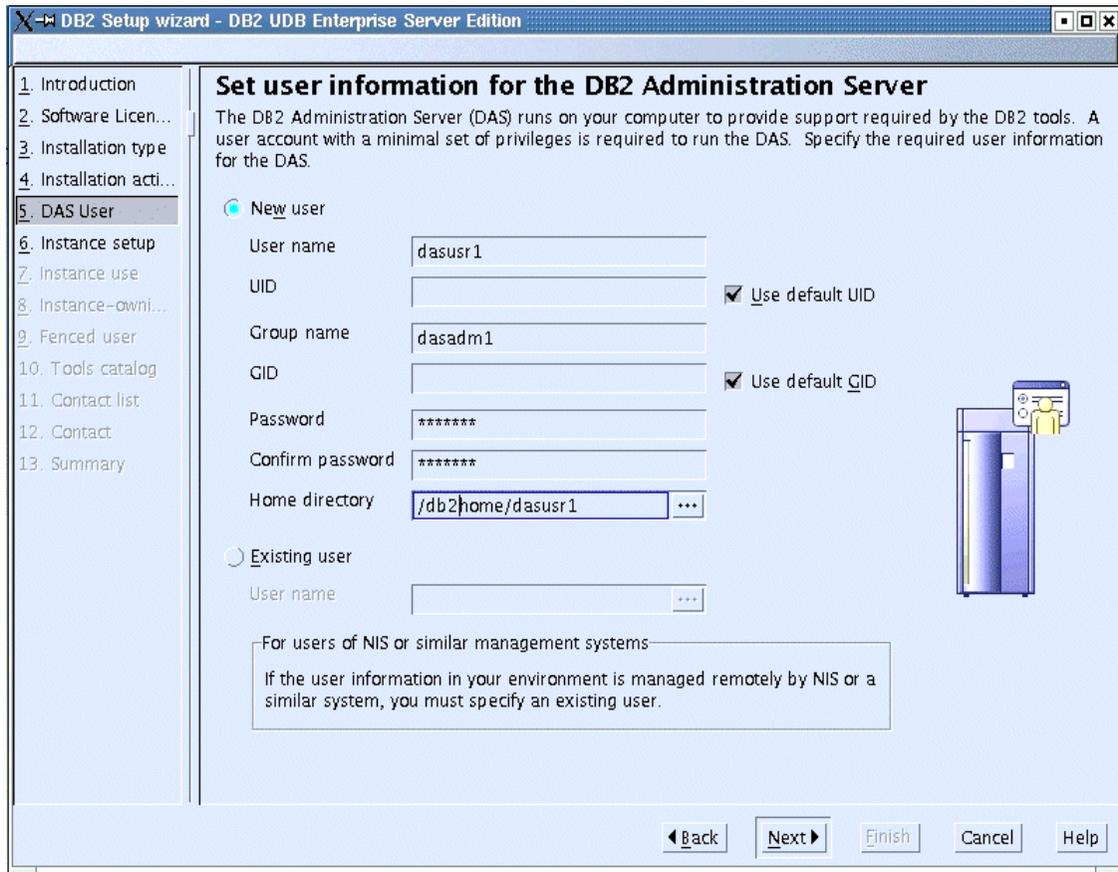


Figure 2-3 Set user information for the DB2 Administration Server

- Set up a DB2 instance. If you want to create a new instance, check off the "Create a DB2 instance" option (Figure 2-4). Otherwise you will need to create a new instance using `db2icrt` after installation.

Create a DB2 instance

Do not create a DB2 instance

Figure 2-4 Create a DB2 instance

10. Select how the instance will be used. The Single-partition instance (Figure 2-5) option means that the instance will only reside on your local machine and will not be used in a partitioned environment.

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.

Partitioned instance

The instance will be used in a partitioned database environment and will reside on multiple database partition servers. This computer will host the instance-owning database partition server. Select this option to use the partitioning capability of DB2.

Figure 2-5 Select how instance will be used

11. Set user information for the DB2 instance owner (Figure 2-6). By default, DB2 Setup creates a new user db2inst1 in group db2grp1. You may change user and group names, or configure an existing user to the new DB2 V8 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.

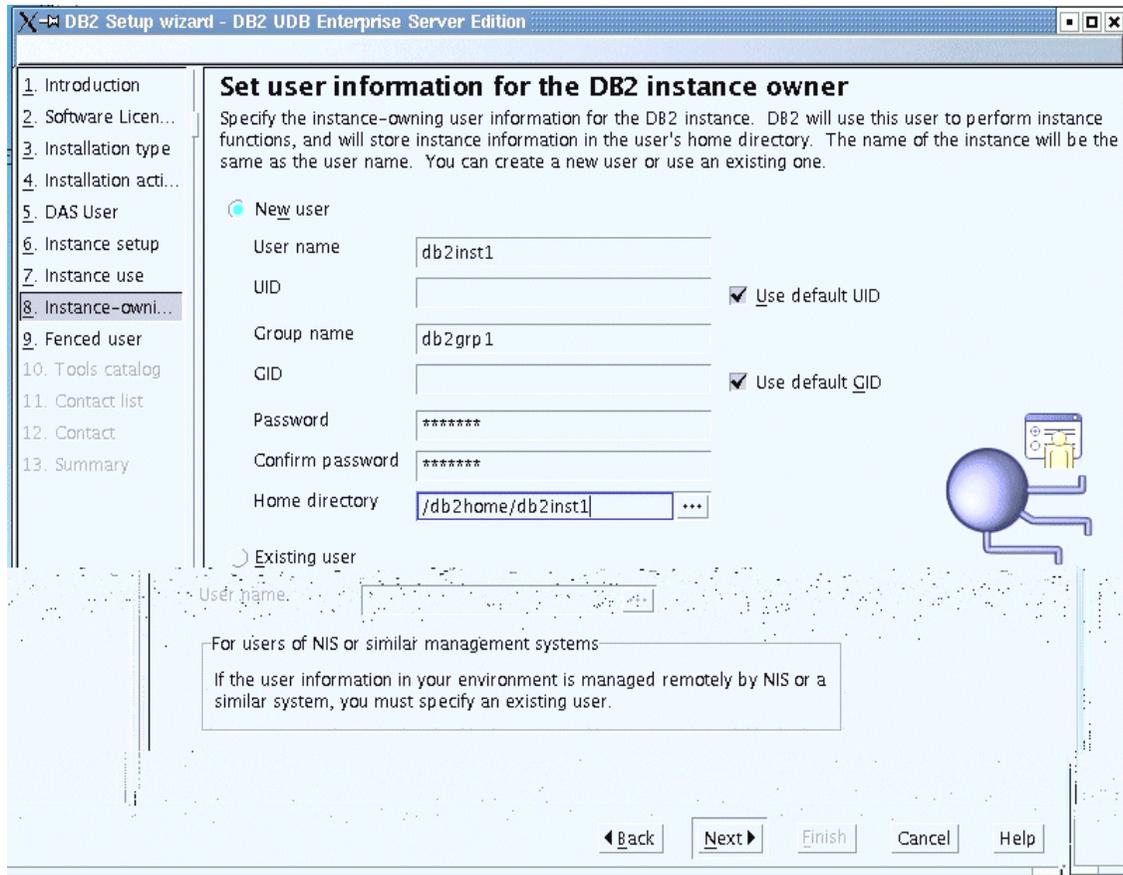


Figure 2-6 Set user information for the DB2 instance owner

12. 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, please note the home directory location.
13. Prepare the DB2 tools catalog (Figure 2-7). Select this option if you plan to use the Task Center or scheduler. This creates a database on your local machine that stores task metadata. The scheduler will not work without this repository. You may also create a DB2 tools catalog after installation using the CREATE TOOLS CATALOG command, but it's much easier to let DB2 Setup create it for you now.

Prepare the DB2 tools catalog

Before you can use certain DB2 tools such as the Task Center and scheduler, you must create the DB2 tools catalog. The DB2 tools catalog contains task metadata. The DB2 Setup wizard can prepare a local database to store this metadata. Specify whether to prepare the DB2 tools catalog.

- Use a local database
- Do not prepare the DB2 tools catalog on this computer

Figure 2-7 Prepare the DB2 tools catalog

- Specify a local database to store the tools catalog (Figure 2-8). Here you specify which instance, database, and schema in which to store the tools catalog. By default, a newly created tools catalog will be placed in the instance owner's home directory.

DB2 Setup wizard - DB2 UDB Enterprise Server Edition

- Introduction
- Software Licen...
- Installation type
- Installation acti...
- DAS User
- Instance setup
- Instance use
- Instance-owni...
- Fenced user
- Tools catalog
- Tools catalog...**
- Contact list
- Contact
- Summary

Specify a local database to store the DB2 tools catalog

The DB2 tools catalog will be stored in a local database. Specify the required information for the database. If the database does not exist, it will be created for you.

Instance: db2inst1

Database:
 New: TOOLSDB
 Existing: /

Schema:
 New: SYSTOOLS
 Existing: /

Back Next Finish Cancel Help

Figure 2-8 Specify local database for DB2 tools catalog

15. Set up the administration contact list (Figure 2-9). The administration contact list stores administrator contact information, and is used for notifying administrators when a database requires attention. You may create a new contact list that is stored locally, or use an existing global contact list that resides on a remote DB2 server. If you check off the Enable notification box, your system will search for an available SMTP server and set it to be used for e-mail notifications.

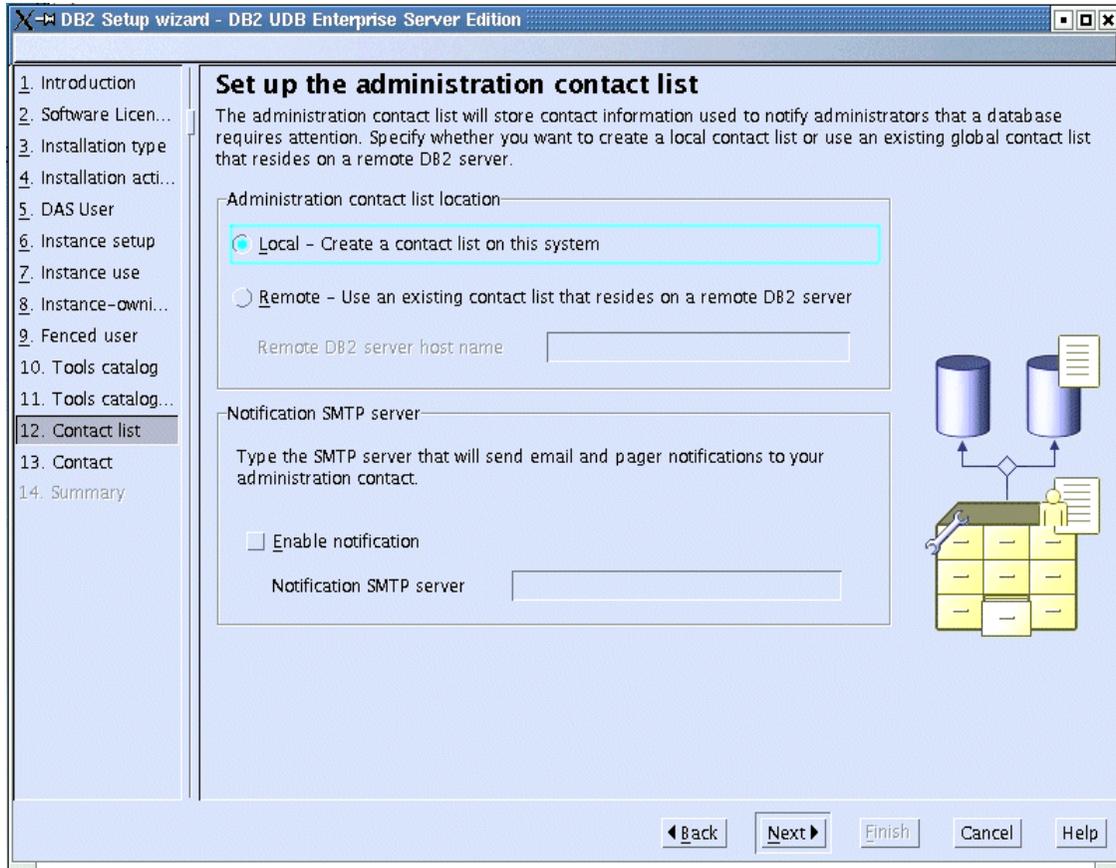


Figure 2-9 Set up the administration contact list

16. Specify a contact for health monitor notification (Figure 2-10). 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 breached. If you check off Address is for a pager, the notification message will be sent to the contact's pager. Refer to page 71 for instructions on how to set up the contact list and SMTP server as a post-install step.

Specify a contact for health monitor notification

By default, a health monitor runs on the DB2 instance you are setting up. When a health indicator threshold is breached, email or pager notification will be sent to an administration contact. Specify a new or existing contact for health monitor notification. If your contact list resides on a remote DB2 server, you must provide a user name and password for logon.

Administration contact for this instance

New contact

Name

Email address

Address is for a pager

Defer this task until after installation is complete

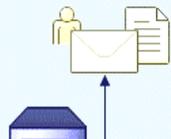


Figure 2-10 Specify contact for Health Monitor notification

Note: If you choose to *Defer task until after installation is complete*, you may specify contacts after installation using the Task Center or Control Center. In the Control Center, click the Contacts icon (which is located in the taskbar roughly under the Tools menu). Refer to page 71 for instructions on how to set up contacts as a post-install step.

17. Start copying files and create response files. This window provides a summary of your installation and configuration settings (Figure 2-11). Scroll through this window to verify that your settings are correct, then click **Finish**.

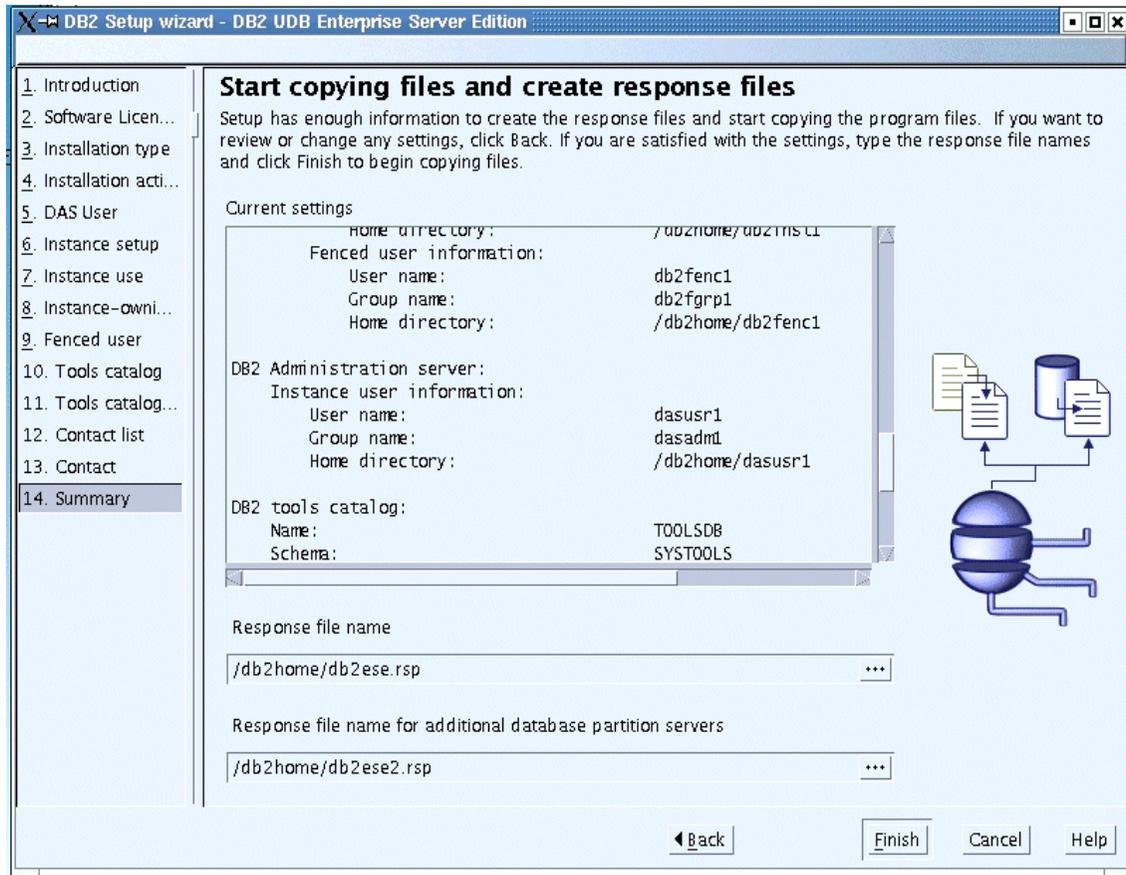


Figure 2-11 Installation summary window

18. After installation, read the Status report tab and check for any errors. If all is well, your next step is to install DB2 documentation. Refer to “Installing DB2 documentation” on page 72.

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

Response file install

Response files that are generated by the DB2 Setup wizard are used to install additional components or products after an initial install. They contain all installation and configuration settings that were specified during the initial install, which allows you to install a consistent configuration across multiple machines. It also saves you time by bypassing the DB2 Setup wizard.

By default, the response file that gets created is called *db2ese.rsp*. However, you have the option to change the name and destination directory while using the DB2 Setup wizard. Refer to Figure 2-11 on page 60.

Creating a response file using the DB2 Setup wizard

Near the beginning of the DB2 UDB ESE Setup wizard, you will see the following window (Figure 2-12).

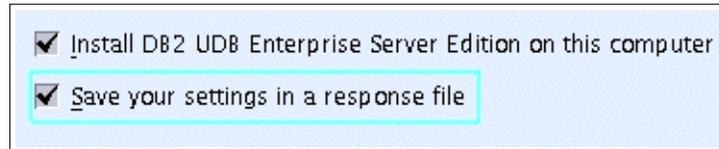


Figure 2-12 Save settings in response file

Simply check off the checkbox beside **Save settings in a response file**. This will save all installation settings to a file called *db2ese.rsp*. We suggest you change the destination directory to an exported file system (for example, /software) available to everyone on the network. The target directory of the response file is specified on the Start copying files window of the DB2 Setup wizard, shown in Figure 2-11 on page 60.

To install DB2 using the response file, refer to the instructions in “Step 2 - Use response file to install DB2 on remaining machines” on page 64. If you want to create a response file by editing a sample response file, refer to “Creating a response file using the sample response file” on page 66.

2.3.2 Multiple partition installation

There are several ways in which you can install DB2 on a multiple-partition environment: (1) 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, or (2) edit a sample response file and use this file to install DB2 on each machine in the cluster, or (3) 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 first.

Multiple partition install using DB2 Setup

With this method, the first step is to use the DB2 Setup wizard on the instance-owning machine to install DB2 and create a new instance. This instance

will be shared across the remaining machines participating in the partitioned database system. The second step is to use a response file that was generated by the DB2 Setup wizard to install DB2 on the remaining machines in the cluster.

Step 1: Use DB2 Setup wizard on instance-owning machine

1. On instance-owning machine, logon as root.
2. Go to the directory where the DB2 install image is located and run **db2setup**. Select **Install Products** on first window.
3. On second window, select **DB2 UDB Enterprise Server Edition** and click **Next**.
4. Click **Next** again. On License Agreement window, read the license agreement and then select **Accept**.
5. Select the installation type. You may choose a Typical, Compact, or Custom installation.
 - *Typical* installs most DB2 components except for Application development and Business Intelligence 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 Application Development Client with DB2 UDB. With the Typical option, the Application Development Client requires a separate installation.
6. On Select the Installation Action window, check off the **Save your settings in a response file** checkbox (Figure 2-13). This will create two response files: *db2ese.rsp* and *db2ese2.rsp*. You will need to use *db2ese2.rsp* to install DB2 on remaining machines participating in the cluster.

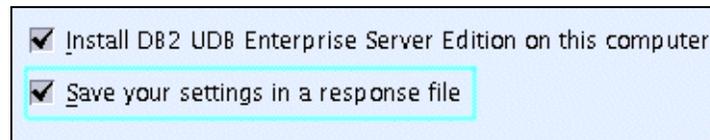
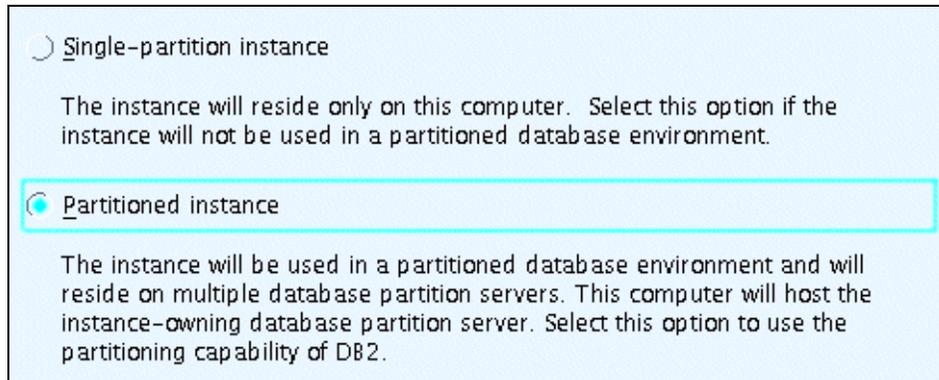


Figure 2-13 Save settings in a response file

7. Set user information for the Database Administration Server. This user administers the Database Administration Server. By default, this is the only user information that is saved in the response file *db2ese2.rsp*. Refer to “DAS user considerations for a partitioned database” on page 51 when deciding on your DAS user, and where to store the DAS user home directory.
8. Set up a DB2 instance. If you want to create a new instance, check off the **Create a DB2 instance** option (for this example, we assume that you

selected this option). Otherwise you will need to create a new instance using *db2icrt* after installation.

9. Select how the instance will be used. Check off **Partitioned instance** (Figure 2-14). This means that the instance will be used in a partitioned database. The machine that you are currently using will host the instance-owning database partition server.



The screenshot shows a dialog box with two radio button options. The first option is 'Single-partition instance' with an unselected radio button. The second option is 'Partitioned instance' with a selected radio button. The 'Partitioned instance' option is highlighted with a red rectangular border. Below each option is a descriptive paragraph.

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.

Partitioned instance

The instance will be used in a partitioned database environment and will reside on multiple database partition servers. This computer will host the instance-owning database partition server. Select this option to use the partitioning capability of DB2.

Figure 2-14 Select how instance will be used

10. Set user information for the DB2 instance owner. By default, DB2 Setup creates a new user *db2inst1* in group *db2grp1*. You may change user and group names, or configure an existing user to the new DB2 V8 instance by selecting the Existing user option. Note that the default home directory is in */home*. You must change the instance home directory to a DB2-specific directory, such as */db2home*, that will be NFS-exported to remaining machines participating in the partitioned database.
11. Set user information for the fenced user. This user is responsible for executing fenced user defined functions, such as UDFs and stored procedures. You should also place this home directory in a directory that is shared across the partitioned servers, such as */db2home*.
12. Prepare the DB2 tools catalog. You require a DB2 tools catalog if you plan to use the Task Center or scheduler. It is a database that stores task metadata. You may also create a DB2 tools catalog after installation using the CREATE TOOLS CATALOG command, but it's much easier to let DB2 Setup create it for you now.
13. Specify a local database to store the tools catalog. Here you specify which instance, database, and schema in which to store the tools catalog.
14. Set up the administration contact list. The administration contact list stores administrator contact information, and is used for notifying administrators when a database requires attention. You may create a new contact list that is

stored locally, or use an existing global contact list that resides on a remote DB2 server. If you check off the `Enable notification` box, your system will search for an available SMTP server that will be used for e-mail notifications.

Note: If there is no SMTP server set up, you may set it up after installation by using the DB2 CLP (Command Line Processor). Refer to page 71 for setup instructions.

15. Specify a contact for health monitor notification. 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 e-mail address when a health indicator threshold is breached. If you check off `Address is for a pager`, the notification message will be sent to the contact's pager.

Note: If you choose to *Defer task until after installation is complete*, you may specify contacts after installation using the Task Center or Control Center. In the Control Center, click the `Contacts` icon which is located in the taskbar under the `Tools` menu. Refer to page 71 for setup instructions.

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**.

Step 2 - Use response file to install DB2 on remaining machines

After performing a partitioned installation, the DB2 Setup wizard creates two response files: `db2ese.rsp` and `db2ese2.rsp`. The `db2ese.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 `db2ese2.rsp` response file only installs the appropriate DB2 filesets, specifies DAS properties, and specifies languages to install. Only use the `db2ese2.rsp` response file when installing database partition servers on participating computers.

The secondary response file, `db2ese2.rsp`, looks something like this:

```
*-----
* Generated response file used by the DB2 Setup wizard
* generation time: 10/21/02 10:43 AM
*-----
* Product Installation
LIC_AGREEMENT      = ACCEPT
PROD               = ENTERPRISE_SERVER_EDITION
INSTALL_TYPE       = TYPICAL
*-----
* Das properties
*-----
```

```

DAS_CONTACT_LIST      = REMOTE
DAS_CONTACT_LIST_HOSTNAME = udblnx01
DAS_SMTP_SERVER       = null
* DAS user
DAS_USERNAME          = dasusr1
DAS_GROUP_NAME        = dasadm1
DAS_HOME_DIRECTORY   = /home/dasusr1
DAS_PASSWORD          = 235262333285355231346
ENCRYPTED              = DAS_PASSWORD
*-----
* Installed Languages
*-----
LANG                  = EN

```

Tip: It is much faster to perform a response file install from a file system network drive rather than a CD-ROM drive. If you are planning on installing multiple clients, you should set up a mounted file system on a code server to improve performance.

To perform a response file install, locate the `db2ese2.rsp` file and do the following:

1. As root, enter the `db2setup` command using this format:

```
<cd-rom>/db2setup -r <responsefile_directory>/<response_file>
```

Where:

- `<cd-rom>` represents the location of the DB2 install image;
- `<responsefile_directory>` represents the directory where the response file is located; and
- `<response_file>` represents the name of the response file (for example, `db2ese2.rsp`).

In our setup, the DB2 install image is located in `/software` and the response file `db2ese2.rsp` is located in `/db2home`. Therefore, we executed the following command in `/software`:

```
[root@udb1nx01]/# ./db2setup -r /db2home/db2ese2.rsp
```

2. After installation has finished, check the installation log to ensure that no errors have occurred. The main installation log file is `/tmp/db2setup.log`.

Installing DB2 using a sample response file

Another way to install DB2 on multiple physical nodes is to edit a sample response file and use this file to install DB2 on each machine in the cluster. Sample response files are also useful for single partition installations if you want

to bypass the graphical wizard while taking advantage of DB2 Setup's configuration capabilities.

Creating a response file using the sample response file

The DB2 CD-ROM contains a ready-to-use sample response file with default entries. This file, called `db2ese.rsp`, is located in:

```
<cd-rom>/db2/linux/samples
```

Where, `<cd-rom>` represents the location of the DB2 install image. This directory also contains sample response files for the Application Development Client and Administration Client, named `db2adcl.rsp` and `db2admcl.rsp`, respectively.

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 "Administration" Guides (listed in "Other publications" on page 301) for guidance on configuring these parameters.
- ▶ Save the file on an exported file system (for example, `/software`) available to everyone on the network.

Installing DB2 using the response file

- ▶ Perform the installation by issuing the `db2setup -r` command as root. For example, if the DB2 install image is located in `/software`, and the `db2ese.rsp` response file is found in `/db2home`, enter the following command as root:

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

Your custom response file is what you make of it. To give you an idea of the final product, the following response file was generated by the DB2 Setup wizard:

```
*-----  
* Generated response file used by the DB2 Setup wizard  
* generation time: 10/18/02 11:51 AM  
*-----  
* Product Installation LIC_AGREEMENT = ACCEPT  
PROD = ENTERPRISE_SERVER_EDITION  
INSTALL_TYPE = TYPICAL  
*-----  
*  
Das properties  
*-----  
DAS_CONTACT_LIST = LOCAL  
* DAS user  
DAS_USERNAME = dasusr1  
DAS_GROUP_NAME = dasadm1
```

```

DAS_HOME_DIRECTORY = /home/dasusr1
DAS_PASSWORD = 235262333285355231346
ENCRYPTED = DAS_PASSWORD
TOOLS_CATALOG_DATABASE = toolsDB
TOOLS_CATALOG_SCHEMA = SYSTOOLS
* toolsDB
properties DATABASE = toolsDB
toolsDB.INSTANCE = inst1
toolsDB.DATABASE_NAME = TOOLSDB
toolsDB.LOCATION = Local
* -----
* Instance properties
* -----
INSTANCE = inst1
inst1.TYPE = ese
inst1.WORDWIDTH = 32
* Instance-owning user
inst1.NAME = db2inst1
inst1.GROUP_NAME = db2grp1
inst1.HOME_DIRECTORY = /db2home/db2inst1
inst1.PASSWORD = 235262333285355231346
ENCRYPTED = inst1.PASSWORD
inst1.AUTOSTART = YES
inst1.AUTHENTICATION = SERVER
inst1.SVCENAME = db2c_db2inst1
inst1.PORT_NUMBER = 50001
inst1.FCM_PORT_NUMBER = 60000
inst1.MAX_LOGICAL_NODES = 4
* Fenced user
inst1.FENCED_USERNAME = db2fenc1
inst1.FENCED_GROUP_NAME = db2fgrp1
inst1.FENCED_HOME_DIRECTORY = /db2home/db2fenc1
inst1.FENCED_PASSWORD = 235262333285355231346
ENCRYPTED = inst1.FENCED_PASSWORD
* Contact properties
CONTACT = contact1
contact1.CONTACT_NAME = db2inst1
contact1.EMAIL = db2inst1@udb1nx01
contact1.PAGER = false
contact1.INSTANCE = inst1
* -----
* Installed Languages
* -----
LANG = EN

```

2.3.3 db2_install utility

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. Login as a user with root authority.
2. Insert and mount the DB2 CD.
3. Enter the **db2_install** command to start the db2_install script. This script is found in the root directory of the CD-ROM.
4. When db2_install prompts you for the product keyword, enter in DB2.ESE.

The installation directory for DB2 on Linux is /opt/IBM/db2/V8.1.

Post-installation tasks

After installing DB2 using db2_install, do the following:

1. Create group and user IDs (if not already done).
2. Create a DB2 instance.
3. Create links for DB2 file (optional).
4. Configure TCP/IP communications for the DB2 instance.
5. Update product license key.
6. Create a DB2 Administration Server (DAS).
7. Create DB2 tools catalog (optional).
8. Set up SMTP server and notification contact list (optional).

Procedure

1. Create group and user IDs; refer to 2.2.4, “User and group setup” on page 48.
2. Create a DB2 instance. Use the **db2icrt** command to create a new instance:
 - a. Log in as root.
 - b. Enter the following command:

```
/opt/IBM/db2/V8.1/instance/db2icrt [-a AuthType] -u FencedID InstNme
```

Where:

- *AuthType* represents the authentication type for the instance. If you do not specify this parameter, the default authentication type, SERVER, is assigned. AuthType can be one of SERVER, CLIENT, DCS, SERVER_ENCRYPT, or DCS_ENCRYPT.
- *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 entered in:

```
./db2icrt -u db2fenc1 db2inst1
```

3. 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.

Links should not be created on systems where multiple versions of DB2 co-exist. If you create links on one version of DB2, the links on the other version will be overwritten.

To create links:

- Log in as root.
- Enter the following command:

```
/opt/IBM/db2/V8.1/cfg/db2ln
```

Note: If you have existing links to `/usr/lib` and `/usr/include` from previous versions of DB2, and want to keep the links to the libraries of the previous version, you must execute the `db2rm1n` command from DB2 V8 before you execute the `db2ln` command from the previous version of DB2. The `db2ln` command removes existing links and replaces them with links to DB2 V8 libraries.

4. 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:

- 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.

- b. Update the database manager configuration file on the server. To do so:
 - i. Log into system as instance owner.
 - ii. Set up the instance environment by running the db2profile script:

```
. INSTHOME/sqllib/db2profile
```
 - iii. Update the SVCENAME parameter in the DBM configuration file. You may specify either the service name or port number. For example, we entered:

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

You can check your SVCENAME by entering:

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

- c. 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
```

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

```
db2stop  
db2start
```

5. Install license key.

You must install a license key on each computer where a DB2 server is installed. 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.

- a. Log in as root user or instance owner.
- b. Enter the following command:

```
/opt/IBM/db2/V8.1/adm/db2licm -a <filename>
```

Where, *filename* is the full pathname and filename for the DB2 ESE license file *db2ese.lic*. This license file is located in /db2/license in the root directory of your CD-ROM. For example, if the CD-ROM is mounted in the directory /cdrom, enter the following command:

```
/opt/IBM/db2/V8.1/adm/db2licm -a /cdrom/db2/license/db2ese.lic
```

On Linux, the DB2 license keys are located in /var/lum in a file called *nodelock*.

6. 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:

- a. Log in as a user with root authority.
- b. Issue the following command to create the DAS:

```
/opt/IBM/db2/V8.1/instance/dascrt -u DASuser
```

For the **-u** parameter, enter the DAS user that you created for this machine, for example, `dasusr1`.

7. Create DB2 tools catalog (optional).

You must create a DB2 tools catalog if you plan to use the Task Center or scheduler. This creates a database on your local machine that stores task metadata. The scheduler will not work without this repository. You may either create a new database or use an existing database.

- a. Log in as instance owner.
- b. Use the `CREATE TOOLS CATALOG` command to create a new tools catalog database, or configure the tools catalog to an existing database.

For example, to create a new tools catalog called `toolscat` in a new database called `toolsdb`, we issued:

```
db2 create tools catalog toolscat create new database toolsdb
```

Note: Refer to the *IBM DB2 UDB Command Reference V8*, SC09-4828 for additional `CREATE TOOLS CATALOG` command options.

8. Set up SMTP server and administration contact list (optional).

By default, a health monitor runs on the DB2 instance you just created. If you set up an SMTP server and administration contact list, the DB2 health monitor will send a notification e-mail to the people on the contact list when a health indicator threshold is breached and a database requires attention. You may create a new contact list that is stored locally, or use an existing global contact list that resides on a remote DB2 server. Another option is to create a contact group. When a notice is sent to the group, each member of the group receives this notice. You can also specify to have the notification sent to a contact's pager. First, you must set up the SMTP server. To do so:

- a. Login as instance owner and issue the following command:

```
db2 update admin cfg using smtp_server <host_name>
```

Where, `host_name` is the TCP/IP hostname of the machine that contains the SMTP server used for e-mail notifications.

- b. Next, decide whether you want the contact list to be stored locally or on a remote server. By default, the contact list is stored locally. If you want to set up a remote contact list (which is highly recommended for a partitioned environment, for it shares a common contact list across multiple DB2 Administration Servers), set the DAS configuration parameter *contact_host* to the TCP/IP hostname of the remote DAS where the contact list is to be stored. For example, to store our contact list on host `udblnx05`, we entered:

```
db2 update admin cfg using contact_host udblrx05
```

- c. Now add contacts to the administration contact list:
 - i. Open up the Control Center and click the Contacts icon. This icon is located in the taskbar roughly under the Tools menu and looks like this:

 - ii. Select the name of your system from the System name pull-down menu.
 - iii. Click the **Add Contact** button. Type in the name and e-mail address of the administration contact. If you want the notification sent to the contact's pager, check off *Address is for a pager*.
 - iv. Click the **Test** button to verify that the e-mail notification functions correctly. Press **OK** to save settings and return to Contacts window.
 - v. If you want to create a Contact Group, click the **Add Group** button, and enter the group's name in the Name field. Next, select names from the Available contacts list and press the arrow key to add these contacts to the group.

You can also add contacts using the Task Center.

- d. Finally, turn the scheduler on using:

```
db2 update admin cfg using sched_enable on
```

Then re-start the DB2 Administration Server for changes to take effect.

Installing DB2 documentation

DB2 Version 8 provides HTML and PDF documentation on separate CDs. These CDs are shipped with your client and server CDs.

- ▶ *HTML documentation CD*. This CD contains DB2 online documentation, including Help screens and the Information Center. It is installed separately from other DB2 products from its own CD-ROM, which means that you can install DB2 HTML documentation immediately after installing DB2, or at a later time. You can also install the documentation and 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.

- ▶ *PDF documentation CD*. Unlike the HTML documentation, 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.

Installing HTML documentation using the DB2 Setup wizard

1. Insert the CD-ROM labeled *DB2 HTML Documentation*.
2. Log onto system as user with root authority.
3. Go to the directory where the documentation CD is mounted by entering:

```
cd /cdrom
```

Where, /cdrom represents the mount point of the CD-ROM.
4. Enter **./db2setup**. This launches the DB2 Setup wizard.
5. Once the launchpad opens, click **Install Products** to install the DB2 documentation onto your machine. You may also want to use this launchpad to view the installation prerequisites, view the Release Notes, or take a Quick Tour to view the features of DB2 UDB Version 8.
6. Select the product you would like to install. Select **DB2 HTML Documentation** and click **Next**.
7. Welcome to the DB2 Setup Wizard. Click **Next**.
8. Select the HTML documentation components to install (Figure 2-15). By default, db2setup installs only the core documentation. Select any additional documentation you would like to install, then click **Next**.

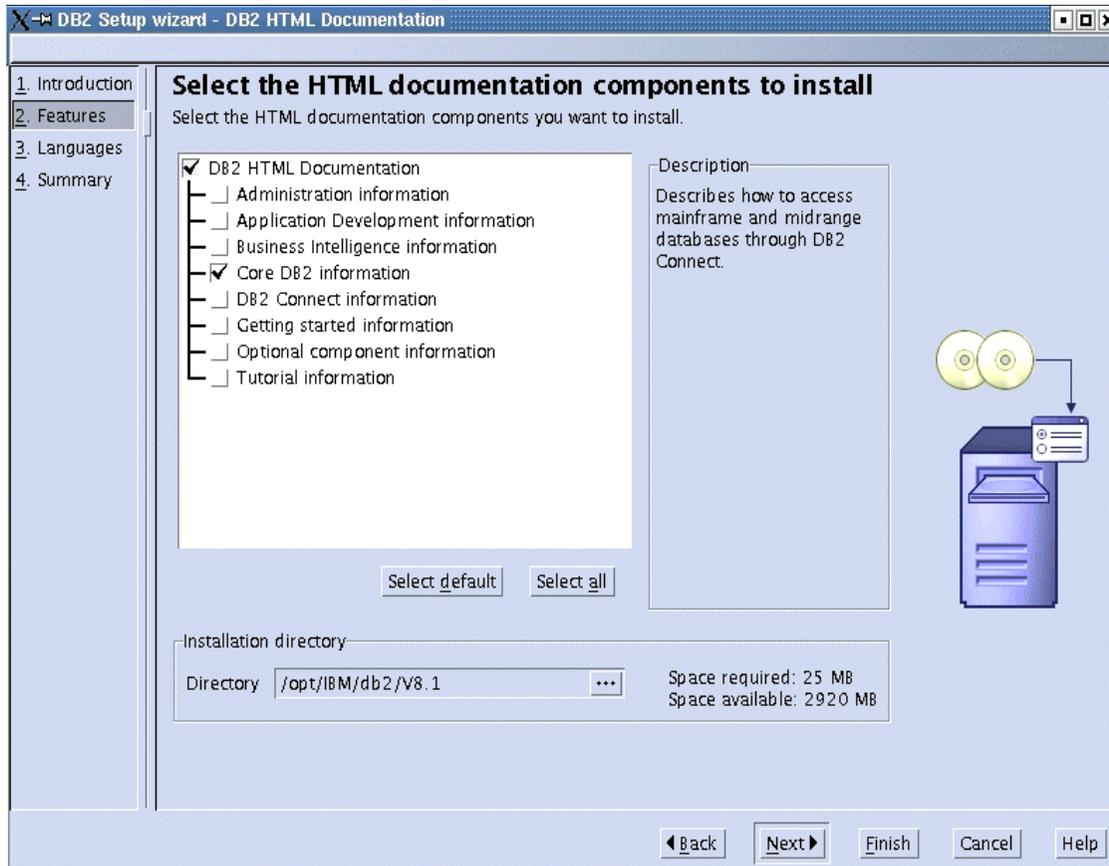


Figure 2-15 Select HTML documentation components to install

9. Languages (Figure 2-16). By default, DB2 Setup only installs English documentation. You may install additional languages by selecting it in the left menu and clicking the arrow (>) key.

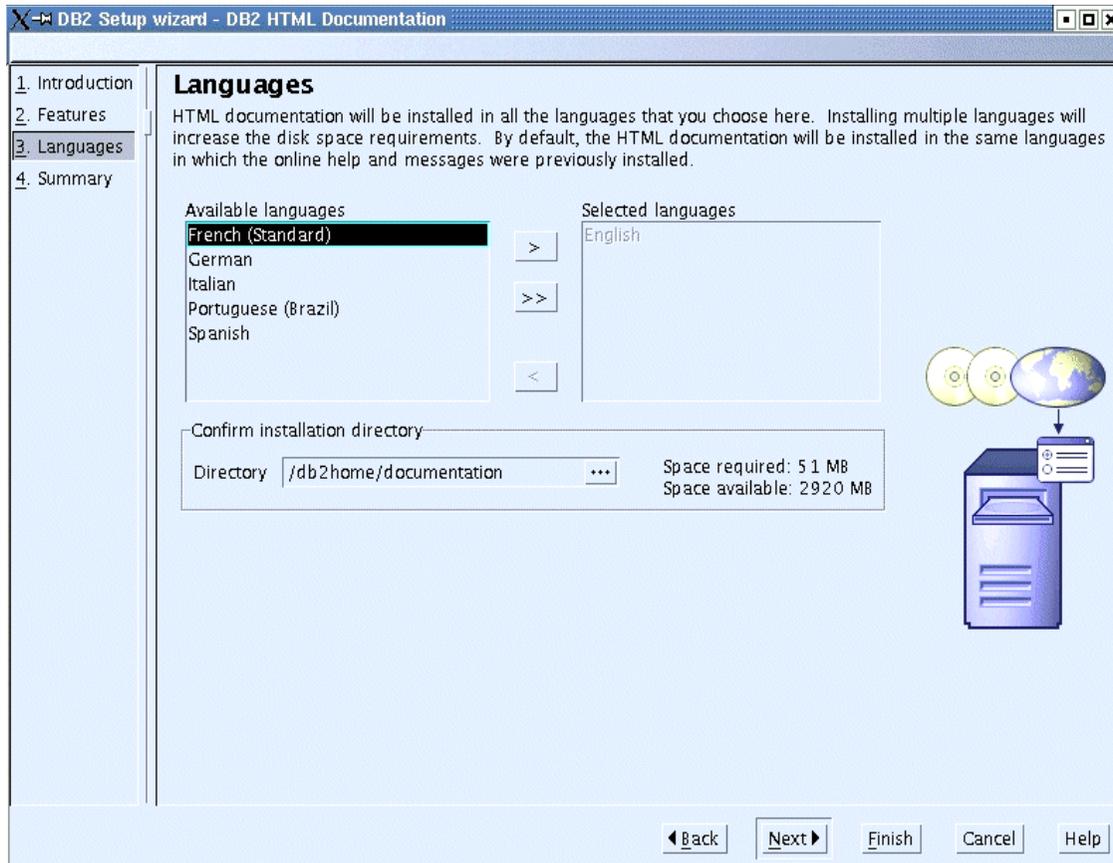


Figure 2-16 Select languages

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

Installing DB2 HTML documentation using doc_install

To install DB2 HTML documentation manually, use the `doc_install` script.

1. Login as a user with root authority.
2. Decide where you want to want to install the documentation. In our partitioned environment, we created a directory called `/documentation` which is located in the shared `/db2home` directory (for example, `/db2home/documentation`).
3. Insert and mount the DB2 HTML Documentation CD-ROM.
4. Enter the `./doc_install` command using the following format:

```
doc_install -l <language> -t <topic> [-p <path>] [-d]
```

To install multiple topics, enter in each topic keywords separated by a space.

The language code for English is *en_US*. Type in `./doc_install` or refer to the Readme file on your CD to view the language codes for other languages.

Valid topics are:

core	Core DB2 information
admin	Administration information
ad	Application development information
wareh	Business Intelligence information
conn	DB2 Connect information
start	Getting started information
tutr	Tutorial information
opt	Optional component information

For example, to install core, administration, and getting started documentation in English into the `/db2home/documentation` directory, enter the following:

```
./doc_install -l en_US -t core admin start -p /db2home/documentation
```

Note: There is no `db2_deinstall` program to uninstall the documentation. Simply use `rm -rf` to remove the documentation files.

2.3.4 Adding an additional partition

To add an additional partition on a separate server/machine at a later time, we suggest you use a response file to install DB2. You may use the response file you created during the initial installation or you can generate a new one by editing a sample response file. The sample response file, `db2ese.rsp`, is found in `/<installation source>/ese/db2/linux/samples`. This is the best method to use, as it can configure DBM parameters in addition to installing the server. Refer to “Installing DB2 using a sample response file” on page 65 for more for information about editing the sample response file. The step-by-step procedure of adding database partition is provided in 3.4, “Add database partition” on page 100.

2.3.5 Installing on NIS

You may use DB2 Setup or `db2_install` to install DB2 on systems using Network Information System (NIS) or NIS+. However, there are some factors to consider when using DB2 Setup to install DB2 on environments that use NIS (or other external security programs that do not allow the DB2 installation program to modify user characteristics). This is because some of the DB2 Setup installation scripts will attempt to update the users and groups that are controlled by NIS, and will not be able to do so. This is not meant to discourage you from using DB2 Setup on NIS systems — you will still benefit from all of its features except for

user and group creation feature. If DB2 Setup detects NIS on your machine, it will not give you the option to create new users during installation. Instead it only will allow you to choose existing users.

Simply abide by the following installation prerequisites in order to use DB2 Setup successfully on NIS systems:

- ▶ Users and groups must first be created on the NIS server before running the DB2 Setup wizard or performing a response file install.
- ▶ Secondary groups must be created for the DB2 instance owner and the DB2 Administration Server on the NIS server. You must then add the primary group of the instance owner to the secondary DB2 Administration Server group. Likewise, you must add the primary DB2 Administration Server group to the secondary group for the instance owner.
- ▶ With ESE, you must add an entry for an instance in the `/etc/services` file *before* creating the instance.



Post installation tasks

In the previous chapter, we provided a detailed process of installing DB2 Version 8 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
Control Center is JAVA-based graphical tool that is used to help you administer your DB2 instances and databases. We explain how to get this tool launched for the first time.
- ▶ Preparation for database creation
This section discusses Linux-specific settings for single and multiple database partitions.
- ▶ Creating databases
This section provides instructions for creating a database using the database wizard and command line processor.
- ▶ DB2 configuration
Here we discuss some important configuration parameters for the instance and database.
- ▶ Client access setup
This section discusses installation and configuration of DB2 clients to access DB2 ESE.

3.1 Control Center setup

DB2 UDB 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 23. 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 either type in the **db2cc &** command or use the **Start Applications** → **IBM DB2** menu to start the Control Center. If you started your X-Window as root user, you need to switch to the DB2 instance owner ID and set up the display before you can start the Control Center. The sample commands are displayed below.

Example 3-1 Start Control Center

```
xhost +  
su - db2inst1  
export DISPLAY=:0.0  
db2cc &
```

The **&** sign after the command returns the shell prompt, which allows you to use this shell for other commands.

Note: If you get a JAVA error, verify that you have installed and are using the correct JDK Version, which is Java(TM) 2 Runtime Environment, Standard Edition 1.3.1, and verify that the path setting is correct. The JDK is described in 2.1.6, “Additional software requirements” on page 31.

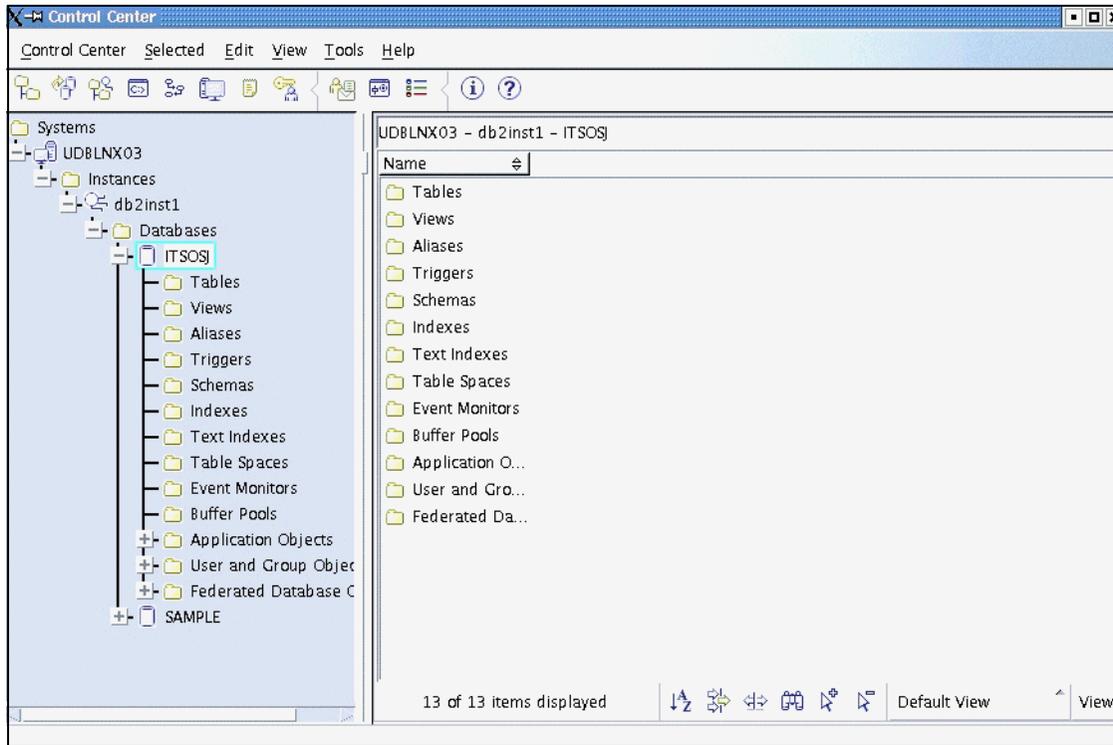


Figure 3-1 Control Center main window

Figure 3-1 shows the Control Center which is started on a Linux system. The Systems tree is expanded to display all databases within the *db2inst1* instance.

3.2 Preparation for database creation

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

- ▶ Linux specific configuration
- ▶ Multi-partition specific configuration

3.2.1 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 *sqllib* 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/sql1lib/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/sql1lib/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/sql1lib` for *cs*h users to set up the DB2 environment.

3.2.2 Multiple partitioned specific configuration

If you have decided to install DB2 in a multi-partitioned environment, then you must prepare the DB2 instance to run on more than just one node. In a partitioned database system, each database partition server must have the authority to perform remote commands on all of the other database partition servers participating in an instance. To enable this, create an *.rhosts* file as instance owner in the `$INSTHOME` directory and change permissions to 600. To create the *.rhosts* file, enter the following command:

```
vi /db2home/db2inst1/.rhosts
```

The format of *.rhosts* is as follows:

```
hostname instance_owner or user_name
```

Where, *hostname* is either the hostname or IP address of the physical machine. In Example 3-2, the user *db2inst1* has permission to perform remote commands on the hosts *udb1nx02* and *udb1nx05*.

Example 3-2 .rhost file

```
udb1nx02 db2inst1  
udb1nx05 db2inst1
```

To change the *.rhosts* file permissions to 600, enter the following command:

```
chmod 600 /db2home/db2inst1/.rhosts
```

Each user who will execute remote commands must have his or her own `.rhosts` file.

To define which hosts are participating in a partitioned environment, you have to modify the `db2nodes.cfg` file located in `$INSTHOME/sqllib`. In our example (Example 3-3), we have two machines which each hold two database partitions.

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 partition on the same host.

Example 3-3 db2nodes.cfg file

```
0 udb1nx02 0
1 udb1nx02 1
2 udb1nx05 0
3 udb1nx05 1
```

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`. If you install DB2 with the `db2setup` wizard, a port range is automatically reserved on the primary (instance-owning) computer.

Logon to each participating computer as a root user and add the identical entries into `/etc/services`. Example 3-4 shows four ports reserved for four local partitions on this host.

Example 3-4 Ports in /etc/services

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

You can check your settings using the `db2_a11` command. In Example 3-5, the `db2_all` command is used to query the date on two physical nodes.

Example 3-5 db2_all sample command

```
db2inst1@udb1nx02:~/sqllib> db2_a11 date

Mon Oct 21 17:06:28 PDT 2002
udb1nx02: date completed ok
```

```
Mon Oct 21 17:06:28 PDT 2002
udblnx02: date completed ok
```

```
Mon Oct 21 17:16:35 PDT 2002
udblnx05: date completed ok
```

```
Mon Oct 21 17:16:35 PDT 2002
udblnx05: date completed ok
```

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 3-6.

Example 3-6 Start of a DB2 instance with four partition

```
db2inst1@udblnx02:~> db2start
11-01-2002 15:10:51    1  0  SQL1063N  DB2START processing was successful.
11-01-2002 15:10:51    0  0  SQL1063N  DB2START processing was successful.
11-01-2002 15:10:52    3  0  SQL1063N  DB2START processing was successful.
11-01-2002 15:10:53    2  0  SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.
```

Note: **db2_a11** is a shell script provided by DB2 that allows you to send commands to one or all database partitions.

3.3 Creating a 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 tablespaces: SYSCATSPACE, USERSPACE1 and TEMPSPACE1. The tablespace 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 tables for holding user created tables. A system temporary tablespace (TEMPSPACE1) is required by DB2 to support activities like sorting data, join tables create indexes.

- ▶ 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.

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

Before you create a multi-partitioned database, you should consider which partition will be the catalog node. In a partitioned database environment, the **create database** command affects all database partitions that are listed in the `db2nodes.cfg` file. The database partition node 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 partition on this host by default. To run the commands or SQL statements on the other partitions, you need to specify the node by setting the `DB2NODE` variable.

For example, host `udblnx02` has partition numbers 0 and 1. If you issue the **export DB2NODE=1** command, and it proceeds or is followed by the **db2 terminate** command, every statement will now run against partition 1.

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 variable `dftdbpath`. You can change this to your desired default path by entering following command:

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

3.3.1 Create Database wizard

If you have DB2 installed in a single partition mode, 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 52. The Create Database wizard can also be used to create a

multi-partitioned database. You can start the Create Database wizard from the Control Center, as shown in Figure 3-2.

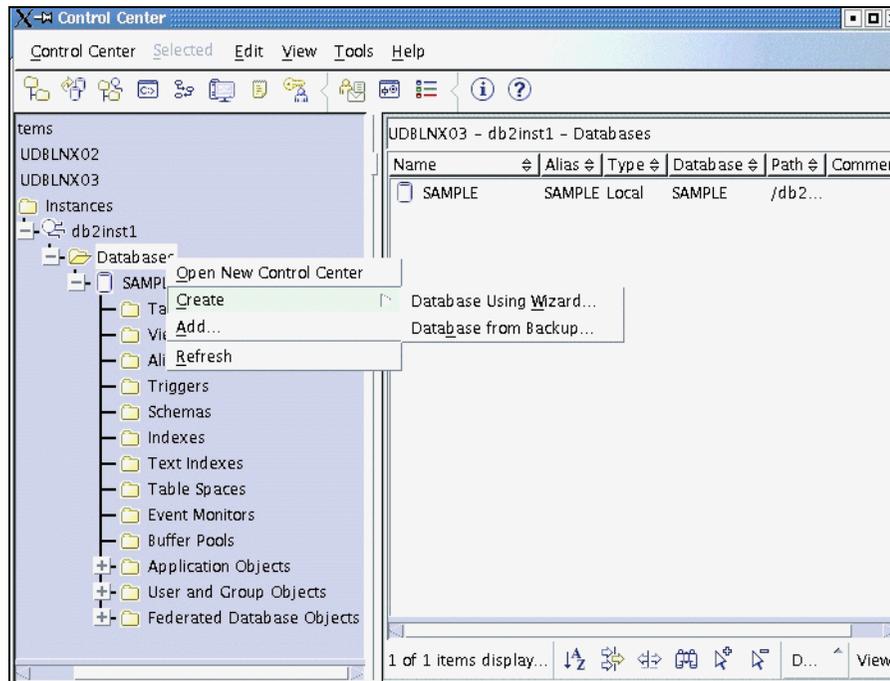


Figure 3-2 Create database wizard in Control Center

- ▶ Expand the object tree until you find the Databases folder.
- ▶ Right-click the Databases folder, and select **Create** → **Database Using Wizard** from the pop-up menu. See Figure 3-2.

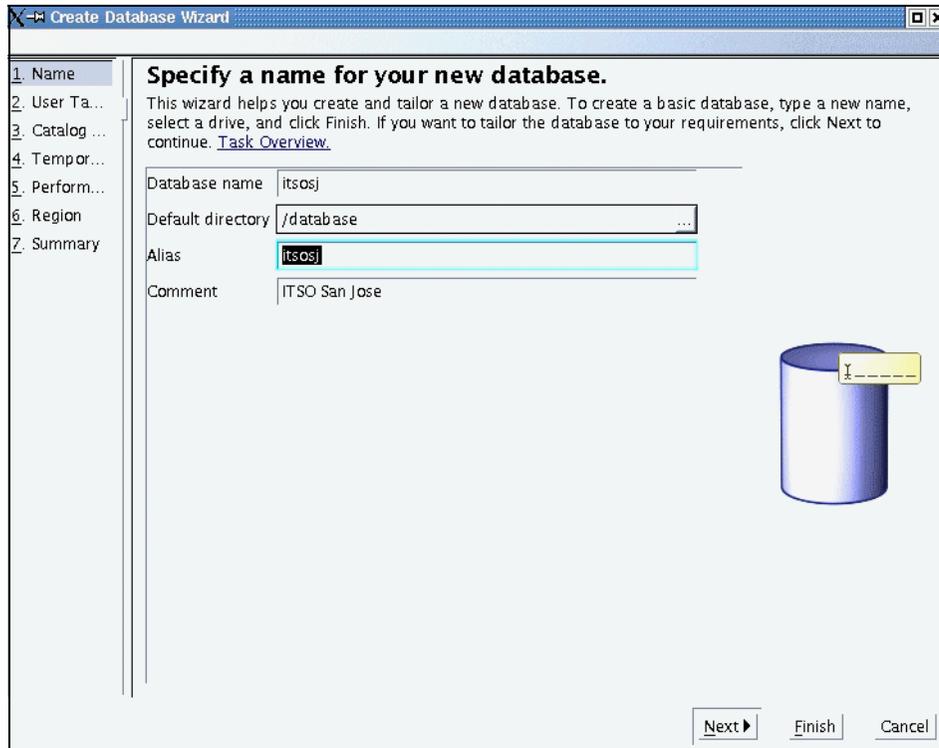


Figure 3-3 Start window of create database wizard

Figure 3-3 shows you the Create Database Wizard window where we have to name the database and database directory. All files related to the database will be stored under this directory path, for example, for the first database in this instance:

```
/database/$DB2INSTANCE/NODE0000/SQL00001/
```

After a database is created, issue the following command to see in which directory the database is stored:

```
db2 list database directory on /database
```

Example 3-7 Output of list database directory

```
db2inst1@udb1nx02:~> db2 list db directory on /database
```

```
Local Database Directory on /database
```

```
Number of entries in the directory = 1
```

```
Database 1 entry:
```

Database alias = ITSOSJ
 Database name = ITSOSJ
 Database directory = SQL00001
 Database release level = a.00
 Comment = ITSO San Jose
 Directory entry type = Home
 Catalog database partition number = 0
 Database partition number = 0

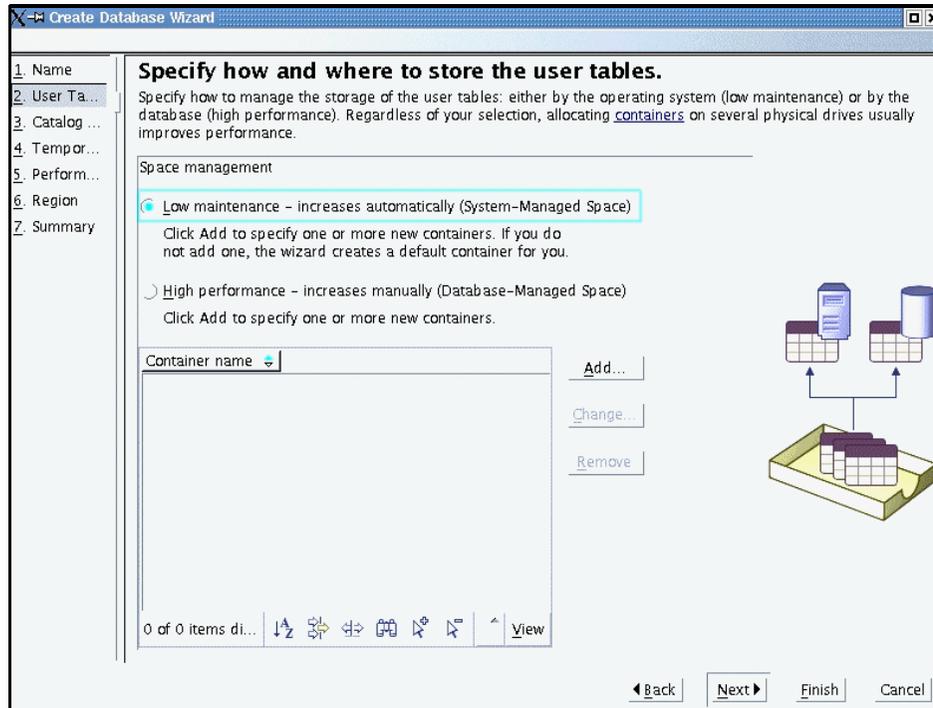


Figure 3-4 Tablespace container definition

In this window (Figure 3-4) we have to decide where DB2 should store the default tablespace *userpace1* and what type it should be. SMS (System Managed Storage) means that the tablespace container is a directory in the given path and all data will be stored in them as files. DMS (Data Managed Storage) means that the tablespace container is a pre-allocated file which is controlled by DB2.

In the next two windows we have to decide where the catalog tablespace *syscatspace* and temporary tablespace *tempspace1* will be located. We recommend that you allocate *tempspace1* in a different file system. For more details, refer to the description in 2.2.2, “Storage planning” on page 36.

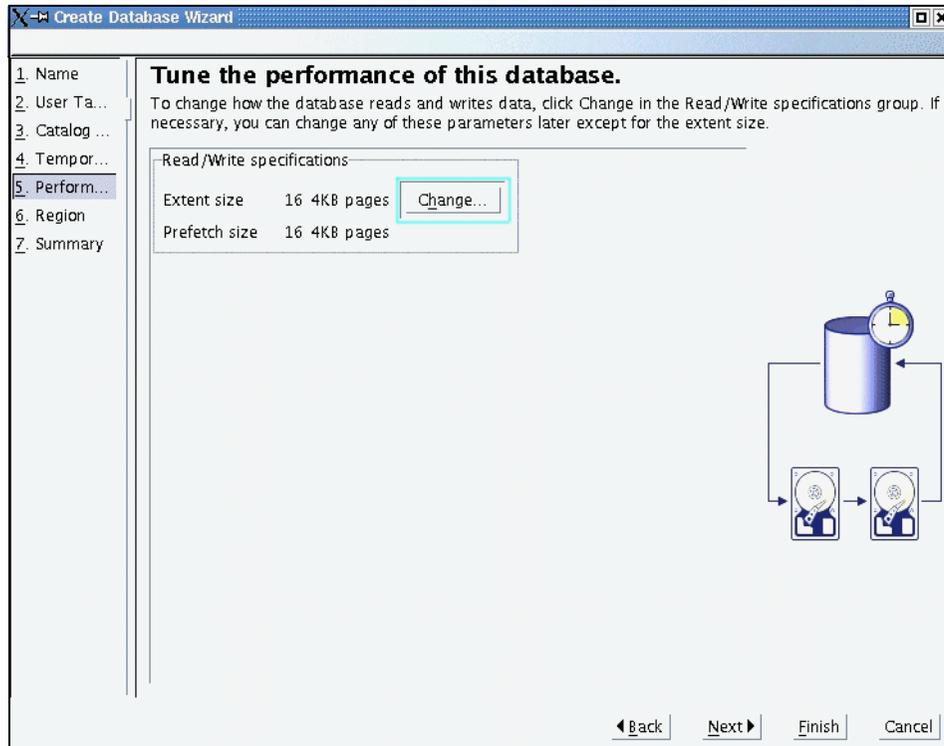


Figure 3-5 Tablespace I/O definition

Tune the performance of this database (Figure 3-5) is an important window where you have to decide how DB2 should read and write data. We recommend that you choose for *Prefetch size* a multiple of *Extend size* to allow DB2 to read data from disk before it is needed (prefetching). Every time DB2 needs data from your table it starts an I/O request. Such a request asks for an extent from disk. If you allow DB2 to get more data at once, then you will reduce I/O wait time.

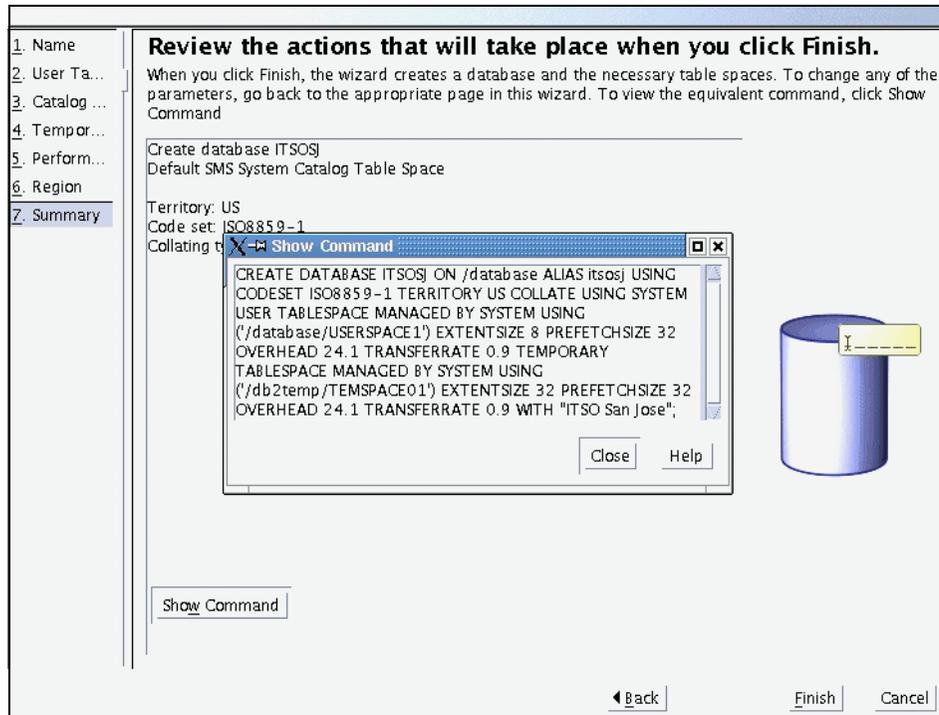


Figure 3-6 Verify generated SQL

After finishing all preview windows, if you select Show Command, the wizard presents a window (Figure 3-6) with the created database command and all the settings you specified. By closing this window and clicking **Finish**, DB2 creates the database for you. A successful database creation confirmation message with the configuration option will be presented, as shown in Figure 3-7.

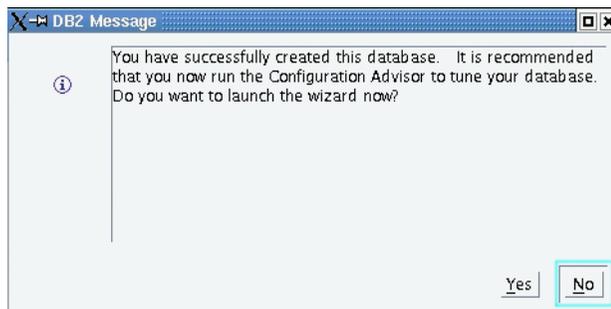


Figure 3-7 Confirmation of create database

If you want to configure the database you just created, click **Yes** and the Configuration Advisory will start. You also can configure the database later through the Control Center under **Tools** → **Wizards**. This will be discussed later more in detail.

Change log path

By default DB2 sets the log path to:

```
$DBPATH/$INSTANCE/NODE0000/SQL0001/SQLLOGDIR
```

To improve performance and avoid I/O contention, you should change this log path to a different file system on different disks. To change the log path, open the Control Center and drill down to your database. Right-click and select **Configure Database Logging** (Figure 3-8).

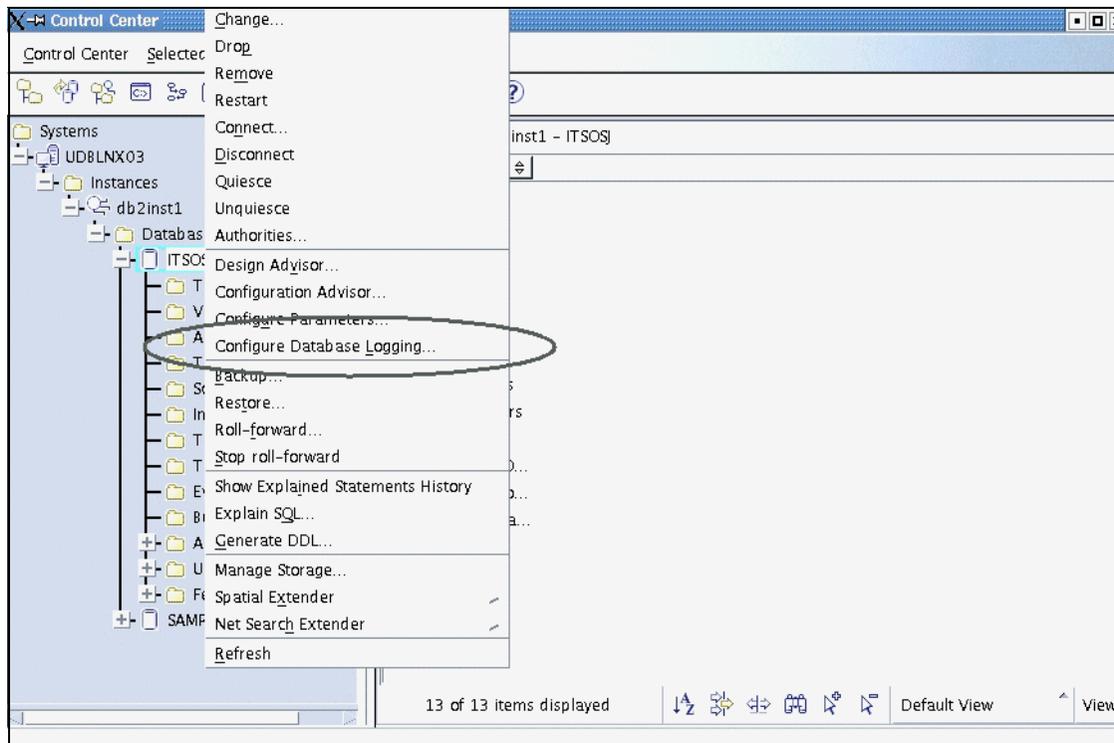


Figure 3-8 Configuring DB2 Logging

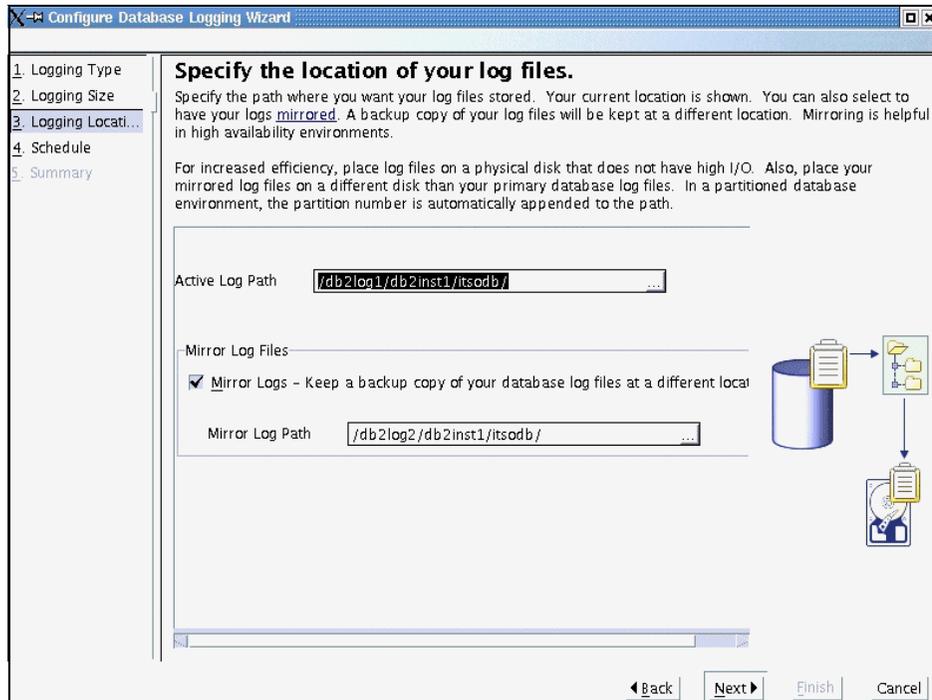


Figure 3-9 Specify new log path

In Figure 3-9 we change the log path to a new location and turn on the new mirror logging feature at the same time. We choose different file systems for the primary and secondary log paths.

The new path will be in effect after all connections have been released and a new connection or activate database command is issued. A faster way to change and activate the new path is to issue following commands:

```
db2 connect to itsosj
db2 update db cfg for itsodb using NEWLOGPATH /db2log1/db2inst1/itsosj
```

```
db2 update db cfg for itsodb using MIRRORLOGPATH
/db2log2/db2inst1/itsosj
```

```
db2 deactivate database itsosj
db2 activate database itsosj
```

To finalize the database configuration, open Control Center again and configure your database as you desire.

3.3.2 Command line

For a multi-partitioned environment, we recommend that you use the DB2 command line, instead of the Create Database wizard to create a database. Although this requires some experience and manual work, it gives you more control of where and how DB2 files are allocated. On the following pages, we provide an example of how to setup a partitioned database over two physical hosts with two logical partitions on each host.

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. To do so, enter the following command:

```
echo $DB2INSTANCE
```

Where, \$DB2INSTANCE represents the current DB2 environment.

Example 3-8 Verify DB2 environment

```
db2inst1@udb1nx02:~/sql1lib> echo $DB2INSTANCE
db2inst1
db2inst1@udb1nx02:~/sql1lib>
```

Generate directory structure

In our example, we do not want to use the default file structure provided by DB2. Instead, we want to use the customized file structure that we created in “File system configuration” on page 38. In this setup, our database, table and index spaces and log files are stored on different file systems. To create the directory structure for log files, temp space and tablespaces, open a shell window as instance owner and issue the **db2_a11** command. See Example 3-9, Example 3-10, and Example 3-11.

Example 3-9 Creation of directory structure for log files

```
db2_a11 'mkdir /db2log/db2inst1'
db2_a11 'mkdir /db2log/db2inst1/itsodb'
or combine both commands in one as:
db2_a11 'mkdir -p /db2log/db2inst1/itsodb'
```

Example 3-10 Creation of directory structure for temp space

```
db2_a11 'mkdir /db2temp/db2inst1'
db2_a11 'mkdir /db2temp/db2inst1/itsodb/'
db2_a11 'typeset -Z4 DB2NODE; mkdir /db2temp/db2inst1/itsodb/NODE$DB2NODE'
or combine all three commands in one as:
db2_a11 'typeset -Z4 DB2NODE; mkdir -p /db2temp/db2inst1/itsodb/NODE$DB2NODE'
```

Example 3-11 Creation of directory structure for tablespaces

```
db2_all 'mkdir /tablespaces/db2inst1'
db2_all 'mkdir /tablespaces/db2inst1/itsodb'
db2_all 'typeset -Z4 DB2NODE; mkdir /tablespaces/db2inst1/itsodb/NODE$DB2NODE'
or combine all three commands in one as:
db2_all 'typeset -Z4 DB2NODE; mkdir -p
/tablespaces/db2inst1/itsodb/NODE$DB2NODE'
```

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

We have decided to use partition number 0 to hold the DB2 catalog. After issuing the command `export DB2NODE=0` and `db2 terminate`, DB2 will run subsequent commands against partition 0.

To create database `itsodb` in database path `/database` use a text editor like `VI` and write the following example:

Example 3-12 Creation of ITSODB database

```
create database itsodb on /database
using codeset ISO8859-1
territory US
with "ITS0 San Jose";

connect to itsodb;

create bufferpool BPTMP all nodes size 5000 PAGESIZE 4k ;

connect reset;
```

Save this file as `any_name` and execute it with:

```
db2 -tvf any_name
```

After executing this file you will have a database called ITSODB stored in `/database/db2inst1/NODE0000`, where the catalog tablespace and all relevant database files are located. In a partitioned environment, the same directory structure will be created on every node except for the catalog tablespace. This tablespace exists only on the partition, where the `create database` command is issued.

At the same time we have created a new buffer pool for our dedicated temp space. Because DB2 allocates many resources at activation time of a database,

we have to stop and start our database to enable the new buffer pool. To do so, enter following commands:

```
db2 deactivate database itsodb
db2 activate database itsodb
```

Reallocate temp and user space1

Now we want to create a new temporary tablespace and drop the default one. This has to be done in this order, because DB2 requires at least one temp space.

Use a text editor like *VI* and write the following example:

Example 3-13 Creation of new temp space and drop default

```
connect to itsodb;

create system temporary tablespace TEMPSPACE01 in IBMTEMPGROUP pagesize 4k
managed by SYSTEM
using ('/db2temp/db2inst1/itsodb/NODE000 $N /temp space01') on dbpartitionnum
(0)
using ('/db2temp/db2inst1/itsodb/NODE000 $N /temp space01') on dbpartitionnum
(1)
using ('/db2temp/db2inst1/itsodb/NODE000 $N /temp space01') on dbpartitionnum
(2)
using ('/db2temp/db2inst1/itsodb/NODE000 $N /temp space01') on dbpartitionnum
(3)
extentsize 8 prefetchsize 32
bufferpool bptemp ;

drop tablespace temp space1;
```

Save this file as *any_name* and execute it with:

```
db2 -tvf any_name
```

To move the default tablespace userspace1 to the new location under */tablespace*, use a text editor like *VI* and write following example:

Example 3-14 Change the location of tablespace userspace1

```
drop tablespace userspace1;

create tablespace userspace1 in IBMDEFAULTGROUP pagesize 4k
MANAGED BY SYSTEM
using ('/tablespaces/db2inst1/itsodb/NODE000 $N /userspace1') on dbpartitionnum
(0)
using ('/tablespaces/db2inst1/itsodb/NODE000 $N /userspace1') on dbpartitionnum
(1)
```

```
using ('/tablespaces/db2inst1/itsodb/NODE000 $N /userspace1')on dbpartitionnum
(2)
using ('/tablespaces/db2inst1/itsodb/NODE000 $N /userspace1')on dbpartitionnum
(3)
extentsize 8 prefetchsize 32;
```

Save this file as *any_name* and execute it with:

```
db2 -tvf any_name
```

The temp space in our example is created in:

```
/db2temp/db2inst1/itsodb/NODE000x/temp space01
```

Where, temp space01 is a directory. The same is true for userspace in:

```
/tablespaces/db2inst1/itsodb/NODE000x/userspace1
```

Tip: The variable \$N in the create tablespace statement in Example 3-14 will be replaced with the partition number during runtime.

Note: The keyword **autoconfigure** allows you to overwrite any default database settings at creation time. This is not supported in a multi partitioned database environment.

For more detailed information about **autoconfigure**, refer to the *IBM DB2 UDB Administration Guide: Implementation V8*, SC09-4820, and *IBM DB2 UDB Command Reference V8*, SC09-4828.

After successful database creation you will have following file structure (see Example 3-15 through Example 3-18):

Example 3-15 Directory structure on host udblnx02 partition 0

```
/database/db2inst1/NODE0000/sqlbdir
/database/db2inst1/NODE0000/SQL00001/db2event/db2detaildeadlock
/database/db2inst1/NODE0000/SQL00001/db2event
/database/db2inst1/NODE0000/SQL00001/SQLLOGDIR
/database/db2inst1/NODE0000/SQL00001/SQLT0000.0
/db2temp/db2inst1/itsodb/NODE0000/temp space01
/tablespaces/db2inst1/itsodb/NODE0000/userspace1
/db2log/db2inst1/itsodb/NODE0000/
```

Example 3-16 Directory structure host udblnx02 partition 1

```
/database/db2inst1/NODE0001/sqlbdir
```

```
/database/db2inst1/NODE0001/SQL00001/db2event  
/database/db2inst1/NODE0001/SQL00001/SQLLOGDIR  
/db2temp/db2inst1/itsodb/NODE0001/temp space01  
/tablespaces/db2inst1/itsodb/NODE0001/userspace1  
/db2log/db2inst1/itsodb/NODE0001/
```

Example 3-17 Directory structure host udblnx05 partition 2

```
/database/db2inst1/NODE0002/sqlbdir  
/database/db2inst1/NODE0002/SQL00001/db2event  
/database/db2inst1/NODE0002/SQL00001/SQLLOGDIR  
/db2temp/db2inst1/itsodb/NODE0002/temp space01  
/tablespaces/db2inst1/itsodb/NODE0002/userspace1  
/db2log/db2inst1/itsodb/NODE0002/
```

Example 3-18 Directory structure host udblnx05 partition 3

```
/database/db2inst1/NODE0003/sqlbdir  
/database/db2inst1/NODE0003/SQL00001/db2event  
/database/db2inst1/NODE0003/SQL00001/SQLLOGDIR  
/db2temp/db2inst1/itsodb/NODE0003/temp space01  
/tablespaces/db2inst1/itsodb/NODE0003/userspace1  
/db2log/db2inst1/itsodb/NODE0003/
```

Note: The directory /database/db2inst1/NODE0000/SQL00001/SQLT0000.0 which holds the catalog tablespace only exists for NODE0000.

Change log path

Our next step is to reallocate the DB2 log files to a separated file system. To show the current log path, type the command:

```
db2 get db cfg for itsodb | grep Path
```

This command affects only the partition against which you are attached. In our environment the initial log path is:

```
Path to log files = /database/db2inst1/NODE0000/SQL00001/SQLLOGDIR/
```

To change the log path on each partition issue the following command:

```
db2_a11 'db2 connect to itsodb;db2 update db cfg for itsodb using NEWLOGPATH  
/db2log/db2inst1/itsodb'
```

This will change the log path on each node to:

```
/db2log/db2inst1/itsodb/NODE000x
```

The subdirectory will be automatically added by DB2, but will not change dynamically. This change only takes effect after reactivating the database. Issue the following commands:

```
db2 deactivate db itsodb
db2 activate db itsodb
```

Then verify the change by using 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'
```

```
Path to log files = /db2log/db2inst1/itsodb/NODE0000'/
```

Create tablespaces and tables

The next step, of course, is to create tables for your application. When a database is created, DB2 creates three default tablespaces. The USERSPACE1 is used for place the data tables. Based on the database design, more tablespaces can be created. The tables usually will be placed to a particular tablespace based on the application and database design. It is common to create your own tablespaces spread across all partitions or only a part of them. Every tablespace has to be created in a database partition group. We show you in Example 3-19 the creation of a tablespace over all partitions defined in db2nodes.cfg, using the default database partition group IBMDEFAULTGROUP.

Example 3-19 Create multi partition tablespace

```
connect to itsodb;

create tablespace sample_mpl in IBMDEFAULTGROUP pagesize 4k
MANAGED BY SYSTEM
using ('/tablespaces/db2inst1/itsodb/NODE000 $N /sample_mpl')on dbpartitionnum
(0)
using ('/tablespaces/db2inst1/itsodb/NODE000 $N /sample_mpl')on dbpartitionnum
(1)
using ('/tablespaces/db2inst1/itsodb/NODE000 $N /sample_mpl')on dbpartitionnum
(2)
using ('/tablespaces/db2inst1/itsodb/NODE000 $N /sample_mpl')on dbpartitionnum
(3)
extentsize 8 prefetchsize 32;
```

Example 3-20 shows a creation of a database partition group sng_01 and a single partition tablespace on partition 1.

Example 3-20 Create single partition tablespace

```
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 ('/tablespaces/db2inst1/itsodb/NODE000 $N /sample_sng')on dbpartitionnum
(1)
extentsize 8 prefetchsize 32;

```

To create tables in the newly created tablespace, use the command line to execute DDL-scripts like Example 3-21 using the **db2 -tvf** command.

Example 3-21 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 SAMPL_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**.

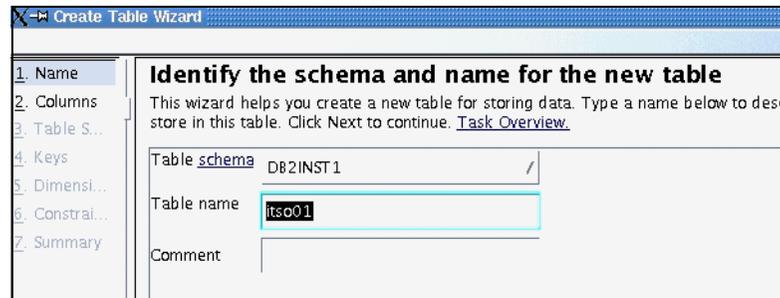


Figure 3-10 Create Table wizard in Control Center

The Create Table Wizard, Figure 3-10, guides you through many windows where you can create a new table using the predefined column definitions, or by defining your own column definitions.

Note: Detailed information about creation of tablespaces, tables or indexes can be found in Chapter 5 of the “*SQL References*” (listed in “Other publications” on page 301.

3.4 Add database partition

You can use the **add node** command to add a logical database partition to a database which is already populated with data. The add node command has two options:

- ▶ LIKE NODE option
- ▶ WITHOUT TABLESPACES option

The add node command creates the database directory structure in the database path (for example, /database/..) on the node where the new database partition has been created. Without using the 'like node'-parameter, temp table space definitions are retrieved from the catalog node.

If you add a partition on a different host, install DB2 base code either by using `db2_install` or `db2setup` and the response file utility. Verify that `rsh`, `rlogin`, `db2_all` and so on, work fine. See 3.2.2, “Multiple partitioned specific configuration” on page 82. Create on the new system the File Systems and directory structure to hold all database files. This is explained in “Generate directory structure” on page 93.

Check /etc/services that sufficient ports are reserved for the DB2 instance on the target host. On a node, you should reserve one port per logical partition.

Example 3-22 /etc/services with for ports reserved for db2inst1

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

Create a file system structure like the existing partition and set the right permissions.

Shutdown your DB2 instance (`db2stop`) so that you can modify `db2nodes.cfg` file. Add a new partition in `$INSTHOME/sqllib/db2nodes.cfg`.

Example 3-23 db2nodes.cfg file

```
0 udb1nx02 0
1 udb1nx02 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 following command, DB2 will update the db2nodes.cfg automatically after the next stop of the instance.

```
db2start dbpartitionnum 2 add dbpartitionnum hostname udb1nx02 port 2 without tablespaces
```

To create the directory structure used by DB2 we have to issue the add node command if it hasn't already been done by starting the database manager. Issue the following commands shown in Example 3-24.

Example 3-24 Add dbpartition utility command

```
db2inst1@udb1nx02:~/sql1lib> export DB2NODE=2  
db2inst1@udb1nx02:~/sql1lib> db2 terminate  
db2inst1@udb1nx02:~/sql1lib> db2 add dbpartitionnum without tablespaces
```

```
DB20000I The ADD NODE command completed successfully.  
udb1nx02: db2 add dbpartitionnum without tablespaces completed ok
```

DB2 has now created the directory structure in */database*, but with the keyword *without tablespaces*, the existing tablespace in partitions 0 and 1 has not been created in the new partition. The database partition nodegroup for IBMTEMPGROUP is expanded with the new database partition 2 and we have to add a tablespace container for TEMPSPACE01 on the new partition.

Now we have to start the remaining partition by entering **db2start** and establishing a connection with the command **db2 connect to itsodb**.

Example 3-25 Add tablespace container for temp space

```
db2inst1@udb1nx02:~> db2 "alter tablespace temp space01 add ('/db2temp/db2inst1/itsodb/NODE0002/temp space01') 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. Example 3-26 shows the statement for MPL_01 in our database ITSODB.

Example 3-26 Alter database partition group MPL_01

```
db2inst1@udblnx02:~> db2 'alter database partition group mpl_01 add
dbpartitionnum(2) without tablespaces'
SQL1759W Redistribute nodegroup is required to change data partitioning for
objects in nodegroup "MPL_01" to include some added nodes or exclude some
dropped nodes.  SQLSTATE=01618
db2inst1@udblnx02:~>
```

The next step will be to add for each tablespace within this database partition group the appropriate tablespace container on the new partition. You cannot omit individual tablespaces within a database partition group.

In our example we add for each tablespace the appropriate container (Example 3-27).

Example 3-27 Add tablespace container on partition 2

```
db2inst1@udblnx02:~> db2 "alter TABLESPACE CUSTOMER_TSP add
('/tablespaces/db2inst1/itsodb/NODE0002/customer_tsp') on dbpartitionnum (2)"
DB20000I The SQL command completed successfully.
```

```
db2inst1@udblnx02:~> db2 "alter TABLESPACE NATION_TSP add
('/tablespaces/db2inst1/itsodb/NODE0002/nation_tsp') on dbpartitionnum (2)"
DB20000I The SQL command completed successfully.
```

```
db2inst1@udblnx02:~> db2 "alter TABLESPACE ORDERS_TSP add
('/tablespaces/db2inst1/itsodb/NODE0002/orders_tsp') on dbpartitionnum (2)"
DB20000I The SQL command completed successfully.
```

```
SQL1759W Redistribute nodegroup is required to change data partitioning for
objects in nodegroup "MPL_01" to include some added nodes or exclude some
dropped nodes.  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.

The **redistribute** command affects all tablespaces 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 MPL_01, log on to the host where the catalog partition resides, connect to the database, and issue the **redistribute** command.

Example 3-28 Redistribute command for MPL_01

```
db2inst1@udblnx02:~> db2 list db directory on /database
```

Local Database Directory on /database

Number of entries in the directory = 1

Database 1 entry:

```
Database alias           = ITSODB
Database name           = ITSODB
Database directory      = SQL00001
Database release level  = a.00
Comment                 = ITS0 San Jose
Directory entry type    = Home
Catalog database partition number = 0
Database partition number = 0
```

```
db2inst1@udblnx02:~> db2 redistribute database partition group mpl_01 uniform
DB20000I The REDISTRIBUTE NODEGROUP command completed successfully.
db2inst1@udblnx02:~>
```

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 3-29.

Example 3-29 Redistribute messages file

Data Redistribution Utility

```
The following options have been specified:
Nodegroup name           : MPL_01
Data Redistribution option : U
Redistribute Nodegroup   : uniformly
No. of nodes to be added : 1
List of nodes to be added :
  2
No. of nodes to be dropped : 0
List of nodes 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
15.29.38		DB2INST1.CUSTOMER
	15.32.13	
15.32.16		DB2INST1.NATION
	15.32.16	

15.32.16 DB2INST1.ORDERS

16.42.45

--All tables in the nodegroup 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 to a table which was redistributed are set as invalid. They will be automatically 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 the *IBM DB2 UDB Command Reference V8*, SC09-4828.

An alternate method to add a logical database partition is to use the Add database partition wizard. See Figure 3-11.

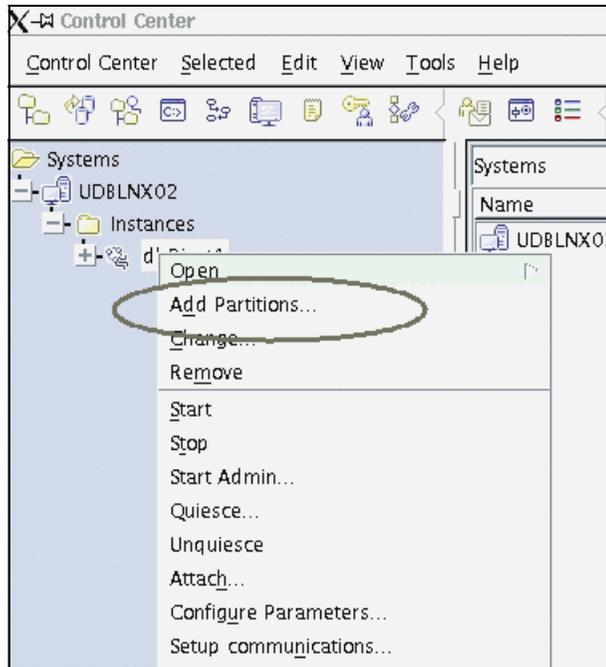


Figure 3-11 Add database partition in Control Center

Go through all windows and fill out the necessary information. In Figure 3-12 you can see the generated commands to add a database partition.



Figure 3-12 Add database partition generated command

After executing these commands, all tables within the affected tablespaces have to be redistributed. This can be done in the next window that appears. 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. Figure 3-13 shows you our example.

Partition Number	Maximum Log Size (MB)	Available Log Size (MB)	Data To Be Moved (MB)	Log Space R
0	273	273	19	19
1	273	273	19	19
2	273	273	38	38

Figure 3-13 Estimates about log space needs to redistribute MPL01

Go through all the windows and execute the generated command, as shown in Figure 3-14 or store it in the Task Center to schedule the script later.

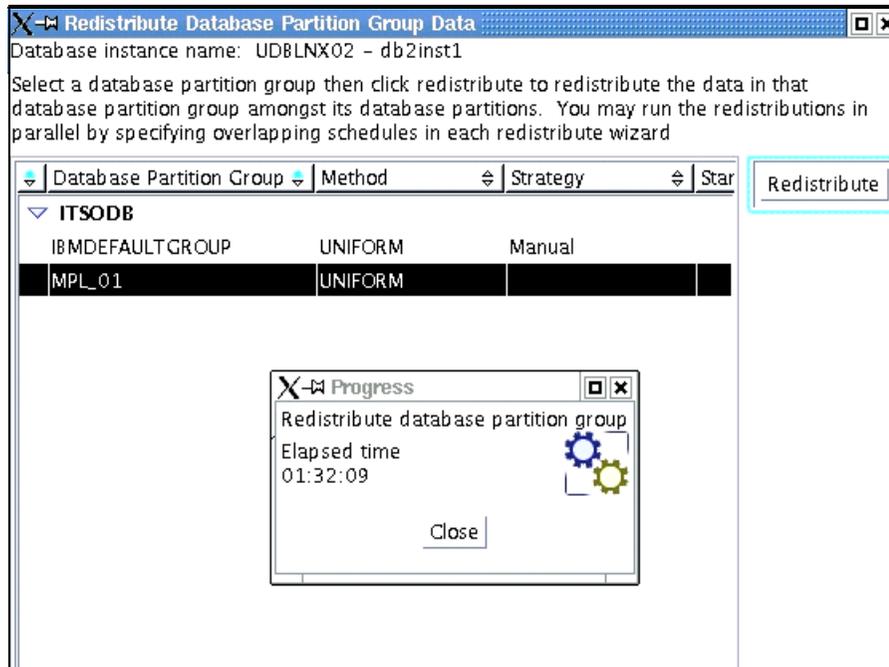


Figure 3-14 Redistribute in progress

Repeat this for all database partition groups that should be extended over the added partition. Figure 3-15 shows you the redistribute statements and completion messages for database partition group IBMDEFAULTGROUP. These messages are stored as well in:

```
$INSTHOME/sql1lib/redist/<dbname>.<part-group>.<time>
```

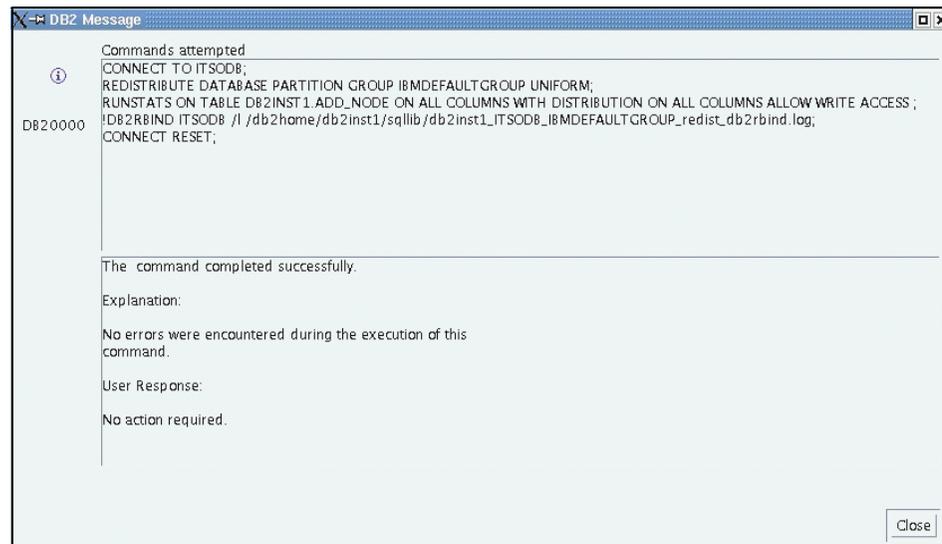


Figure 3-15 Redistribute completion messages

3.5 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. They are stored in different places, see Figure 3-16.

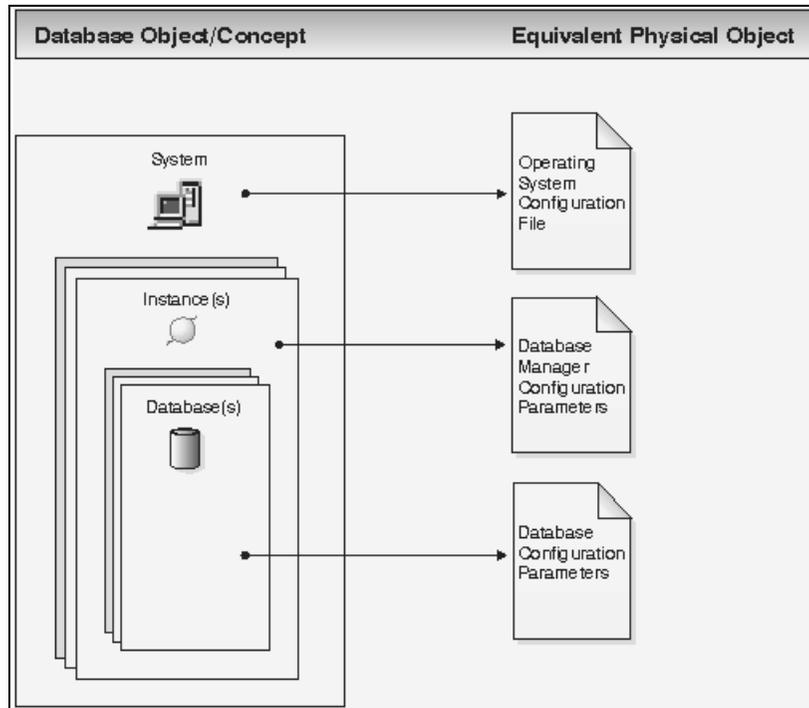


Figure 3-16 Configuration overview

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 Performance Configuration wizard or the AUTOCONFIGURE command.

3.5.1 DBM configuration

The database manager configuration 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 in *\$INSTHOME/sql/lib/db2dump/*. The default is **3**.

▶ **mon_heap_sz**

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.

▶ **query_heap_sz**

The initial default value is mostly sufficient. As a minimum, you should set *query_heap_sz* to a value at least five times larger than *aslheapsz*. This will allow for queries larger than *aslheapsz* and provide additional memory.

▶ **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**

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.

▶ **maxagents**

Number of maximum available db2agent processes. Use the **get snapshot for dbm** command and look at the value *Agents stolen from another application*. If greater than zero then increase maxagents.

▶ **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.

Note: For a complete and detailed description of all the DBM parameters, refer to Chapter 13 “Configuring DB2” in *IBM DB2 UDB Administration Guide: Performance V8*, SC09-4821.

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

- 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 3-17).

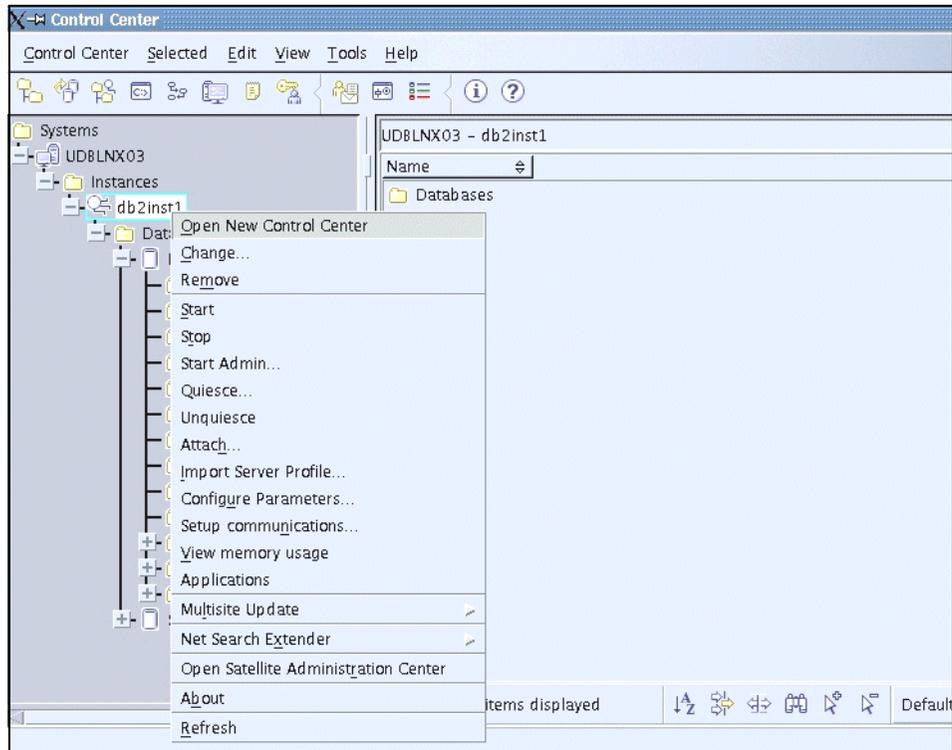


Figure 3-17 Configure instance in Control Center

- 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 3-18).

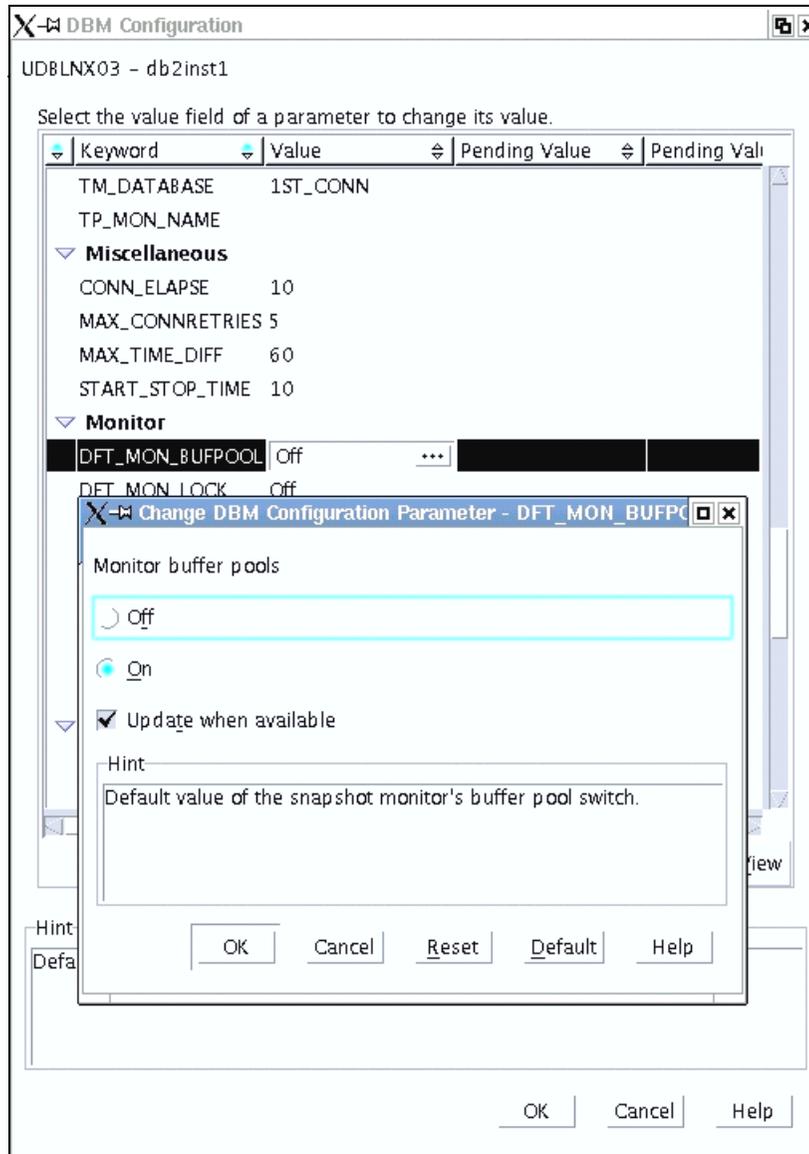


Figure 3-18 Sample of changing value

- Once all the desired changes have been made, click OK and these changes will be applied (Figure 3-19).

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.

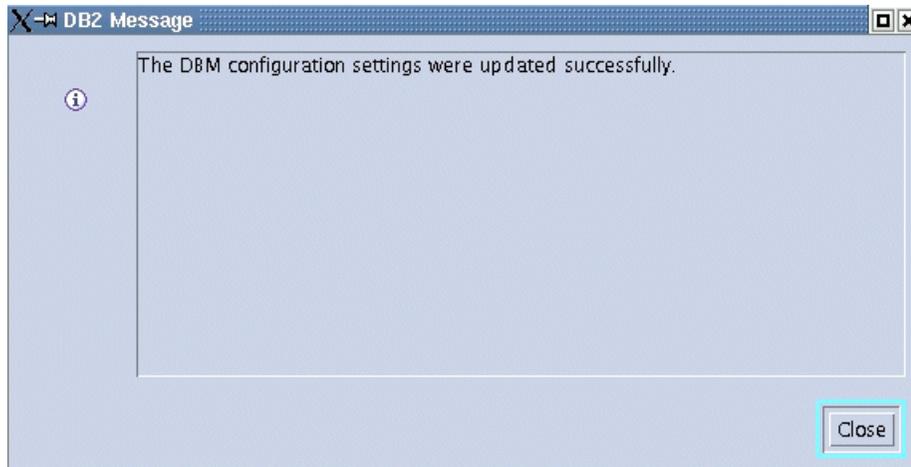


Figure 3-19 Confirmation window

Command line

Commands to change the settings can be quickly and conveniently entered by using the command line processor: Use

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 using your input about the system, application behavior, and so on.

Example 3-30 Using the command line to change DBm CFG

```
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
```

3.5.2 DB 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 a partitioned environment, the SQLDBCON exists in every database partition. Therefore, it is possible to have different settings for each partition. To modify these parameters, use the graphical interface Control Center or use the DB2 command line.

In a partitioned environment you have to issue this command on each partition:

```
db2 update db cfg for xy using <parameter>
```

Important DB cfg parameters

This section describes some important database configuration parameters and the recommended settings.

► avg_appls

This lets DB2 know how many concurrent applications run in your system. When set to 1, DB2 gives every application all current available memory in the bufferpool.

► catalogcache_sz

This holds database, tablespace, 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 - ((Catalog cache inserts X100) / 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 *dbheap*.

► **sortheap**

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* parameter in the database manager configuration file also needs to be adjusted.

Note: For a complete and detailed description of all the Database configuration parameters, refer to Chapter 13 “Configuring DB2” in *IBM DB2 UDB Administration Guide: Performance V8*, SC09-4821.

Monitor and update DB configuration

Configuring DB2 database settings can be done either through the Control Center or the DB2 Command line.

Control Center

- 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 3-20).

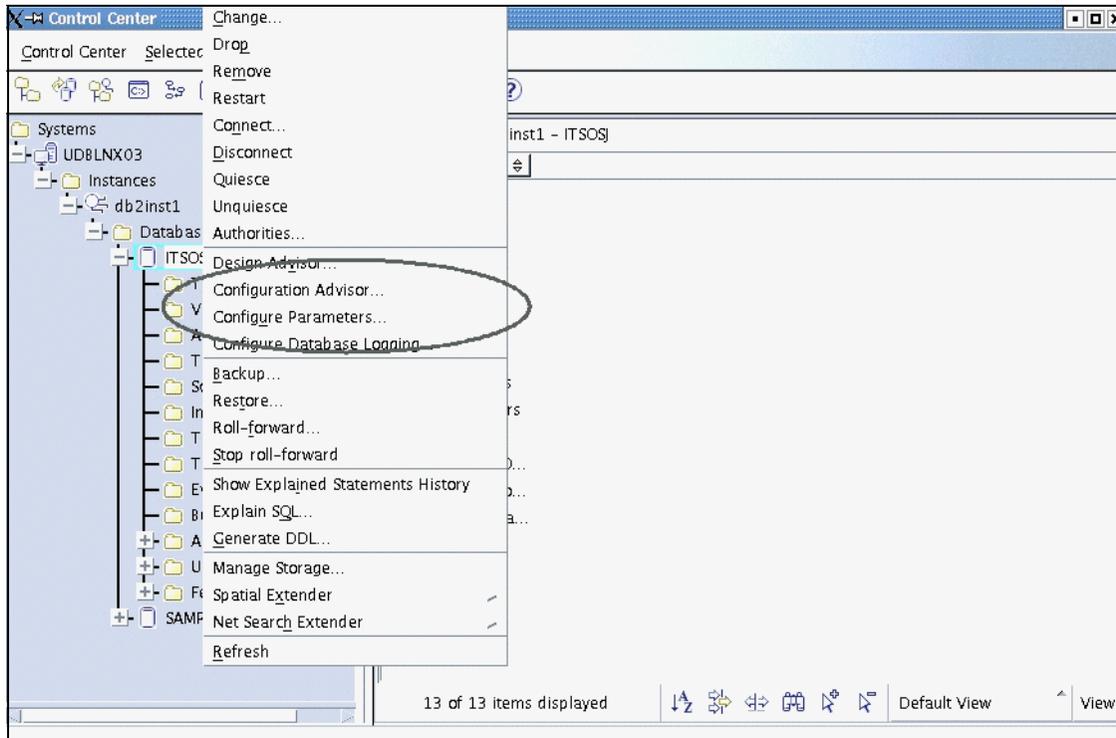


Figure 3-20 Select configuration method in Control Center

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 3-21). After selecting a parameter, a brief description of this parameter is displayed at the bottom under *Hint*.

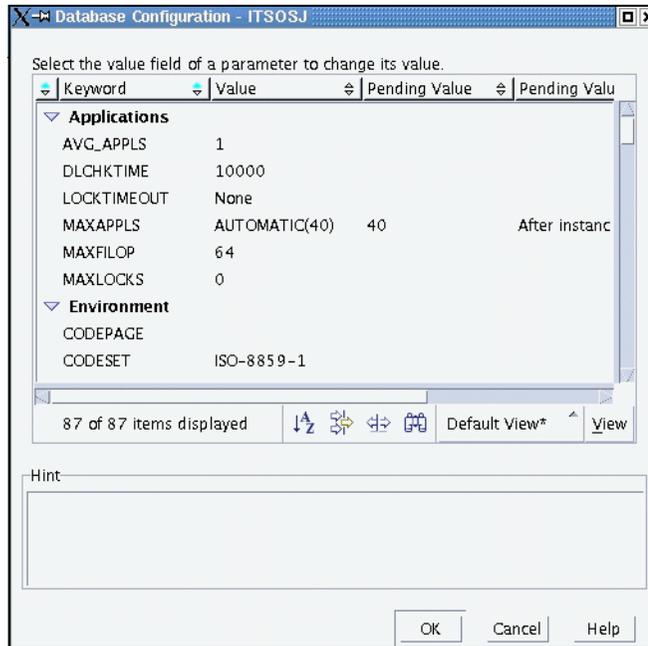


Figure 3-21 Select parameter to change

Figure 3-22 shows the change of the value for parameter *maxfilop*.

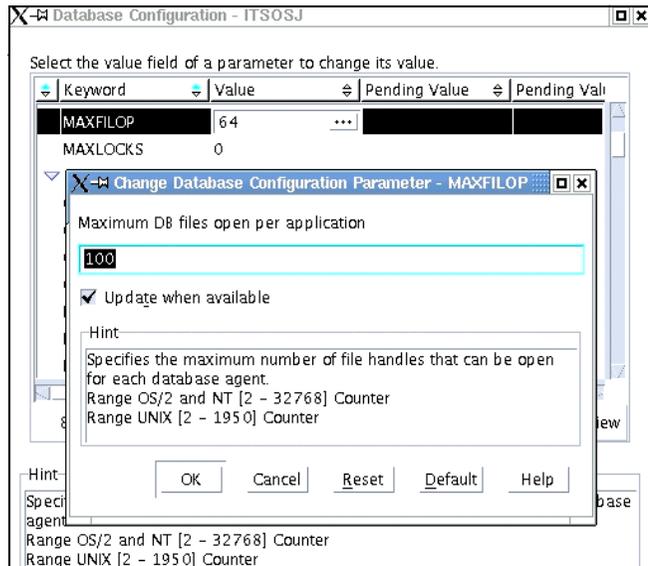


Figure 3-22 Sample of change *maxfilop* parameter

After all desired changes are done, press **OK** and all changes will be applied to the database (Figure 3-23).

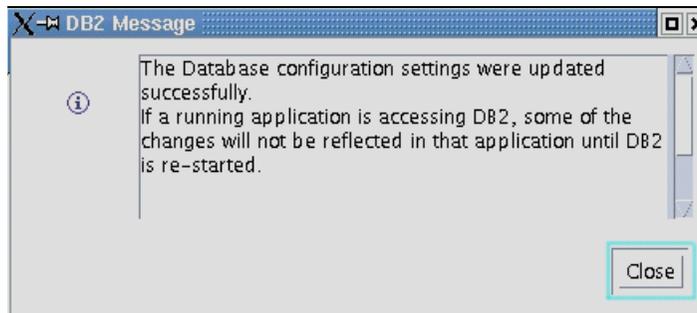


Figure 3-23 Confirmation after update database configuration

Using the Configuration Advisor option

The Configuration Advisor is a new 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.

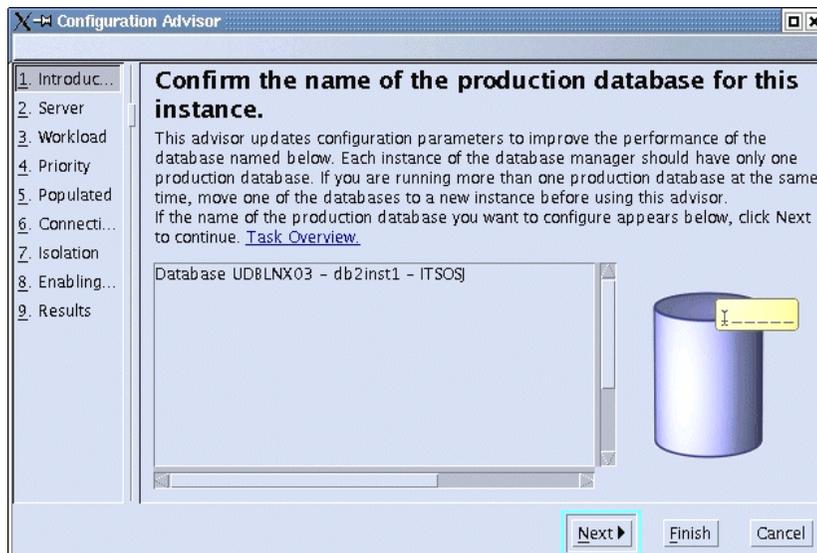


Figure 3-24 Configuration Advisor

Figure 3-24 shows you the first window of the Configuration Advisor GUI. 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. The following example shows database configuration update commands (Example 3-31):

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 3-31 Sample of update cfg

```
db2 get db cfg for itsosj
db2 update db cfg for itsosj using DBHEAP 1200
db2 get db cfg for itsosj show detail
```

3.5.3 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/sqllib/*.

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 tablespace or all tablespaces. 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.
- ▶ **DB2_HASH_JOIN** Default=YES
This specifies hash join as a possible join method when compiling an access plan.
- ▶ **DB2MEMMAXFREE** Default= 8 388 608 bytes
This is the maximum number of bytes of unused private memory that is retained by DB2 processes before unused memory is returned to the operating system.
- ▶ **DB2_NEWLOGPATH2** Default=0
This parameter allows you to specify whether a secondary path should be used to implement dual logging. The secondary path name is generated by appending a “2” to the current value of the logpath database configuration parameter.

Note: Details of all the registry variables are described in Appendix A “DB2 Registry and Environment Variables” in the *IBM DB2 UDB Administration Guide: Performance V8*, SC09-4821.

db2set on command line

db2set is the command to set registry variables. The command can be executed using the command prompt (Example 3-32).

Example 3-32 Sample of db2set command

```
db2set -a11
db2set -lr
db2set -i db2inst1 DBCOMM=tcPIP
db2set -g
db2set -g DB2ADMINSERVER=dasusr1
```

db2ca graphical tool

To start the graphical tool, issue the command **db2ca &** or open it in the application folder **IBM DB2 -> Configuration Assistant** (Figure 3-25).

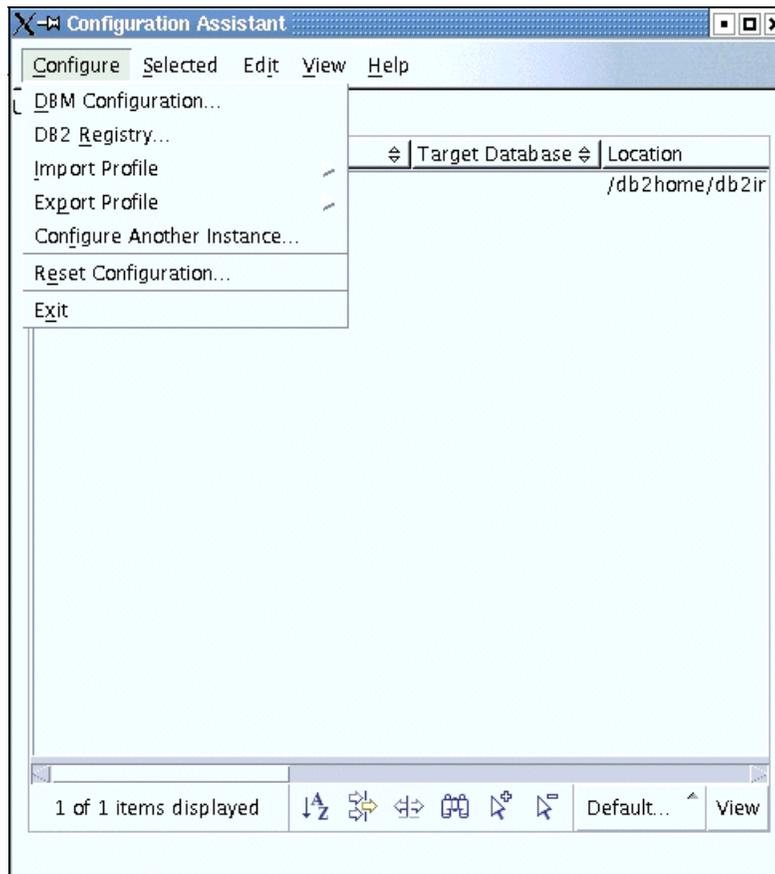


Figure 3-25 Configuration Assistant

Select DBM Configuration or DB2 Registry and go to the next following window and make your changes. After selecting a parameter to change, a description of this parameter appears at the bottom of the window.

Multipage allocation

For system managed tablespaces (SMS) we recommend that you enable multipage 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 tablespaces 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 **db2mpfa** on every database partition.

3.6 Client configuration

In this section we discuss how to install and configure DB2 clients to access DB2 ESE using the graphical tool Configuration Assistant or the DB2 command line processor.

There are three types of DB2 clients available:

► Run-Time Client

This client provides you access to DB2 UDB servers with application interfaces, such as JDBC, SQLJ, ODBC, CLI and OLE DB. Use this client if you don't need to administer DB2 servers.

► Administration Client

The Administration Client has all features of the DB2 Run-Time client plus tools to administer a DB2 Server.

► Application Development Client

This client provides a collection of graphical and non-graphical tools for developing applications. It includes all components of the DB2 Administration Client.

All Client versions are available on the following platforms: AIX, HP-UNIX, Linux, Solaris and Windows operating system.

The client selection depends on your application environment. At least you must have one Administrator Client to bind all DB2 CLI packages against the DB2 database that you want to use.

Note: Installation requirements details are described in *IBM DB2 UDB Quick Beginnings for DB2 Clients V8*, GC09-4832.

On the server side you have to configure the DB2 instance so that clients can connect to any database within this instance.

Verify on the server side these 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:

```
DB2SET -a11 and look at DB2COMM=tcPIP
```

- ▶ TCP/IP port name in database manager configuration should be set:

```
db2 get dbm cfg | grep SVCENAME
```

```
TCP/IP Service name (SVCENAME) = db2c_db2inst1
```

Installing DB2 Administrator Client on Linux

Once you have checked the prerequisites as specified in 2.1, “Basic requirements” on page 24, do the following tasks:

1. Log in as root user.
2. Mount your CD-ROM or copy the DB2 Code to disk.
3. Enter the command **./db2setup** and the DB2 setup wizard will start.
4. Follow the instructions under the wizard to install the client.

After installing your DB2 Client, you should configure the client to access a remote DB2 server. DB2 Version 8 only supports TCP/IP to administer a remote database.

To configure the client, start the Configuration Assistant graphical tool by typing **db2ca &** as a valid DB2 user or open it in the DB2 folder (Figure 3-26).

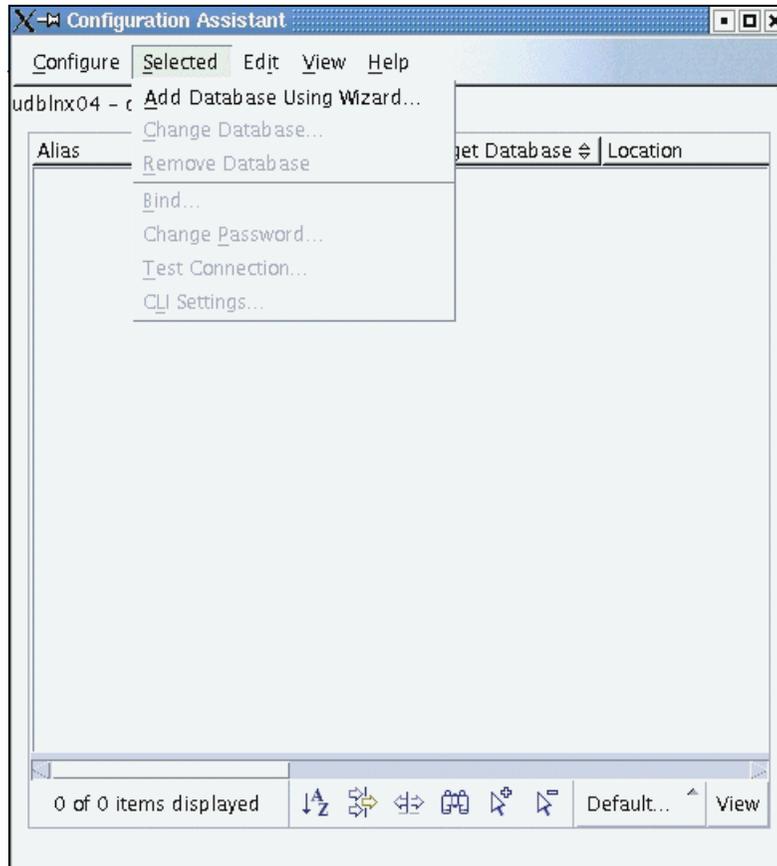


Figure 3-26 Add Database Wizard in CA

In the next window, select **Search Network** and click **Next**. Expand the tree under **Known systems** until you see all databases on this system. Select your desired database. After filling out all required fields, you can test a connection to this database (Figure 3-27).

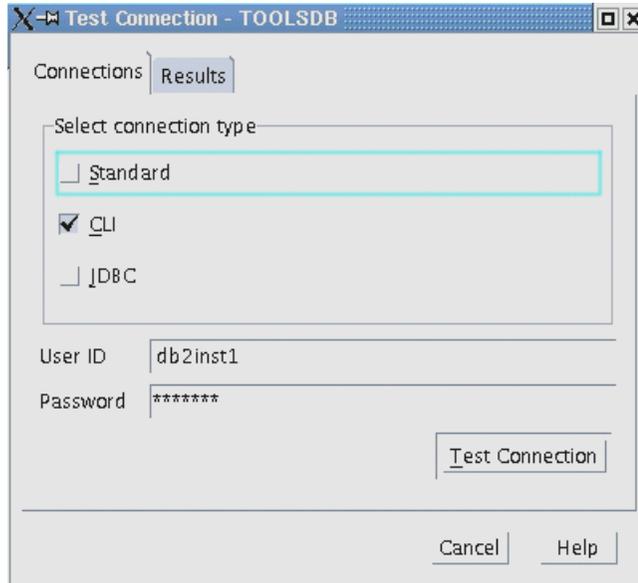


Figure 3-27 Connection test window

To configure many clients in your company, perform all configurations on one client and export the definition to a flat file. The file can be transferred to each client and imported under Configuration Assistant (Figure 3-28).

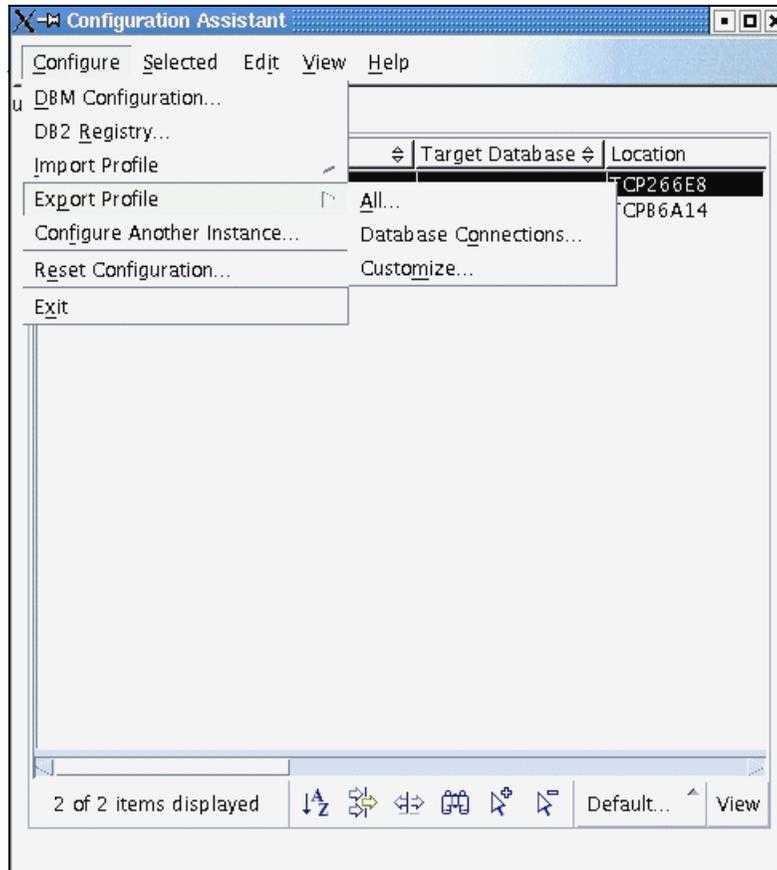


Figure 3-28 Export client definition

To configure clients you can use the DB2 command line processor as well. Example 3-33 shows you the commands to configure a client to access our sample database ITSODB on server udblnx02.

Example 3-33 Configure client on command line

```
db2 catalog tcpip node udblnx02 remote udblnx02 server 50001 ostype LINUX with
'DB Server udblnx02'
db2 catalog database ITSODB as ITSODB at node udblnx02 authentication server
with "Test database for ITS0"
```

```
db2 list node directory
```

Node 1 entry:

```
Node name                = UDBLN02
```

```
Comment                = DB Server udblnx02
Directory entry type    = LOCAL
Protocol                = TCPIP
Hostname                = udblnx02
Service name            = 50001
```

db2 list db directory

Database 1 entry:

```
Database alias          = ITSODB
Database name           = ITSODB
Node name               = UDBLN02
Database release level  = a.00
Comment                 = Test database for ITS0
Directory entry type    = Remote
Authentication          = SERVER
Catalog database partition number = -1
```

db2 connect to itsodb user db2inst1 using <password>

Database Connection Information

```
Database server        = DB2/LINUX 8.1.0
SQL authorization ID   = DB2INST1
Local database alias   = ITSODB
```

Note: For more details and explanation, refer to the *IBM DB2 UDB Command Reference V8*, SC09-4828.

3.7 Configuring licensing

DB2 is a licensed product. There are two tools that you can use to configure the license policy: the DB2 License Center and the *db2licm* utility. The License Center is a graphical tool found within the Control Center and *db2licm* is its command line counterpart. This section provides instructions about how to configure licensing using both of these tools. The following topics will be discussed:

- ▶ Installing the license key
- ▶ Setting up the Registered user policy
- ▶ Updating the number of licensed processors
- ▶ Changing the enforcement policy

All of these tasks are optional, as they depend on the version of DB2 you are using (*Try and Buy* or fully licensed) and the type of license policy your company purchased.

3.7.1 DB2 License Center

The DB2 License Center is a graphical tool found within the Control Center that manages licensing. You can use this tool to view license information, add and remove license keys, specify user and enforcement policies, and specify the number of licensed processors. 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 which 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.

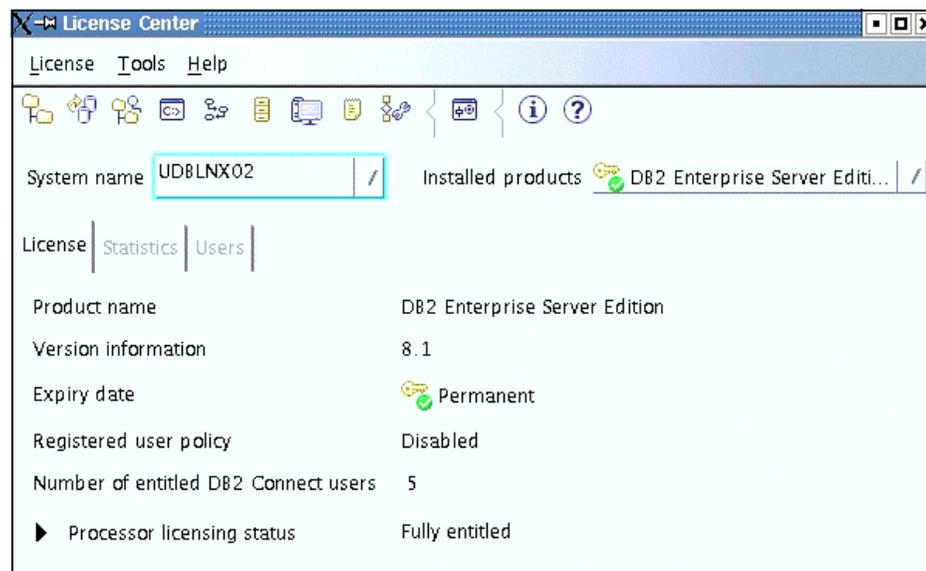


Figure 3-29 DB2 License Center

Installing a license key

To install a license key using the License Center, you must add the *db2ese.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**.

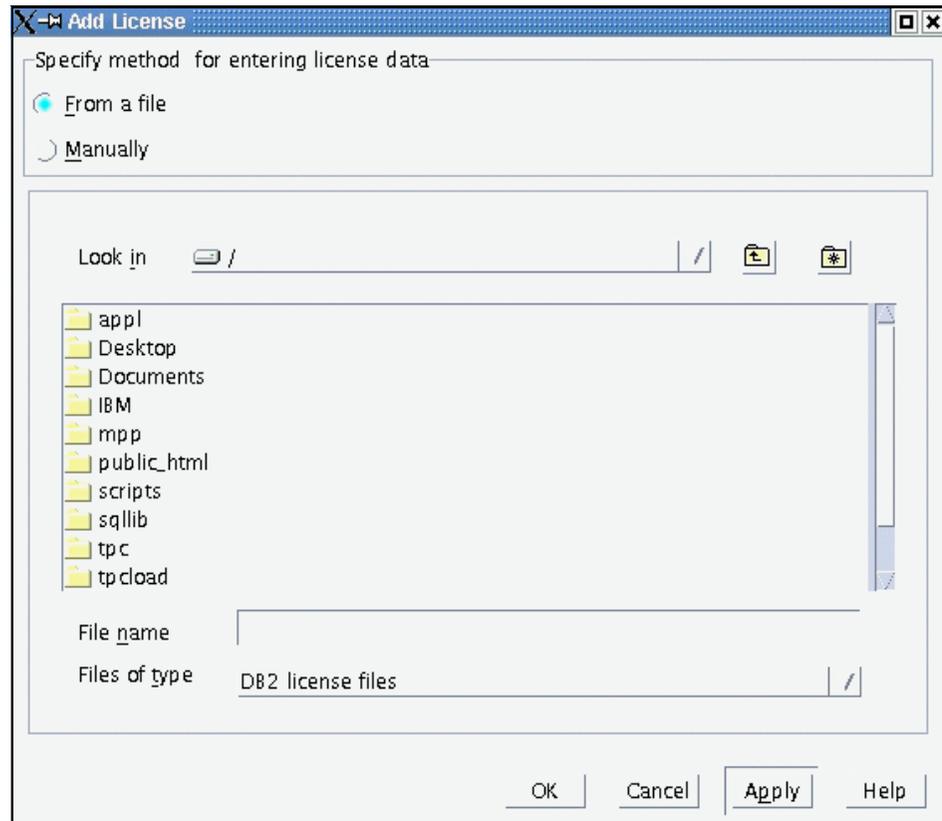


Figure 3-30 Add license file

To remove a license file, select **Remove** from the License menu.

Setting up a Registered user policy

If you have purchased Registered User licenses, you need to add Registered users to the Registered users list. To do this:

1. Select **Change** from the **License** menu.
2. Check off the box beside Registered DB2 Connect users, then press **OK**. This will enable the Users tab in the License Center.
3. Select the **Users** tab in the License Center, then click **Add**. Enter in the user ID and click **Apply**. After you have added all Registered users, click **OK**.

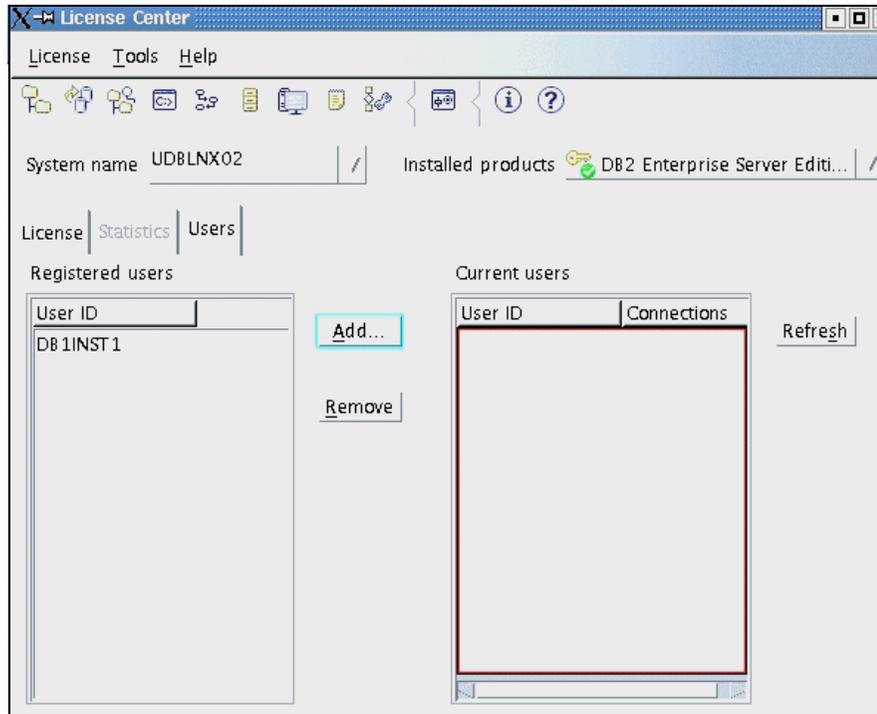


Figure 3-31 Add Registered users

Updating the number of licensed processors

To change the number of licensed processors for your DB2 product, do the following:

1. Select **Change** from the License menu.
2. In the *Number of licensed processors* box, use the up and down arrows to set the number, and then click **OK**.

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*.

3.7.2 db2licm and db2licd

The `db2licm` and `db2licd` utilities are command line tools that manage licensing. The `db2licm` command is used to view license information, add and remove license keys, specify user and enforcement policies, and specify the number of licensed processors. The `db2licd` utility displays connection information for current users.

To view all of your licensing information, enter `db2licm -l`. For example, this is our output:

```
db2inst1@udb1nx03:~/sql1lib> db2licm -l
Product Name                = "DB2 Enterprise Server
Edition"
Product Password            = "DB2ESE"
Version Information          = "8.1"
Expiry Date                  = "Permanent"
Registered Connect User Policy = "Enabled"
Number Of Entitled Users     = "5"
Enforcement Policy          = "Soft Stop"
Number of processors         = "1"
Number of licensed processors = "1"
Annotation                   = ""
Other information            = ""
```

To view connection information for currently connected users, run `db2licd`:

```
db2licd -q
```

Installing a license key

To install a license key, you must add the `db2ese.lic` license file using the `db2licm` command. To add a DB2 license file:

1. Login as instance owner.
2. Enter the following command:

```
/opt/IBM/db2/V8.1/adm/db2licm -a <filename>
```

Where, *filename* is the full path name and filename for the DB2 ESE license file `db2ese.lic`. This license file is located in `/db2/license` in the root directory of your CD-ROM. For example, if the CD-ROM is mounted in the directory `/cdrom`, enter the following command:

```
/opt/IBM/db2/V8.1/adm/db2licm -a /cdrom/db2/license/db2ese.lic
```

If you are setting up a multi-partitioned database system, perform this task on each machine.

On Linux, the DB2 license keys are located in `/var/lum` in a file called `nodelock`.

To remove a license file, use `db2licm -r`. For example: `db2licm -r db2ese`.

Setting up a Registered user policy

Perform this task if your company purchased Registered user licenses.

1. Log in as the instance owner.
2. Enable the Registered users policy for DB2 ESE by entering the following command:

```
db2licm -p db2ese registered
```

3. Go to the `/opt/IBM/db2/V8.1/misc` directory.
4. In this directory, create a text file called `db2ese.reg`. Using a text editor, add registered user entries to this file. For example:

```
db2inst1  
user2  
user3
```

5. Restart the `db2licd` process for the changes to take effect

Run the following command to stop the license daemon:

```
db2licd -end
```

The license daemon will be restarted automatically during connection requests or license center updates, where `db2licd` is required.

Updating the number of licensed processors

Use the `db2licm -n` command. For example, to change the number of licensed processors to 4, enter the following command:

```
db2licm -n db2ese 4
```

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
```



Migration

When migrating DB2 from Version 6 or 7 to Version 8, there are several important steps to perform before and after installation to ensure a safe and successful migration.

In this chapter, we discuss:

- ▶ Migration planning
- ▶ Installing the new DB2 V8 server
- ▶ Migrating your instances and databases

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 Version 8, 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.

Migration requirements

Make sure your system satisfies the following software and space requirements prior to Version 8 migration.

- ▶ Migrate all of your DB2 servers to Version 8 before you migrate any of your DB2 clients to Version 8. Some problems may occur if you migrate your clients first.
- ▶ If you are migrating from Version 6 to Version 8 on Linux, you must be at least at Version 6 FixPak 2.
- ▶ 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.
- ▶ Ensure that you have sufficient space in the filesystem where the tablespaces 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.

Table 4-1 Space recommendations for migration

Table space	Space recommendation
System catalog space (SYSCATSPACE)	<ul style="list-style-type: none">▶ You should allocate 2x the space currently occupied▶ For DMS tablespaces, free pages should be equal to or greater than used pages
Temporary table space (TEMPSPACE1, by default)	<ul style="list-style-type: none">▶ For DMS tablespaces, total pages should be twice the amount of total pages for the system catalog table space

To check the size of your DMS tablespaces, enter in the following commands as instance owner:

```
db2 list database directory
db2 connect to database_alias
db2 list tablespaces show detail
```

To increase space in a DMS tablespace, you can add additional containers.

- ▶ 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.

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
```

Migration test consideration

The entire migration process consists of two parts: (1) Installing DB2 Version 8 code, and (2) Migrating instances and databases. Installing DB2 Version 8 does not require system down time. Migrating instances and databases is the actual step which converts your DB2 system from Version 6 or 7 to Version 8. You need to stop DB2 to perform the instances and databases migration.

We recommend that you install and test Version 8 before migrating any Version 6 or Version 7 instances to Version 8. If you have a test environment which is a mimic of the production system, you can install DB2 V8, migrate DB2 instances and databases, test 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 Version 8 while the application is still up and running under Version 6 or 7. You can then create Version 8 test instances and databases to test your applications while production transactions are running.

When you are convinced to move your production systems up to Version 8, 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.

If you want to skip testing Version 8 and go directly to migration, you still can consider installing Version 8, while your existing instance(s) continue to run, again to reduce system down time.

4.2 Installing the new DB2 V8 server

This section discusses migration considerations when installing DB2 on single and multi-partitioned systems using the DB2 Setup, `db2_install`, and response file installation methods.

4.2.1 Installing DB2 V8 using DB2 Setup wizard

In this section, we provide an example of how to migrate DB2. In our example, we do not want to create new users or new instances. We simply want to migrate our existing Version 7 instances and databases to Version 8.

Here we only discuss migration-specific considerations that were not covered in Chapter 2. For details about how to install DB2 using the DB2 Setup wizard, refer to 2.3.1, “DB2 Setup” on page 52.

- ▶ As root, run `db2setup` from the directory where the DB2 install image is located.
- ▶ The Set user information for the Database Administration Server window appears (Figure 4-1). *Do not* select the existing V6 or V7 DAS user (for example, `db2as`) for the Version 8 DAS user. Always select a new user for the Version 8 DAS.

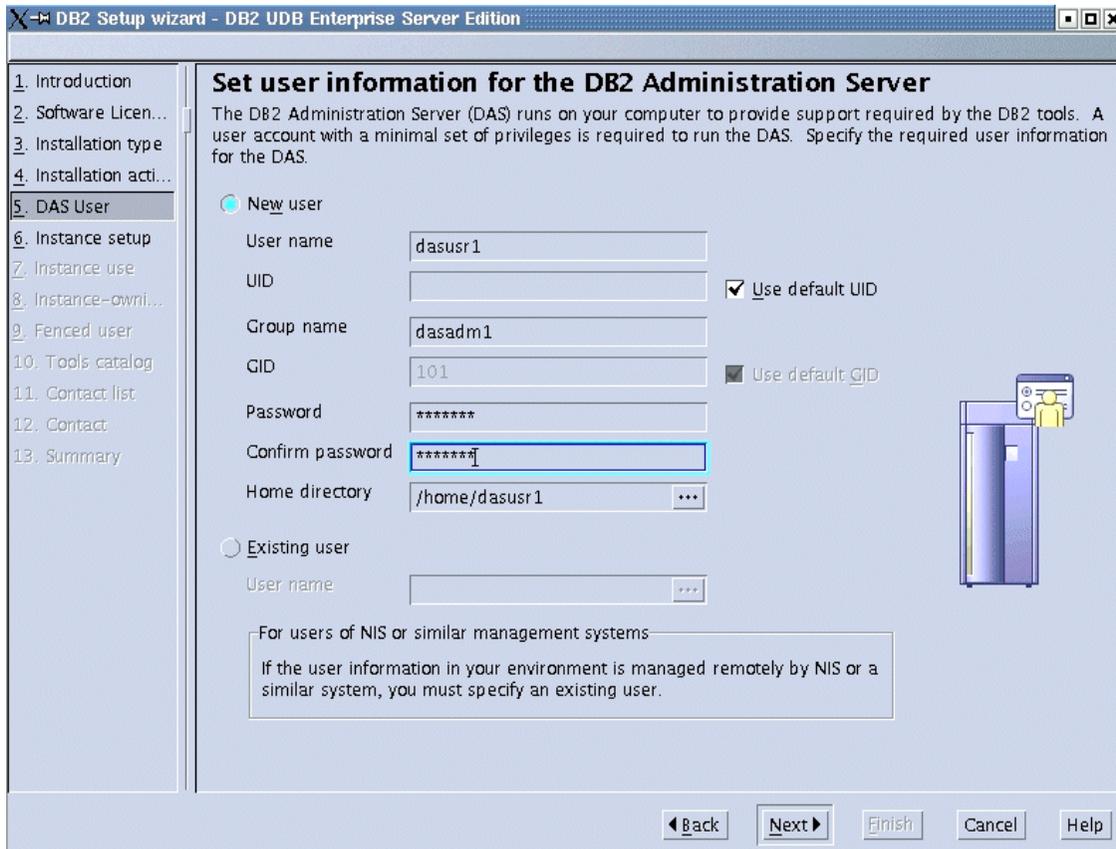


Figure 4-1 Create new DAS user

- ▶ The Set up a DB2 instance window appears (Figure 4-2). If you only want to migrate your existing instances, select the **Do not create a DB2 instance** option. Select **Create a new DB2 instance** if you want to create a new instance in addition to the old ones you will be migrating. If you want to create the V8 tools catalog on a separate instance, then we recommend you select **Create a new DB2 instance** at this point in time. This option allows you to use the DB2 Setup wizard to create the tools catalog on the new instance you are creating.

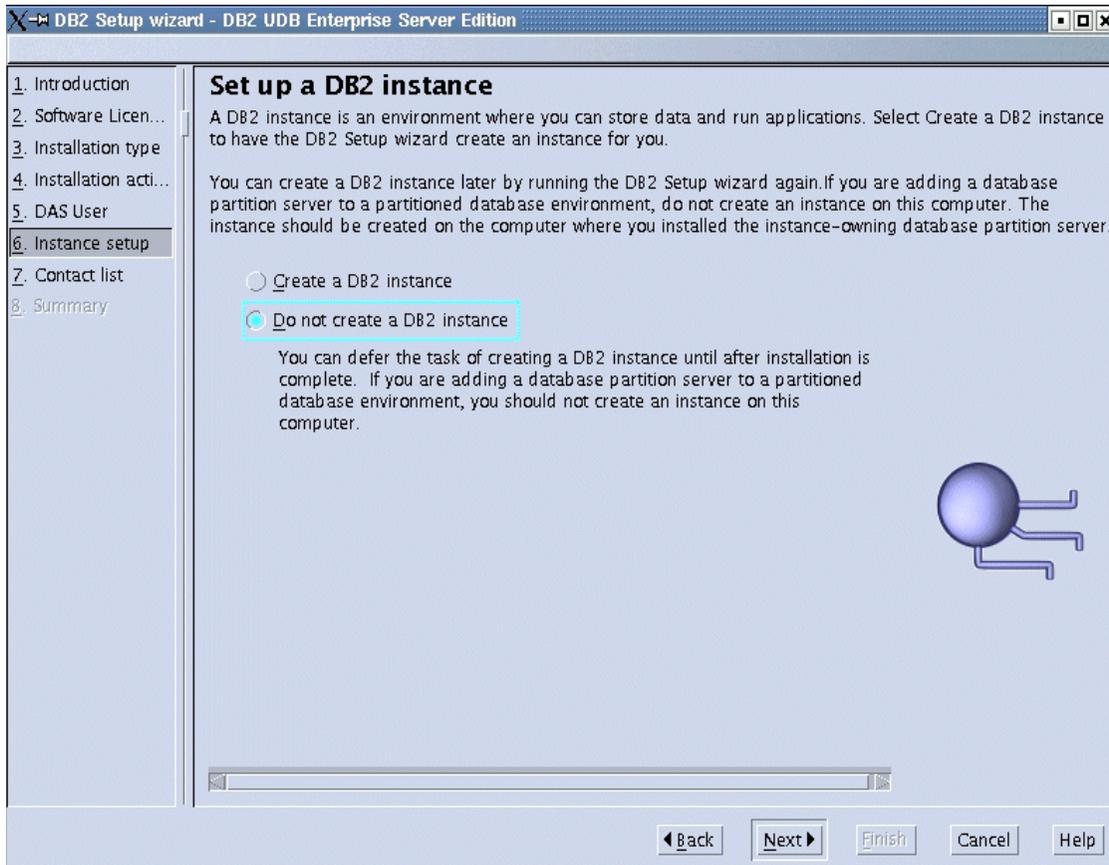


Figure 4-2 Migrating existing instance only

- ▶ The Prepare the DB2 tools catalog window only appears if you choose to create a new instance. This will create a local database in the new instance you are creating. This database will be used to store task metadata that is used by the scheduler and Task Center. You may also migrate pre-Version 8 scripts and schedules (for the Task Center) to this tools catalog after the installation, using the `dasmigr` command.

4.2.2 Installing DB2 V8 using `db2_install`

If you want to use `db2_install` to migrate DB2 to Version 8, perform the same pre-migration tasks described in 4.3.1, “Pre-migration tasks” on page 139, run the `db2_install` utility, and then perform all of the post-migration tasks described in 4.3, “Migrating your instances and databases” on page 139. Refer to 2.3.3, “`db2_install` utility” on page 68 for information on using `db2_install`.

One thing to be careful about is the version of JDK residing on your system. Because db2_install does not install the latest JDK for you, your system will only have an older version of the IBM JDK. This will cause problems when you try to use the V8 graphical tools or development tools. We recommend you drop the old version of the IBM JDK and install IBM Java Developer's Kit Version 1.3.1. Refer to 2.1.6, "Additional software requirements" on page 31 for instructions on how to install the newer JDK.

4.2.3 Installing DB2 V8 using a response file

Refer to Chapter 2 for instructions on how to install DB2 using a response file. Because the response file install is essentially the same thing as the DB2 Setup wizard, read the migration considerations and recommendations for DB2 Setup in 4.2.1, "Installing DB2 V8 using DB2 Setup wizard" on page 136.

4.2.4 Creating test instance

We recommend that you create a test V8 instance after installing DB2 UDB V8 server, and test it to ensure your V7 applications will work with your test data in your test instance. Refer to Chapter 2, "Installation" for instance user ID, group setup, and creating instances. If you install using the db2setup GUI or the response file install, you can create your test instance normally through them.

4.3 Migrating your instances and databases

Now that you have successfully installed DB2 UDB V8 server, it is time to migrate the instances and databases. This section provides a list of pre-migration tasks and the migration procedures.

4.3.1 Pre-migration tasks

Perform the following steps to prepare for a safe and successful migration:

1. Save DBM and DB configuration settings. You should compare these settings before and after migration to check for any migration errors.
2. Save tablespace and package information using the **db2 list tablespaces** and **db2 list packages** commands. You may also want to compare this information before and after migration.
3. Change diagnostic error level to 4. Diaglevel 4 records all errors, warnings, and informational messages to make problem determination easier for you if any migration errors occur. Issue the following command:

```
db2 update dbm cfg using diaglevel 4
```

4. Take the V6 or V7 DB2 Server offline:
 - a. Stop the DB2 license service by entering **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**.
5. 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. Use the following syntax:

```
db2 archive log for database <database_alias> user <user_name> using <password>
```
6. Back up databases. We suggest you do an offline backup from each local database. Use the following command:

```
db2 backup database <database_alias> user <user_name> using password <password>
```
7. As instance owner, stop the instance by issuing **db2stop**.
8. 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.

4.3.2 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 8 format. The **db2 migrate database** command is used to migrate the databases.

In case you are wondering what happens during instance migration, the **db2imigr** command performs the following:

- ▶ Checks cataloged databases to make sure they are ready for migration
- ▶ Migrates your instance to a V8 instance
- ▶ Updates system and local database directories to a Version 8 format
- ▶ Merges pre-V8 DBM configuration settings with V8 DBM configuration settings

In the remainder of this section, we lead you through a procedure of completing your migration.

1. Login as the instance owner.
2. Run **db2ckmig** to verify that a database can be migrated. In the following example, we scanned all cataloged databases and saved results to text file *mig.txt*.

Example 4-1 db2ckmig command

```
db2inst1@udb1nx02:/opt/IBM/db2/V8.1/instance> ./db2ckmig -e -1  
/db2home/db2inst1/mig.txt
```

```
db2ckmig was successful. Database(s) can be migrated.
```

3. Next, we looked at *mig.txt* to check for any errors and to ensure that the version of DB2CKMIG being run was Version 8, and not a previous version.

Example 4-2 db2ckmig output

```
db2inst1@udb1nx02:/opt/IBM/db2/V8.1/instance> more /db2home/db2inst1/mig.txt
```

```
Version of DB2CKMIG being run: VERSION 8.
```

4. Now it is safe to migrate the instance. Login as root and run the **db2imigr** command to migrate your instance to Version 8. Use the following syntax:

```
./db2imigr [-u <fenced_user>] instance_name
```

Where, *fenced_user* represents the fenced user ID, and *instance_name* represents the name of the instance you are migrating. The fenced user ID is only required if you are migrating from a client instance to a server instance. Our input and output is shown in the following example.

Example 4-3 Migrating the instance

```
udb1nx02:/opt/IBM/db2/V8.1/instance # ./db2imigr db2inst1
```

```
db2ckmig was successful. Database(s) can be migrated.  
DBI1070I Program db2imigr completed successfully.
```

5. At this point in time, you may want to check the DB2 level to ensure that your instance is now a Version 8 instance. Our db2level results are shown in Example 4-4. Fortunately, there were no unexpected surprises.

Example 4-4 Check db2level

```
db2inst1@udb1nx02:~/sql1lib/db2dump> db2level  
DB21085I Instance "db2inst1" uses "32" bits and DB2 code release "SQL08010"  
with level identifier "01010106".
```

Informational tokens are "DB2 v8.1.0.8", "s021112", "", and FixPak "1".
Product is installed at
"/opt/IBM/db2/V8.1".

6. The next step is to migrate all of your databases to Version 8 format. To do this, login as the instance owner to the machine where the database catalogs are stored, issue **db2start**, and then run the **db2 migrate database** command.

Before doing so, check the catalog database partition number to ensure that you are working on the correct partition.

Example 4-5 Viewing catalog information

```
db2inst1@udblnx02:~> db2 list db directory on /database
Local Database Directory on /database

Number of entries in the directory = 1
Database 1 entry:
Database alias           = ITSODB
Database name           = ITSODB
Database directory      = SQL00001
Database release level  = a.00
Comment                 = ITSO San Jose
Directory entry type    = Home
Catalog database partition number = 0
Database partition number = 0
```

Next, we started the instance.

Example 4-6 Starting the instance

```
db2inst1@udblnx02:~> db2start
11-15-2002 16:17:55    1  0  SQL1063N  DB2START processing was successful.
11-15-2002 16:17:55    0  0  SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.
```

Now we are ready to migrate the database. To migrate a database, use the following command syntax:

```
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, as shown in Example 4-7.

Example 4-7 Migrating the database

```
db2inst1@udb1nx02:~/sql1lib/db2dump> db2 migrate database itsodb
SQL1103W The Migrate Database command processing was successful.
```

7. Update statistics.

DB2 Version 8 has some statistics that are modified or do not exist in previous versions of DB2. To take advantage of these statistics, 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.

8. Rebind packages.

Any packages stored in a database are invalidated during migration, but then rebuilt after migration when the V8 database manager uses them for the first time. After migration, run the **db2rbind** command to rebind these packages.

9. 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 *userid* and *password* parameters are optional.

The **db2exmig** command does the following:

- Renames the V6/V7 explain tables.
- Creates a new set of explain tables using EXPLAIN.DDL. This EXPLAIN.DDL is under \$INSTHOME/sql1lib/misc directory.
- Copies the contents of the old tables to the new tables.
- Drops old explain tables.

10. Compare DBM and DB configuration settings, and tablespace and package records from before and after migration to verify that the migration was successful.

4.3.3 Multiple partition migration

To migrate a partitioned database system, you need to (1) install the new DB2 V8 server on each machine, (2) migrate the instance on the instance-owning machine, (3) migrate the databases on the catalog node, and (4) create the new V8 DAS on each physical machine in the partitioned database system.

4.3.4 Migrating the DB2 Administration Server

In Version 8, the DB2 Administration Server (DAS) has changed significantly. It is no longer an instance; instead, it is now a service. 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. Due to these changes, there are several things to be aware of when migrating the DB2 to Version 8.

- ▶ Do not use an existing DAS user (for example, the V7 DAS user `db2as`) for Version 8. You must create a new DAS user for the Version 8 DAS.
- ▶ If V7 and V8 server installations are *not* to co-exist on the same machine, then drop the V7 DAS instance before migrating to Version 8. However, do not drop the V7 DAS if you want to migrate V7 scripts and schedules to the V8 tools catalog. If this is the case, only drop the V7 DAS after installing DB2 V8, creating the V8 tools catalog, and running the `dasmigr` command.
- ▶ If V7 and V8 server installations are to co-exist, then the V7 DAS is to be used to administer V7 instances, and the V8 DAS is to be used to administer V8 instances. The V8 DAS can only run on port 523; therefore, the V7 DAS will have to run on a private port. To set up the V7 DAS on a private port, perform the following steps:
 - a. Login as the V7 DAS user.
 - b. Stop the DB2 Administration Server by issuing `db2admin stop`.
 - a. Unset the Version 7 DB2ADMINSERVER registry value by setting it to blank:

```
db2set DB2ADMINSERVER=
```
 - b. Update the port number by entering the following command:

```
db2 update dbm cfg using SVCENAME <port_number>
```

Where, *port_number* is a new port number assigned to the V7 DAS.

- c. Reset the V7 DB2ADMINSERVER registry variable to the Version 7 DB2 Administration Server instance name. For example:

```
db2set DB2ADMINSERVER=db2as
```
- d. Restart the DB2 Administration Server by issuing **db2admin start**.
- e. Using the Control Center, catalog a new administration node at the client to use the V7 DAS.

Migrating V7 scripts and schedules

If you want to use existing pre-V8 scripts and schedules (created in the old DAS) in DB2 Version 8, you will need to migrate the DAS using the **dasmigr** command. Migrating the DAS will migrate these scripts and schedules to the new V8 DB2 tools catalog. You need to have an existing V8 DB2 tools catalog prior to performing this task. If you have not yet created a tools catalog, refer to 2.3.3, “db2_install utility” on page 68 for instructions. To migrate the DAS, follow this procedure:

1. Log in as root user.
2. Go to: `/opt/IBM/db2/V8.1/instance`
3. Enter the command:

```
dasmigr previous_das_name new_das_name
```

Where, *previous_das_name* represents the name of the pre-V8 DAS instance, and *new_das_name* represents the name of the new V8 DAS.

For example, we migrated our old DAS db2as to the new DAS dasusr1:

```
./dasmigr db2as dasusr1
```

4. Our scripts that were created in the old DAS (db2as) were migrated to the new DAS. By default, scripts and schedules are put in the `/db2home/dasusr1/das/tmp/migr/` directory and are administered by the Task Center.

4.3.5 Client and server compatibility

Be aware of the following DB2 Version 7 and Version 8 client and server compatibility restrictions:

- ▶ Avoid having V8 clients access V7 servers. There are known restrictions that are documented, and there may be some unknown restrictions.
- ▶ You can use a V7 32-bit client to access a V8 32-bit server, but if you want to use the DB2 GUI tools (for example, Control Center, Health Center, and so on) or database commands (for example, BACKUP, RESTORE) against a V8 server, then you require a V8 client.

- ▶ You can use either a V8 32-bit client or V8 64-bit client to access a V8 64-bit server without using DB2 Connect.
- ▶ A V7 32-bit client can access to V8 64-bit server as long as you use a Version 8 32-bit DB2 Connect gateway:

V7 32-bit client <--> V8 32-bit DB2 Connect Gateway <--> V8 64-bit server



Administering databases

This chapter contains information regarding some important database maintenance tasks for DB2 UDB 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

We discuss factors to consider when choosing database and table space recovery methods, including backing up and restoring a database or table space, and using rollforward recovery. We also discuss the utilities' basic usage, for example, BACKUP, RESTORE, ROLLFORWARD, and so on.

- ▶ Table and index reorganization and statistics collection

We discuss the REORG utility, which can be used to defragment the data pages occupied by tables and indexes, and the RUNSTATS utility for statistics collection, as well as the REBIND task to make the latest statistics information applicable to existing packages. We also discuss the different methods that can be used for table reorganization and index reorganization.

- ▶ Data movement via EXPORT, IMPORT and LOAD

We discuss the utilities that can be used to move data between tables or databases, or other desirable formats; for example, exporting data into the format of a spread program or operating system files with the format of ASC, DEL, or IXF, and so on. We also discuss the LOAD utility in this section, especially for the new online load feature which takes locks at the table level only now. These new load features significantly improve the availability of the

data and help customers deal with the maintenance of large data volumes and reduces the maintenance window time required.

- ▶ Job scheduling using the Task Center

We discuss job scheduling by using the Task Center, and also mention the DB2 Tools Catalog Database, which is introduced in DB2 UDB Version 8 and is used to store scheduling-related data.

5.1 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 backups of the entire database as well as saving the recovery log files. 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. 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.

DB2 logs every transaction, unless told not to. Load is a DB2 tool, and when using that tool to load data, DB2 will not log the transaction. The log files are used to recover the database. DB2 has two types of logging mechanisms:

- ▶ Circular logging

Circular logging is the default logging method for a newly created database. The database is a non-recoverable database when circular logging is used. In circular logging, a set of log files is used in round-robin fashion. Only active logs exist in the log-path which are used by crash recovery. Crash recovery is the process where all uncommitted changes are rolled back to a consistent state of the data. After a full offline database backup, all changes in the log files will be set as invalid. New changes after backup will be written to the active logs.

- ▶ Archive logging

By using archive logging, the database is considered as a recoverable database. Active logs will be retained as archive logs after taking a full offline database backup. To enable archive logging, set the LOGRETAIN parameter in the database configuration. This allows you to recover your database or any table spaces to end of log or any point in time. For crash recovery, DB2 only uses the active logs. To prevent a filesystem full condition, enable USEREXIT so that DB2 will automatically save the archived logs to either TSM or another filesystem after filling an active log and releasing the space in the log path.

Since Version 7.2, DB2 has provided you with 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. In DB2 UDB Version 8, the mirror log path is enabled by updating the database configuration, as shown below.

```
db2 update db cfg for <db_alias> using MIRRORLOGPATH <log path>
```

Another high availability feature is to set the configuration parameter for transaction log blocking. In case of a log full condition, DB2 cannot allocate more logs in the log path. Read only access is possible.

```
db2 update db cfg for <db_alias> using BLK_LOG_DSK_FUL YES
```

DB2 provides the following backup mechanisms.

Offline 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.

Online backup

An online backup allows applications to connect to the database and read and write to tables during database backup. Log retain or userexit must be enabled for an online backup. This type of backup can backup the entire database or only one or more tablespaces.

Full backup

A full backup is a image copy of the entire database (Figure 5-1). In a partitioned environment, each database partition backup is written in a separate output file.

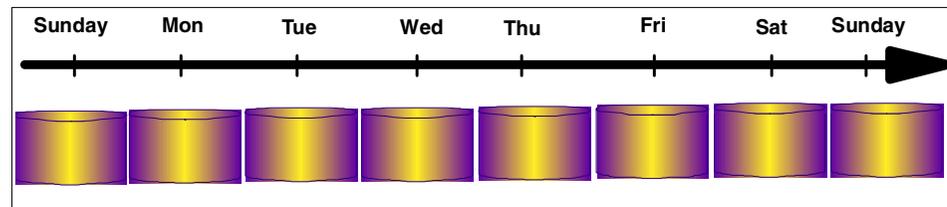


Figure 5-1 Full backup

Incremental 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 5-2. The TRACKMOD parameter (Track modified pages) in the database configuration must be set to ON.

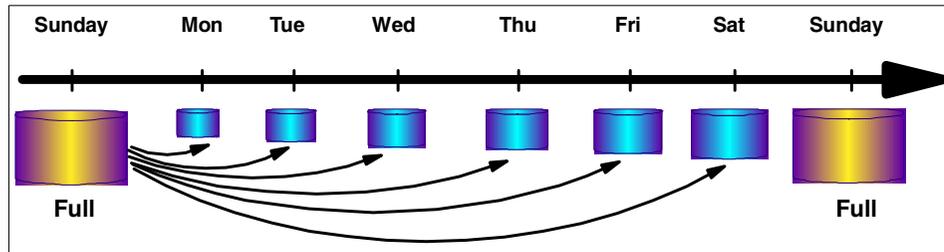


Figure 5-2 Incremental backup

Delta 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 5-3).

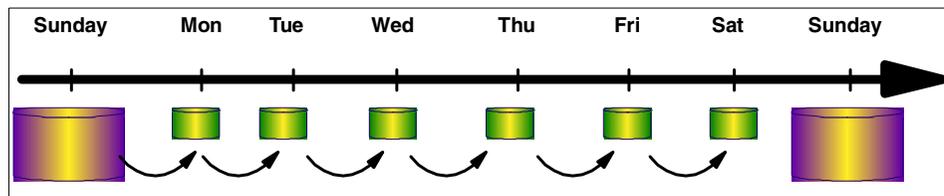


Figure 5-3 Incremental delta backup

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.

5.1.1 Enable roll forward and log archiving

To use all functionality available in DB2 to prevent data loss, we will enable both roll forward recovery and log archive files. This allows us to recover the database to the point in time of failure. The following example walks you through this task in a multi-partitioned database environment.

DB2 provides sample user exit programs to archive DB2 logs either to disk, Tivoli Storage Management (TSM) server or to Tape. 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.

Create directory `/db2archive/db2inst1` and set read/write permissions for the instance owner user ID, so that DB2 can access the directory.

```
chown db2inst1:db2grp1 /db2archive/db2inst1
```

Generate the directory structure for each partition in which to hold the archived logs:

```
db2_a11 'mkdir /db2archive/db2inst1/ITSODB'
```

```
db2_a11 'typeset -Z4 DB2NODE;mkdir /db2archive/db2inst1/ITSODB'/NODE$DB2NODE'
```

Copy from `$INSTHOME/sqllib/samples/c` the sample userexit `db2uext2.cdisk` to any work directory and rename it to `db2uext2.c`. Modify the definition variables to fit your environment. Refer to Example 5-1.

Example 5-1 USEREXIT definition to archive log to disk

```
#define ARCHIVE_PATH "/db2archive/db2inst1/" /* path must end with a slash */
#define RETRIEVE_PATH "/db2archive/db2inst1/" /* path must end with a slash */
#define AUDIT_ACTIVE 1 /* enable audit trail logging */
#define ERROR_ACTIVE 1 /* enable error trail logging */
#define AUDIT_ERROR_PATH "/db2home/db2inst1/" /* path must end with a slash */
#define AUDIT_ERROR_ATTR "a" /* append to text file */
#define BUFFER_SIZE 32 /* # of 4K pages for output buffer */
```

If the variable `ARCHIVE_PATH` is set to `/db2archive/db2inst1/` then userexit archives the log files to `/db2archive/db2inst1/ITSODB/NODE000x/`.

The variable `RETRIEVE_PATH` points to a filesystem where the userexit has to place the retrieved archive logs in case of a log apply is performed.

To archive logs to TSM storage, copy the sample userexit `db2uext2.ctsm` from `$INSTHOME/sqllib/samples/c` to any work directory and rename it to `db2uext2.c`. Modify `AUDIT_ERROR_PATH` variable to set the error message file; see Example 5-2.

Example 5-2 USEREXIT definition to archive log to TSM

```
#define BUFFER_SIZE 32768 /* transmit or receive the log */
#define AUDIT_ACTIVE 1 /* enable audit trail logging */
#define ERROR_ACTIVE 1 /* enable error trail logging */
#define AUDIT_ERROR_PATH "/db2home/db2inst1/uexttsm/" /* path must end with */
#define AUDIT_ERROR_ATTR "a" /* append to text file */
#define DSMI_DIR NULL /* use value from the environment */
#define DSMI_CONFIG NULL /* use value from the environment */
#define DSMI_LOG NULL /* use value from the environment */
#define MGMT_CLASS NULL /* use default management class */
```

Note: The subdirectory */db2archive/db2inst1/ITSODB/NODE000x* must exist before USEREXIT is turned on.

The variable `AUDIT_ERROR_PATH` sets the path where the userexit writes its messages file.

Now we have to compile this sample userexit as root user.

For archiving to disk:

```
cc -D_INCLUDE_POSIX_SOURCE db2uext2.c -o db2uext2
```

For archiving to TSM:

```
cc -D_INCLUDE_POSIX_SOURCE db2uext2.c -o db2uext2  
-I/opt/tivoli/tsm/client/api/bin/sample  
/opt/tivoli/tsm/client/api/bin/libApiDS.so -lApiDS -lpthread -lcrypt -ldl
```

Copy the output file *db2uext2* to *\$INSTHOME/sqllib/adm*.

Turn on log retain and userexit by updating database configuration on each partition.

```
db2 update db cfg for itsodb using logretain on userexit on
```

The database is now in a backup pending state. Do a full offline backup from every partition, starting with the catalog partition. For example,

```
db2_a11 '<<<+0<db2 backup db itsodb to /db2backup'
```

```
db2_a11 '<<<-0<db2 backup db itsodb to /db2backup'
```

The database is now enabled to take database backups in online mode as well as to backup individual tablespaces. To verify this, use the following command:

```
db2 get db cfg for itsodb | grep enabled
```

And, look at:

```
Log retain for recovery enabled          (LOGRETAIN) = RECOVERY  
User exit for logging enabled           (USEREXIT) = ON
```

To verify the customized USEREXIT, enter the following command:

```
db2 archive log for itsodb
```

The USEREXIT writes all output data to */db2home/db2inst1/ARCHIVE.LOG* as specified in the `audit_error_path`.

Example 5-3 Output of USEREXIT program

Time Started: Fri Oct 25 14:26:45 2002

Parameter Count: 8
Parameters Passed:
Database name: ITSODB
Logfile name: S0000000.LOG
Logfile path: /db2log/db2inst1/itsodb/NODE0000/
Node number: NODE0000
Operating system: Linux
Release: SQL08010
Request: ARCHIVE
System Action: ARCHIVE from /db2log/db2inst1/itsodb/NODE0000/ file
S0000000.LOG to /database/db2inst1/archlog1/ITSODB
Media Type: disk
User Exit RC: 0
Time Completed: Fri Oct 25 14:26:45 2002

Commands such as **backup** or **restore** writes detailed backup/restore information in the Recovery History File. In a multi-partitioned database, this file exists in every partition. To see the entries in the recovery history file, use the command **db2 list history**. Sample output is shown in Example 5-4.

Example 5-4 Backup history

db2inst1@udblnx02:~> db2 list history backup all for itsodb

List History File for itsodb

Number of matching file entries = 14

Op	Obj	Timestamp+Sequence	Type	Dev	Earliest Log	Current Log	Backup ID
----	-----	--------------------	------	-----	--------------	-------------	-----------

B	D	20021028162533001	F	D	S0000008.LOG	S0000008.LOG	
---	---	-------------------	---	---	--------------	--------------	--

Contains 11 tablespace(s):

00001 SYSCATSPACE
00002 USERSPACE1
00003 TEST01
00004 CUSTOMER_TSP
00005 LINEITEM_TSP
00006 NATION_TSP
00007 ORDERS_TSP
00008 PART_TSP
00009 PARTSUPP_TSP
00010 REGION_TSP
00011 SUPPLIER_TSP

```

-----
Comment: DB2 BACKUP ITSODB OFFLINE
Start Time: 20021028162533
End Time: 20021028162555
-----

```

```
00006 Location: /db2home
```

5.1.2 Enable incremental backup

DB2 supports incremental and delta backups which is a copy of all data pages that have been updated since the last backup. To enable incremental or delta backup, you need to set the TRACKMOD database configuration parameter to ON, as shown:

```
db2 update db cfg for itsodb using TRACKMOD on
```

In a multi-partitioned environment you have to do this on each database partition. Once the TRACKMOD is set to ON, you need to do a full database backup once before starting incremental/delta backup. This mode only becomes active after a full backup of the database has been performed.

5.1.3 Enable usage of TSM

DB2 supports database backups and archive log backups using TSM storage. The TSM system consists of TSM server and TSM client. Our test was based on the following configuration:

- ▶ Server: WIN 2000 server with TSM server version 4.2.1
- ▶ Client: Linux SuSE V8.1 with TSM client version 4.2.3

Each host has to be registered on a TSM server.

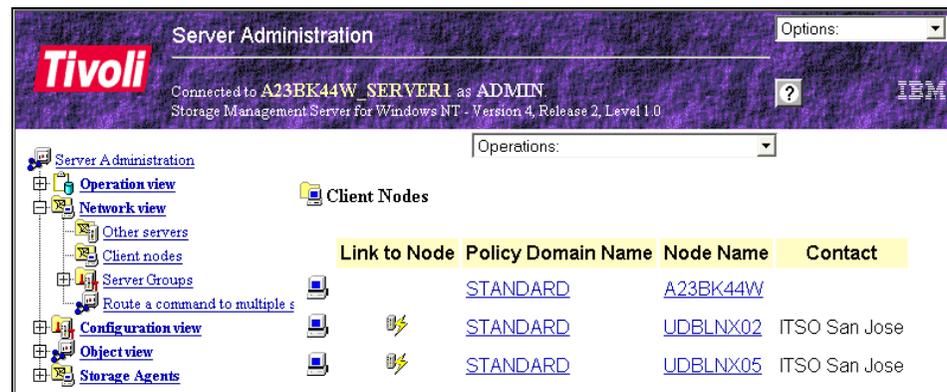


Figure 5-4 Registered clients on TSM server

On the TSM server, Tivoli needs certain storage pools to store 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 USEREXIT program and a backup storage pool (BACKUPPOOL) is used to store all DB2 backup files (Figure 5-5).

The screenshot shows the Tivoli Server Administration window. The title bar reads "Server Administration" and "Options:". Below the title bar, it says "Connected to A23BK44W_SERVER1 as ADMIN" and "Storage Management Server for Windows NT - Version 4, Release 2, Level 1.0". The main area is titled "Operation Results" and contains a table with the following data:

Storage Pool Name	Device Class Name	Estimated Capacity (MB)	Pct Util	Pct Migr	High Mig Pct	Low Mig Pct
ARCHIVEPOOL	DISK	100.0	0.1	0.1	90	70
BACKUPPOOL	DISK	800.0	46.0	46.0	90	70
DISKPOOL	DISK	0.0	0.0	0.0	90	70
FILPOOL1	FILEDEV1	0.0	0.0	0.0	90	70
SPACEMGPOOL	DISK	0.0	0.0	0.0	90	70

Figure 5-5 TSM List of storage pools

After installing the TSM client on each Linux server, we have to configure DB2 so that DB2 can communicate with TSM. Add `$INSTHOME/sql/lib/userprofile` to the path where TSM is installed:

Example 5-5 TSM definition in file userprofile

```
export DSM_CONFIG=/opt/tivoli/tsm/client/api/bin/dsm.opt
export DSM_DIR=/opt/tivoli/tsm/client/api/bin
```

The TSM client must be configured as well. Define which TSM server is to be used, in the file:

`/opt/tivoli/tsm/client/api/bin/dsm.sys`

Example 5-6 shows the file in our test environment.

Example 5-6 dsm.sys configuration file on Linux client

```
db2inst1@udb1nx02:/opt/tivoli/tsm/client/api/bin> more dsm.sys
```

```
SERvername A23BK44W_SERVER1
COMMmethod TCPip
TCPPort 1500
TCPServeraddress 9.1.39.125
PasswordAccess generate
```

The next step is to set a permanent TSM password. Change to the directory \$INSTHOME/sql1lib/adsm as root user and issue following command (Example 5-7):

```
./dsmapiw
```

Example 5-7 Set password for TSM

```
udblnx02:/db2home/db2inst1/sql1lib/adsm # ./dsmapiw
```

```
*****  
* Tivoli Storage Manager *  
* API Version = 4.2.3 *  
*****
```

```
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** command to manage the backup files in TSM. You can view and delete DB2 backup files, and bring backup files from TSM storage to the database server. Example 5-8 shows a simple query using db2adutl.

Example 5-8 db2adutl query sample

```
db2inst1@udblnx02:~> db2adutl query full
```

```
Query for database ITSODB
```

```
Retrieving FULL DATABASE BACKUP information.
```

```
1 Time: 20021104103617 Oldest log: S0000043.LOG DB Partition Number: 1 Sessions: 1  
2 Time: 20021104103534 Oldest log: S0000046.LOG DB Partition Number: 0 Sessions: 1  
3 Time: 20021104103044 Oldest log: S0000045.LOG DB Partition Number: 0 Sessions: 1
```

```
db2inst1@udblnx02:~> db2adutl query logs
```

```
Query for database ITSODB
```

```
Retrieving LOG ARCHIVE information.
```

```
Log file: S0000045.LOG, DB Partition Number: 1, Taken at: 2002-11-04-11.59.11  
Log file: S0000046.LOG, DB Partition Number: 1, Taken at: 2002-11-04-11.59.11  
Log file: S0000050.LOG, DB Partition Number: 0, Taken at: 2002-11-04-11.59.11  
Log file: S0000051.LOG, DB Partition Number: 0, Taken at: 2002-11-04-11.59.14
```

Note: A complete description for using db2adutl is written in the *IBM DB2 UDB Command Reference V8*, SC09-4828.

5.1.4 Backup utility

We have completed all the steps, as described in 5.1.1, “Enable roll forward and log archiving” on page 151, we now discuss how to take database backups.

- ▶ **Full offline:** A full offline database backup allows you to restore the database image without applying any log records. Example 5-9 shows the offline backup command.

Example 5-9 Full offline backup to disk

```
db2inst1@udblnx02:/db2backup> db2 backup db itsodb to /db2backup
```

Backup successful. The timestamp for this backup image is : 20021029163312

```
udblnx02: db2 backup db itsodb to /db2backup completed ok
```

- ▶ **Full online and incremental:** An incremental backup saves backup time and resources. During online backup, the database is still accessible for applications. Example 5-10 provides a sample of a full online backup.

Example 5-10 Incremental backup to disk

```
db2inst1@udblnx02:~> db2 backup db itsodb online incremental to /db2backup
```

Backup successful. The timestamp for this backup image is : 20021029165004

```
udblnx02: db2 backup db itsodb online incremental to /db2backup completed ok
```

- ▶ **Delta backup:** Example 5-11 shows the delta database backup after taking a full database backup.

Example 5-11 Delta backup

```
db2inst1@udblnx02:> db2 backup db itsodb online incremental delta to /db2backup
```

Backup successful. The timestamp for this backup image is : 20021029165624

```
udblnx02: db2 backup db itsodb online incremental to /db2backup completed ok
```

5.1.5 Backup database with Control Center

The backup utility is fully integrated within the DB2 Control Center (Figure 5-6).

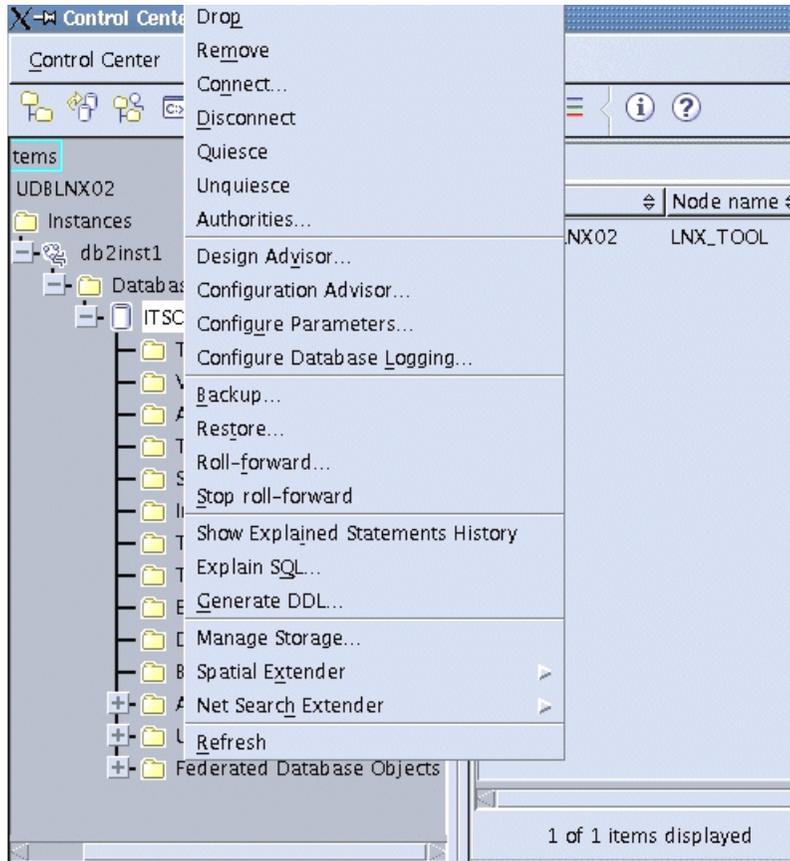
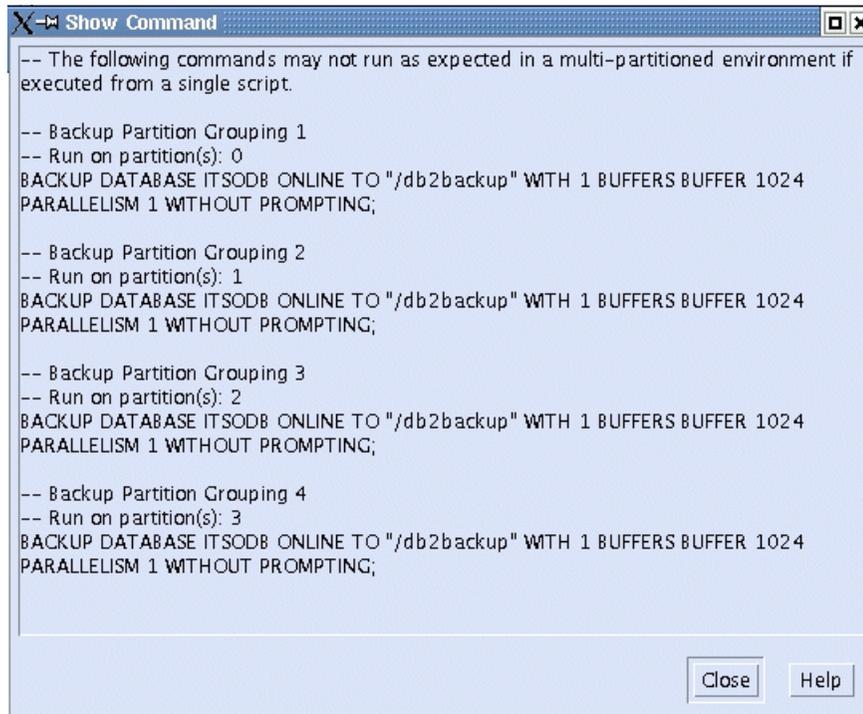


Figure 5-6 Backup in Control Center

Go through all of the windows and fill out all needed parameters. At the end you can list the generated backup statements, as shown in Figure 5-7. 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 5-8



```
-- The following commands may not run as expected in a multi-partitioned environment if
executed from a single script.

-- Backup Partition Grouping 1
-- Run on partition(s): 0
BACKUP DATABASE ITSODB ONLINE TO "/db2backup" WITH 1 BUFFERS BUFFER 1024
PARALLELISM 1 WITHOUT PROMPTING;

-- Backup Partition Grouping 2
-- Run on partition(s): 1
BACKUP DATABASE ITSODB ONLINE TO "/db2backup" WITH 1 BUFFERS BUFFER 1024
PARALLELISM 1 WITHOUT PROMPTING;

-- Backup Partition Grouping 3
-- Run on partition(s): 2
BACKUP DATABASE ITSODB ONLINE TO "/db2backup" WITH 1 BUFFERS BUFFER 1024
PARALLELISM 1 WITHOUT PROMPTING;

-- Backup Partition Grouping 4
-- Run on partition(s): 3
BACKUP DATABASE ITSODB ONLINE TO "/db2backup" WITH 1 BUFFERS BUFFER 1024
PARALLELISM 1 WITHOUT PROMPTING;
```

Close Help

Figure 5-7 Generated backup statements

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.

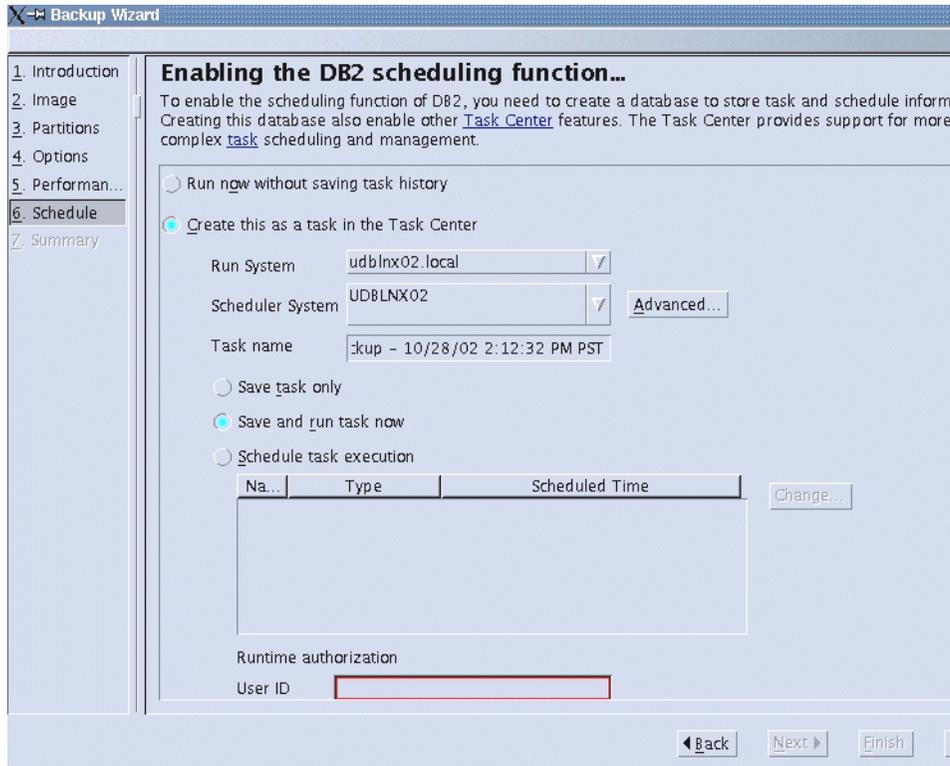


Figure 5-8 Scheduling backup scripts

5.1.6 Backup database in DB2 command line processor

In Example 5-12 we show you a script to backup our sample database ITSODB, which is partitioned. Call this script with the db2_all command:

```
db2_all '/db2home/db2inst1/database_backup.ksh db2inst1 itsodb'
```

Example 5-12 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/sql1lib/db2profile ]]
then

    echo "File not found: " /$DBHOME/sql1lib/db2profile
    echo "please check host- and instancename"
    echo "      ***** script abnormally ended *****"
"
    exit 1

fi

. /$DBHOME/sql1lib/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 failed after 3 trials
$szDate " | tee -a /$DBHOME/$DB_ALIAS.log
echo ""
exit 8

```

Modify this sample so that it fits your environment and requirements.

Note: For a detailed explanation of backing up databases, refer to Chapter 2 “Database Backup” in *IBM DB2 UDB Data Recovery and High Availability Guide and Reference V8*, SC09-4831.

5.1.7 DB2 database recovery utility

A full or partial database restore may be required in many situations. The recovery method that is available depends on your backup and DB2 log handling. DB2 supports the following recovery methods:

► Crash recovery

DB2 uses this method to set a database to a consistent state after an unexpected failure. Crash recovery will rollback all the uncommitted changes. It can be forced with the **db2 restart database** command.

► Full database restore offline

This will restore the entire database to the point in time when the database backup was taken, or in conjunction with roll forward to a specific point in time. It is the only restore method to recover a database with circular logging. A full database restore example is:

```
db2 restore db itsodb from /db2backup taken at 200210291525
```

► Full database restore with incremental

Once your database is enabled for recovery, that is, log retain on, and for incremental backup, that is, trackmod is on, you can restore the entire or partial database to a point in time where it is needed.

```
db2 restore db itsodb online incremental auto from /db2backup taken at 200210291525
```

► Tablespace restore

If you have enabled roll forward recovery, it is possible to restore any tablespace to a point in time or to the end of log.

```
db2 restore db itsodb "tablespace(nation_tsp)" online incremental auto from /db2backup taken at 200210291525
```

► Rollforward recovery

This process applies changes to the tablespaces from the archive or active logs. This command must be issued from the catalog partition.

```
db2 'rollforward db itsodb to end of logs on dbpartitionnum(0,1) and complete online'
```

- ▶ Recovering a dropped table

DB2 supports the recovery of a dropped table by using tablespace level restore, if the tablespace where the table resides is enabled for dropped table recovery. This can be done during tablespace creation or by altering an existing table using the DROPPED TABLE RECOVERY ON option.

Note: For a detailed explanation of backing up databases, refer to Chapter 2 “Database Backup” in *IBM DB2 UDB Data Recovery and High Availability Guide and Reference V8*, SC09-4831.

5.1.8 Backup and restore sample scenario

The following scenario is applied to our four partition database ITSODB:

- ▶ After creating and loading many tables into the database, we take a full offline backup.
- ▶ Roll forward, userexit, and trackmod are enabled.
- ▶ Normal usage of the database starts which includes updates, delete, and so forth against tables.
- ▶ Online incremental backups are regularly scheduled.

We perform the following activities in sequence to demonstrate the database restore functions:

1. Full offline database backup of ITSODB
2. Updates against table CUSTOMER
3. Full online incremental database backup of ITSODB
4. Updates against table CUSTOMER
5. Full online incremental database backup of ITSODB
6. Updates against table CUSTOMER
7. Table CUSTOMER is no longer accessible after a system outage of host udblnx05, where database partitions 2 and 3 are located.

To restore our table to the point in time right before it was corrupted, you have to find out when the last backup was taken. To do this, we issued the following command on each partition (which is partition 2 and 3 in our example):

```
db2 list history backup all for itsodb
```

Example 5-13 List database backup history output

```
Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log Backup ID
-----
B D 20021029165624001 0 D S0000020.LOG S0000021.LOG
-----
Contains 9 tablespace(s):

00001 USERSPACE1
00002 CUSTOMER_TSP
00003 LINEITEM_TSP
00004 NATION_TSP
00005 ORDERS_TSP
00006 PART_TSP
00007 PARTSUPP_TSP
00008 REGION_TSP
00009 SUPPLIER_TSP
-----
Comment: DB2 BACKUP ITSODB ONLINE
Start Time: 20021029165624
End Time: 20021029165635
-----
00003 Location: /db2backup
```

Next, get the timestamp from the output (Example 5-13) and issue the following command:

```
db2 restore db itsodb "tablespace(customer_tsp)" online incremental auto from /db2backup taken at 20021029165624
```

Repeat these steps for each partition where the tablespace needs to be restored. After restoring the tablespace, the tablespace will be in roll forward pending state on the partition where you have performed the restore; see Example 5-14.

Example 5-14 List tablespaces

```
Tablespace ID          = 4
Name                   = CUSTOMER_TSP
Type                   = System managed space
Contents               = Any data
State                  = 0x0080
Detailed explanation:
Roll forward pending
```

Issue the **roll forward** command for the database on partitions 2 and 3. See Example 5-15 for the command syntax and output.

Example 5-15 Rollforward database for partition 2 and 3

```
db2 'rollforward db itsodb to end of logs on dbpartitionnum(2,3) and complete
tablespace(customer_tsp) online'
```

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
-----
-----
                2 not pending                -
2002-10-30-01.20.36.000000
                3 not pending                -
2002-10-30-01.20.36.000000
```

```
DB20000I The ROLLFORWARD command completed successfully.
```

After restoring databases or tablespaces, you should consider taking a new backup to get a new consistent image copy of your database.

5.1.9 Recovery of a dropped table

To recover a dropped table that was located within a tablespace set for dropped table recovery, follow this procedure:

1. Restore the tablespace where the table was stored before the table was dropped from a database or tablespace level backup. In our example, the dropped table was in tablespace *nation_tsp* which resides in partition 1, 2, and 3.

Example 5-16 Tablespace restore for all partition

```
db2 restore db itsodb "tablespace(nation_tsp)" online incremental auto from
/db2backup taken at 20021031151430
```

```
DB20000I The RESTORE DATABASE command completed successfully.
```

```
db2inst1@udb1nx02:~/scripts> db2_all '<<+1<db2 restore db itsodb
"tablespace(nation_tsp)" online incremental auto from /db2backup taken at
20021031151455'
```

```
DB20000I The RESTORE DATABASE command completed successfully.
```

```
udb1nx02: db2 restore db itsodb "tablespace(nation_tsp)" online incremental
auto from /db2backup taken at 20021031151455 completed ok
```

```
db2inst1@udblnx02:~/scripts> db2_all '<<+
db2inst1@udblnx02:~/scripts> db2_all '<<+2<db2 restore db itsodb
"tablespace(nation_tsp)" online incremental auto from /db2backup taken at
20021031152514'
```

```
DB20000I The RESTORE DATABASE command completed successfully.
udblnx05: db2 restore db itsodb "tablespace(nation_tsp)" online incremental
auto from /db2backup taken at 20021031152514 completed ok
db2inst1@udblnx02:~/scripts> db2_all '<<+3<db2 restore db itsodb
"tablespace(nation_tsp)" online incremental auto from /db2backup taken at
20021031152528'
```

```
DB20000I The RESTORE DATABASE command completed successfully.
udblnx05: db2 restore db itsodb "tablespace(nation_tsp)" online incremental
auto from /db2backup taken at 20021031152528 completed ok
```

-
2. Create an export directory in a shared file system or create the directory on every partition.

```
db2_all 'mkdir /db2backup/rest_table'
```

3. Determine the table ID by looking in the history file (Example 5-17). In a multi-partitioned environment this has to be done on the catalog partition.

Example 5-17 List history file

```
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 20021031154206
000000000000460500050002
-----
"DB2INST1"."NATION" resides in 1 tablespace(s):

00001 NATION_TSP
-----
Comment: DROP TABLE
Start Time: 20021031154206
End Time: 20021031154206
-----
00001
```

```
DDL: CREATE TABLE "DB2INST1"."NATION" ( "N_NATIONKEY" INTEGER NOT NULL ,
"N_NAME" CHAR(25) NOT NULL , "N_REGIONKEY" INTEGER NOT NULL , "N_COMMENT"
VARCHAR(152) ) PARTITIONING KEY ("N_NATIONKEY") IN "NATION_TSP" ;
```

- Rollforward the database with the correct table ID. The table ID is listed in the backup ID column (Example 5-18).

Example 5-18 Rollforward with table ID

```
db2 'rollforward db itsodb to end of logs on all dbpartitionnums and complete
tablespace(nation_tsp) online recover dropped table 00000000000460500050002
to /db2backup/rest_table'
```

```

                                Rollforward Status

Input database alias              = itsodb
Number of nodes have returned status = 4

Node number  Rollforward          Next log          Log files
processed    Last committed transaction
              status              to be read
-----
-----
              0 not pending              -
2002-10-31-23.52.11.000000
              1 not pending              -
2002-10-31-23.52.11.000000
              2 not pending              -
2002-10-31-23.52.11.000000
              3 not pending              -
2002-10-31-23.52.11.000000
```

```
DB20000I The ROLLFORWARD command completed successfully.
```

- If rollforward has finished, DB2 has restored the table data to the export directory in the subdirectory NODEnnnn. Now get DDL from the history file to create the table and load all data from the export file into the table. See the history output in Example 5-17.

Example 5-19 Restored table data as loadable file

```
db2inst1@udblnx05:/db2backup/rest_table/NODE0002> more data
23,"Italien"          ",1,"IBM Europe"
5,"ETHIOPIA"         ",0,"fluffily ruthless"
7,"GERMANY"          ",3,"blithely ironic foxes grow
5,"MOROCCO"          ",0,"ideas according to the fluffily final"
16,"MOZAMBIQUE"      ",0,"ironic courts wake fluffily even, bold"
```

5.2 Table/index reorganization and statistics collection

The tables and/or indexes will become fragmented and the physical data pages for tables and/or indexes will become discontinuous when considerable changes have been made to the table. Furthermore, 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. The 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. One of the tasks you need to do for better performance is to reorganize the tables and indexes.

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 can take advantage of these statistics when determining the access path to the data to gain better performance for subsequent database operations.

5.2.1 Table reorganization

DB2 UDB provides two methods for reorganizing tables:

► **Online**

Online 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. Online table reorganization is only allowed on tables with type-2 indexes and without extended indexes.

► **Offline**

The offline method provides faster table reorganization, especially if you do not need to reorganize LOB or LONG data. LOBS and LONG data are no longer 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.

Both online and offline reorganizations have been enhanced to improve support for multi-partition databases. You can reorganize a single partition, a set of partitions, or all partitions.

Note: From the point of view of command syntax for REORG TABLE, the methods of reorganizing tables provided by DB2 include classic and in-place (online). In general, classic table reorganization is faster, but should 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 in-place 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 more details regarding classic and in-place table reorganization, please refer to *DB2 Administration: Performance V8*, SC09-4821.

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 the online book, *DB2 Administration: Performance V8*.

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.

5.2.2 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 command REORG INDEXES is new for DB2 Version 8. It has the ability to read and update a table and its existing indexes during an index reorganization.

During online index reorganization, the entire index objects (that is, all indexes on the table) are rebuilt. A shadow copy of the index object is made, leaving the original indexes and the table available for read and write access. Any concurrent transactions that update the table are logged. Once the logged table changes have been forward-fitted and the new index (the shadow copy) is ready, the new

index is made available. While the new index is being made available all access to the table is prohibited.

You can reduce the need of index reorganization via a variety of means, for example, setting PCTFREE for indexes, or enabling the online index defragmentation by setting MINPCTUSED (it 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.

The default behavior of the REORG INDEXES command is ALLOW NO ACCESS, which places an exclusive lock on the table during the reorganization process. But you can also specify ALLOW READ ACCESS or ALLOW WRITE ACCESS to permit other transactions to read from or update the table.

5.2.3 Statistics information collection

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.

5.2.4 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 has been executed, thereby taking advantage of the new statistics. This is especially important for static SQL. Without the rebind, the static SQL application won't 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.

5.2.5 Examples

This section provides examples demonstrating the usage of the utilities mentioned in this section: REORG TABLE, REORG INDEXES, REORGCHK and RUNSTATS. We have created a table named "EMP" across three partitions and populated with 100,000 rows of data. An unique index named "U_EMP_ENO" is associated with this table.

Here are more details for the table EMP (Figure 5-9):

```
db2inst1@UDBLN08:~> db2 "describe table emp"

Column          Type      Type
name            schema   name
-----
ENO              SYSIBM   INTEGER          4      0 Yes
LASTNAME        SYSIBM   VARCHAR          30      0 Yes
HIREDATE        SYSIBM   DATE              4      0 Yes
SALARY          SYSIBM   INTEGER          4      0 Yes

  4 record(s) selected.

db2inst1@UDBLN08:~> db2 "describe indexes for table emp"

Index      Index      Unique      Number of
schema     name       rule         columns
-----
DB2INST1   U_EMP_ENO  U              1

  1 record(s) selected.

db2inst1@UDBLN08:~> db2 "select dbpartitionnum(eno) PART_NO,count(*) PART_ROWS from emp
group by dbpartitionnum(eno)"

PART_NO    PART_ROWS
-----
          0      33269
          1      33338
          2      33393

  3 record(s) selected.
```

Figure 5-9 More details for sample table EMP

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 table “EMP”.
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 “EMP”.

```
db2inst1@UDBLN08:~/test/scripts> db2 "runstats on table db2inst1.emp with distribution on all columns and detailed indexes all allow read access"
DB20000I The RUNSTATS command completed successfully.
db2inst1@UDBLN08:~/test/scripts> db2 "reorgchk current statistics on table db2inst1.emp"

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
-----
DB2INST1 EMP                99807  0 1020 1020    - 4092087  0 98 100 ---

Index statistics:

F4: CLUSTERRATIO or normalized CLUSTERFACTOR > 80
F5: 100 * (KEYS * (ISIZE + 9) + (CARD - KEYS) * 5) / ((NLEAF - NUM EMPTY LEAFS) * INDEXPAGESIZE) > 50
F6: (100 - PCTFREE) * ((INDEXPAGESIZE - 96) / (ISIZE + 12)) ** (NLEVELS - 2) * (INDEXPAGESIZE - 96) / (KEYS * (ISIZE + 9) + (CARD - KEYS) * 5) < 100
F7: 100 * (NUMRIDS DELETED / (NUMRIDS DELETED + CARD)) < 20
F8: 100 * (NUM EMPTY LEAFS / NLEAF) < 20

SCHEMA  NAME                CARD  LEAF  ELEAF  LVLS  ISIZE  NDEL  KEYS  F4  F5  F6  F7  F8 REORG
-----
Table: DB2INST1.EMP
DB2INST1 U_EMP_ENO          99807  390    0    2    6    0 99807 100 93  0  0  0 -----
```

Figure 5-10 Using RUNSTATS and REORGCHK to gather and check statistics

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 “EMP”. You can also use the SELECT statement to get this information from system

catalog views directly, for example, SYSSTAT.TABLES and SYSSTAT.INDEXES, and so on.

Attention: The RUNSTATS utility is run against one partition only due to performance considerations, so some global statistics information may be inaccurate here. For example, the CARD should be 100,000 in reality, but here it says 99,807. Another command INSPECT CHECK can be used to generate accurate information about the pages occupied by the table and index objects.

The RUNSTATS utility is also available in the DB2 Control Center. If you want to use the Schedule function provided by DB2, please use the DB2 Control Center instead. Figure 5-11 and Figure 5-12 for RUNSTATS and Scheduler are provided for your reference.

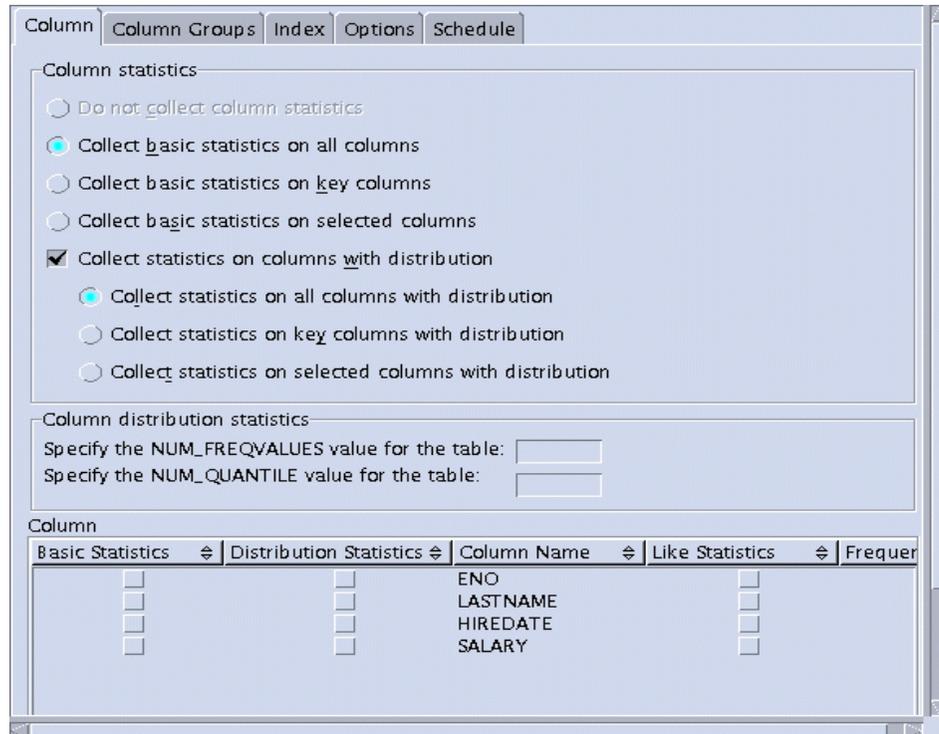


Figure 5-11 RUNSTATS in DB2 Control Center on Linux

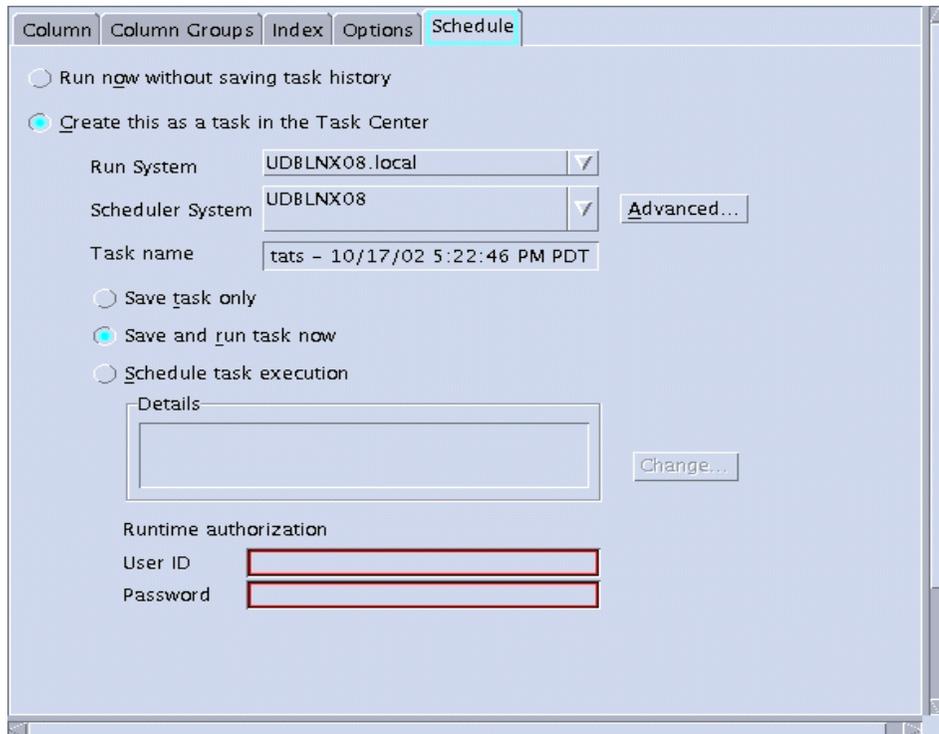


Figure 5-12 Schedule in DB2 Control Center

2. Delete rows from the table EMP.

Example 5-20 Delete rows from sample table EMP

```

db2inst1@UDBLNx08:~/test/scripts> cat empdel.sql
delete from emp where eno>=20000 and eno<=30000;
delete from emp where eno>=40000 and eno<=50000;
delete from emp where eno>=60000 and eno<=70000;
delete from emp where eno>=80000 ;
db2inst1@UDBLNx08:~/test/scripts> db2 -tvf empdel.sql
...
DB20000I The SQL command completed successfully.

```

3. Use REORGCHK with UPDATE STATISTICS option to obtain the new statistics and check the table again.

```

db2inst1@UDBLNX08:~/test> db2 "reorgchk update statistics on table db2inst1.emp"

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
-----
DB2INST1    EMP                49701   0  522  1020   - 2037741  0  49  51 -**

Index statistics:

F4: CLUSTERRATIO or normalized CLUSTERFACTOR > 80
F5: 100 * (KEYS * (ISIZE + 9) + (CARD - KEYS) * 5) / ((NLEAF - NUM EMPTY LEAFS) * INDEXPAGESIZE) > 50
F6: (100 - PCTFREE) * ((INDEXPAGESIZE - 96) / (ISIZE + 12)) ** (NLEVELS - 2) * (INDEXPAGESIZE - 96) / (KEYS * (ISIZE + 9) + (CARD - KEYS) * 5) < 100
F7: 100 * (NUMRIDS DELETED / (NUMRIDS DELETED + CARD)) < 20
F8: 100 * (NUM EMPTY LEAFS / NLEAF) < 20

SCHEMA      NAME                CARD  LEAF  ELEAF  LVLS  ISIZE  NDEL   KEYS  F4  F5  F6  F7  F8 REORG
-----
Table: DB2INST1.EMP
DB2INST1 U_EMP_ENO      49701  213    9     2     6  2151  49701  100  89   0   4   0  ----

```

Figure 5-13 Using REORGCHK to update statistics for table

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 output from REORGCHK, please refer to the *IBM DB2 UDB Command Reference V8*, SC09-4828.

4. Reorganize the index and table and monitor the procedure of reorganization via the DB2 command GET SNAPSHOT.

In the “Index statistics” part from the output shown in Figure 5-13, we can see that both ELEAF and NDEL are greater than 0. We can use the following commands to physically remove the leaf pages occupied by pseudo delete (for more information about pseudo delete and type 2 index, please refer to *DB2 Administration: Performance V8*).

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 for deletion. 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.

```
db2inst1@UDBLN08:~/test> db2 "reorg indexes all for table emp allow write access cleanup only all
on all dbpartitionnums"
DB20000I The REORG command completed successfully.
db2inst1@UDBLN08:~/test> db2 "reorgchk update statistics on table db2inst1.emp"

Doing RUNSTATS ....

Table statistics:
...

Index statistics:

SCHEMA   NAME                CARD  LEAF  ELEAF  LVLS  ISIZE  NDEL  KEYS  F4  F5  F6  F7  F8  REORG
-----
Table: DB2INST1.EMP
DB2INST1 U_EMP_ENO      49701  201    0     2     6     0  49701 100  90   0   0   0  -----
```

Figure 5-14 REORG INDEXES and REORGCHK to update statistics

Then we find that ELEAF and NDEL goes back to 0 as the pseudo deleted pages are physically removed.

Now we reorganize the table as recommended. In the first example (Example 5-21) we use classic table reorganization method with ALLOW READ ACCESS option.

Example 5-21 REORG TABLE with ALLOW READ ACCESS option

```
db2inst1@UDBLN08:~/test> db2 "reorg table emp index u_emp_eno allow read
access on all dbpartitionnums"
DB20000I The REORG command completed successfully.
```

Then we use the in-place table reorganization method with ALLOW WRITE ACCESS option. Before issuing the command of REORG TABLE, to observe the process of REORG TABLE, you can open another window and keep “GET SNAPSHOT” command running with specified interval. The length of interval depends on the data volume you are operating against and your system’s

capability. You can do it with a shell script or manually. In our example (Example 5-22), a shell script named “run” is used to run the “GET SNAPSHOT” command at an interval of 2 seconds.

Example 5-22 Start a monitor program to observe table reorganization procedure

```
db2inst1@UDBLN08:~/test> run
Usage: run <command_name> [ <sleeping_interval> [ < result_file > ] ]
db2inst1@UDBLN08:~/test> run "db2 get snapshot for tables on sample" 2
snap_reorgtab_write.out
Program name: "db2 get snapshot for tables on sample"
This program will keep running every 2 seconds for ever,
pls interrupt it with Ctrl+C if needed ...
```

Command "db2 get snapshot for tables on sample" is running NOW ...

Note: The output of “get snapshot” is gathered into the file “snap_reorgtab_write.out”.

Follow the startup of monitor program, now we can start in-place table reorganization by using the command shown in Example 5-23.

Example 5-23 REORG TABLE with INPLACE and ALLOW WRITE ACCESS mode

```
db2inst1@UDBLN08:~/test> db2 "reorg table emp index u_emp_eno 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 in-place table reorganization is asynchronous and may not be effective immediately. To determine if the in-place table reorganization is finished or what the current status is, you can use GET SNAPSHOT FOR TABLES ON SAMPLE (here the SAMPLE database is used in our example) to obtain the detailed information for the reorganization. Example 5-24 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.

Example 5-24 The snapshot information for table reorganization

Table Snapshot

```
First database connect timestamp    = 10-17-2002 13:46:35.708637
Last reset timestamp                =
Snapshot timestamp                  = 10-17-2002 14:54:53.136558
Database name                       = SAMPLE
Database path                       =
/db2home/db2inst1/db2inst1/NODE0000/SQL00001/
```

Input database alias = SAMPLE
Number of accessed tables = 1

Table List

Table Schema = DB2INST1
Table Name = EMP
Table Type = User

Table Reorg Information:

Node number = 0

Reorg Type =
Reclustering
Inplace Table Reorg
Allow Write Access

Reorg Index = 1
Reorg Tablespace = 2
Start Time = 10-17-2002 14:54:52.336284
Reorg Phase =
Max Phase =
Phase Start Time =
Status = Started
Current Counter = 6
Max Counter = 169
Completion = 0
End Time =

Table Snapshot

First database connect timestamp = 10-17-2002 13:46:35.708637
Last reset timestamp =
Snapshot timestamp = 10-17-2002 14:55:17.020074
Database name = SAMPLE
Database path =
/db2home/db2inst1/db2inst1/NODE0000/SQL00001/
Input database alias = SAMPLE
Number of accessed tables = 1

Table List

Table Schema = DB2INST1
Table Name = EMP
Table Type = User

Table Reorg Information:

Node number = 0

Reorg Type =
Reclustering
Inplace Table Reorg
Allow Write Access

Reorg Index = 1
Reorg Tablespace = 2
Start Time = 10-17-2002 14:54:52.336284

```

Reorg Phase      =
Max Phase       =
Phase Start Time =
Status        = Truncate
Current Counter  = 169
Max Counter     = 169
Completion      = 0
End Time        =

```

Table Snapshot

```

First database connect timestamp = 10-17-2002 13:46:35.708637
Last reset timestamp            =
Snapshot timestamp              = 10-17-2002 14:55:19.130922
Database name                    = SAMPLE
Database path                    =
/db2home/db2inst1/db2inst1/NODE0000/SQL00001/
Input database alias             = SAMPLE
Number of accessed tables        = 1

```

Table List

```

Table Schema      = DB2INST1
Table Name        = EMP
Table Type        = User
Table Reorg Information:
Node number       = 0
  Reorg Type      =
    Reclustering
    Inplace Table Reorg
    Allow Write Access
  Reorg Index     = 1
  Reorg Tablespace = 2
  Start Time      = 10-17-2002 14:54:52.336284
  Reorg Phase     =
  Max Phase       =
  Phase Start Time =
Status        = Completed
  Current Counter = 169
  Max Counter     = 169
  Completion      = 0
  End Time        = 10-17-2002 14:55:17.746016

```

If you are not willing to use the command line to execute REORG TABLE or REORG INDEXES, these commands are also available in DB2 Control Center. Figure 5-15 and Figure 5-15 are provided for your reference.



Figure 5-15 REORG TABLE in DB2 Control Center

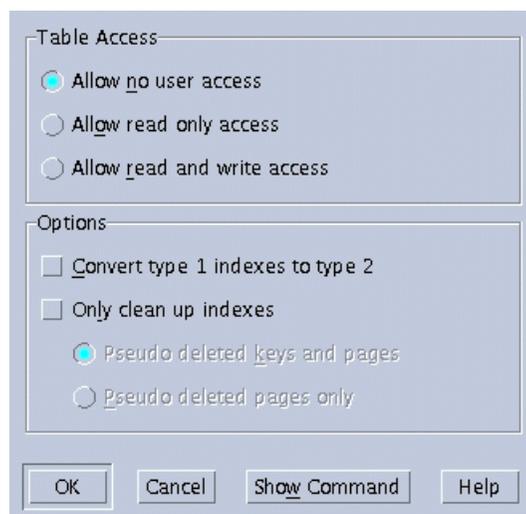


Figure 5-16 REORG INDEXES in DB2 Control Center

- Use the REORGCHK with UPDATE STATISTICS option to obtain the new statistics after the index and table reorganization.

```
db2inst1@UDBLN08:~/test> db2 "reorgchk update statistics on table db2inst1.emp"

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
-----
DB2INST1 EMP                49701   0  510  510    - 2037741   0  98 100 ---

Index statistics:

F4: CLUSTERRATIO or normalized CLUSTERFACTOR > 80
F5: 100 * (KEYS * (ISIZE + 9) + (CARD - KEYS) * 5) / ((NLEAF - NUM EMPTY LEAFS) * INDEXPAGESIZE) > 50
F6: (100 - PCTFREE) * ((INDEXPAGESIZE - 96) / (ISIZE + 12)) ** (NLEVELS - 2) * (INDEXPAGESIZE - 96) / (KEYS * (ISIZE + 9) + (CARD - KEYS) * 5) < 100
F7: 100 * (NUMRIDS DELETED / (NUMRIDS DELETED + CARD)) < 20
F8: 100 * (NUM EMPTY LEAFS / NLEAF) < 20

SCHEMA   NAME                CARD  LEAF  ELEAF  LVLS  ISIZE  NDEL   KEYS  F4  F5  F6  F7  F8 REORG
-----
Table: DB2INST1.EMP
```

Figure 5-17 Using REORGCHK to update statistics after table reorganization

Here you may find that FP (File Pages) for table and LEAF (Leaf Pages) for index are lowered. For table statistics, you may find that the recommendation part has turned back to “---”. It means no REORG is required at present.

- 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 purposes. An example of using the rebind command is shown in Example 5-25.

Example 5-25 Re-binding packages

```
db2inst1@UDBLN08:~/sampc> db2 "rebind tbread"
DB20000I The REBIND PACKAGE command completed successfully.
# db2inst1@UDBLN08:~/test> db2rbind sample -l db2rbind.log all
```

Attention: Please 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 designated package name is recommended.

5.3 Data movement via EXPORT, IMPORT, and LOAD

In this section, we discuss DB2's data movement using the EXPORT, IMPORT and LOAD utilities. We can use EXPORT to export data from database tables into files with different file formats, and also use IMPORT to write data into database tables from a source data file. Furthermore, when large volumes of data need to be moved into the database tables, the LOAD utility can also be used for better performance. In addition, we describe the basic usage and 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.

5.3.1 Export data to files from database tables

The DB2 UDB 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 an SQL 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) or DB2 Control Center (CC), or via an Application Programming Interface (API). Example 5-26 is a sample to export data from a table named EMP which is spread across three partitions by using the DB2 CLP.

Example 5-26 Export all data from sample table "EMP"

```
db2inst1@UDBLN08:~/itso/datmov> db2 "export to emp.ixf of ixf select * from
emp"
SQL3104N The Export utility is beginning to export data to file "emp.ixf".
SQL3105N The Export utility has finished exporting "100000" rows.
```

Number of rows exported: 100000

The data across all the three partitions has been exported. If you want to export data for only one partition, for example, data residing on current database partition, you can do it as in Example 5-27.

Example 5-27 Export data for current partition from sample table "EMP"

```
db2inst1@UDBLN08:~/itso/datmov> db2 "export to emppart0.del of del messages
exppart0.msg select * from emp where dbpartitionnum(eno) = current
dbpartitionnum"
```

Number of rows exported: 33269

If you want to export data in parallel across different partitions, you can use `db2_all` or `db2batch`. For more details regarding the use of `db2batch` to export data in parallel, refer to *IBM DB2 UDB Data Movement Utilities Guide and Reference V8*, SC09-4830.

You can also use the DB2 Control Center to invoke EXPORT for tables on Linux platforms. Figure 5-18 is provided for your reference. In the Control Center, 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. Also under the Messages file you can specify the messages file to be created that will contain the error, warning, and informational messages associated with the EXPORT operation. In addition, if you want to know the command associated with your choices, click **Show Command**. A pop-up window appears with detailed commands inside. Furthermore, you may use the Schedule function provided by DB2 for Export operations by choosing the Schedule tab on the window.

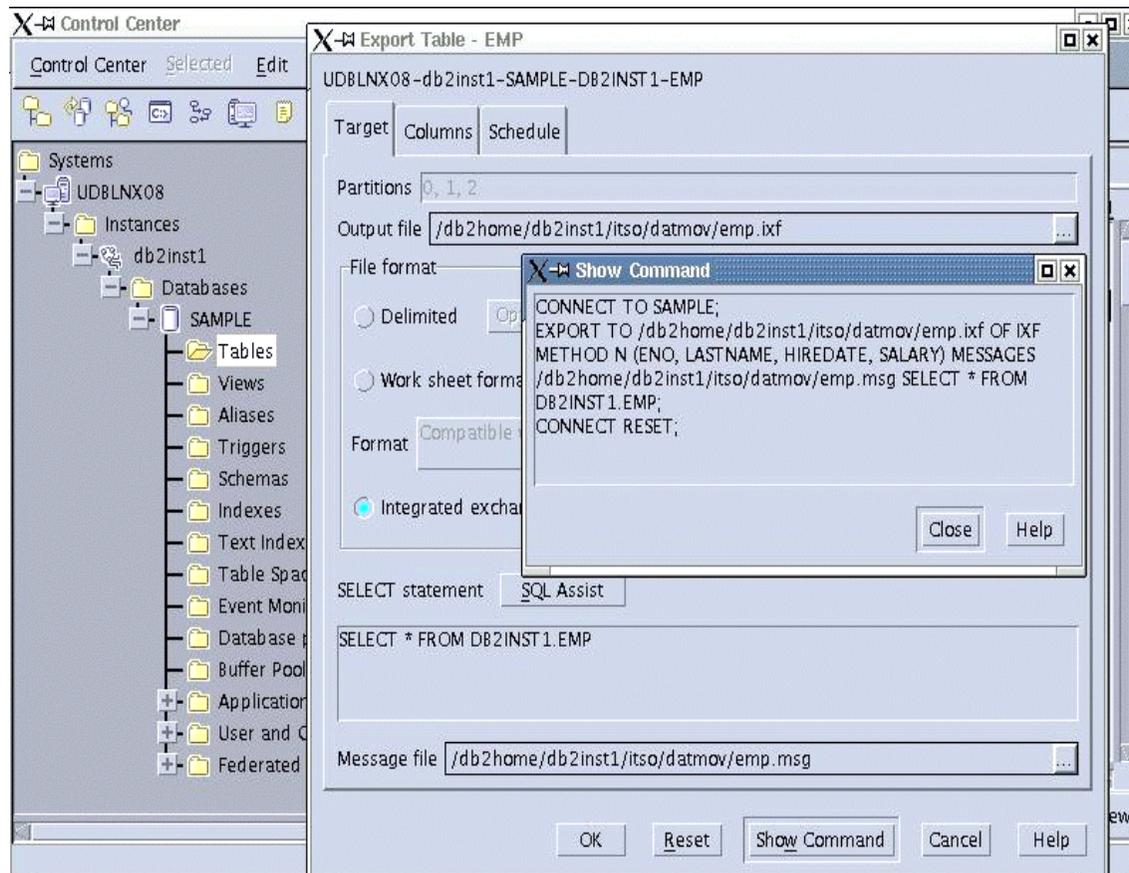


Figure 5-18 Using EXPORT in DB2 Control Center

A table can be saved by using the export utility and specifying the IXF file format. The saved table (including its indexes) can then be recreated while importing data via the import utility.

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. In this case, you should limit the amount of data selected by specifying conditions on the WHERE clause, so that the export file will fit on the target file system. You can invoke the export utility multiple times to export all of the data.

For information about using EXPORT API within your own program, and more details regarding the EXPORT utility, refer to *IBM DB2 UDB Data Movement Utilities Guide and Reference V8*, SC09-4830, and *IBM DB2 UDB Command Reference V8*, SC09-4828.

5.3.2 Import 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 receiving the imported data already contains 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 IXF (Integrated eXchange Format).

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 can be invoked through the command line processor (CLP), the Import notebook in the Control Center, or an application programming interface (API), sqluimpr. You also 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.

Example 5-28 shows the IMPORT command issued through the CLP.

Example 5-28 Using IMPORT via DB2 CLP

```
db2inst1@UDBLN08:/db2home/db2inst1> db2 "create table emp4imp like emp"
DB20000I The SQL command completed successfully.
db2inst1@UDBLN08:/db2home/db2inst1> db2 "import from ./itso/datmov/emp.ixf of
ixf commitcount 20000 messages emp4imp.msg insert into emp4imp"

Number of rows read          = 100000
Number of rows skipped       = 0
Number of rows inserted     = 100000
Number of rows updated       = 0
Number of rows rejected      = 0
Number of rows committed    = 100000
```

Within the preceding example (Example 5-28), the COMMITCOUNT parameter is used. It means that DB2 will perform a COMMIT after every n records are imported. If the import is interrupted for some reason when using the

COMMITCOUNT parameter, the committed rows will remain in the target table despite the failure of the import procedure. For such a situation, you may use the REPLACE or REPLACE_CREATE option to delete all existing data from the table by truncating the data object and inserts the imported data. Or you can also use INSERT option with RESTARTCOUNT parameter specified to avoid re-importing the rows already existed in the table. In Example 5-29, the MESSAGES parameter is used to monitor the status of import operation. You can check the generated message file which is in ASCII format.

Example 5-29 Monitor IMPORT process via message file

```
db2inst1@UDBLN08:~> tail -f emp4imp.msg
SQL3153N The T record in the PC/IXF file has name "emp.ixf", qualifier "",
and source "          ".

SQL3109N The utility is beginning to load data from file
"./itso/datmov/emp.ixf".

SQL3221W ...Begin COMMIT WORK. Input Record Count = "20000".
SQL3222W ...COMMIT of any database changes was successful.

SQL3221W ...Begin COMMIT WORK. Input Record Count = "40000".
SQL3222W ...COMMIT of any database changes was successful.

SQL3221W ...Begin COMMIT WORK. Input Record Count = "60000".
SQL3222W ...COMMIT of any database changes was successful.

SQL3221W ...Begin COMMIT WORK. Input Record Count = "80000".
SQL3222W ...COMMIT of any database changes was successful.

SQL3221W ...Begin COMMIT WORK. Input Record Count = "100000".
SQL3222W ...COMMIT of any database changes was successful.

SQL3110N The utility has completed processing. "100000" rows were read from
the input file.

SQL3221W ...Begin COMMIT WORK. Input Record Count = "100000".
SQL3222W ...COMMIT of any database changes was successful.

SQL3149N "100000" rows were processed from the input file. "100000" rows
were successfully inserted into the table. "0" rows were rejected.
```

Similar to EXPORT, we also can use the DB2 Control Center to invoke the IMPORT utility. To open the Import notebook within 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 window (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.

Figure 5-19 is a sample for using IMPORT within DB2 Control Center.

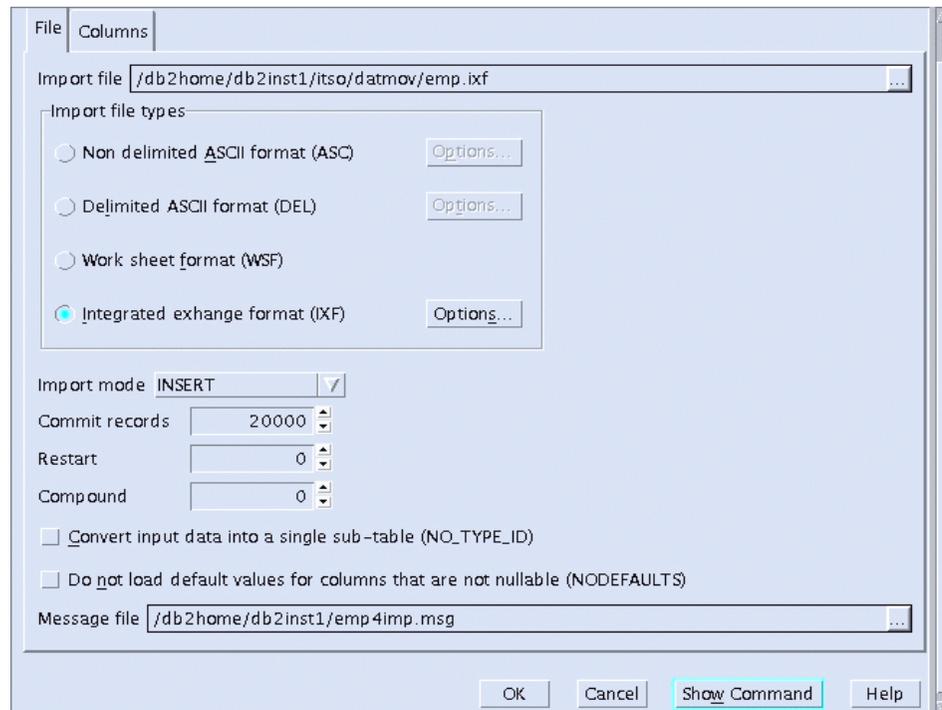


Figure 5-19 Using IMPORT within DB2 Control Center

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 5-30.

Example 5-30 Enable INSERT BUF for IMPORT in a partitioned database

```
db2inst1@UDBLN08:~> db2 connect to sample
db2inst1@UDBLN08:~> db2 bind $HOME/sql1lib/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 or identity columns, refer to *IBM DB2 UDB Data Movement Utilities Guide and Reference V8*, SC09-4830, and *IBM DB2 UDB Command Reference V8*, SC09-4828.

5.3.3 Load data into existing tables

The load utility is capable of efficiently moving large quantities of data into newly created tables or into tables that already contain data. The utility can handle most data types including large objects (LOBs) and user-defined types (UDTs). 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 key violation or a DATALINK violation are removed from the table.
- ▶ 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.

Several enhancements have been made to the load utility in Version 8. New functionality has been added to simplify the process of loading data into both single partition and multi-partition database environments.

Load operations now take place at the table level. This means that the load utility no longer requires exclusive access to the entire table space, and concurrent access to other table objects in the same table space is possible during a load operation. Further, table spaces that are involved in the load operation are not quiesced. When the COPY NO option is specified for a recoverable database, the table space will be placed in the backup pending table space state when the load operation begins.

Another feature that has been added to the load utility is the ability to query pre-existing data in a table while new data is being loaded. You can do this by specifying the READ ACCESS option of the LOAD command.

The LOCK WITH FORCE option has also been introduced in this release. It allows you to force applications to release the locks they have on a table, allowing the load operation to proceed and to acquire the locks it needs.

You can now load data in partitioned database environments using the same commands (LOAD, db2load) and APIs (db2load, db2LoadQuery) used in single partition database environments. The AutoLoader utility (db2atld) and the AutoLoader control file are no longer needed.

Through the use of the new CURSOR file type, you can now load the results of an SQL query into a database without having to export them to a data file first.

Prior to Version 8, following a load operation the target table remained in check pending state if it contained generated columns. The load utility will now generate column values, and you no longer need to issue the SET INTEGRITY statement after a load operation into a table that contains generated columns and has no other table constraints.

The functionality of the LOAD QUERY command has also been expanded. It now returns the table state of the target table into which data is being loaded as well as the status information it previously included on a load operation in progress. The LOAD QUERY command can be used to query table states whether or not a load operation is in progress on a particular table.

The Control Center now has a load wizard to assist you in the set up of a load operation.

The load utility can be invoked through the command line processor (CLP), the Load notebook in the Control Center, or an application programming interface (API), db2Load.

The following are some examples for using load through CLP.

Examples for partitioned table loading

First, we create a sample table named LOADTAB. Its table structure is a copy of EMPLOYEE table, and, a unique index is created for this table. A data file exported in DEL format will be the external source file for the subsequent data loading examples. In addition, both EMPLOYEE and LOADTAB table are defined on partition 0 through 3 in a partitioned database environment.

Example 5-31 Sample table preparation for LOAD testing

```
db2inst1@UDBLN08:~/itso> db2 create table loadtab like employee
DB20000I The SQL command completed successfully.
b2inst1@UDBLN08:~/itso> db2 "create unique index u_lt_empno on loadtab(empno)"
DB20000I The SQL command completed successfully.
db2inst1@UDBLN08:~/itso> db2 "export to empl.del of del select * from
employee"
SQL3104N The Export utility is beginning to export data to file "empl.del".
SQL3105N The Export utility has finished exporting "32" rows.
Number of rows exported: 32
```

In the following sample, we will use two methods to load data into the LOADTAB, using external source file and using CURSOR answer set as the source for loading separately.

Example 5-32 Load data from source DEL file into a partitioned table

```
db2inst1@UDBLN08:~/itso> db2 "load from empl.del of del insert into loadtab"
Agent Type      Node      SQL Code      Result
-----
LOAD            000      +00000000     Success.
LOAD            001      +00000000     Success.
LOAD            002      +00000000     Success.
PARTITION       001      +00000000     Success.
PRE_PARTITION   000      +00000000     Success.
RESULTS:        3 of 3 LOADs completed successfully.
```

Summary of Partitioning Agents:
Rows Read = 32
Rows Rejected = 0
Rows Partitioned = 32

Summary of LOAD Agents:
Number of rows read = 32
Number of rows skipped = 0
Number of rows loaded = 32
Number of rows rejected = 0
Number of rows deleted = 0
Number of rows committed = 32

```
db2inst1@UDBLN08:~/itso> db2 "select dbpartitionnum(empno),count(*) from
loadtab group by dbpartitionnum(empno)"
```

```

1          2
-----
          0          10
          1          11
          2          11

```

3 record(s) selected.

In Example 5-32, the source file is loaded into LOADTAB which is spread across three partitions. The command is just like the one in a single partition environment. To verify the load result, within the same example, we used an SQL statement to count the number of rows distributed on different partitions, from the output of SQL execution, we find that the rows are distributed to the partitions evenly.

Example 5-33 Load data from CURSOR into a partitioned table

```

db2inst1@UDBLN08:~/itso> db2 "declare mycur cursor for select * from employee"
DB20000I The SQL command completed successfully.
db2inst1@UDBLN08:~/itso> db2 "load from mycur of cursor replace into loadtab"

```

Agent Type	Node	SQL Code	Result
LOAD	000	+00000000	Success.
LOAD	001	+00000000	Success.
LOAD	002	+00000000	Success.
PARTITION	001	+00000000	Success.
RESULTS:	3 of 3 LOADs completed successfully.		

Summary of Partitioning Agents:

```

Rows Read           = 32
Rows Rejected       = 0
Rows Partitioned    = 32

```

Summary of LOAD Agents:

```

Number of rows read      = 32
Number of rows skipped   = 0
Number of rows loaded    = 32
Number of rows rejected  = 0
Number of rows deleted   = 0
Number of rows committed = 32

```

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 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 partition table (Example 5-33). If you try to load source file in IXF format to a 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 defined on a multi partition nodegroup.

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.

If there is no partitioned database-specific configuration options specified with a LOAD command in a partitioned database environment (when DB2 registry variable DB2_PARTITIONEDLOAD_DEFAULT is NOT set to NO), the default values will be used. For example, the MODE will default to PARTITION_AND_LOAD.

In the preceding two examples, the default options for loading in a partitioned database environment are used.

Example 5-34 is a simple example using PARTITIONED DB CONFIG parameter to specify partitioned database-specific options for LOAD.

Example 5-34 Using PARTITIONED DB CONFIG parameter in LOAD command

```
db2inst1@UDBLN08:~/itso> db2 "load from emp1.del of del replace into loadtab
partitioned db config mode partition_only part_file_location
/db2home/db2inst1/itso partitioning_dbpartnums (1,2)"
```

Agent Type	Node	SQL Code	Result
LOAD_TO_FILE	000	+00000000	Success.

LOAD_TO_FILE	001	+00000000	Success.
LOAD_TO_FILE	002	+00000000	Success.
PARTITION	001	+00000000	Success.
PARTITION	002	+00000000	Success.
PRE_PARTITION	000	+00000000	Success.

Summary of Partitioning Agents:
 Rows Read = 32
 Rows Rejected = 0
 Rows Partitioned = 32

Examples for ALLOW READ ACCESS when loading

The ALLOW READ ACCESS option is very useful when loading large amounts of data because it gives users access to table data at almost all times, even 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 applications that have locks on the table, release them. 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 5-35, first we create a table named LOADREAD then insert three rows as the pre-existing data, then we will use ALLOW READ ACCESS to load additional rows from a source file in DEL format, and when the load is taking place, we will try to access the table from another session.

Example 5-35 Load data with ALLOW READ ACCESS

```
db2inst1@UDBLN08:~/itso> db2 "create table loadread(col1 int not null primary
key, col2 char(10))"
DB20000I The SQL command completed successfully.
db2inst1@UDBLN08:~/itso> db2 "insert into loadread
values(1,'aaaa'),(2,'bbbb'),(3,'cccc')"
```

DB20000I The SQL command completed successfully.

```
db2inst1@UDBLN08:~/itso> db2 "load from loadapp.del of del insert into
loadread allow read access"
```

Agent Type	Node	SQL Code	Result
LOAD	000	+00000000	Success.
LOAD	001	+00000000	Success.
LOAD	002	+00000000	Success.
PARTITION	001	+00000000	Success.
PRE_PARTITION	000	+00000000	Success.
RESULTS:	3 of 3 LOADs completed successfully.		

Summary of Partitioning Agents:

```
Rows Read           = 2
Rows Rejected       = 0
Rows Partitioned    = 2
```

Summary of LOAD Agents:

```
Number of rows read      = 2
Number of rows skipped   = 0
Number of rows loaded    = 2
Number of rows rejected  = 0
Number of rows deleted   = 0
Number of rows committed = 2
```

When the load command is issued and load is running, we can open another session to access the same table with read-only operation. Before the very end of the load operation, the SELECT statement in the our example returns the answer set very quickly. But when at the last stage of the load operation and before load returns the successful message as in the preceding example, the SELECT statement cannot succeed due to resource contention with load operation. At this time the load utility needs to acquire an exclusive lock (Z-lock) on the table before data is committed (Example 5-36).

After the load operation is finished, the newly loaded data is visible to other applications.

Example 5-36 Concurrent access to the table where load is taking place

```
db2inst1@UDBLN08:~/itso> db2 "select * from loadread"
```

```
COL1          COL2
```

```
-----  
1 aaaa  
2 bbbb  
3 cccc
```

3 record(s) selected.

```
db2inst1@UDBLN08:~/itso> db2 "select * from loadread"
```

```
COL1      COL2  
-----  
1 aaaa  
2 bbbb  
3 cccc
```

3 record(s) selected.

```
db2inst1@UDBLN08:~/itso> db2 "select * from loadread"
```

```
SQL0911N The current transaction has been rolled back because of a deadlock  
or timeout. Reason code "2". SQLSTATE=40001
```

```
db2inst1@UDBLN08:~/itso> db2 "select * from loadread"
```

```
COL1      COL2  
-----  
4 dddd  
2 bbbb  
3 cccc  
1 aaaa  
5 eeee
```

5 record(s) selected.

The LOCK WITH FORCE option may be used to force off conflicting applications with load operation, and allow the load operation to proceed without having to wait. Example 5-37 is an example for load with the ALLOW READ ACCESS and LOCK WITH FORCE options.

Example 5-37 Load with ALLOW READ ACCESS and LOCK WITH FORCE

```
db2inst1@UDBLN08:~/itso> db2 "delete from loadread where col1>3"  
DB20000I The SQL command completed successfully.  
db2inst1@UDBLN08:~/itso> db2 "load from loadapp.del of del insert into  
loadread allow read access lock with force"
```

And in another session we keep trying to access the table LOADREAD. It will result in error SQL1224N as the application is forced off by the application where the load is running (Example 5-38).

Example 5-38 Concurrent application be forced off by LOAD operation

```
db2inst1@UDBLNX08:~/itso> db2 "select * from loadread"
```

```
COL1          COL2
-----
          1 aaaa
          2 bbbb
          3 cccc
```

3 record(s) selected.

```
db2inst1@UDBLNX08:~/itso> db2 "select * from loadread"
```

```
SQL1224N A database agent could not be started to service a request, or was
terminated as a result of a database system shutdown or a force command.
SQLSTATE=55032
```

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 truncates 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 load operations initiated from the CLP, the message files will not be displayed to the console or retained. 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, /db2home/db2inst1/myload.msg.

If the MESSAGES option is used, once the load operation has completed the message files on each partition will be transferred to the client machine and stored in files using 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 5-1.

Table 5-1 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>

For more information regarding the agent type listed in the preceding table, refer to *IBM DB2 UDB Data Movement Utilities Guide and Reference V8*, SC09-4830, and *IBM DB2 UDB Command Reference V8*, SC09-4828.

Note: We strongly recommend that the MESSAGES option be used for partitioned database load operations initiated from the CLP.

The LOAD QUERY command can be used to check the status of a load operation during processing and returns the table state. Even if a load is not processing, the table state can also be returned. The output of the LOAD QUERY command in Example 5-39 is gathered when previous load example (*Example 5-35 Load data with ALLOW READ ACCESS on page 194*) is running.

Example 5-39 Using LOAD QUERY to monitor LOAD status

```
db2inst1@UDBLN08:~> db2 "load query table loadread"
SQL3530I The Load Query utility is monitoring "LOAD" progress on partition
"0".

SQL3501W The table space(s) in which the table resides will not be placed in
backup pending state since forward recovery is disabled for the database.

SQL3109N The utility is beginning to load data from file
"/db2home/db2inst1/itso/loadapp.del".

SQL3500W The utility is beginning the "LOAD" phase at time "10-22-2002
17:14:25.692869".

SQL3519W Begin Load Consistency Point. Input record count = "0".
SQL3520W Load Consistency Point was successful.

SQL3110N The utility has completed processing. "1" rows were read from the
input file.

SQL3519W Begin Load Consistency Point. Input record count = "1".

SQL3520W Load Consistency Point was successful.

SQL3515W The utility has finished the "LOAD" phase at time "10-22-2002
17:14:27.993262".

SQL3500W The utility is beginning the "BUILD" phase at time "10-22-2002
17:14:28.312771".

SQL3213I The indexing mode is "INCREMENTAL".

SQL3532I The Load utility is currently in the "LOAD" phase.

Number of rows read          = 1
Number of rows skipped       = 0
Number of rows loaded        = 1
```

```
Number of rows rejected    = 0
Number of rows deleted    = 0
Number of rows committed  = 1
Number of warnings        = 0
```

Tablestate:

Load in Progress

Read Access Only

```
db2inst1@UDBLN08:~> db2 "load query table loadread"
```

```
SQL3532I The Load utility is currently in the "UNKNOWN" phase.
```

Tablestate:

Load in Progress

Read Access Only

```
db2inst1@UDBLN08:~> db2 "load query table loadread"
```

Tablestate:

Normal

LOAD WIZARD in 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 window (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 window opens.

Detailed information about the Control Center is available in its online help.

The following is an example of how to use the Load Wizard to load data into a sample table named EMP which is defined on partition 0 through 2, so this is a partitioned load. Detailed steps are as below:

1. Invoke the Load Wizard from DB2 Control Center (Figure 5-20).

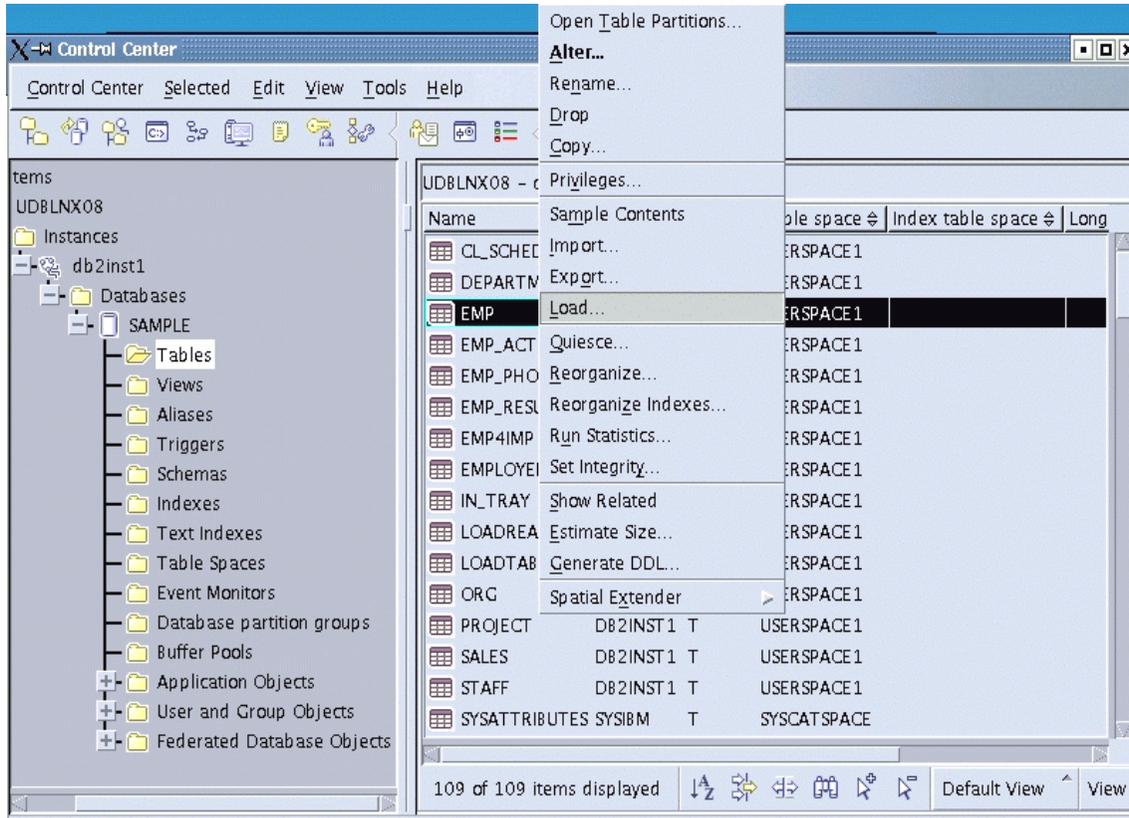


Figure 5-20 Invoking Load Wizard from DB2 Control Center

2. Choose the load operation (Figure 5-21).

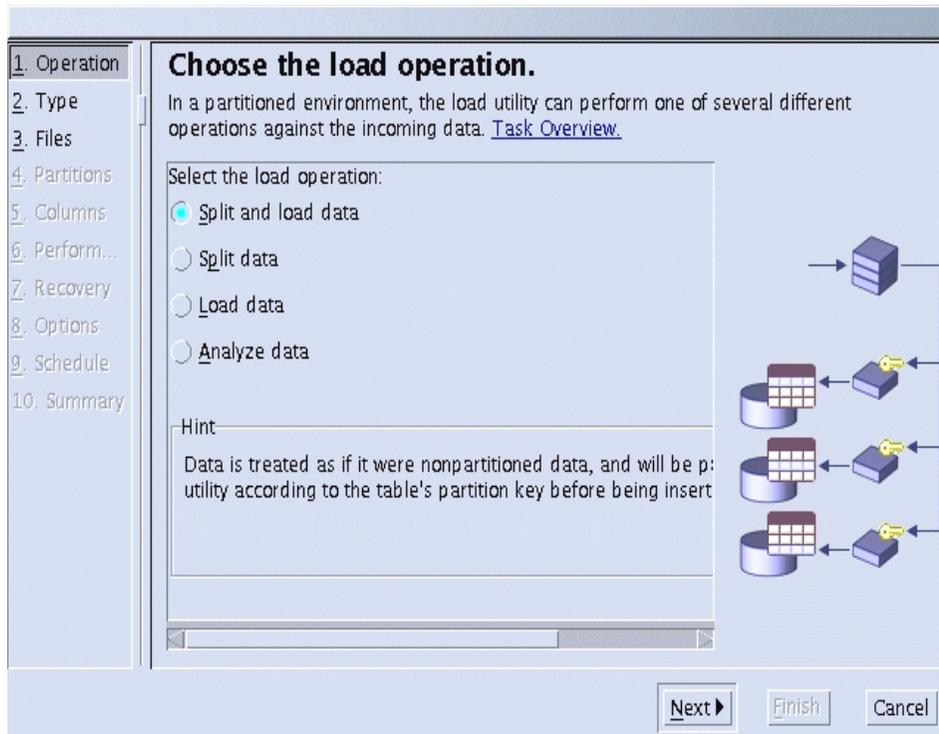


Figure 5-21 Choose the load operation mode

3. Choose whether the original table data will be kept (Figure 5-22).

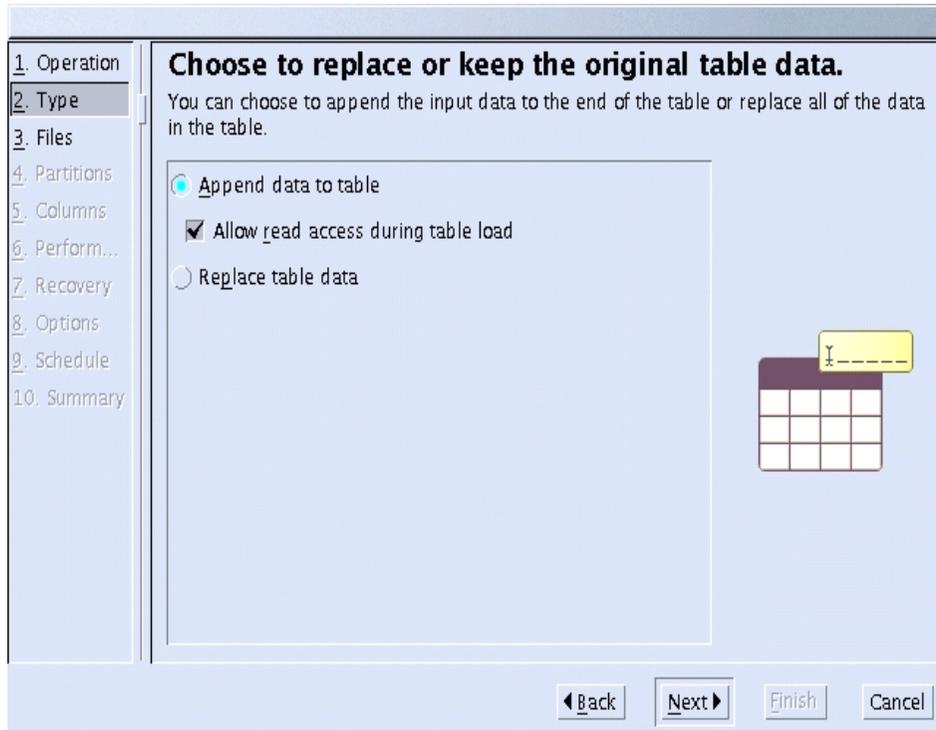


Figure 5-22 Choose if the original table will be kept

4. Specify input and output files (Figure 5-23).

In this window, you can specify the format for input files, and options specific of the format you are using. For example, you can click the DEL Options button 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 in the window which will pop up when you click the "...” button.

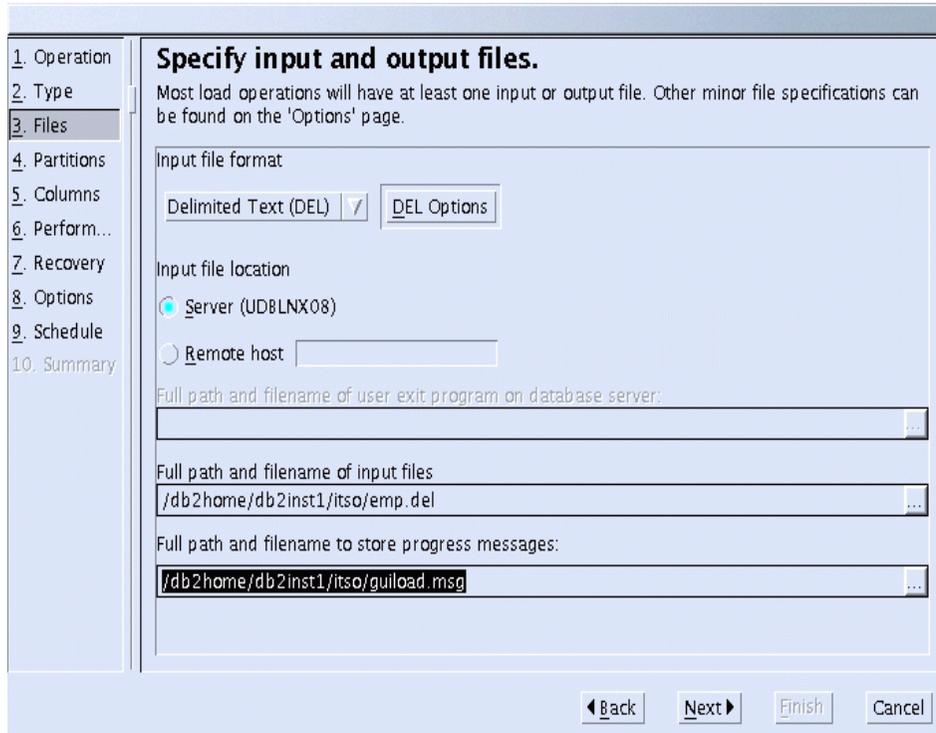


Figure 5-23 Specify input and output files

5. Specify which partitions participate in the load (Figure 5-24).

You can use the default choice of “Load data into all table partitions” in the window, if you just want to load data into all the partitions where the table resides. Under some conditions, you may want to load data for a specific partition or partitions; then you can choose “Load data into selected table partitions”. In addition, in the “Partitioning partitions” group within the window, you also can specify which partitions will be used to do the data partitioning work; in general, you can just leave it and “Let DB2 decide which partitions participate”.

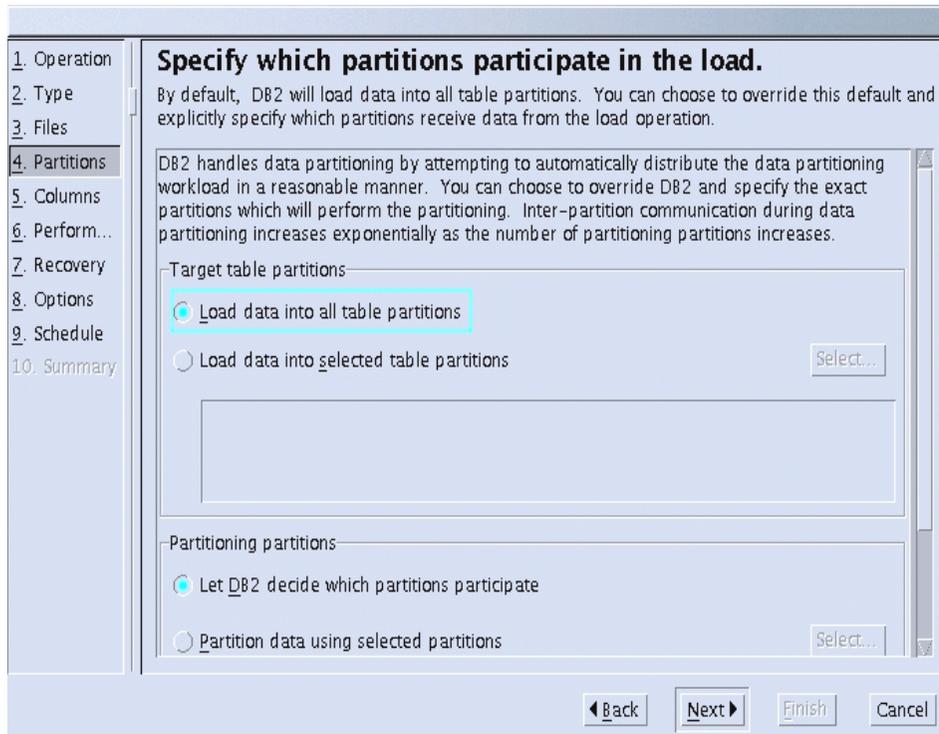


Figure 5-24 Specify which partitions participate in the load

6. Define the input columns and their mapping to the output columns (Figure 5-25).

You can simply use the default mapping if you think this is appropriate for your situation, or you can make some modifications in the window, if needed. For example, if large object (LOB) is involved in your load, then you may need to specify LOB related options in the window.

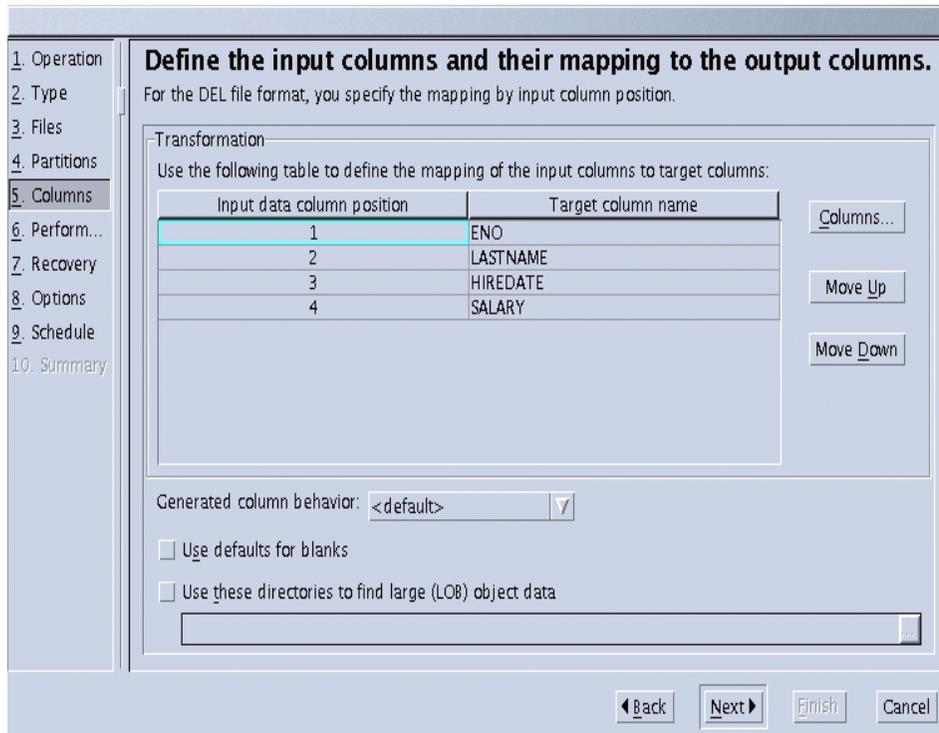


Figure 5-25 Define column related properties for load

7. Specify performance and statistics collection options (Figure 5-26).

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. But 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 window. If you don't want to make any modifications, leaving the defaults is alright.

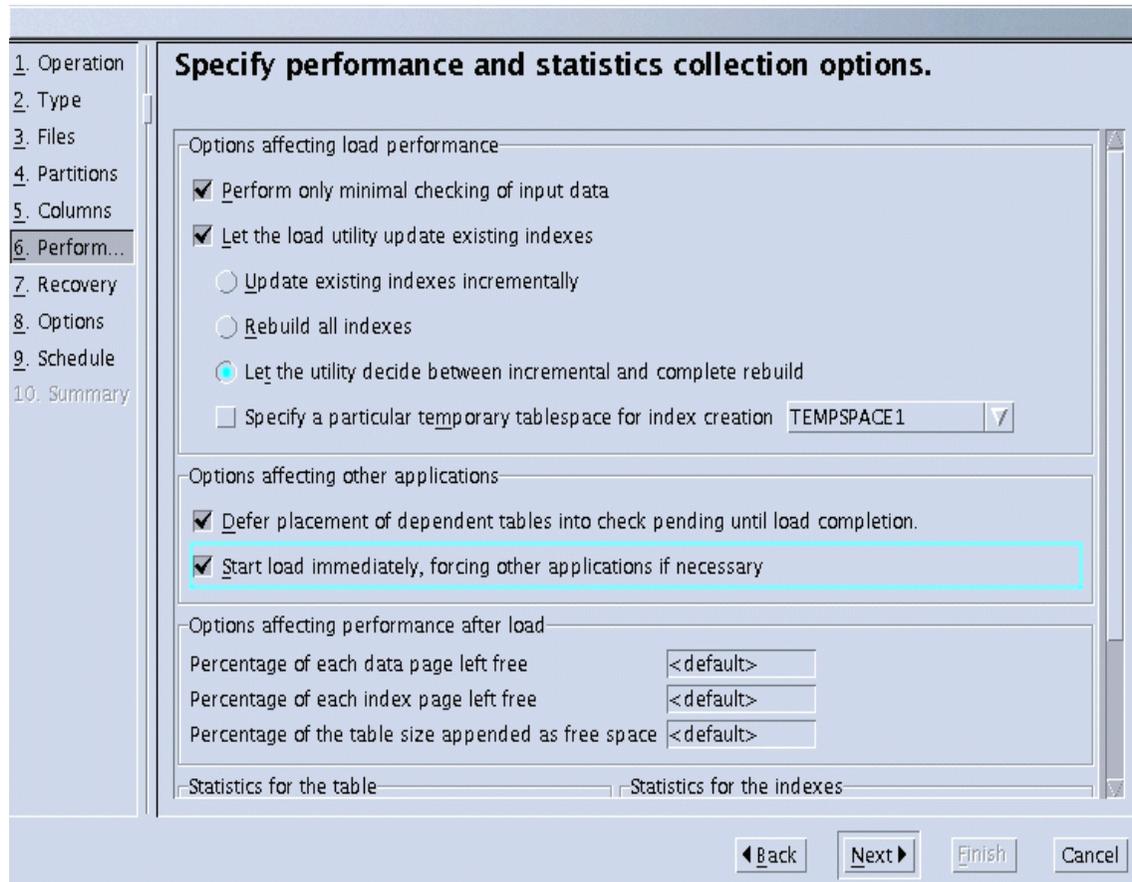


Figure 5-26 Specify performance and statistics collection options

8. Specify failure options and recovery strategy.

Options regarding failure handling and recovery strategy are covered in this window. 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 will rollback in the Crash recovery group in the window, as shown in Figure 5-27.

Select failure options and recovery strategy.

Use this page to specify failure and recovery options for a load failure. You can also specify forward recovery options if the database that contains the target table is configured for forward recovery.

Crash recovery

Establish consistency points and generate row count messages during data load

Number of rows between consistency points

Load job fails if any partition encounters errors in the setup phase

Load will rollback immediately on all partitions if any fail after setup is complete

Forward recovery

The following options will be valid only if the database is configured for forward recovery at the time the load task runs. If you choose to perform an unrecoverable load, a subsequent roll forward through this load will succeed but render the table unusable.

Perform a recoverable load

Do not make a copy of the input data. This will put the tablespace into backup pending.

Save a copy of the input data. The table will be usable after load completion.

Copy target:

Directory names:

◀ Back Next ▶ Finish Cancel

Figure 5-27 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 windows, and you can get hints by clicking on the option listed in the window (Figure 5-28).

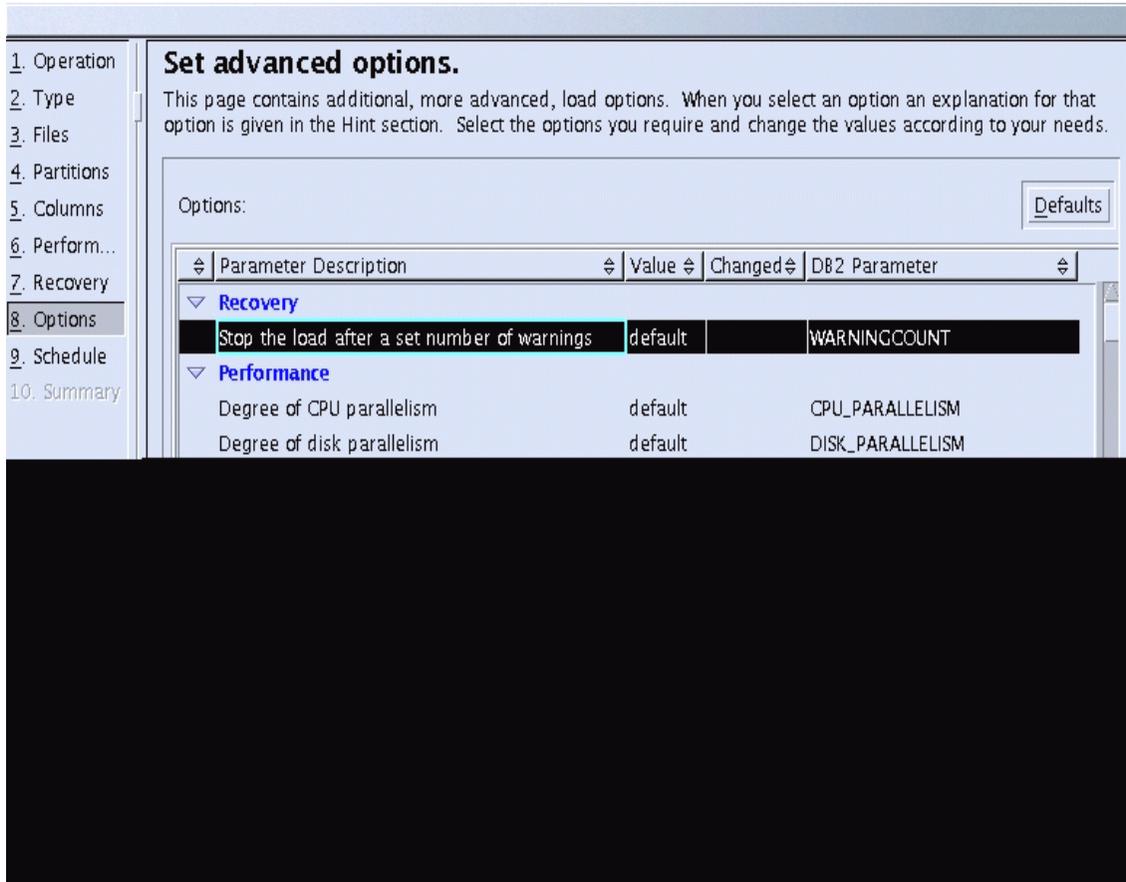


Figure 5-28 Set advanced options for load operation

10. Scheduling task execution.

If you want to run the load operation without saving task history, choose the Run now option as displayed in the sample window. Or if you want to create it as a task in the Task Center, then you can change the choices in the window (Figure 5-29).

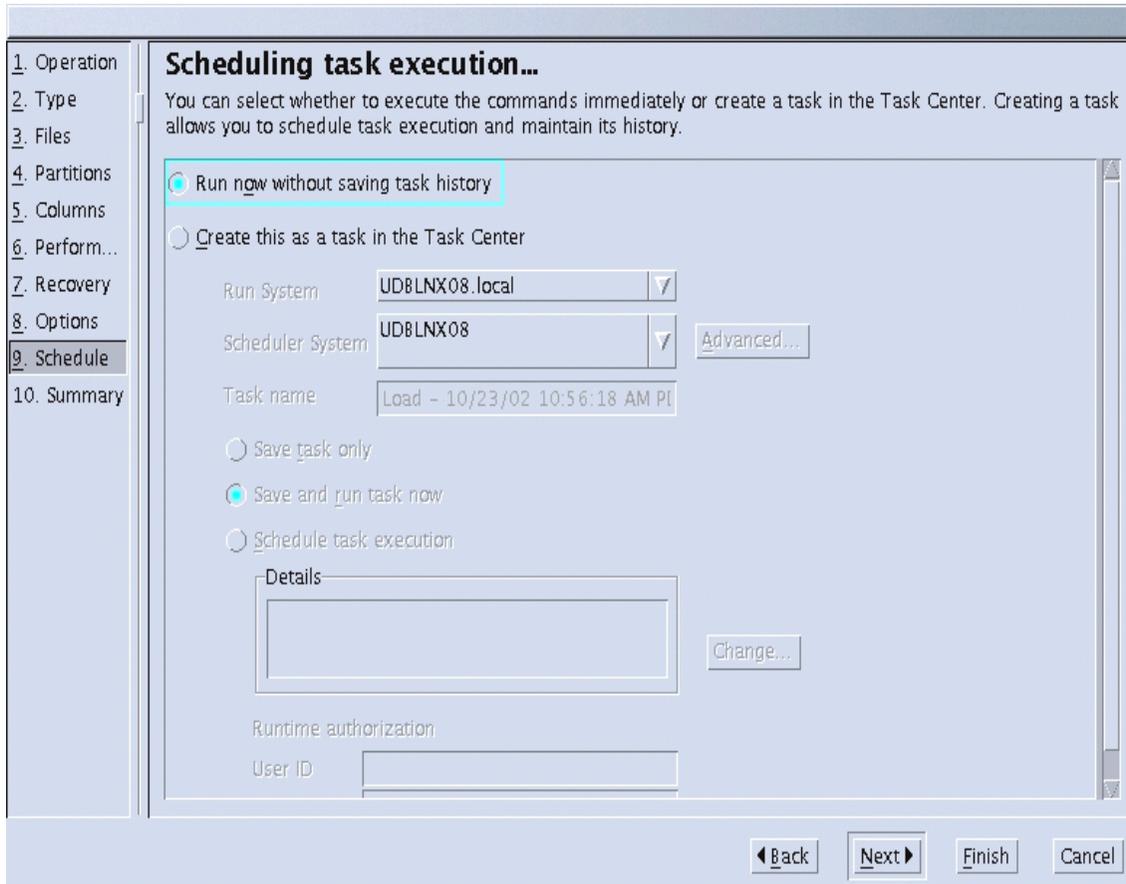


Figure 5-29 Scheduling task execution

11. View the task summary and execute it.

In Figure 5-30, the DB2 CLP commands which were generated based on your choices in the previous windows are displayed. You may click Finish to submit the command, or click Back for further modification of your choices.

You can check the message files for more details regarding the load operation within the DB2 Control Center. The message files are *guiload.msg.prep.000* for pre-partitioning, *guiload.msg.part.001* for partitioning, and *guiload.msg.load.000*, *guiload.msg.load.001*, *guiload.msg.load.002* for loading. All those files are located under the directory */db2home/db2inst1/itso*, which was specified in the previous Specifying input and output files window.

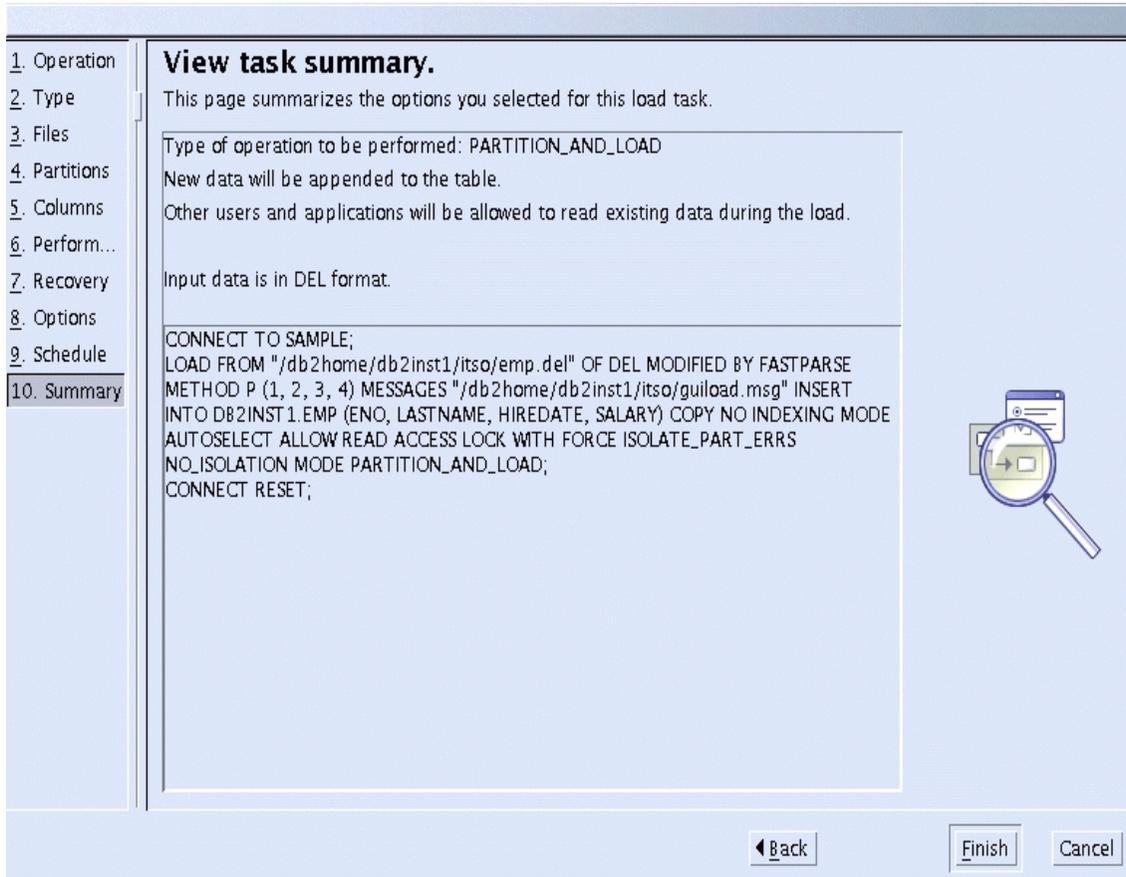


Figure 5-30 View task summary

For detailed information about using load, refer to *IBM DB2 UDB Data Movement Utilities Guide and Reference V8*, SC09-4830, and *IBM DB2 UDB Command Reference V8*, SC09-4828.

5.3.4 Using db2move utility

This tool facilitates the movement of large numbers of tables between DB2 databases located on workstations. It calls the DB2 export, import, and load APIs, depending on the action requested by the user, and operates the files in PC/IXF format. For details regarding this utility, refer to *IBM DB2 UDB Command Reference V8*, SC09-4828.

5.4 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 in the following pages as the preparation work for enabling the Task Center and Scheduler.

5.4.1 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 like the Task Center, the Control Center, Configuration Assistant, or the Development Center. 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 UDB, 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 via DB2 CLP commands, refer to Appendix B, “DB2 Tools Catalog creation” on page 295.

So 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 5-31 shows the relationship between DB2 tools, DB2 instances, DAS, Scheduler, and the Tools Catalog database.

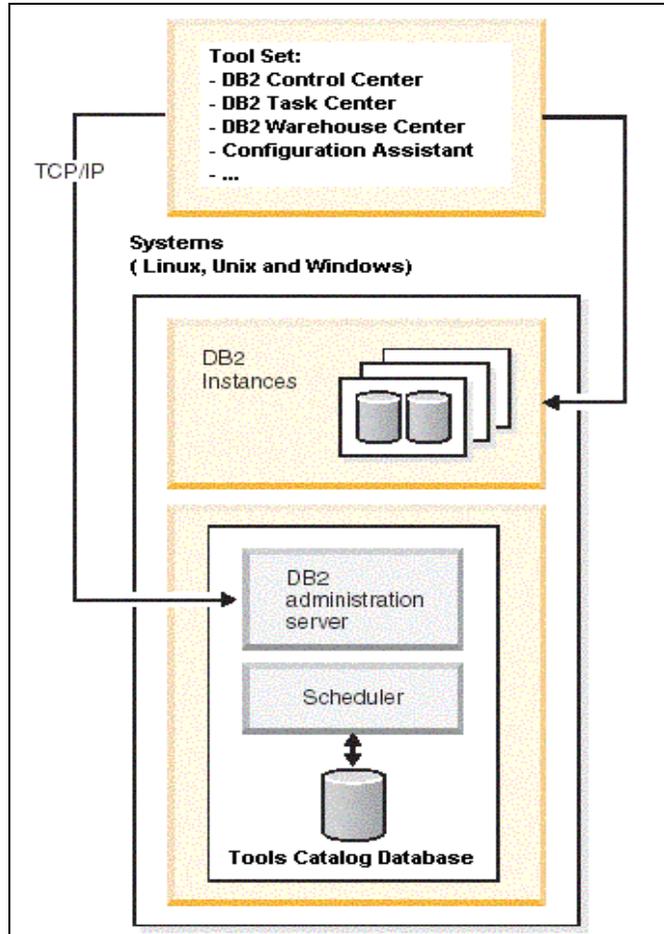


Figure 5-31 How DAS and Tools Catalog Database relate to other parts of DB2

5.4.2 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 Task Center and create a new task (Figure 5-32):

Right-click in the content pane, then click **New**, and a new window will open.

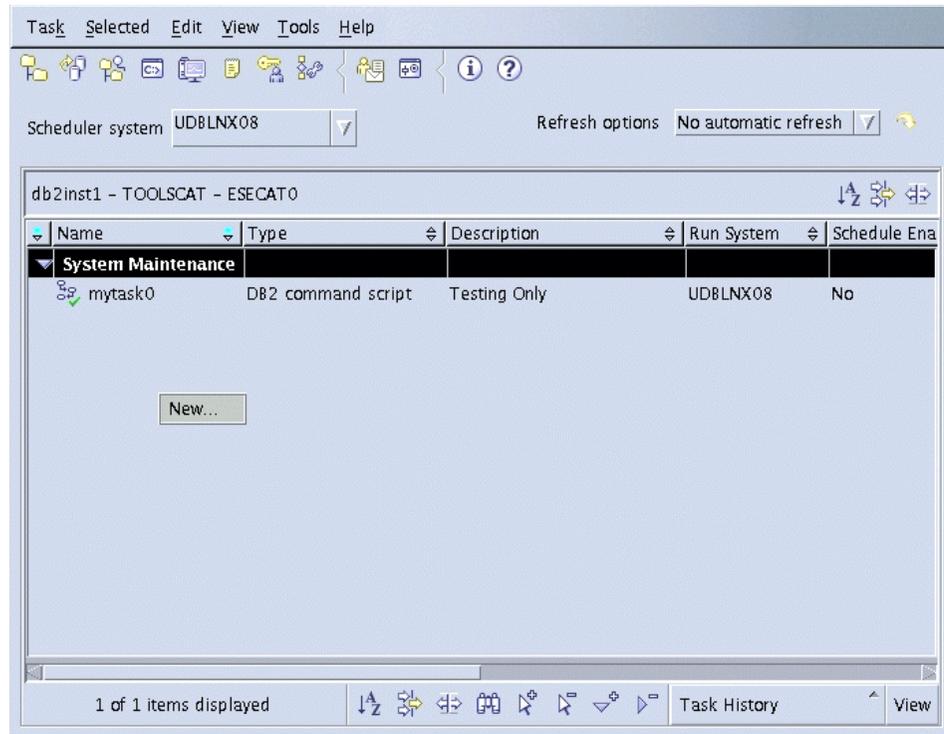


Figure 5-32 Task Center main window

2. General info input for a new task.

You can input general info for a task in this window (Figure 5-33). For example, the name and type for the task, the instance and partition 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.

UDBLNX08.DB2INST1.<New Task>

Task	Command Script	Run properties	Group	Schedule	Notification	Task Actions	Security
Name	Testing Task 1						
Type	DB2 command script						
Description	Testing task for demonstration purpose only.						
Task category	Redbook Related						
Run system	UDBLNX08						
DB2 instance and partition	db2inst1_0						

Figure 5-33 General info input for a new task

3. Command script for the new task.

Here you can input the script for the task, and by default, the semicolon sign “;” will be used as the DB2 statement termination character. If you want, you can change it in this window (Figure 5-34).

Task	Command Script	Run properties	Group	Schedule	Notification	Task Actions	Security
	<pre>connect to sample; runstats on table db2inst1.employee; connect reset;</pre>						

Figure 5-34 Command script input for a 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 chose **Stop execution** when failure happens (Figure 5-35).

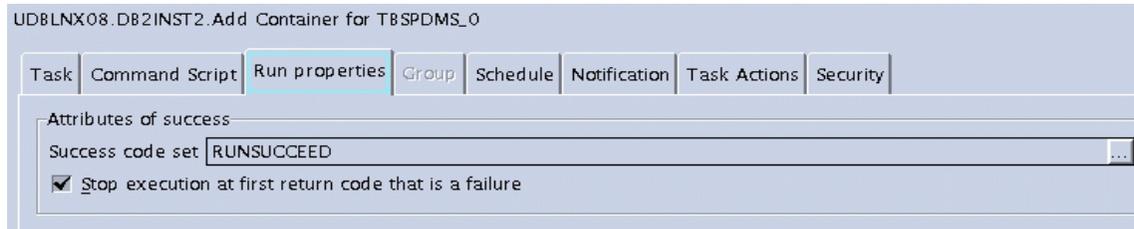


Figure 5-35 Run properties for the task

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 5-36 provides a sample to specify running a weekly task, on every Sunday, and with an end date set.

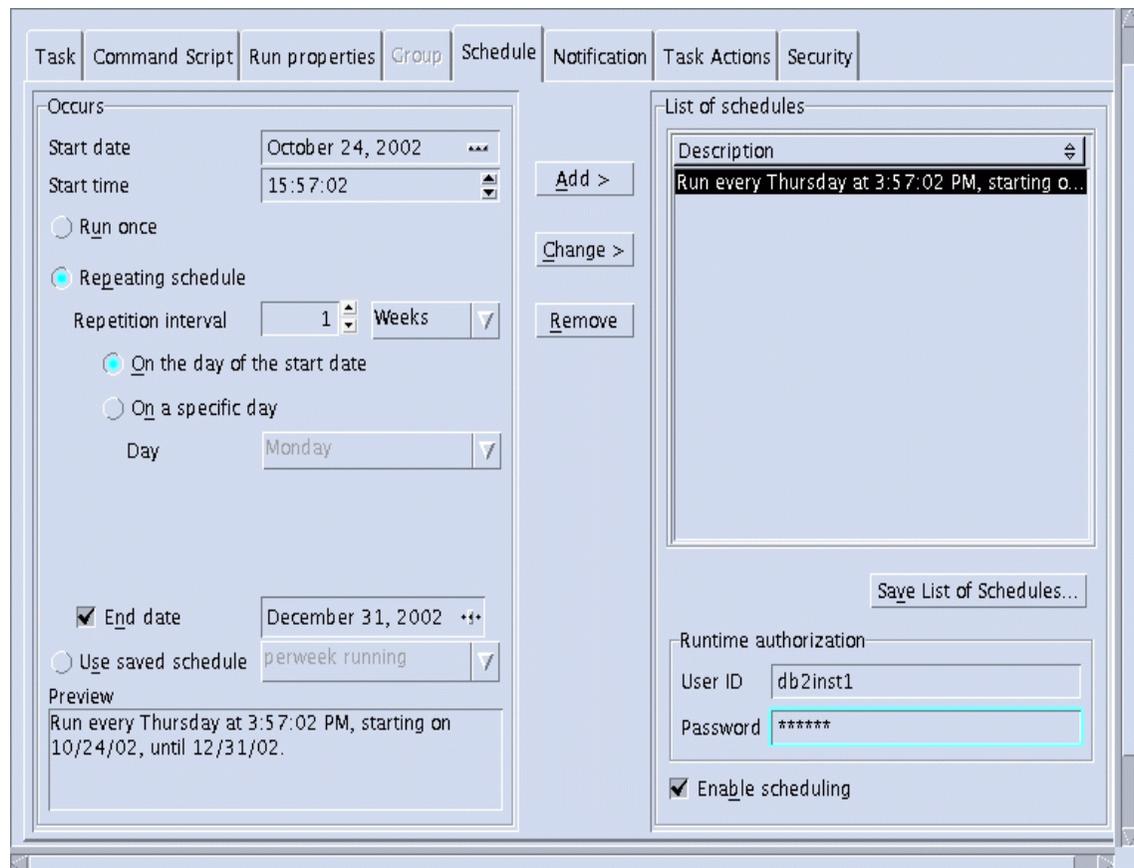


Figure 5-36 Scheduling tasks for repetitive running

Figure 5-37 is another example which is used to run a task 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 window. Once saved, the next time you are scheduling a task, you can choose **Use saved schedule** with the desired name to schedule a task.

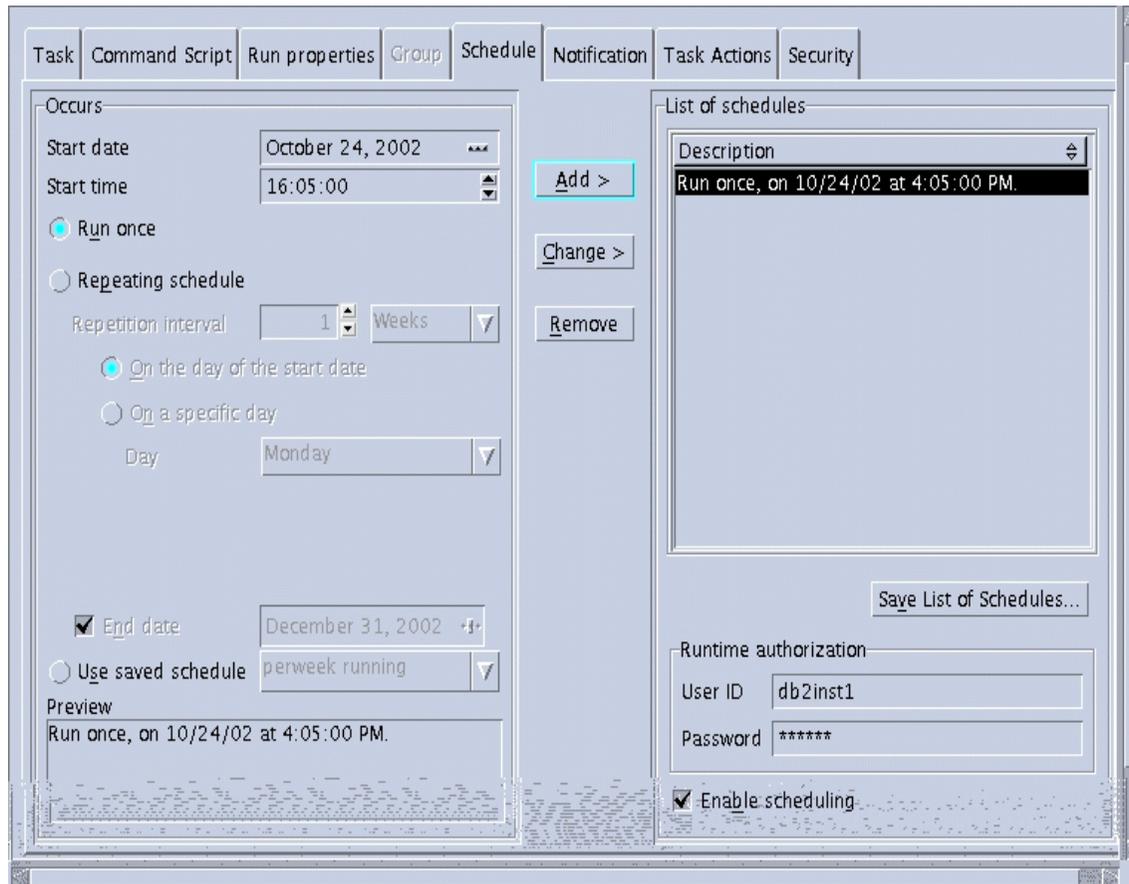


Figure 5-37 Scheduling tasks for running once

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 or failure, or both. You also can specify multiple notifications for different conditions for only one task.

In Figure 5-38, the notification will be sent to DB2's Journal as a message entry.

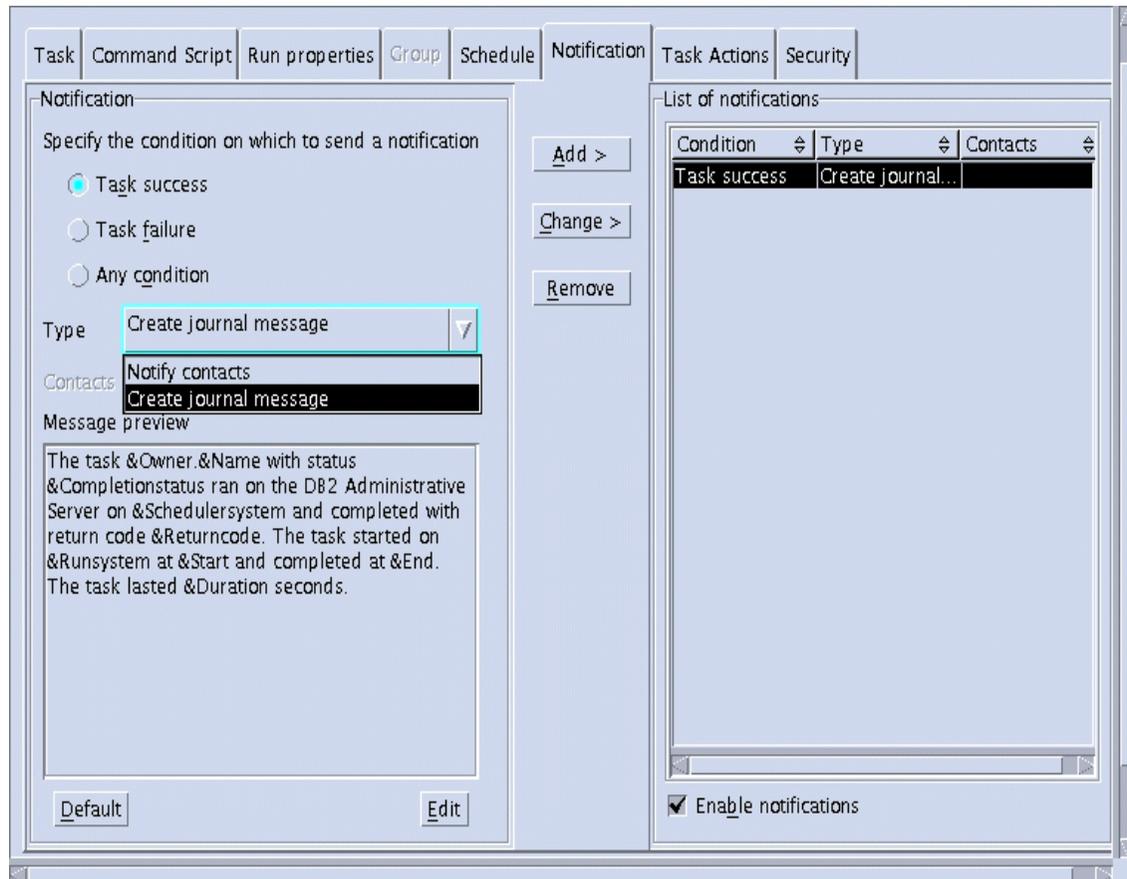


Figure 5-38 Notification setting for different task running result

7. Follow-up actions for different task running result (Figure 5-39).

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.

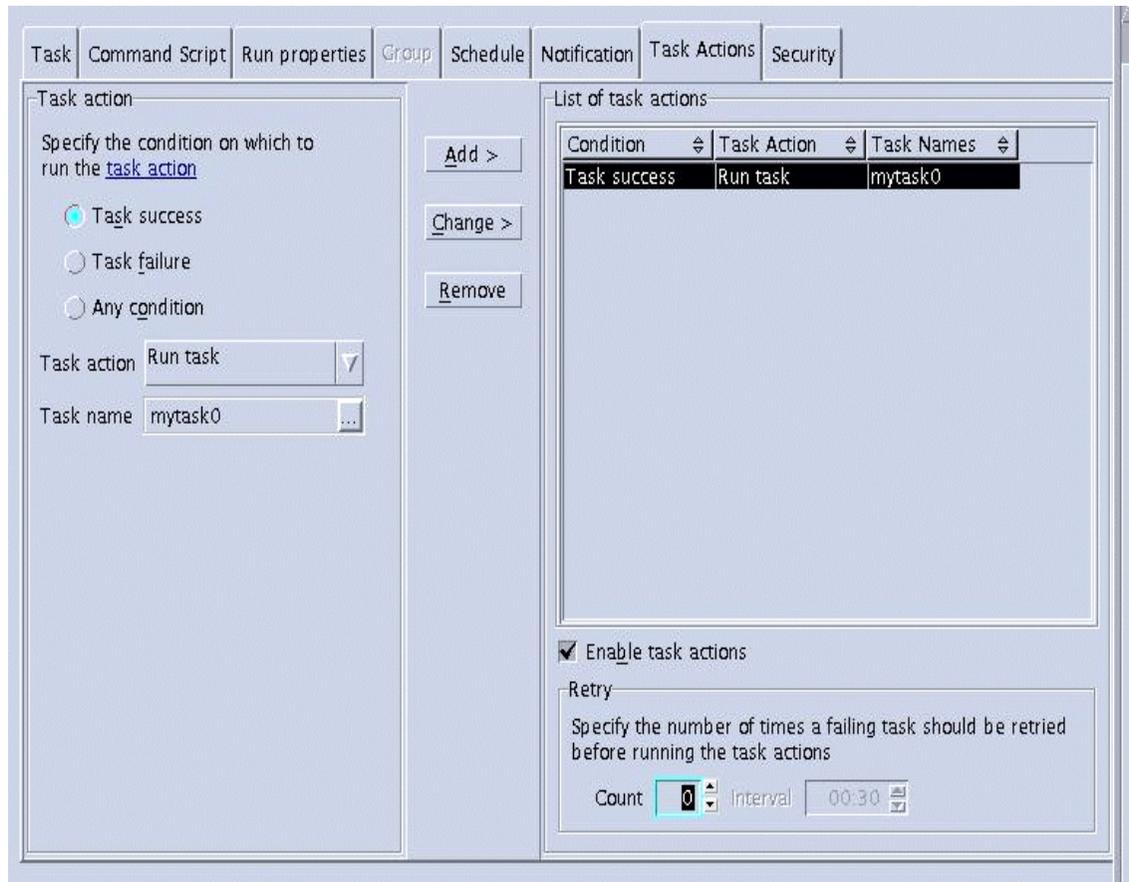


Figure 5-39 Follow-up actions setting for a task

You can also specify access privileges to the task, such as read, run, or execute for different users and groups by clicking the Security tab within the window.

Now you can submit the task to the Task Center by clicking OK.

As in our example, the notification is specified to be sent to DB2's journal, so we can check the Journal for the running results.

To invoke the Journal, you can select **Tools** —> **Journal** in the menu bar of the Task Center or Control Center, or other DB2 GUI tools where the Tools menu is available.

Here is the result for our sample task. You can get more details by right-clicking the result entry which corresponds to your task, then choose **Show Results**. Another window will pop up, and within that, you can get detailed information for

the task execution. For example, the command script and running output of the commands specified in the task are shown in Figure 5-40.

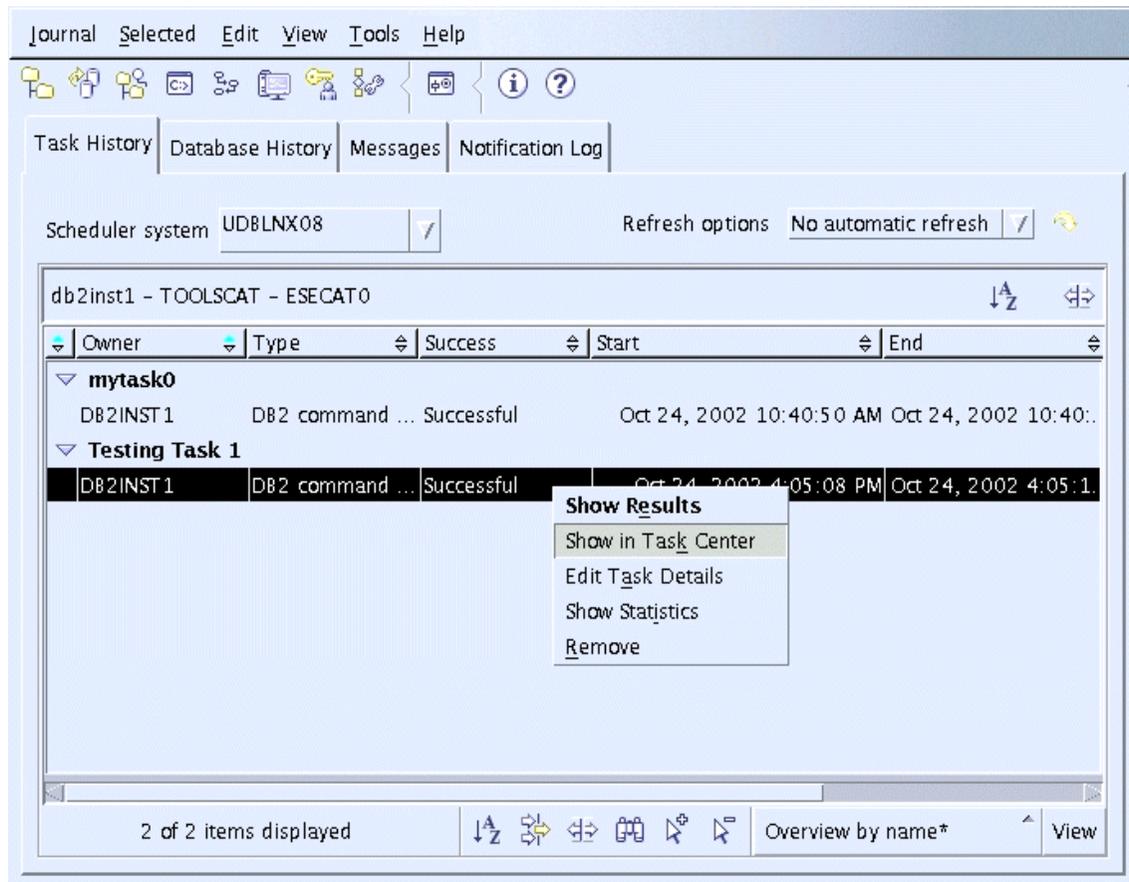


Figure 5-40 Using Journal to check task running results

You can also show the task entry within the Task Center by choosing **Show in Task Center** in the pop-up window as the preceding figure shows. Then 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, just as the following example shows in Figure 5-41.

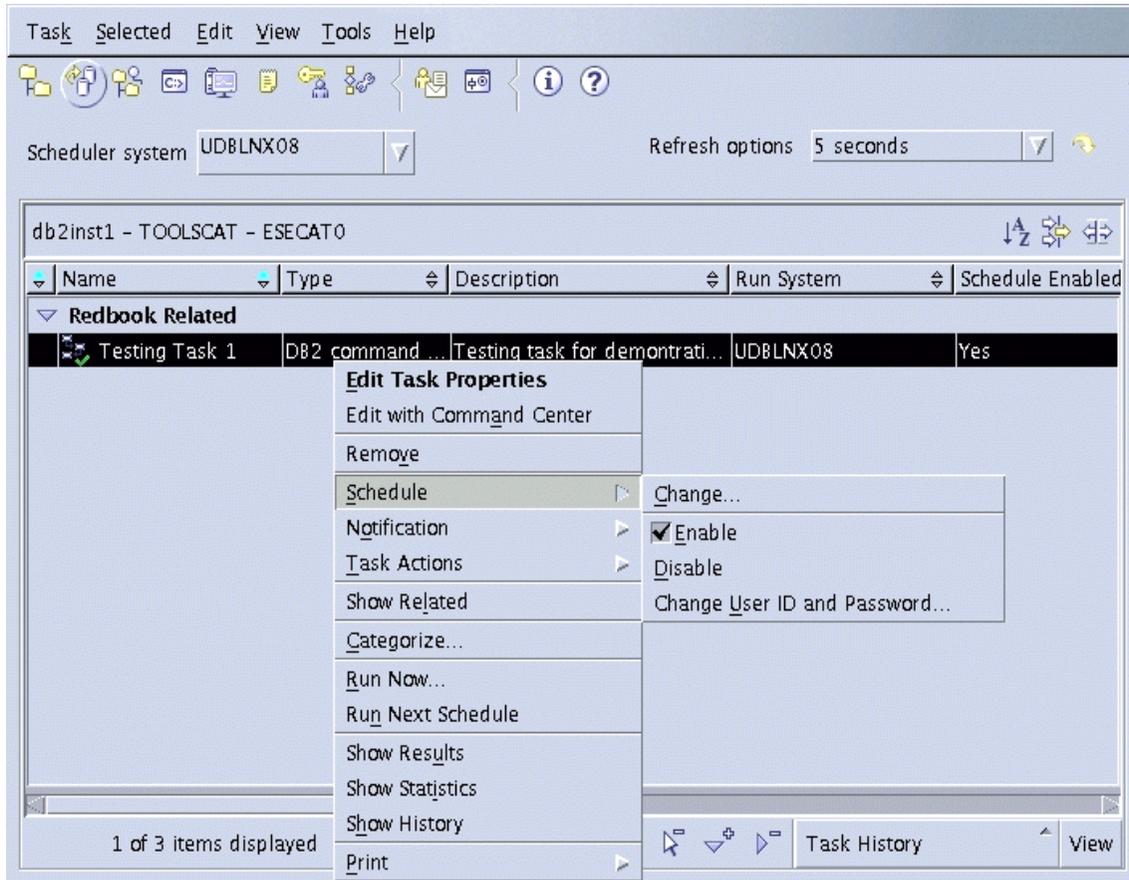


Figure 5-41 Modifying task within Task Center

The scheduling function is also available for many other DB2 Wizards or GUI tools such as export, import, load wizard, runstats, reorg, backup and restore. You can take advantage of these useful functions for task automation purposes in your environment.



Monitoring DB2

This chapter discusses DB2's monitoring features provided by Health Center and Health Monitor.

DB2 Universal Database Version 8 adds self-managing and resource tuning (SMART) database technology that lets database administrators choose to configure, tune and manage their databases with enhanced automation. SMART database management means administrators spend less time managing routine tasks and more time focussing on tasks that help enterprises gain and maintain a sustainable competitive advantage. Several tools are added or redesigned to support self-managing, self-optimizing and self-tuning requirements, such as Configuration Advisor, Health Center, Performance Expert, Recovery Expert, and so forth. These new tools make DB2's configuration and maintenance tasks more smart and easy-to-use, and finally results in significant reduction of human intervention and cost reduction for the database system operation and maintenance.

This chapter gives an in-depth discussion of the Health Center and Health Monitor. Chapter 3, "Post installation tasks" on page 79 has detailed information regarding Configuration Advisor. For other DB2 monitoring tools, refer to their online help or specific documents for more information.

In addition, this chapter also discusses DB2 UDB and operating system log files which may provide more detailed information about certain problems and can help you further determine what the problem is and what the cause is. Afterwards you can take corrective actions based on the collected information.

Furthermore, some useful and popular tools from the operating system are also available to assist in monitoring the system resource usage and perform troubleshooting, and we also provide brief information about these tools.

6.1 Health Monitor and Health Center

With Version 8, DB2 introduces two new features to help you monitor the health of your DB2 systems: Health Monitor and Health Center. These tools add a management by exception capability to DB2 Universal Database by alerting you to potential system health issues. This enables you to address health issues before they become real problems that affect your system performance. Prior to using the Health Monitor and Health Center, you need to understand the concept of a health indicator which is briefed in the following paragraph.

6.1.1 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. You can also 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.

6.1.2 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 automatically evaluates a set of health indicators, even without user interaction. If the current value of a health indicator is outside the acceptable operating range defined by its warning and alarm thresholds, the health monitor generates a health alert.

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 is enabled by default when a instance is created; you can deactivate it using the database manager configuration parameter `health_mon`. If the Health Monitor finds that a defined threshold has been exceeded (for example, 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 will raise an alert.

The Health Monitor, not only alerts DBAs via e-mail, pager, or PDA if a database system is likely to experience a fatal situation before it occurs (such as, running out of memory, or a table space is becoming full, and so on), and its autonomic capabilities can also take or recommend corrective actions. Therefore, it reduces reliance on the skill level of DBAs, and improves their productivity.

6.1.3 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.

DB2 Version 8 also provides a new Web Health Center that can be used to access the Health Monitor information from a Web browser or PDA.

6.1.4 Using Health Center and Health Monitor

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. You can also start the Health Center by clicking the health center status beacon when it appears on the status line of the DB2 Control Center window.

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 (Figure 6-1). Similarly, the Stop Health Monitor menu item will appear if `health_mon` is set to ON and the Health Monitor is started.

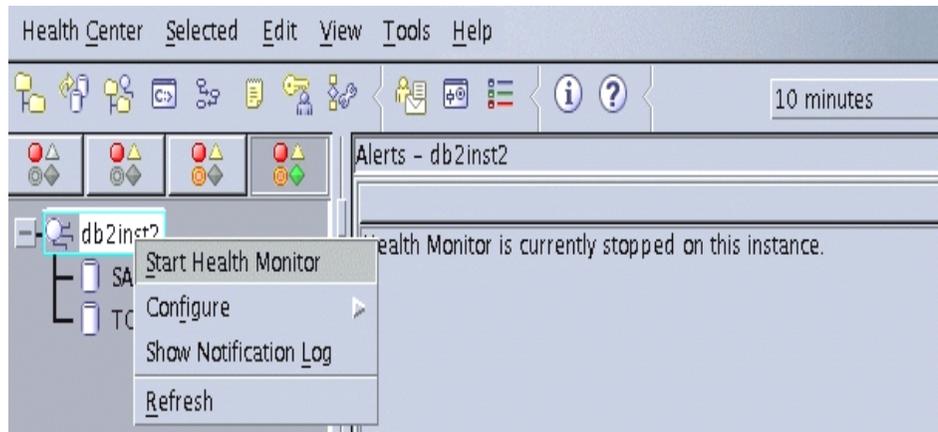


Figure 6-1 Start Health Monitor for instance using Health Center

You can use the toggle buttons at the top of the navigation bar to filter the alerts according to their severity (Figure 6-2). The Health Center will launch the third toggle, display alarms, warnings, and attentions. If the Health Center is opened for the first time, you can choose **Display all alerts** to view the navigation tree with all catalogued LUW instances on the left pane of the window.

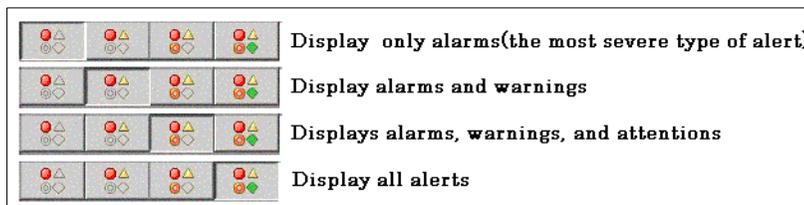


Figure 6-2 Using toggle buttons to filter the alerts

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 6-1 Check if Health Monitor is enabled for instance

```
db2inst2@UDBLNX08:/db2home/db2inst2> db2 update dbm cfg using health_mon on
DB20000I The UPDATE DATABASE MANAGER CONFIGURATION command completed
successfully.
db2inst2@UDBLNX08:/db2home/db2inst2> 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 thresholds or object state, the notifications can be sent and actions can be taken automatically.

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, another submenu with three items is displayed, as shown in Figure 6-3. You can use **Notifications** to set up the notification for the Health Monitor, or you can choose other items to view or change the Global Health Indicator Default Settings and Database Object Health Indicator Settings.

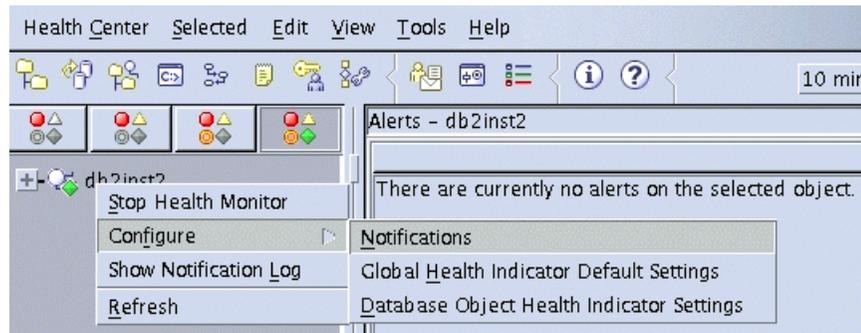


Figure 6-3 Configure menu in Health Center

If you want to notify people via e-mail or pager when an alert is generated, you need to define a contact list in the DAS first. You can add a contact or contact group in the Contacts management window or by using DB2 CLP commands, for example:

```
ADD CONTACT contname TYPE EMAIL ADDRESS contname@compname.com
```

If you want to add the contact via the Contacts management GUI tool, select **Tools** —> **Contacts** within the Health Center (Figure 6-4).



Figure 6-4 Start Contacts management

Within the Contacts window, 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 specified for the contact is reachable by pressing a Test button (Figure 6-5).

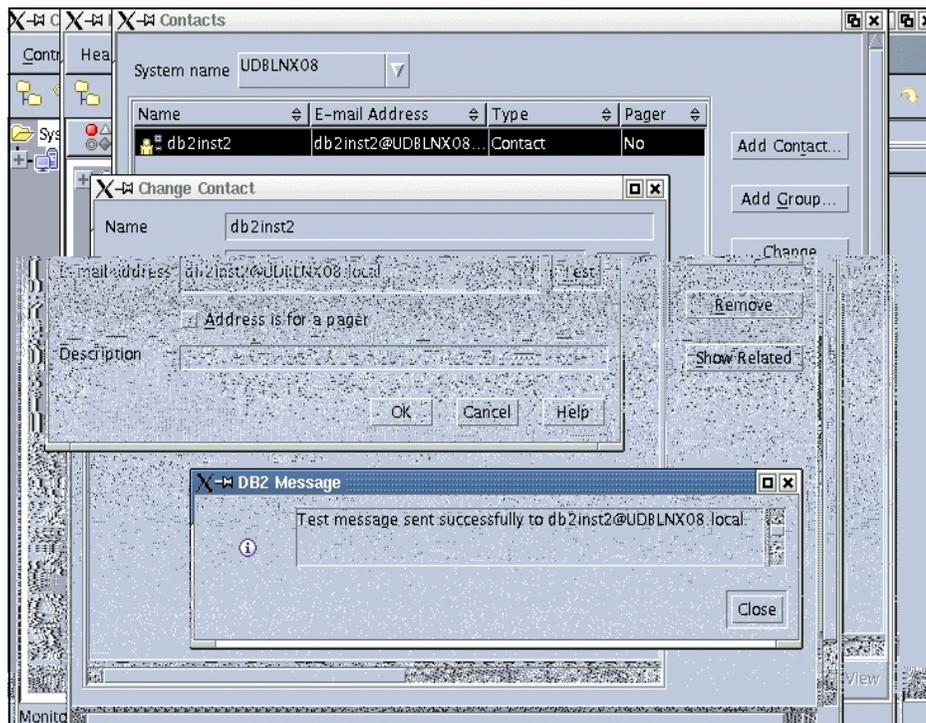


Figure 6-5 Contacts management

After pressing the Test button, if the DB2 Message returned shows that the test is successful, then an e-mail will be sent to the specified e-mail account (Figure 6-6).

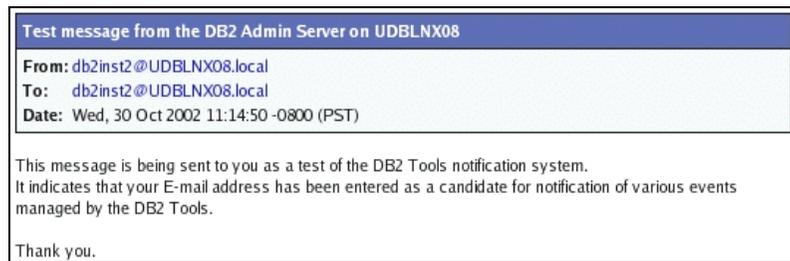


Figure 6-6 The testing e-mail sent by DB2 automatically

Note: Before using the e-mail notification function, you need to make sure the e-mail system which is residing on the server (specified by SMTP_SERVER in administration configuration) is up and running normally.

After your contact list is added, you can select **Configure -> Notifications** from the pop-up window, as shown in Figure 6-3, to add or change a contact or contact group. 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/sql/lib/db2dump, file name is <instance name>.nfy. For details regarding the notification log as well as its setting, refer to 6.2.1, "DB2 administration notification log" on page 242.

Usage Sample

The following sample shows you 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 or pager. 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 tablespace and table.

2. Review health indicator for the DMS tablespace
3. Populate a desired amount of data into the tablespace
4. Check the notification sent by DB2 system
5. Set up additional follow-up actions

These steps are given in detail:

1. Prepare DMS tablespace and table.

In our example, a DMS tablespace with raw device container is used. For information about how to use raw device in Linux environment for DB2 UDB, refer to Chapter 2, “Installation” on page 23 in this book. The size of the tablespace is 12 MB at its first creation. After the tablespace is created, a table with a unique index is created within (Example 6-2).

Example 6-2 Prepare tablespace and table for Health Center

```
db2inst2@UDBLNX08:/db2home/db2inst2/test> cat crt_tbspdms_0.db2
create regular tablespace tbspdms_0
managed by database
using (DEVICE '/dev/raw/raw1' 12M);
db2inst2@UDBLNX08:/db2home/db2inst2/test> db2 -tvf crt_tbspdms_0.db2
db2inst2@UDBLNX08:/db2home/db2inst2/test> cat emp.crt.sql
drop table emp;
CREATE TABLE EMP (ENO INTEGER, LASTNAME VARCHAR(30),
                   HIREDATE DATE, SALARY INTEGER)
                   in tbspdms_0;
CREATE UNIQUE INDEX u_emp_eno on EMP(ENO);
db2inst2@UDBLNX08:/db2home/db2inst2/test> db2 -tvf emp.crt.sql
```

2. Review health indicator settings for the DMS tablespace.

To configure the health indicator for DMS tablespaces in the Health Center, you need to start Database Object Health Indicator setting window first (Figure 6-7).



Figure 6-7 Start database object health indicator setting

Within the Configure Database Object Health Indicator Setting window as shown in Figure 6-8, the health indicators are listed on the left side of the window. You can select whether or not to monitor them by toggling the check

box underneath the Evaluate bar, setting the threshold values of warning or alarm, respectively. You can review the commands generated based on your choices by clicking **Show Command**.

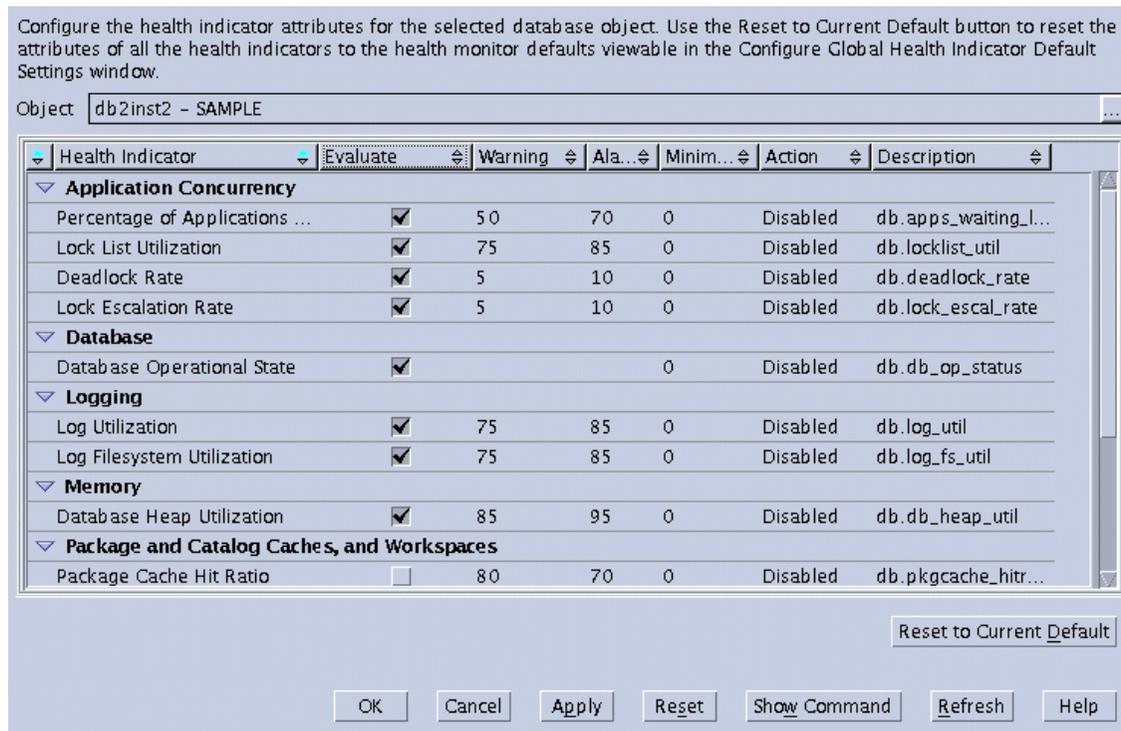


Figure 6-8 Configuring database object health indicators

Furthermore, you can set actions when the threshold values reach the warning or alarm level. To set up the action, choose a specific health indicator and click on this health indicator's action property button "...", as shown in Figure 6-9 highlighted in red ellipse. Another window appears, and this window can be used to enable the action property for the health indicator, and run a script or a task which is predefined in the task center.

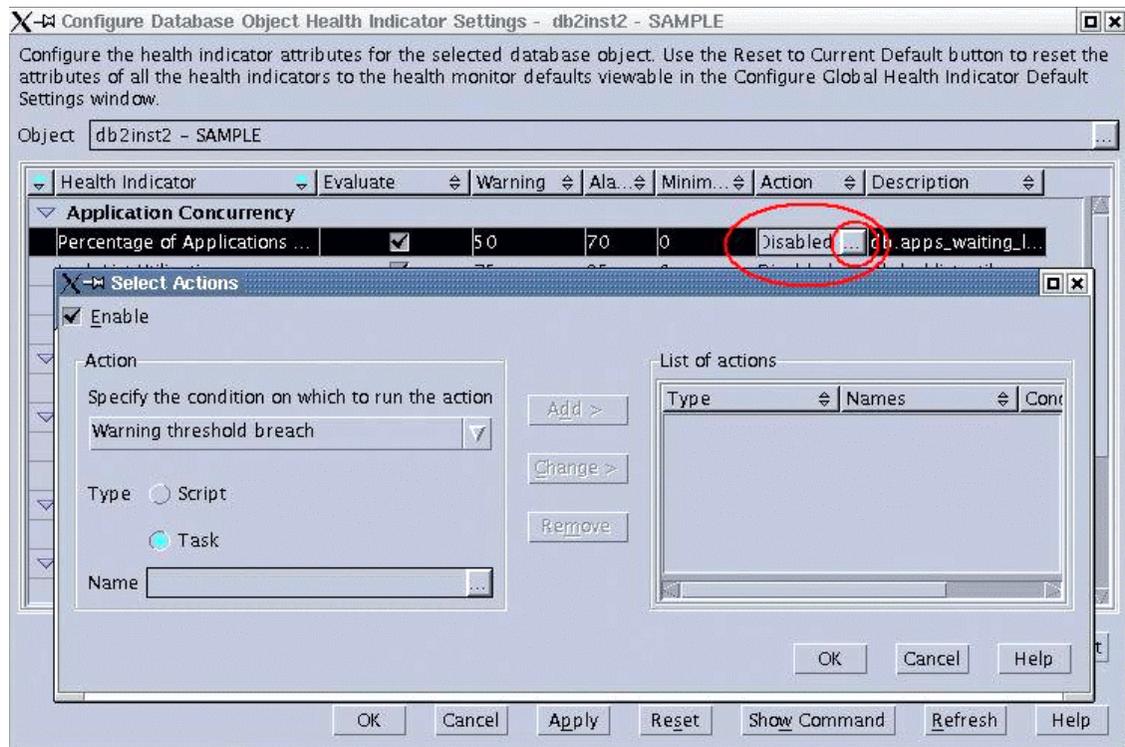


Figure 6-9 Setting up Action for specific health indicator

The configuration dialog launches in the context of the database selected in the Health Center. To view the tablespace health indicators, we need to choose the database object first. By clicking the “...” button beside Object as shown in the Figure 6-10 (where the button is highlighted in red), the database object list window will pop up. In the pop-up windows, keep expanding the object tree until the desired object is shown, and then select the object and click OK. Here we chose the DMS tablespace which is created beforehand in Step 1 of this example.

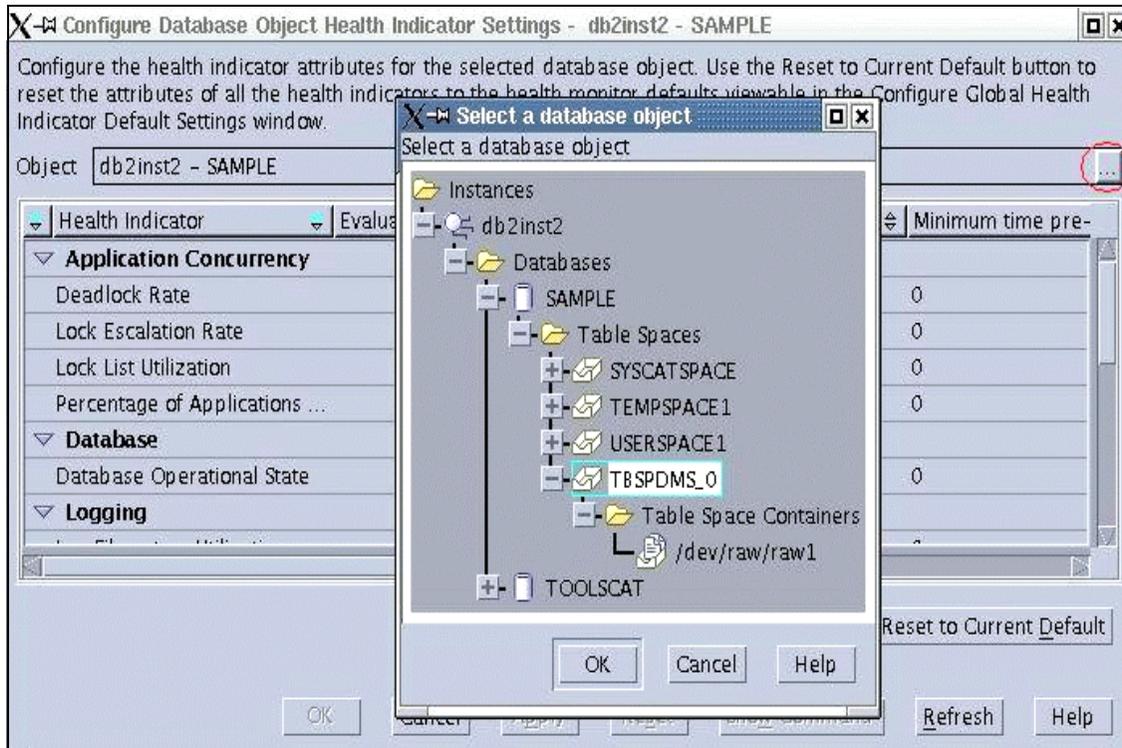


Figure 6-10 Choosing database object for health indicator setting

Once the object is chosen, you can modify the health indicators settings for database objects. Figure 6-11 shows the default setting from the tablespace level global settings. In our example, a threshold value of 80 is set for a warning. Likewise, another threshold value, 90, is set for alarm, and no action is defined. Both default values are expressed in percentage. To change the threshold value, click the number and a new button will appear for you to specify the new value.

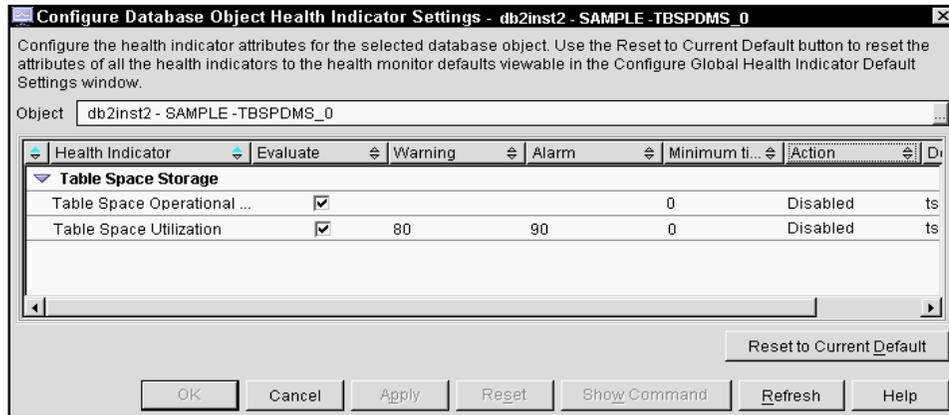


Figure 6-11 Database object health indicator setting

You can click Show Command to get the command based on the above choices, as shown in Figure 6-12. You can run the command in DB2 CLP to change configurations for the health indicator, or click OK to submit your changes.

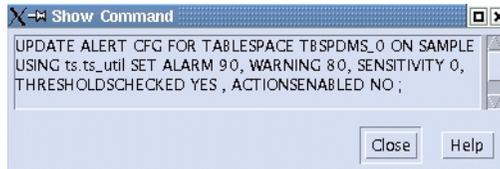


Figure 6-12 Set health indicator value command

You can use the DB2 CLP command GET ALERT CFG to verify the setting, as shown in Example 6-3.

Example 6-3 Using GET ALERT CFG to view health indicator status

```
db2inst2@UDBLN08:/db2home/db2inst2> db2 "get alert cfg for tablespaces "
```

Alert Configuration

```
Indicator Name           = ts.ts_op_status
Type                     = State-based
Sensitivity               = 0
Formula                  = ts.ts_status;
Actions                  = Disabled
Threshold or State checking = Enabled
```

```
Indicator Name           = ts.ts_util
Type                     = Threshold-based
```

```

Warning                = 80
Alarm                  = 90
Sensitivity             = 0
Formula = ((ts.ts_used_pages/ts.ts_usable_pages)*100);
Actions                = Disabled
Threshold or State checking = Enabled

```

3. Populate a desired amount of data into the tablespace.

We populate a certain amount of data into the tablespace to trigger the threshold. DB2 common table expression is used in our example to generate specified number of rows (Example 6-4). In our example 200,000 rows are inserted into the table and consume more than 11 MB space for table data and index pages. The total space for the table space is around 12 MB so the threshold values will be exceeded as expected.

Example 6-4 Populating data to trigger threshold for table space utilization

```

db2inst2@UDBLN08:/db2home/db2inst2/test> cat empins.sql
INSERT INTO EMP
WITH DT(ENO) AS (VALUES(1) UNION ALL SELECT ENO+1 FROM DT WHERE ENO < 200000 )
SELECT ENO, TRANSLATE(CHAR(INTEGER(RAND()*1000000)),
CASE MOD(ENO,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()*10957) DAYS AS HIREDATE,
INTEGER(10000+RAND()*200000) AS SALARY
FROM DT;

```

```

db2inst2@UDBLN08:/db2home/db2inst2/test> db2 -tvf empins.sql
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 via 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 window, 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 db2inst2. The notification log for the instance will be shown in the bottom pane of the window. You also can use the Notification Log Filter to set up a

filter to show health monitor notifications only. Figure 6-13 shows the notification log of our example. It shows that the value of current table space utilization is 98 percent and severity level is Alarm. The timestamp at that time the alarm condition was triggered is also shown.

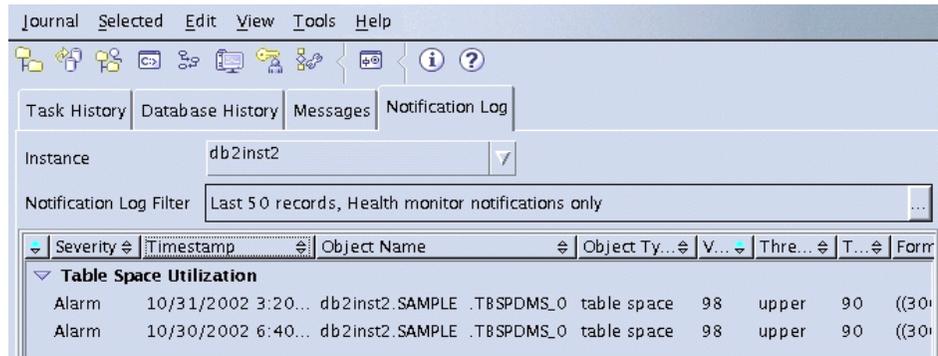


Figure 6-13 Administration notification log for health monitor indicator

An e-mail is sent to designated contact with content like the following e-mail shows (Figure 6-14).

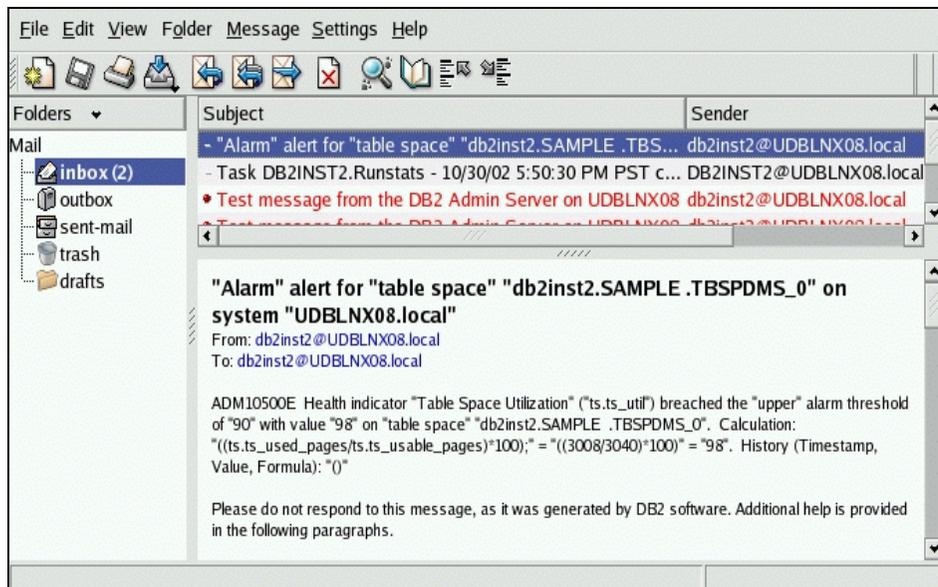


Figure 6-14 Notification for health indicator by e-mail

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 6-15, 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 tablespaces and containers. In the right pane of the window, the Health Center lists all the alerts.

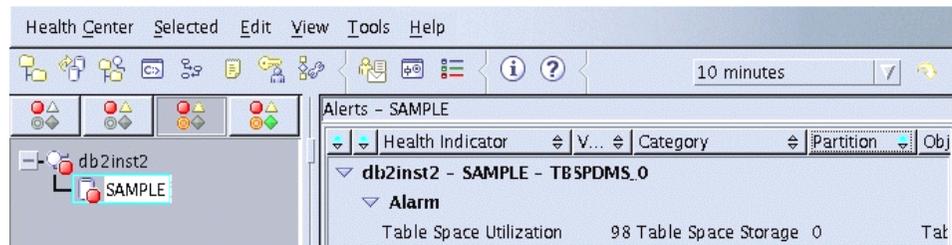


Figure 6-15 Monitor health indicators in Health Center

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 including the recommendation for this alert, choose Show Details; to disable the health indicator choose Disable Evaluation. Figure 6-16 shows you these options.

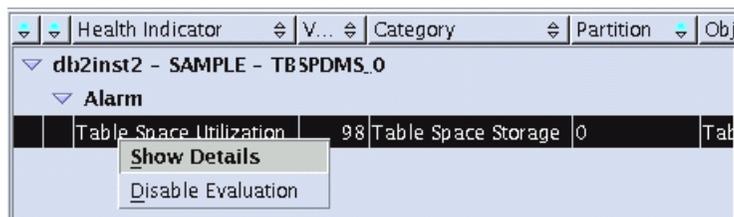


Figure 6-16 Choosing the options to handle an alert

If you chose Show Details, the following window (Figure 6-17) appears.

Details Recommendations

Object db2inst2 - SAMPLE - T8SPDMS_0 - 0

Alert value 98 [View History](#)

Formula ((3008/3040)*100)

Timestamp 10/31/2002 4:10:01 PM

Severity Alarm

Category Table Space Storage

Thresholds

Warning [Apply](#)

Alarm [Reset](#)

Additional Information

The table space growth rate is "0" from "10-31-2002 15:10:00.000695" to "10-31-2002 16:10:00.000695", and is "N/A" from "10-30-2002 16:10:00.000695" to "10-31-2002 16:10:00.000695". Time to fullness is projected to be "N/A" and "N/A" respectively.

Description

This health indicator tracks the consumption of storage for each DMS table space. The DMS table space is considered full when all containers are full. The indicator is calculated using the formula: (ts.used / ts.useable) * 100 where ts.used and ts.useable are the system monitor data elements Used Pages in Table Space and Useable Pages in Table Space, respectively. Table space utilization is measured as the percentage of space consumed, where a high percentage indicates less than optimal function for this indicator. The short term and long term growth rates, included in the additional information for this indicator, can be used to determine if current rate of growth is a short term phenomenon or consistent with long term growth. The calculation of time remaining as fullness is

Figure 6-17 Detailed information for an alert within Health Center

You can click the Recommendations tab to get recommendation actions for the current situation. In our example, three recommended actions are available for fixing our problem (Figure 6-18).

Details Recommendations

Number of actions 3

Action [<Select>](#)

- Investigate storage utilization
- Add a new stripe set
- Add new table space containers

Figure 6-18 Get recommendations for alert

You can choose any one of the actions which may be appropriate for your real conditions. In our example, we chose **Add new table space containers**. DB2 now provides a detailed explanation about the chosen action and function button to execute the action. See Figure 6-19.

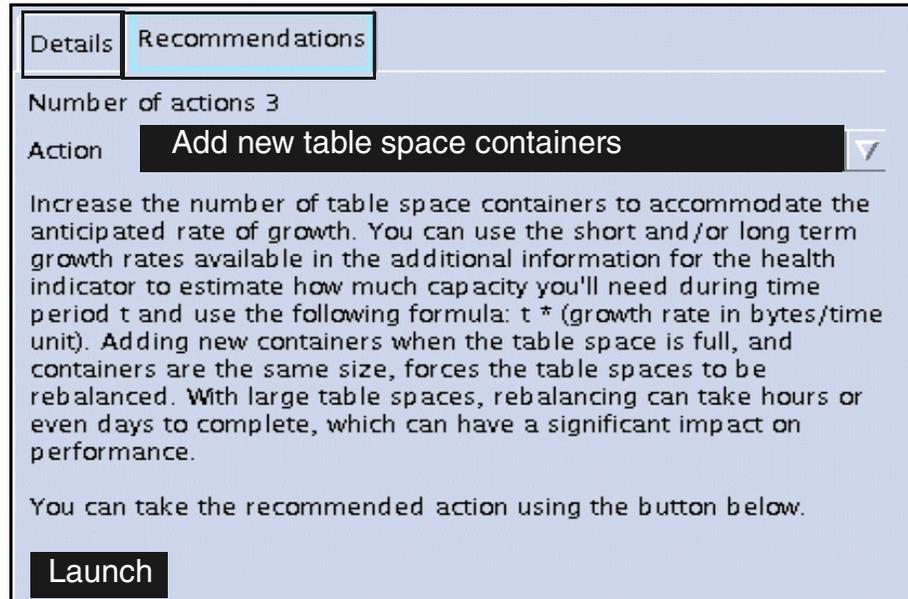


Figure 6-19 Recommendation details for specific recommendation choice

If you want to perform the recommended action right now, click **Launch**. For our sample, a window which can be used to add containers appears. You can choose the Containers tab, and then click **Add** in the window to add containers for the DMS tablespace. See Figure 6-20.

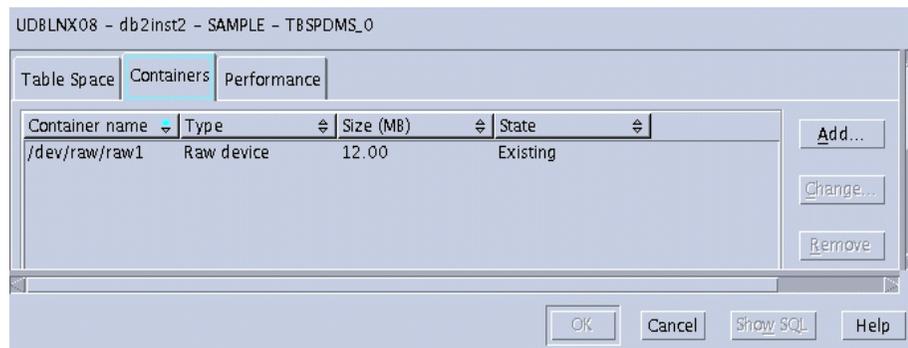


Figure 6-20 Take action according to recommendations given by DB2

Another way to get recommendations is via 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 *IBM DB2 UDB System Monitor Guide and Reference V8*, SC09-4847.

5. Set up additional follow-up actions.

There are situations when you need a more complex action plan for the error condition. This can be accomplished by using a predefined script or task. To enable the predefined script or task, choose a specific health indicator, click the health indicator’s action property, and then click the “...” button (refer to Figure 6-9 Setting up Action for specific health indicator on page 233 for an illustration). Another window will appear to allow you to choose a predefined script or task as the follow-up action. Figure 6-21 shows that a predefined task is chosen as the action for an alarm threshold breach condition for the tablespace health indicator.

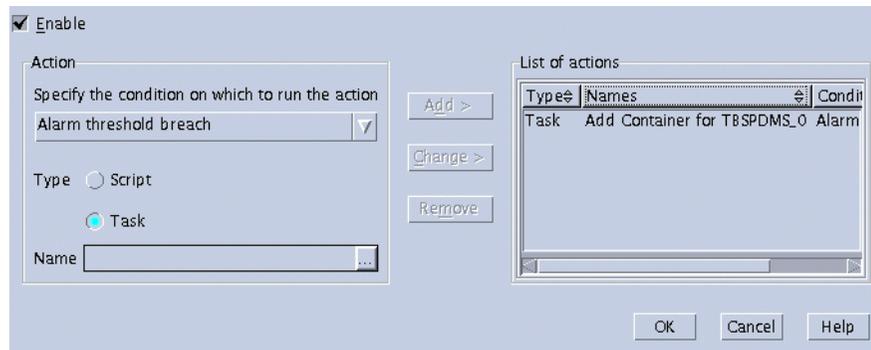


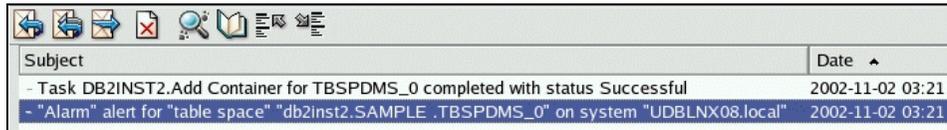
Figure 6-21 Setting action property for alert in Health Center

The task chosen in this window is predefined in the Task Center. Example 6-5 shows the commands within that task used to add an additional container.

Example 6-5 Commands used in the task for follow-up action

```
db2inst2@UDBLN08:/db2home/db2inst2/test> cat alt_tbspdms_0.db2
connect to sample;
alter tablespace tbspdms_0 add (DEVICE '/dev/raw/raw2' 12M);
connect reset;
```

The action plan specified in the Health Center is triggered by warning or alarm condition. If we had set up the action plan from the beginning, the add container task would have been executed once we populated the data. The result is shown in Figure 6-22.



Subject	Date ▲
- Task DB2INST2.Add Container for TBSPDMS_0 completed with status Successful	2002-11-02 03:21
- "Alarm" alert for "table space" "db2inst2.SAMPLE.TBSPDMS_0" on system "UDBLNX08.local"	2002-11-02 03:21

Figure 6-22 Notification for alert and follow-up task execution

This figure is captured from the e-mail system and it shows an alarm has been triggered on the table space. In our example, we have defined notification to contacts for the task, so the follow-up task's execution message was also sent.

It shows that once the alarm for the tablespace is triggered, the follow-up corrective action is executed automatically without user intervention. Meanwhile, DB2 also automatically notifies the related contacts with the detailed information associated to the alert. After corrective actions take effect, the health indicator will return to the normal state and keep monitoring for the system.

6.2 Log files for troubleshooting

In DB2 UDB Version 8, the former diagnostic log file, `db2diag.log`, which contains most information for troubleshooting is separated into two files. The first one is the DB2 Administration Notification Log (also known as DB2 Notify Log) which is intended for database and system administrators use and the content is easy to understand. The second one is DB2 diagnostic log which is intended primarily for DB2 customer support staff or someone who is familiar with viewing diagnostic log file.

The alert log is depreciated and not recommended to use in DB2 UDB Version 8. If you still want to use alert facility in DB2 UDB Version 8, you can enable it by setting DB2 registry variable `DB2_ALERT_LOG` to true.

6.2.1 DB2 administration notification log

The Notify 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. This also is the directory where db2diag.log resides. The name for notify log is <instance owner>.nfy. For example, /home/db2inst1/sqllib/db2dump/db2inst1.nfy, here the instance owner is db2inst1, and DIAGPATH is /home/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:

- 0: No administration notification messages captured. (Not 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 may 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 6-6 is part of the messages extracted 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 6-6 DB2 administration notification log

```
2002-11-01-12.15.23.061333 Instance:db2inst2 Node:000
PID:15683(db2agent (SAMPLE)) TID:1024 Appid:*LOCAL.db2inst2.041A71201519
data management sqlEscalateLocks Probe:2 Database:SAMPLE
```

```
ADM5500W DB2 is performing lock escalation. The total number of locks
currently held is "11382", and the target number of locks to hold is "5691".
^^
```

```
2002-11-01-12.15.23.114660 Instance:db2inst2 Node:000
PID:15683(db2agent (SAMPLE)) TID:1024 Appid:*LOCAL.db2inst2.041A71201519
data management sqlEscalateLocks Probe:3 Database:SAMPLE
```

```
ADM5502W The escalation of "11373" locks on table "DB2INST2.EMP" to lock
intent "X" was successful.
```

6.2.2 DB2 diagnostic log (db2diag.log)

Besides the notify log, you can also use the DB2 diagnostic log to help identify the cause of a problem that you are having with your Linux-based system. Once the data is collected, it can be examined by someone who is familiar with the problem, or provided to DB2 Customer Support staff for analysis, because sometimes this log file is not intuitive to understand — it may contain dumps or cryptic messages.

The file db2diag.log is located under the directory which is specified by the database manager configuration parameter DIAGPATH. 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, please 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.

The following log messages (Example 6-7) are extracted from db2diag.log. They are written into the log file when running the DB2 CLP command as below:

```
db2 update db cfg for sample using newlogpath /db2log
```

The part of db2diag.log shows that DB2 is trying to create a directory name /db2log/NODE0000 under /db2log, but the operating system prohibits DB2 from doing that and returned an error number "0d". It means permission denied (you can get the meaning of error number from errno.h include file). 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.db2grp1 /db2log** with root login, the problem is resolved.

Example 6-7 DB2 diagnostic log file

```
2002-11-06-21.34.00.213398 Instance:db2inst1 Node:000
PID:4373(db2agent (instance)) TID:8192 Appid:*NO.db2inst1.047006133400
oper system services sqlomkdirp Probe:100

errno: 0d00 0000                                     ....

2002-11-06-21.34.00.263248 Instance:db2inst1 Node:000
PID:4373(db2agent (instance)) TID:8192 Appid:*NO.db2inst1.047006133400
oper system services sqlomkdirp Probe:10

directory:2f64 6232 6c6f 672f 4e4f 4445 3030 3030      /db2log/NODE0000
...
```

6.2.3 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 haven't 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.

6.3 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 UDB, 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.

6.3.1 Top

Top 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 5 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 6-1.

Table 6-1 Frequently used command options for Top

Command option	Description
d	Specifies the delay between screen updates.
p	Monitors only processes with given process ID. This flag can be given up to twenty times.
i	Starts 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/on the display of CPU states, or sort the processes list by the memory occupancy percentage, or kill a process in the list. Table 6-2 shows some commonly used commands.

Table 6-2 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	Sorts tasks by memory usage.
W	Writes current setup to top configuration file ~/.toprc.

Example 6-8 shows using Top, where some interactive commands are used, such as 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 6-8 Using Top to monitor system resource usage

```

2:21pm up 2:18, 3 users, load average: 1.75, 1.77, 1.37
74 processes: 71 sleeping, 3 running, 0 zombie, 0 stopped
CPU states: 1.1% user, 0.9% system, 0.0% nice, 97.8% idle
Mem: 578608K av, 448292K used, 130316K free, 0K shrd, 78904K buff
Swap: 1753304K av, 0K used, 1753304K free 290556K cached

```

```

PID USER      PRI  NI  SIZE  RSS SHARE STAT %CPU %MEM  TIME COMMAND
 820 root        5  -10 23288 18M  5284 S <   0.1  3.2 16:34 X
1538 root       15   0 14136 13M  9328 S    0.0  2.4  0:12 nautilus
1545 root       15   0 12608 12M  8604 S    0.1  2.1  4:36 rhn-applet-gui
1536 root       15   0 10676 10M  8172 R    0.3  1.8 17:36 gnome-panel
1556 root       15   0  8648 8644  6752 S    0.0  1.4  0:08 gnome-terminal
1551 root       15   0  8468 8464  6828 S    0.0  1.4  0:02 gweather-applet

```

1418	root	15	0	8212	8208	6324	S	0.0	1.4	0:00	gnome-session
1496	root	15	0	6988	6984	5576	S	0.0	1.2	0:01	gnome-settings-
1494	root	15	0	6360	6360	5204	S	0.0	1.0	0:02	metacity
1540	root	16	0	5892	5888	4952	S	0.7	1.0	35:23	magicdev
1490	root	15	0	5088	5088	1972	S	0.0	0.8	0:01	gconfd-2
1543	root	15	0	4064	4064	3440	S	0.0	0.7	0:03	pam-panel-icon
738	xfst	15	0	3276	3276	884	S	0.0	0.5	0:00	xfst
771	root	15	0	2920	2920	2804	S	0.0	0.5	0:00	gdm-binary
1492	root	15	0	2248	2248	1848	S	0.0	0.3	0:00	bonobo-activati
1557	root	16	0	1520	1520	1152	S	0.0	0.2	0:00	bash
560	root	18	0	1468	1468	1224	S	0.0	0.2	0:00	sshd
3855	root	15	0	1448	1412	848	S	0.0	0.2	0:00	db2fmc
6561	root	15	0	1412	1412	1116	S	0.0	0.2	0:00	bash
695	postfix	15	0	1324	1324	1060	S	0.0	0.2	0:00	nqmgr

Regarding 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 designated processes only. You can specify up to 20 processes for Top to monitor by using the “p” command option. In addition, by taking advantage of the function provided by the UNIX/Linux shell, you can use “pgrep db2sampl” as the example shows to get the process ID for process db2sampl and add it into the Top monitor process list directly. 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 1 second.

Example 6-9 Using Top to monitor specified processes

```
[root@localhost root]# top d 1 p `pgrep db2sampl` p 3766 p 3767
```

9:16pm up 1:07, 3 users, load average: 2.54, 0.99, 0.38
3 processes: 3 sleeping, 0 running, 0 zombie, 0 stopped
CPU states: 6.3% user, 37.3% system, 0.6% nice, 55.6% idle
Mem: 255448K av, 242584K used, 12864K free, 0K shrd, 16156K buff
Swap: 265064K av, 0K used, 265064K free 148216K cached

PID	USER	PRI	NI	SIZE	RSS	SHARE	STAT	%CPU	%MEM	TIME	COMMAND
4365	db2inst1	15	0	7744	7728	3856	S	0.4	3.0	0:00	db2sampl
3766	db2inst1	15	0	40732	39M	40564	S	0.0	15.9	0:00	db2sysc
3767	db2inst1	15	0	24308	23M	24152	S	0.0	9.5	0:00	db2sysc

6.3.2 Vmstat

This tool 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 uninterrupted sleep.
- w: The number of processes swapped out but otherwise can be run. This field is calculated, but Linux never desperation swaps.

▶ **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**

These are percentages of total CPU time.

- us: user time
- sy: system time
- id: idle time

A sample report generated by `vmstat` is shown in Example 6-10.

Example 6-10 Using vmstat to observe virtual memory statistics

```
[root@UDBLN06 instance]# vmstat -n 1
```

procs			memory				swap		io			system			cpu	
r	b	w	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	
0	0	0	47144	8576	16800	466852	0	5	37	121	569	159	3	1	96	
4	0	2	47140	8572	16668	466336	4	0	2572	1160	2557	2225	48	20	33	
3	1	0	47140	8568	16672	466636	0	0	2560	2012	2190	1754	75	17	9	
2	0	0	47140	8568	15644	467908	0	0	3076	1536	2444	2065	62	19	20	
3	0	0	47140	8568	15656	467680	0	0	2564	1728	2535	2018	67	17	16	
3	1	0	47136	8572	15212	467792	4	0	3076	1684	2574	2030	68	22	11	
5	0	1	47136	8568	15112	467936	0	0	2564	1696	2559	2045	65	20	16	
4	1	0	47136	8568	15096	468916	0	0	2564	1412	2190	1678	59	15	26	
4	1	0	47136	8568	15092	468340	0	0	3072	1828	2573	2023	68	20	12	
1	0	0	47136	8572	14956	468048	0	0	2564	1720	2561	2117	66	22	13	
3	1	0	47132	8580	14968	468040	4	0	3080	1684	2543	1975	69	17	15	
3	1	0	47132	8572	14476	468560	0	0	3076	1720	2610	2046	66	20	15	
2	0	0	47132	8568	14476	469168	0	0	2560	1444	2194	1690	56	15	29	
2	1	0	47132	8568	14488	468792	0	0	2564	1540	2487	1965	63	21	17	
3	1	0	47128	8572	14504	468532	4	0	2572	1712	2522	2003	66	18	17	

6.3.3 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.

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.

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.

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 6-11 shows using `iostat` command to monitor CPU and device I/O statistics.

Example 6-11 Using iostat to get CPU and I/O statistics

```
[root@UDBLN06 root]# iostat 1
Linux 2.4.18-14 (UDBLN06)      11/06/2002

avg-cpu:  %user   %nice    %sys  %idle
           1.98    0.00    0.66  97.36

Device:            tps   Blk_read/s   Blk_wrtn/s   Blk_read   Blk_wrtn
dev3-0              5.44        50.72        119.68    1269102    2994466
dev3-1              0.00         0.01         0.00         304         0

avg-cpu:  %user   %nice    %sys  %idle
           64.00    0.00    19.00  17.00

Device:            tps   Blk_read/s   Blk_wrtn/s   Blk_read   Blk_wrtn
dev3-0            184.00       5120.00       3216.00     5120       3216
dev3-1              0.00         0.00         0.00         0         0

avg-cpu:  %user   %nice    %sys  %idle
           65.00    0.00    20.00  15.00

Device:            tps   Blk_read/s   Blk_wrtn/s   Blk_read   Blk_wrtn
dev3-0            208.00       6144.00       3328.00     6144       3328
dev3-1              0.00         0.00         0.00         0         0

avg-cpu:  %user   %nice    %sys  %idle
           69.00    0.00    18.00  13.00

Device:            tps   Blk_read/s   Blk_wrtn/s   Blk_read   Blk_wrtn
dev3-0            180.00       5136.00       3192.00     5136       3192
dev3-1              0.00         0.00         0.00         0         0
```

6.3.4 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 6-3 gives some frequently used command options for sar which are extracted from sar's man pages.

Table 6-3 Frequently used command options for sar

Options	Description
-A	This is equivalent to specifying -bBcdqrRuvvWy -I SUM -I PROC -n FULL -U 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.
-U	Reports CPU utilization for a given processor.
-v	Reports status of inode, file and other kernel tables.
-w	Reports system switching activity.
-x	Reports statistics for a given process.

Some samples of using sar command are provided below. In Example 6-12, at first, sar is started with “-A” command option and the output is saved in sartest.bin file. For testing purpose, you can generate some CPU load for sar to report; for example, loading data into DB2 database in a multiple partition environment. Then you can obtain the desired report from the saved file for different system activities. For example:

```
sar -u -s 11:09:00 -f sartest.bin|more
```

The above command can be used to get CPU utilization information since a specified starting time.

```
sar -w -c -s 11:09:10 -e 11:09:25 -f sartest.bin|more
```

The above command can be used to get process creation/fork and system context switch information.

And the command below can be used to obtain network statistics for a specific network interface.

```
sar -n DEV -s 11:09:25 -f sartest.bin|egrep "eth0|IFACE"
```

See Example 6-12 for more details.

Example 6-12 Using sar to monitor system activities

```
[root@UDBLN06 root]# sar -A -o sartest.bin 2 20
[root@UDBLN06 root]# sar -u -s 11:09:00 -f sartest.bin|more
Linux 2.4.18-14 (UDBLN06)      11/07/2002

11:09:01 AM      CPU      %user      %nice      %system      %idle
11:09:03 AM      all       0.00       0.00       0.00      100.00
11:09:05 AM      all       0.50       0.00       0.50      99.00
11:09:07 AM      all       0.00       0.00       0.50      99.50
11:09:09 AM      all       0.00       0.00       0.00     100.00
11:09:11 AM      all       1.00       0.00       1.00      98.00
11:09:13 AM      all       8.50       0.00      32.50      59.00
11:09:15 AM      all      14.50       0.00      48.00      37.50
11:09:17 AM      all       5.50       0.00       6.50      88.00
11:09:19 AM      all      17.50       0.00      11.00      71.50
11:09:21 AM      all      11.00       0.00      12.50      76.50
11:09:23 AM      all      23.50       0.00       9.00      67.50
11:09:25 AM      all      67.00       0.00      19.50      13.50
11:09:27 AM      all      62.00       0.00      18.00      20.00
11:09:29 AM      all      65.00       0.00      20.00      15.00
11:09:31 AM      all      63.00       0.00      19.00      18.00
11:09:33 AM      all      63.00       0.00      18.00      19.00
11:09:35 AM      all      66.00       0.00      18.00      16.00
Average:         all      27.53       0.00      13.76      58.71

[root@UDBLN06 root]# sar -w -c -s 11:09:10 -e 11:09:25 -f sartest.bin|more
Linux 2.4.18-14 (UDBLN06)      11/07/2002

11:09:11 AM      proc/s
11:09:13 AM       3.00
11:09:15 AM       4.50
11:09:17 AM       1.50
11:09:19 AM       0.00
11:09:21 AM       2.00
11:09:23 AM       0.00
11:09:25 AM       0.00
Average:         1.83

11:09:11 AM      cswch/s
```

```
11:09:13 AM    190.00
11:09:15 AM   1013.50
11:09:17 AM   1349.00
11:09:19 AM   2071.00
11:09:21 AM   2132.00
11:09:23 AM   1643.50
11:09:25 AM   2194.00
Average:      1399.83
```

```
[root@UDBLN06 root]# sar -n DEV -s 11:09:25 -f sarestest.bin | egrep "eth0|IFACE"
11:09:25 AM    IFACE  rxpck/s  txpck/s  rxbyt/s  txbyt/s
11:09:27 AM    eth0   1187.00  2459.00 1511133.00 3007010.50
11:09:29 AM    eth0   1218.00  2528.50 1558313.00 3078301.00
11:09:31 AM    eth0   1219.00  2534.00 1558488.00 3085869.00
11:09:33 AM    eth0   1209.00  2504.50 1541410.50 3051547.50
11:09:35 AM    eth0   1181.00  2502.50 1504659.50 3050603.50
Average:      eth0   1202.80  2505.70 1534800.80 3054666.30
```

As mentioned in 6.3.3, “iostat” on page 250, the sar utility is contained in sysstat package; before using this command, you need to make sure sysstat package is installed in your system.

For details about using the sar command, refer to the man help pages for sar or other Linux documentation.

6.3.5 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, pstree 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 pkill 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.



DB2 integrated tools and wizards

This chapter discusses some DB2 integrated tools available for the Linux platform for administration and development purposes. DB2 UDB provides a multitude of tools that make it much easier and more efficient to install and administer your database system, to continuously manage increasing volumes of data, to handle the tremendous growth in the number of users, to deliver better performance and to support different kinds of applications, and so forth.

Besides GUI tools, DB2 also provides many command line tools to assist you in administering the database system, such as you can use the DB2 Command Line Processor to execute DB2 administration commands and SQL statements, use `db2mtrk` to track DB2 memory usage by instance and database, as well as `db2` agents, and use `db2look` to extract the needed DDL statements to reproduce the database objects.

Some tools and wizards have been discussed in other chapters of this book, as shown in Table 7-1 on page 256. For information regarding these tools and wizards, refer to those chapters.

Table 7-1 Tools and wizards covered in other chapters

Tool/ Wizard	Usage	Chapter Number
db2setup	DB2 installation and instance creation	2
db2_install	DB2 package installation	2
Configuration Advisor	DB2 instance and database parameter configuration	3
Command Center	GUI tool for db2 commands and SQL statements execution	3
License center	Manage db2 licenses information	3
Backup/ Restore	Database backup and recovery	5
Reorg	Reorganize table and indexes	5
Runstats	Statistics information collection	5
Export/Import	Table data export and import	5
Load	Load data into database	5
Task Center	Task setup and scheduling	5
Health Center	System health monitor configuration	6
Development Center	Integrated tool for stored procedure, UDF development	8

In this chapter, we select a few tools and wizards to discuss in detail. For the entire DB2 UDB tools and wizards collection, refer to *IBM DB2 UDB Guide to GUI Tools for Administration and Development*, SC09-4851. That book focuses on introducing GUI tools, and many tools are covered. In addition, you can also read *IBM DB2 UDB Command Reference V8*, SC09-4828, to acquire more information about tools and utilities provided by system and DB2 CLP commands.

In general, to invoke GUI tools in DB2 UDB for the Linux platform, you can start them from DB2 Control Center. To start DB2 Control Center, click the Control Center icon available on the Linux desktop, or enter the command “db2cc” from the command line, and then press enter. Most GUI tools can be invoked from the Control Center, as Figure 7-1 shows; there is a tools menu available for you to choose the desired tool. In addition, another pop-up window which contains choices of wizards will show up if you clicked the “Wizards...” item within tools

menu. You can choose Create Table Space Wizard to create a tablespace step-by-step under DB2's instruction, or select the Load Wizard to load data to a single or multiple partition environment. Most of the time these tools or wizards are also available in the pop-up menu when you right-click specific objects in the navigation tree; for example, if you choose **Databases**, and right-click it, then Create Database Wizard will be displayed in the pop-up menu.

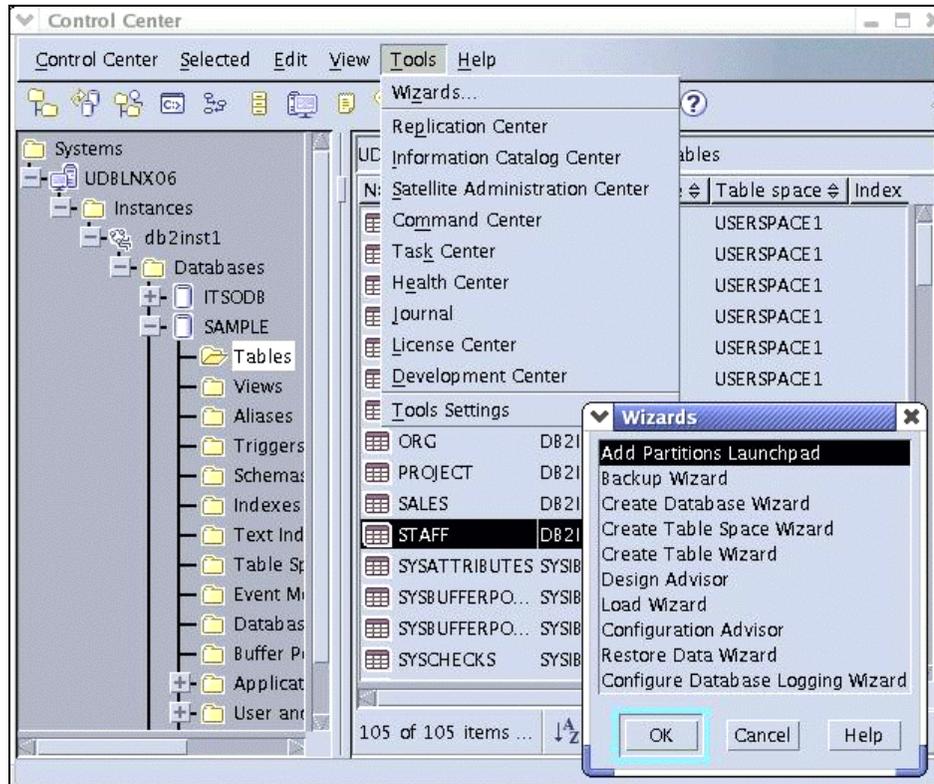


Figure 7-1 Tools and Wizards menu in Control Center

Tip: If the icons for DB2 UDB are not displayed in Linux desktop, you can use the DB2 Icon Generation Tool for Linux to add those icons. To invoke the Icon Generation Tool, enter “db2icons <instance_owner>” from command line, or just enter “db2icons” to get usage help. After you have added the icons for the specified user, if you log in to the system using that user, DB2 Icons will appear to the user on the desktop.

The following tools and wizards are discussed in this chapter:

- ▶ **Design Advisor:** Also available in command line form via `db2adv`, it can be used to recommend suitable indexes for your tables according to the given workload or to verify if a virtual index may be helpful for the given workload.
- ▶ **Visual Explain:** It can be used to obtain access plan for specified SQL statements in a graph and help you tune the SQL statements and your database manager configuration parameters to improve performance. Command line tools are also available to gain access plan information for SQL statements, such as `db2exp1n` (for static SQL), `dynexp1n` (for dynamic SQL), and so on.
- ▶ **Memory Visualizer:** Also available in command line form via `db2mtrk` (Memory Tracker), it can be used to monitor memory usage for DB2 instance.

7.1 Design Advisor

The Design Advisor can assist you in designing and defining suitable indexes for your tables according to the workload you have, or help you verify if a virtual index is appropriate for the database system. It is particularly beneficial for designing indexes for very large tables, as sometimes you cannot make sure if the index you want to create is really helpful for your application and you need to create it first to test if it works. But for tables with a large volume of data, creating an index usually takes long time and considerable disk space is also required; Moreover, sometimes you will find that the index you just created doesn't help anything. In such a situation, the Design Advisor would be the best tool to help you, because the Design Advisor works depending on the virtual indexes with no need to create real indexes.

There are two terms you need to understand before using the Design Advisor.

▶ **Workload**

A workload is a set of SQL statements with specified execution frequency which the database manager has to process during a given period of time. The SQL statements can include SELECT, INSERT, UPDATE, and DELETE statements. For example, during one month your database manager may have to process 1,000 INSERTs, 10,000 UPDATEs, 10,000 SELECTs, and 1,000 DELETEs. The advising engine uses this workload information in conjunction with the database information, such as statistics to recommend indexes. The goal of the advising engine is to minimize the total workload cost.

▶ **Virtual index**

Virtual indexes are indexes that do not exist in the current database schema. These indexes may be either recommendations that the advise facility has

made, or indexes that you want the advise facility to evaluate. Virtual index information is defined and manipulated in the ADVISE_INDEX table.

The Design Advisor is available in both graphical form and command line form. The GUI form that can be started from DB2 Control Center is the recommended method to use the Design Advisor as it provides extensive help pages that can answer your questions, and it can also help you to construct a workload by looking for recently executed SQL statements, or looking through the recently used packages, or let you add SQL statements manually.

An example is provided below to demonstrate the usage of Design Advisor in graphical form. You can start it from DB2 Control Center as shown in Figure 7-2, or you can invoke it from the Wizard pop-up window after you choose **Tools** → **Wizards** in the Control Center.

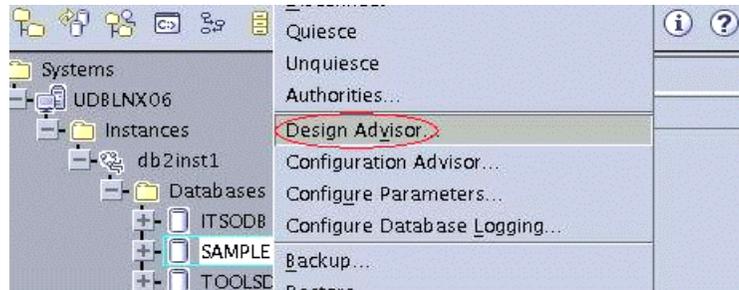


Figure 7-2 Start Design Advisor from Control Center

The first window (Figure 7-3) for the Design Advisor is an introductory window. It tells you what the advisor is for and which database you are working against. You can click **Next** to define the workload.

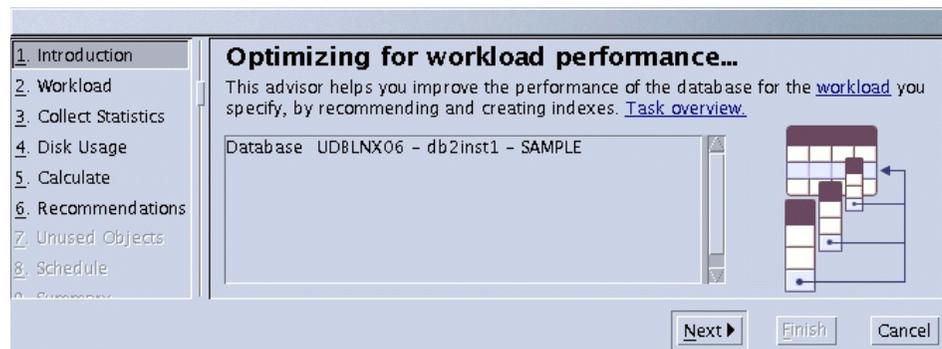


Figure 7-3 Design Advisor introduction

In the workload definition window, as Figure 7-4 shows, at first, you need to specify a name for the workload, then you click **Add** to add a SQL statement into the workload, with SQL statement name and execution frequency specified. Adding multiple SQL statements are also allowed here, you simply need to select **Add** again to add an additional SQL statement.

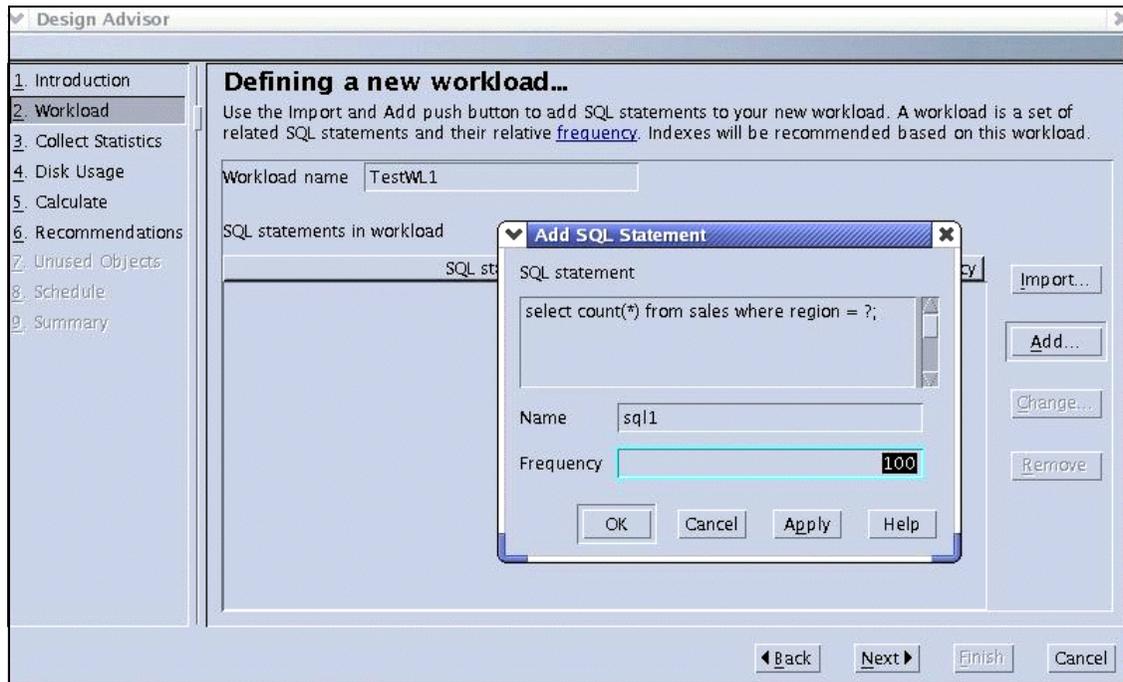


Figure 7-4 Defining workload for Design Advisor

You can also choose **Import**, which is located on the right side of the window to import a workload from explained SQL statements, as Figure 7-5 shows.

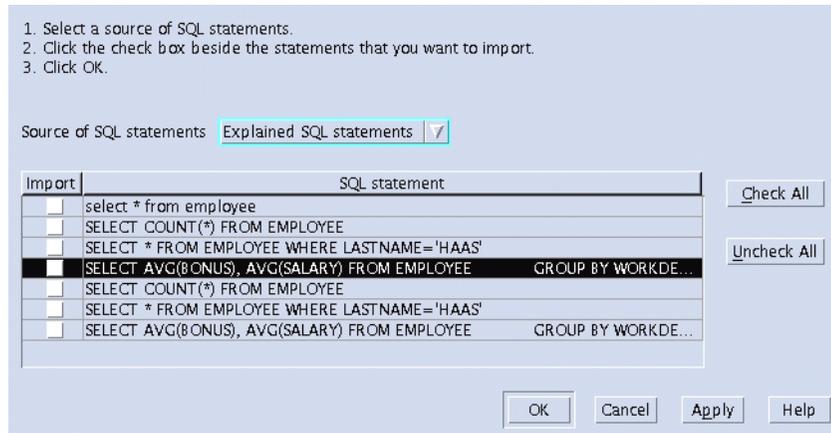


Figure 7-5 Importing explained SQL statements to define a workload

The next step is to specify which tables need to collect statistics. As mentioned in the previous paragraph, the advising engine uses workload information in conjunction with the database information such as statistics to recommend indexes. Therefore, if up-to-date statistics for the involved tables don't exist, you need to collect them first to make the recommended indexes usable. Figure 7-6 shows that some tables are chosen for statistics updating.

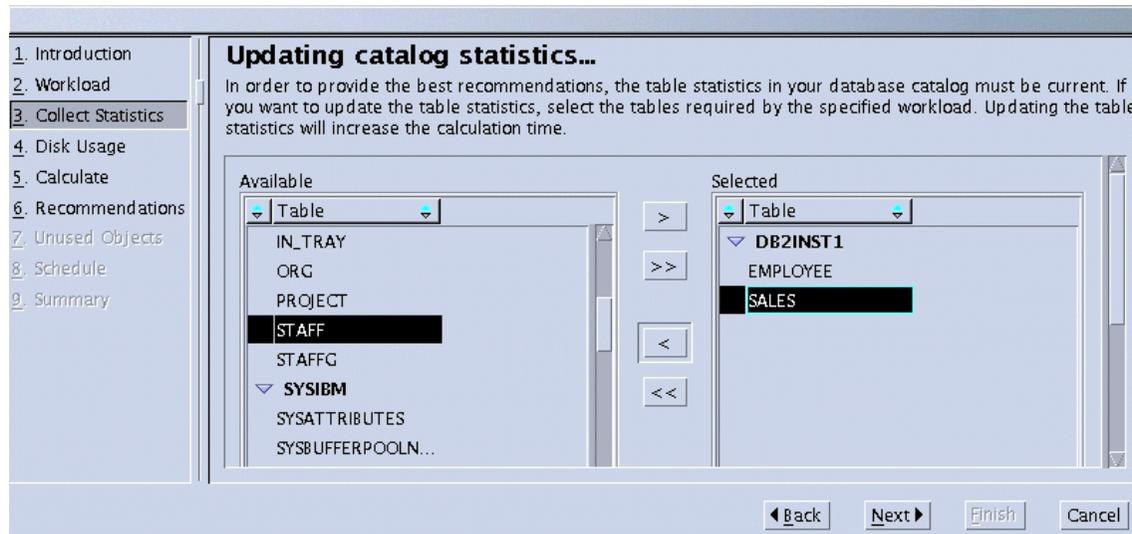


Figure 7-6 Updating catalog statistics for Design Advisor

Then you can specify the default table space and combined maximum disk space for the recommended objects in Disk Usage window. You can specify when the

advisor engine should calculate the recommendations in the Calculate window. If you want to start the calculation at once, choose **Now** in the window, and then recommendations are given in the Recommendations window, as shown in the Figure 7-7.

Here if you want to change the name for the recommended index instead of using the name given by DB2, you can click the original name shown in the Name field, and then input the name you desire. You can click **Show SQL** to obtain detailed SQL statements to create the index, and you can also toggle the check box in the Create field to point out whether or not you want to have the index created by DB2.

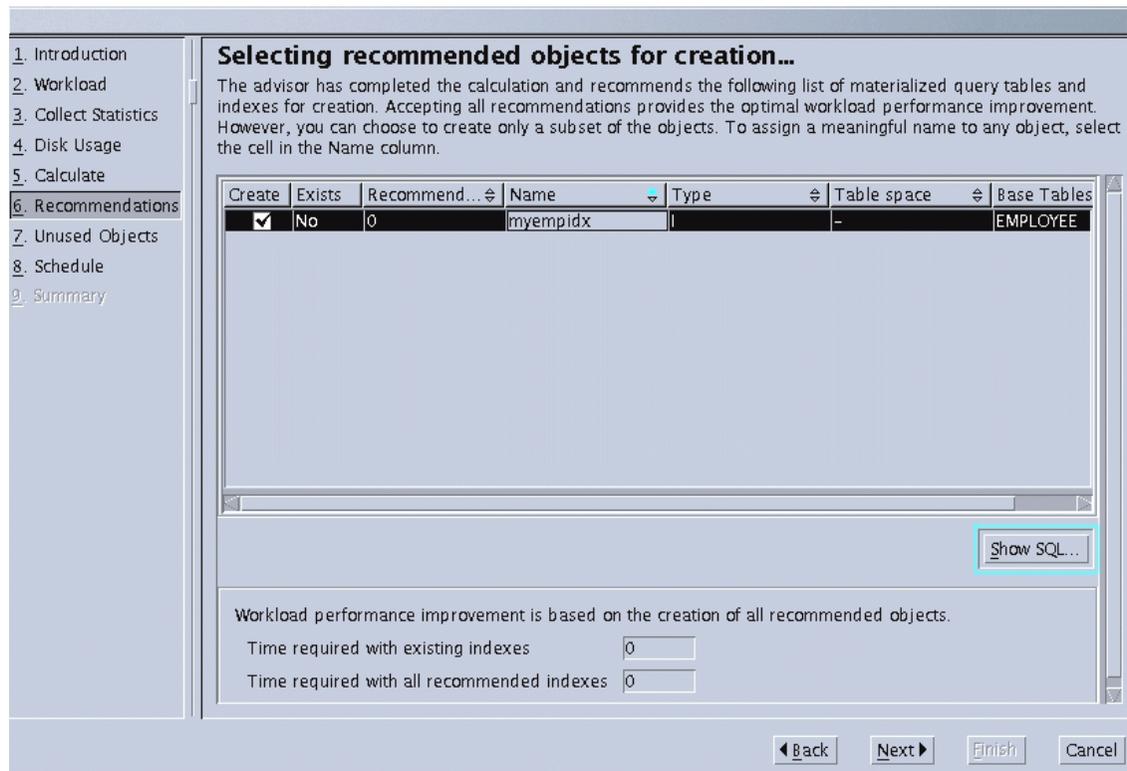


Figure 7-7 Recommended indexes by Design Advisor

If some indexes for the chosen tables already exist, the Design Advisor can tell you if these indexes remain unused in the specified workload. Then you can specify if you want to drop those unused indexes — make sure that dropping the “unused” indexes won’t impact other workloads before you choose to do that.

The remaining step is to schedule the running of recommended SQL statements, including create index and drop index, and then the related indexes will be created or dropped.

You can also run Design Advisor from the command line, if you are familiar with this. Example 7-1 consists of three steps. The first step is to define a workload, the second step is to run statistics for the related tables, and then the third step is to run the db2advis utility from the command line to obtain recommendations. From the output you can see that the timerons for workload running with and without recommended indexes are displayed, and the improvements by using the recommended indexes are also evaluated. In addition, by using the parameter -o, the index creation script of the recommendations can be placed into a file.

Example 7-1 Using db2advis to recommend index creation for specific workload

```
$ cat sample.sql
--#SETFREQUENCY 100
select count(*) from sales where region = ?;
--#SETFREQUENCY 3
select projno, sum(comm) tot_comm from employee, emp_act
where employee.empno = emp_act.empno and
employee.job='DESIGNER'
group by projno
order by tot_comm desc;
--#SETFREQUENCY 50
select * from sales where sales_date = ?;

$db2 "runstats on table db2inst1.employee with distribution allow write access"
DB20000I The RUNSTATS command completed successfully.
$db2 "runstats on table db2inst1.sales with distribution allow write access"
DB20000I The RUNSTATS command completed successfully.
$db2 "runstats on table db2inst1.emp_act with distribution allow write access"
DB20000I The RUNSTATS command completed successfully.

$db2advis -d sample -i sample.sql -t 0 -o ad1.sql
execution started at timestamp 2002-11-11-17.33.32.188732
  found [3] SQL statements from the input file
  recommending indexes...
Initial set of proposed indexes is ready.
Found maximum set of [4] recommended indexes
Cost of workload with all indexes included [1.258857] timerons
total disk space needed for initial set [ 0.035] MB
total disk space constrained to [ -1.000] MB
  4 indexes in current solution
[101.4306] timerons (without indexes)
[ 1.2589] timerons (with current solution)
[%98.76] improvement
```

```

Trying variations of the solution set.
--
--
-- LIST OF RECOMMENDED INDEXES
-- =====
-- index[1],    0.009MB
  CREATE INDEX IDX021111173333163 ON "DB2INST1"."SALES" ("REGION" DESC) ;
  COMMIT WORK ;
  --RUNSTATS ON TABLE SALES FOR INDEX IDX021111173333163 ;
  COMMIT WORK ;
-- index[2],    0.009MB
  CREATE INDEX IDX021111173334209 ON "DB2INST1"."SALES" ("SALES_DATE" ASC,
"SALES" ASC, "REGION" ASC, "SALES_PERSON" ASC) ;
  COMMIT WORK ;
  --RUNSTATS ON TABLE SALES FOR INDEX IDX021111173334209 ;
  COMMIT WORK ;
-- index[3],    0.009MB
  CREATE INDEX IDX021111173333164 ON "DB2INST1"."EMP_ACT" ("PROJNO" ASC,
"EMPNO" ASC) ;
  COMMIT WORK ;
  --RUNSTATS ON TABLE EMP_ACT FOR INDEX IDX021111173333164 ;
  COMMIT WORK ;
-- index[4],    0.009MB
  CREATE INDEX IDX021111173333177 ON "DB2INST1"."EMPLOYEE" ("JOB" ASC, "COMM"
ASC, "EMPNO" ASC) ;
  COMMIT WORK ;
  --RUNSTATS ON TABLE EMPLOYEE FOR INDEX IDX021111173333177 ;
  COMMIT WORK ;
-- =====
DB2 Workload Performance Advisor tool is finished.

```

7.2 Visual Explain

You can use Visual Explain to capture and view information about the access plan chosen by the DB2 optimizer for static or dynamic SQL statements as a graph. Here an access plan is a cost estimate of resource usage for a query, which is based on the available information, such as statistics for tables and indexes, instance and database configuration parameters, bind options and query optimization level, and so on. An access plan also specifies the order of operations for accessing the data.

The access plan acquired from Visual Explain can help you understand how individual SQL statements are executed so that you can tune the statements and the database manager configuration parameters to improve performance.

In general, you collect and use explain data for the following reasons:

- ▶ To understand how the database manager accesses tables and indexes to satisfy your query. For example, you can determine whether or not an index was used to access a table.
- ▶ To evaluate your performance tuning actions when you change some aspect of the database manager, the SQL statements, or the database. You should examine the explain data to find out how your action has changed performance.

Figure 7-8 shows you the procedure for how Visual Explain works. You need to have your tables and indexes ready at first, then reorganize tables and indexes if appropriate, and gather statistics for those tables and indexes. In addition, a change to some configuration parameters may be required. Then you can provide your SQL statements with the specified precompilation or bind option to DB2, and DB2 optimizer will take care of the remaining items for you.

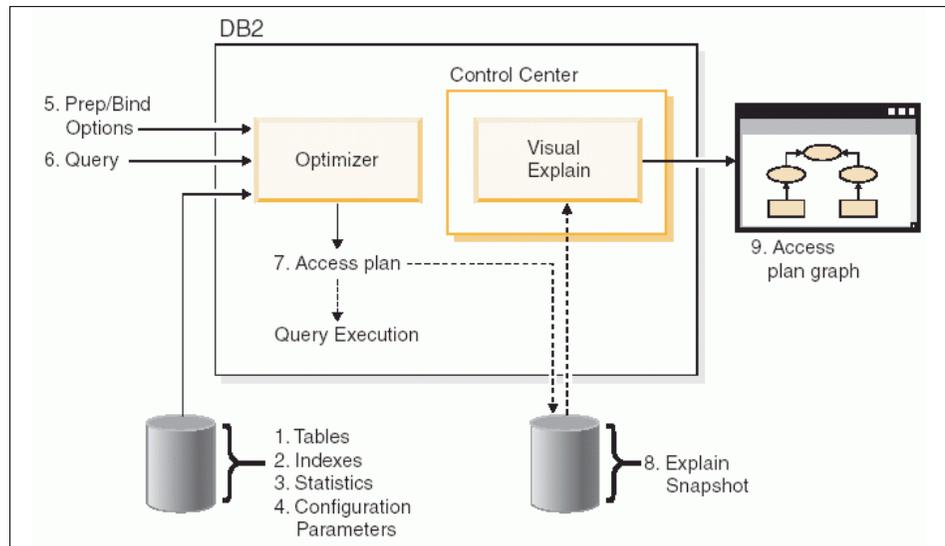


Figure 7-8 Visual Explain working procedure

You can start Visual Explain in the DB2 Control Center by right-clicking a database name and selecting either **Show Explained Statements History** or **Explain SQL**, as shown in Figure 7-9.

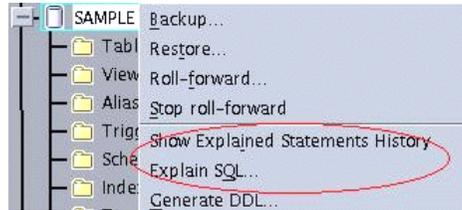


Figure 7-9 Start Visual Explain

You can input an SQL statement manually or import the SQL statement via the Get button available in the Explain SQL window. You can also change the optimization class for the SQL statement in the same window. Then you can click OK to get the result. Another window appears and shows you the access plan for the SQL statement (that you just provided) in a graph, as shown in Figure 7-10. You can right-click the object or operation in the window and then choose a menu item available in the pop-up menu to obtain more details for the object or operation. For example, you can gain the statistics for a table by choosing the Show Statistics menu item from the pop-up menu for a table object.

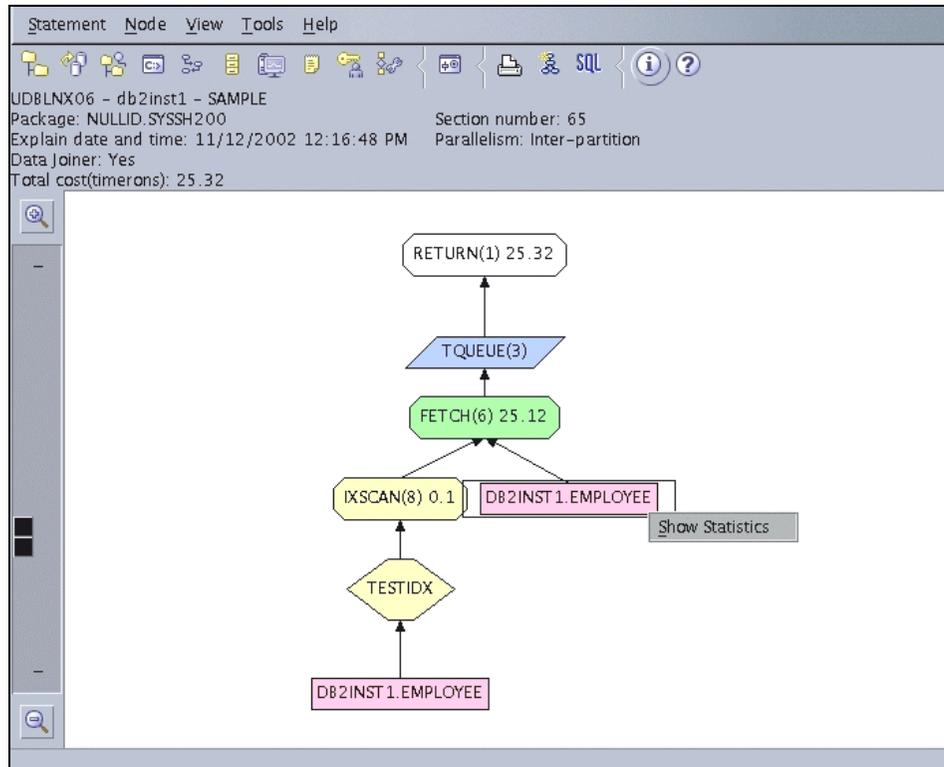


Figure 7-10 Access plan shown in Visual Explain

DB2's access plan can be obtained via the command line tools as well; for details regarding Visual Explain or the command line tools, refer to the *“Administration” Guides* (listed in “Other publications” on page 303).

7.3 Memory Visualizer and Memory Tracker

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 use it on its own as a monitoring tool or invoke it from a Health Center recommendation.

You can start Memory Visualizer in the Control Center by right-clicking an instance and selecting View memory usage. The Memory Visualizer window appears Figure 7-11.

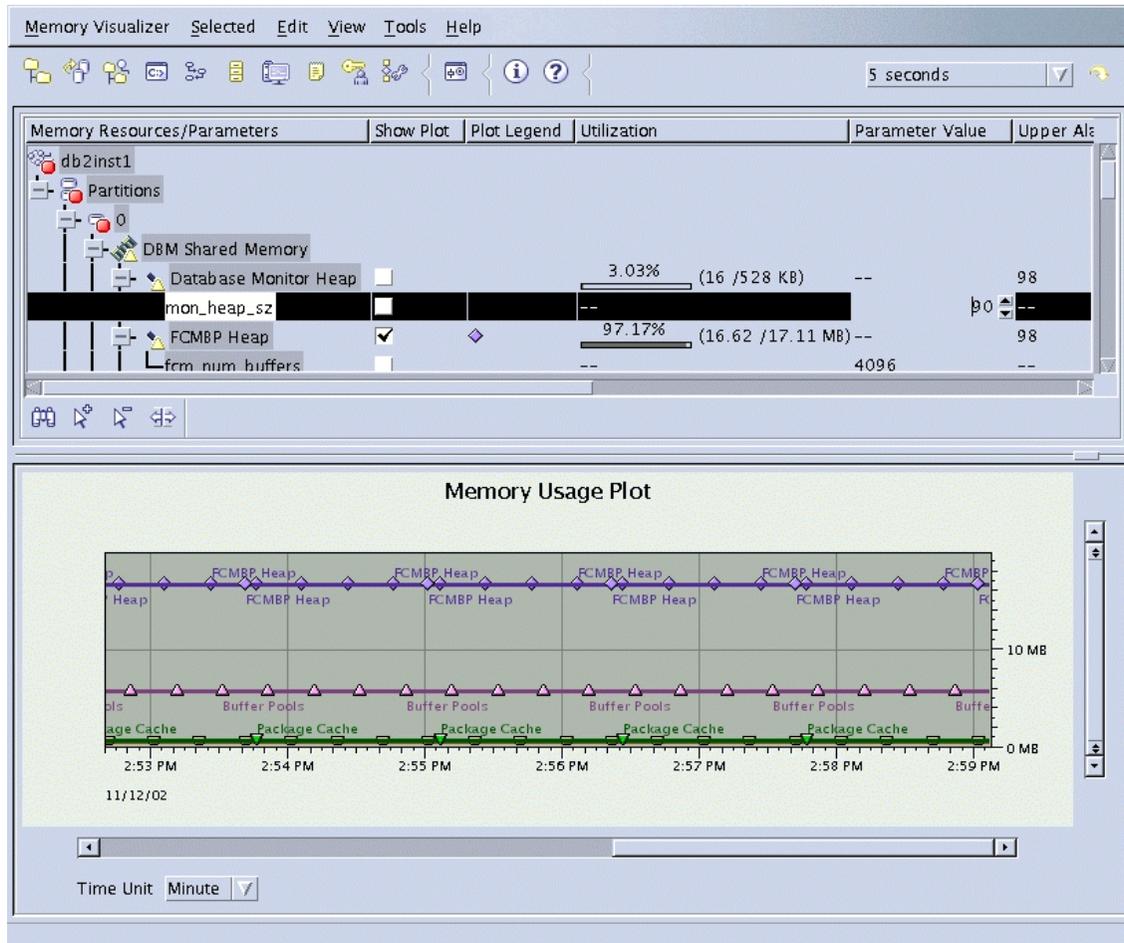


Figure 7-11 Using Memory Visualizer to monitor DB2 memory usage

Within this window, 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 window, 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.

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`. You can obtain detailed help message for this tool by entering the following command in a Linux terminal:

```
db2mtrk -h
```

Example 7-2 shows you how to use `db2mtrk` to monitor memory usage by DB2 instances and databases.

Example 7-2 Using db2mtrk to monitor DB2 memory usage

```
$ db2mtrk -i -d
```

```
Tracking Memory on: 2002/11/12 at 15:29:28
```

```
Memory for instance
```

monh	other	fcmbp
16.0K	6.9M	16.6M

```
Memory for database: SAMPLE
```

utilh	pckcacheh	catcacheh	bph	bph	bph	bph
16.0K	640.0K	160.0K	4.1M	656.0K	400.0K	272.0K
bph	lockh	dbh	other	appctlh	appctlh	appctlh
208.0K	448.0K	1.5M	54.7M	16.0K	16.0K	16.0K
appctlh						
320.0K						

```
$ db2 connect to sample
```

```
Database Connection Information
```

```
Database server      = DB2/LINUX 8.1.0
SQL authorization ID = DB2INST1
Local database alias = SAMPLE
```

```
$ db2 "alter bufferpool ibmdefaultbp immediate size 10000 "
```

```
DB20000I The SQL command completed successfully.
```

```
$ db2mtrk -i -d
```

```
Tracking Memory on: 2002/11/12 at 15:30:12
```

Memory for instance

monh	other	fcmbp
16.0K	6.9M	16.6M

Memory for database: SAMPLE

utilh	pckcacheh	catcacheh	bph	bph	bph	bph
16.0K	640.0K	160.0K	45.3M	656.0K	400.0K	272.0K
bph	lockh	dbh	other	appctlh	appctlh	appctlh
208.0K	448.0K	1.5M	54.7M	16.0K	16.0K	16.0K
appctlh						
320.0K						

In the preceding example, the buffer pool size of IBMDEFAULTBP is changed from default 1,000 4K pages to 10,000 4K pages dynamically. You can monitor the change made by DB2 through db2mtrk very clearly. The first **db2mtrk -i -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” is 4.1 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 -i -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 45.3M (highlighted in bold) now.

Note: 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 hidden buffer pools created by DB2 for internal use.

For a detailed explanation about command parameters and a field description of the output of db2mtrk, refer to the online help message of the Memory Tracker tool.



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 UDB 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

8.1 Application configuration

DB2 supports local and remote application development. The database always appears as a local database to the application. Figure 8-1 illustrates two different development configurations. The left side of the diagram represents a local development environment, in which the database and application reside on the same machine. On the right side there is a representation of a client-server development environment, in which the database and application are located on separate machines. No matter in which configuration, the database always appears as a local database to the application.

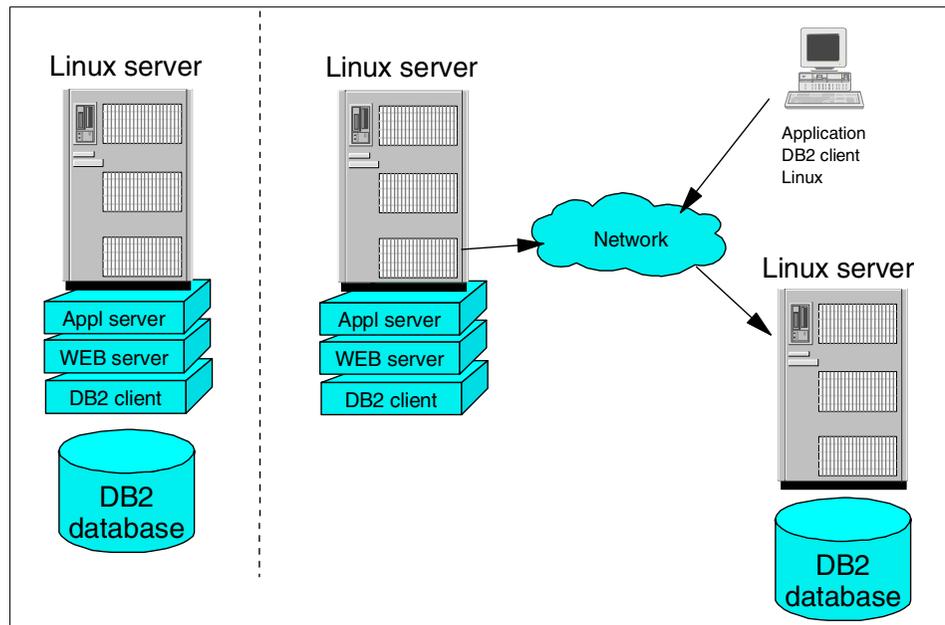


Figure 8-1 Development configuration

One of the special DB2 UDB features is that multiple DB2 UDB versions can coexist in the same system. For example, you can run a DB2 Version 7.2 instance and DB2 Version 8.1 instance on the same machine. Refer to Figure 8-2 for an example of this scenario. With DB2 Version 8, it is now possible to run not only multiple versions of DB2, but also different releases of DB2 on the same machine. For instance, it will be possible to run V8.1 FixPak 1 and V8.1 FixPak 2 in different instances at the same machine. Before Version 8, it was only possible to run one release and FixPak level within a version level, that is, DB2 Version 7.2 FixPak 4 and Version 7.2 FixPak 5 could not run on the same server.

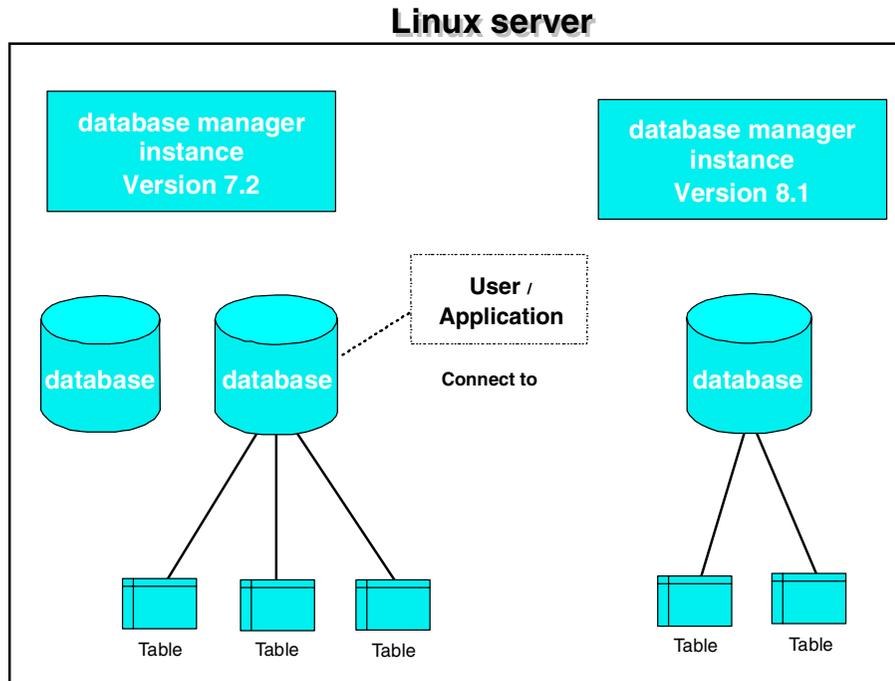


Figure 8-2 One host with multiple database manager instance versions

8.2 DB2 application objects

DB2 UDB supports stored procedures (SP), user defined functions (UDFs) and triggers, all of which are used to develop applications. DB2 V8 also provides a graphical tool called the Development Center, which is a component of the Application Development Client. This tool is used to develop SPs and UDFs. See 8.4, “Application tools” on page 284 for more information about this tool.

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 will be executed when a certain event occurs against a DB2 table. Types of events that may invoke triggers are Insert, Update, or Delete to a given DB2 table. Triggers may be set up 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 Chapter 5, “Statements” in the “*SQL References*” (listed in “Other publications” on page 303).

The following scenario is given in our sample database.

When updating either the *salary* column or *bonus* column in the table EMPLOYEE, the trigger saves the old data in a history table for auditing requirements.

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);
```

Now we modify any 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 Every updates force a insert in to 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@udb1nx02:~/tpc> db2 "select * from employee_hist"
```

EMPNO	SALARY_OLD	SALARY_NEW	BONUS_OLD	BONUS_NEW	CHG_DATE
000110	70000.00	45000.00	900.00	900.00	2002-11-06-14.13.23.287844
000070	65000.00	55000.00	700.00	700.00	2002-11-06-14.13.22.795208
000170	24680.00	55000.00	500.00	500.00	2002-11-06-14.13.23.766978
000070	55000.00	65000.00	700.00	700.00	2002-11-06-14.13.24.254235

4 record(s) selected.

8.2.2 Use defined functions (UDFs)

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 a table function. UDFs can be written in languages such as C or Java, as well as in pure DB2 SQL. Use the **create function** DDL statement to register a function in the database.

Note: The create function statement is explained in more detail in Chapter 5, “Statements” in the “SQL References” (listed in “Other publications” on page 303).

The following example creates a UDF called *salary_diff* which receives an employee ID (EMPNO) as input and calculates the difference between the old and new salary values from the *employee_hist* table which is populated through our trigger (see Example 8-2 on page 274). To get 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 a 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)

```

```

db2inst1@udblnx02:~/tpc> 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: The statement `select max(chg_date)` is needed, because for each `empno` there can exist more than one row in `employee_hist` table, but a UDF can return only one row.

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.

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. In Example 8-4, there is 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`.

Note: The `create procedure` statement is explained in more detail in Chapter 5, “Statements” in the “*SQL References*” (listed in “Other publications” on page 303).

Example 8-4 A simple example of an SQL procedure

```
db2inst1@udb1nx02:~/sql1lib/samples/sqlproc> db2 connect to sample
```

Database Connection Information

```
Database server      = DB2/LINUX 8.1.0
SQL authorization ID = DB2INST1
Local database alias = SAMPLE
```

```
db2inst1@udb1nx02:~/sql1lib/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.
```

To call the SQL procedure from the command line, we entered the following command.

```
db2 "CALL update_salary ('000100', 1)"
```

The output is displayed in Example 8-5.

Example 8-5 Procedure call on command line

```
db2inst1@udb1nx02:~/sql1lib/samples/> db2 "CALL update_salary ('000100', 1)"
```

```
Return Status = 0
```

A stored procedure usually runs in a separate process under the fenced user outside of database processes. 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
```

After installing the Application Development Client either on the database server or on a Windows or Linux client, you will find many examples in the directory `$INSTHOME/sql/lib/samples` about how to create UDFs and stored procedures.

For detailed information about building SQL procedures, refer to *IBM DB2 UDB Application Development Guide: Building and Running Applications V8*, SC09-4825.

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:

- ▶ Perl
- ▶ PHP
- ▶ C++
- ▶ Java

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.

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 5.004_04 or later
- ▶ DBI 0.93 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.

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 UDB
- ▶ DB2 Application Development Client
- ▶ Perl 5.004_04 or later
- ▶ DBI 0.93 or later
- ▶ DBD::DB2 driver

Here is the installation procedure:

1. Install DB2 UDB and the Application Development Client on your workstation. Refer to Chapter 2, “Installation” on page 23 for instructions on how to install DB2 UDB on Linux. The Application Development Client installable image is found on your DB2 UDB ESE CD-ROM, and is also provided on a separate CD in the DB2 Personal Developer's Edition and DB2 Universal Developer's

Edition media packs. Run **db2setup** to install this product using the installation wizard.

2. Install Perl 5. Perl is typically provided by the Linux distributions, but you can also 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. The Perl generic DBI module can be downloaded 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-DB2-0.76.

- Before installing DBD::DB2 driver, set the DB2_HOME environment variable to the installation directory:

```
export DB2_HOME=/opt/IBM/db2/V8.1
```

- Untar DBD-DB2-0.76 by entering the following command. In this example, we will untar to the /db2home/db2inst1/perl directory.

```
tar xzf DBD-DB2-0.76.tar.gz -C /db2home/db2inst1/perl
```

Set permissions on this file to give access to the DB2 user.

- In the DBD-DB2-0.76 directory, login as the DB2 user and 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

- Now login as the root user and run the following command:

```
make install
```

- The **make install** command installs the DBI and then deletes the working directory

5. Enable the DBI module

To enable Perl to load the DBI module, you must include the following line in your DB2 application:

```
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 $dbhhandle = DBI->connect('dbi:DB2:dbalias', $userID, $password);
```

Where:

- *\$dbhhandle* 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

Note: Red Hat 8.0 and SuSE 8.1 include Perl and Perl DBI RPM packages (Red Hat: perl-5.8.0-55.i386.rpm and perl-DBI-1.30-1.i386.rpm; SuSE: perl-5.8.0-57.i586.rpm and perl-DBI-1.28-30.i586.rpm). To create a DB2 application with Perl, The Red Hat 8.0 and SuSE 8.1 users only need to install and compile the DBD::DB2 package.

The rest is up to you. Here are some good references for Perl application development on DB2:

<http://perl.apache.org/docs/1.0/guide/>

Everything you need to know about mod_perl technology.

<http://dbi.perl.org/>

Useful information about DBI.

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

The IBM DB2 Perl database interface Web site.

<http://www7b.boulder.ibm.com/dmdd/library/tutorials/db2linux/db2linux.html>

At the bottom of this Web site is a useful tutorial called “Using Perl to access DB2 for Linux”.

8.3.2 PHP

If you want to create a Web-enabled database application, you should consider using DB2 for Linux with PHP (Hypertext Processor language). PHP is a

server-side, cross-platform scripting language that allows you to embed application logic directly into an HTML file. It can be used to access DB2 from Web-based applications to generate customized dynamic content or for capturing database transactions from a Web browser. It supports DB2 access using the Unified-ODBC access method, in which the user-level PHP communicates with DB2 using ODBC calls through the DB2 CLI layer. PHP is most commonly used with the Apache Web server, but also works well on other Web servers, such as WebSphere, thttpd and AOLServer. The PHP Apache module runs as part of the Apache (httpd) process.

Here we provide references to useful Web sites that will help you setup PHP and DB2 with the Apache server.

- ▶ To download and install the Apache Web server, refer to:

<http://httpd.apache.org>

This Web site tells you everything you need to know about installing and configuring Apache on Linux. It is geared towards Red Hat, but should also work for other distributions.

- ▶ To download and install PHP 4.x for Apache 1.x.x on Linux, go to:

<http://php.apache.org>

This is a great Web site that provides detailed, easy to follow instructions on how to acquire, install, compile, and setup PHP for Apache on Linux.

- ▶ There are a number of configuration steps that are required to get things working. These steps are covered in “Apache Web Development with IBM DB2 for Linux” tutorial on the Internet. Go to the following Web site to download this tutorial:

<http://www7b.boulder.ibm.com/dmdd/library/tutorials/db2linux/db2linux.html>

8.3.3 C++

C++ is a very popular and powerful programming language. Currently, DB2 for Linux supports the following C/C++ programming languages and compilers:

- ▶ **For Linux on Intel x86:** GNU/Linux gcc and g++ Versions 2.95.3 and 2.96
- ▶ **For Linux/390:** GNU/Linux gcc and g++ Version 2.95.3

Visit the DB2 application development Web site for information on gcc and g++ Version 3.0 support for Linux on Intel and for future version support for Linux/390:

<http://www.ibm.com/software/data/db2/udb/ad>

Refer to *IBM DB2 UDB Application Development Guide: Building and Running Applications V8*, SC09-4825, for more information about C/C++ programming on DB2.

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.

In order to build Java applications on Linux with DB2 JDBC support, you need to install and configure the following components on your development machine:

- ▶ IBM Java Developer Kit:
 - **For Linux on Intel:** IBM Developer Kit and Runtime Environment for Linux, Java 2 Technology Edition, Version 1.3.1, 32-bit version. This product gets installed with DB2 if you use DB2 Setup.
 - **For Linux /390:** IBM zSeries Developer Kit for Linux, Java 2 Technology Edition.
 - **For Linux on IA64:** IBM Developer Kit and Runtime Environment for Linux, Java 2 Technology Edition, Version 1.3.1, 64-bit version.
- ▶ DB2 Java Enablement. This is provided with DB2 UDB Version 8 for Linux clients and servers.

In order to run Java stored procedures or user-defined functions, you must first enable DB2 to load the *libjava.so*, *libjvm.so*, and *libhi.so* Java shared libraries. The load program searches for the libraries in */lib* or */usr/lib*. You must set the access path to these libraries by doing one of the following:

- ▶ Add the location of the Java shared libraries to */etc/ld.so.conf*, then run the **ldconfig** command as root to refresh the run-time linker:

```
ldconfig
```

Or,

- ▶ Create symbolic links in */usr/lib* to point to the *libjava.so*, *libjvm.so* and *libhi.so* libraries.

To create symbolic links, go to the */usr/lib* directory and enter the following commands:

```
ln -fs JAVAHOME/jre/bin/libjava.so .
ln -fs JAVAHOME/jre/bin/classic/libjvm.so .
ln -fs JAVAHOME/jre/bin/libhi.so .
```

Where, JAVAHOME is the base directory for the JDK.

You can access sample Java programs in the *INSTHOME/sql/lib/samples/java* directory, where *INSTHOME* is the instance owner's home directory.

Read the README file in this directory for information about how to run a sample program. Pay particular attention to these items:

- ▶ Before doing anything, copy the /sqlib/samples directory to your working directory:

```
cd /db2home/db2inst1/sqlib
mkdir mysamples
cp -r samples/* mysamples/
```

This will give you the correct permissions in the /samples directory so that you can run the sample programs.

- ▶ Ensure that the IBM Java Development Kit is set in your path by entering:

```
echo $PATH
```

If the JDK path is not found there, append it to PATH. For example, if your JDK path is /opt/IBMJava2-131/bin, enter:

```
PATH=$PATH:/opt/IBMJava2-131/bin
```

- ▶ If you use the **make** command, you will most likely have to make changes to the *makefile* file for it to work with your environment.

Refer to *IBM DB2 UDB Application Development Guide: Building and Running Applications V8*, SC09-4825, for more information about Java programming on DB2.

In addition to the programming languages discussed in this section, DB2 also can be accessed using script languages like BASH and AWK. These script languages come with Linux and don't require special setup. They are convenient for simple DB2 tasks or modeling.

8.4 Application tools

DB2 provides many useful tools for developing DB2 applications, such as the Command Center, Script Center, SQL Assist, and the Development Center. These tools become available to you after installing the Application Development Client.

8.4.1 Command Center

The DB2 Command Center (Figure 8-3) is a powerful interactive tool that can be used for executing DB2 SQL statements and DB2 commands as well as for building scripts. You can start the Command Center from the Tools menu in the Control Center or any graphical tool provided by DB2.

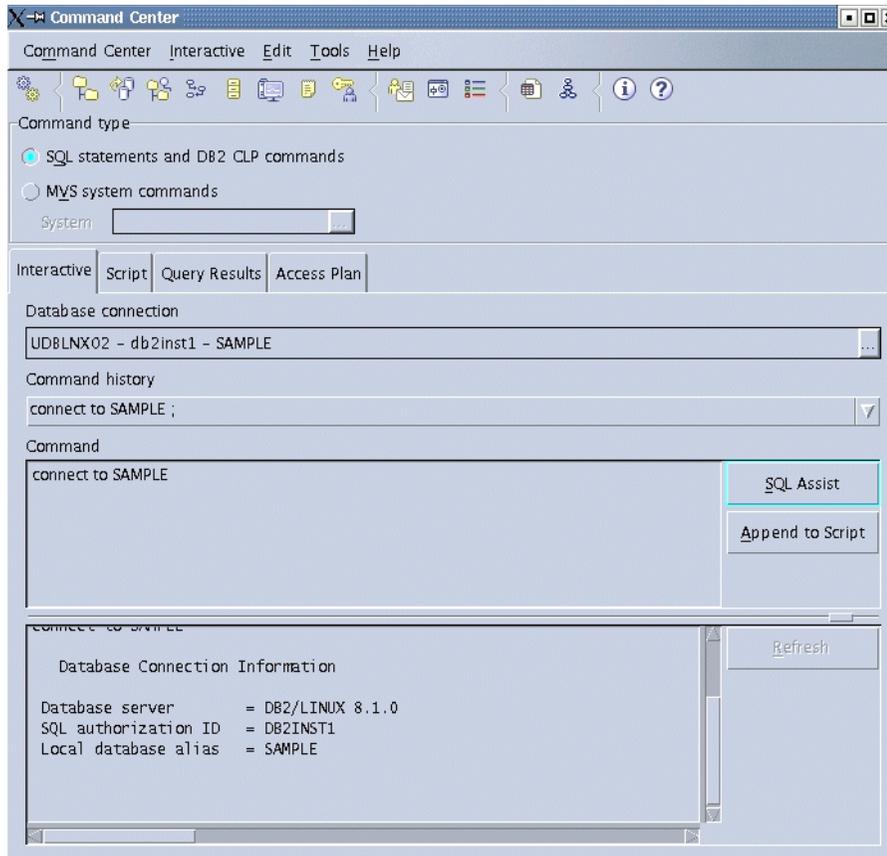


Figure 8-3 Command Center window

On the *Interactive* page you can execute any DB2 commands or SQL statements directly against a DB2 database. For complex SQL statements, the Command Center allows you to store them as scripts if you press the *Append to Script* button. This script can optionally be stored in the Task Center to run it at a specific time.

The *Script* page allows you to execute commands in sequence, import stored scripts for execution, or generate new scripts.

Results from any command or SQL statements are displayed on the *Query Results* page.

To show how DB2 retrieves your data for a specific SQL statement, use the *Access Plan* page.

The *SQL Assist* button starts the SQL Assistant tool (Figure 8-4) which you can use to easily select or modify your data. This tool can be called from many other tools.

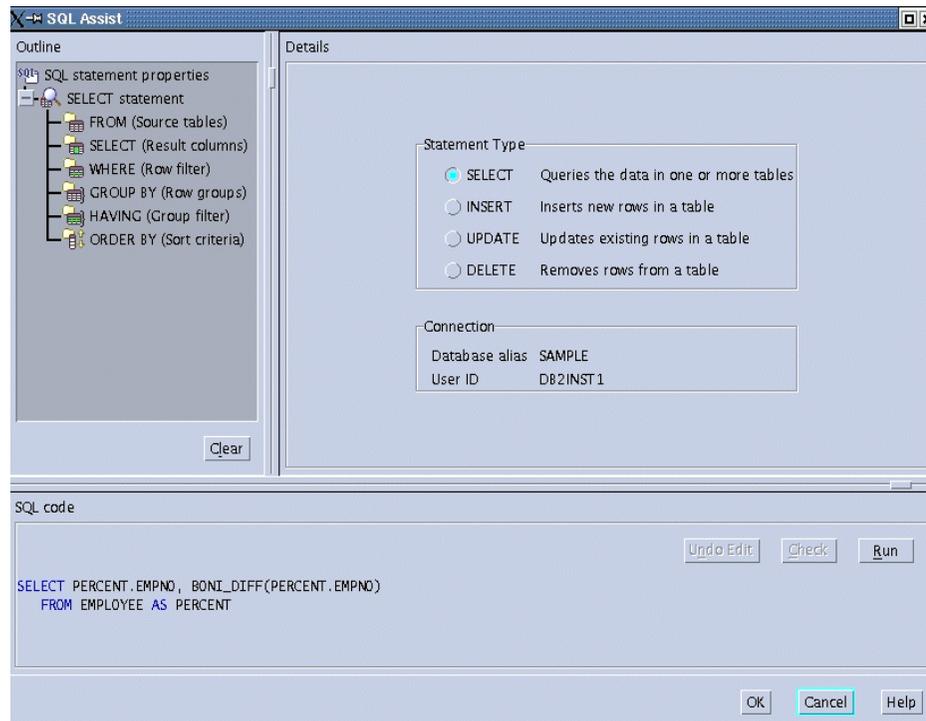


Figure 8-4 SQL Assist window

8.4.2 Development Center

The Development Center is a powerful tool used to develop stored procedures, user defined functions or structured types. To start the Development Center (Figure 8-5), use the command line and enter **db2dc &** or start it from the Tools menu of any graphical tool provided by DB2.

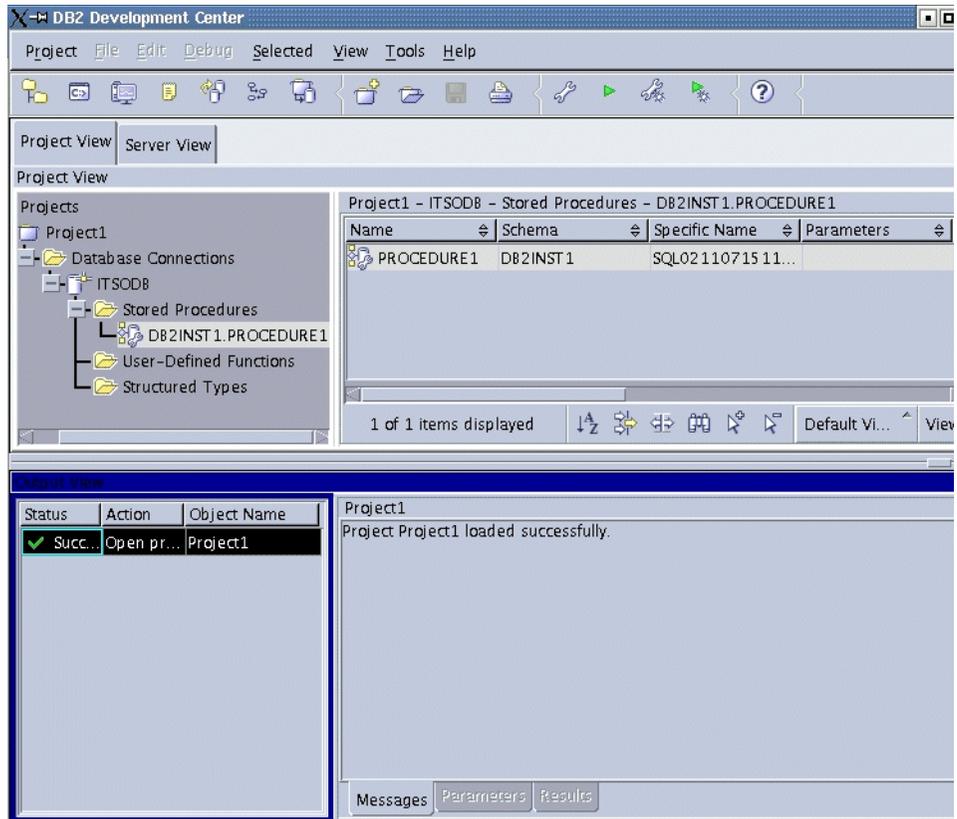


Figure 8-5 Development Center main window

The Development Center provides you with functions as well as procedures to create DB2 application objects with a wizard or manually. By right-clicking on an object, a selection of different actions is displayed. You may edit or run an existing object, create a new object or import an object from another project or file system. In the directory `$INSTHOME/sql/lib/samples` you will find many samples that you can import. Figure 8-6 shows you an example stored procedure which was imported from the sample directory.

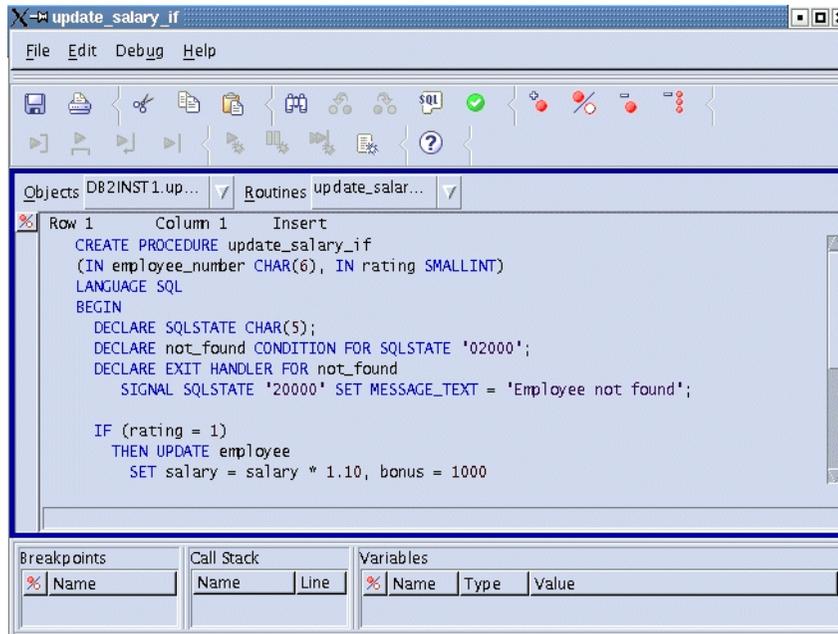


Figure 8-6 Stored Procedure edit window

To compile stored procedures using the Development Center, we have to configure two DB2 environment variables (Example 8-6):

- ▶ DB2_SQLROUTINE_COMPILER_PATH to set the path to the C compiler
- ▶ DB2_SQLROUTINE_COMPILE_COMMAND to define parameters for compiling procedures.

Example 8-6 Set C compiler environment for Linux

```
DB2SET DB2_SQLROUTINE_COMPILER_PATH=/usr/bin
DB2SET DB2_SQLROUTINE_COMPILE_COMMAND=cc -fpic -I$HOME/sqllib/include SQLROUTINE_FILENAME.c
-shared -o SQLROUTINE_FILENAME -L$HOME/sqllib/lib -ldb2
```

Important: The database server on which you want to use the Development Center must have the Application Development Client installed, even if you are developing from a Windows or Linux Client.

In addition to the store procedures, UDF and triggers, you also can use the Development Center to develop an OLEDB application.

The following DB2 UDB Version 8 manuals provide detailed DB2 application development information:

- ▶ *IBM DB2 UDB Application Development Guide: Building and Running Applications V8*, SC09-4825
- ▶ *IBM DB2 UDB Application Development Guide: Programming Client Applications V8*, SC09-4826
- ▶ *IBM DB2 UDB Application Development Guide: Programming Server Applications V8*, SC09-4827

For more information about the Development Center, refer to *IBM DB2 UDB Guide to GUI Tools for Administration and Development*, SC09-4851.

8.4.3 Additional development tools

In addition to the tools shipped out with DB2 there are many others available on the market. The DB2 Development Center also provides Development Add-Ins for the following Microsoft Products:

- ▶ Visual C++
- ▶ Visual Basic
- ▶ Visual InterDev

Here a list of development tools supported by DB2 on Linux:

- ▶ Borland Kylx
<http://www.borland.com/kylx/index.html>
- ▶ IBM Visual Age for Java
<http://www-3.ibm.com/software/ad/vajava/>
- ▶ Net.Data
<http://www-3.ibm.com/software/data/net.data/>
- ▶ WebSphere Integration
<http://www7b.boulder.ibm.com/wsdd/products/>
- ▶ DB2 Extenders
<http://www-3.ibm.com/software/data/db2/extenders/>
- ▶ Bunker Hill
http://www.bunkerhill.com/ei-0_files/WebHelp/2/index.htm



A

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. For a detailed explanation of these prefix sequences or other prefix sequences that can be used, refer to the *"Administration" Guides* (listed in "Other publications" on page 303).

Table A-1 Commonly used command prefix sequences for db2_all and rah

Sequence	Description
	Runs the commands in sequence in the background.
&	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 first command in the following example shows you how to update database configuration parameters for all database partitions in the instance; and the second one shows you how to use prefix sequences for substitution purposes, and using the prefix sequence “ ” (single quotation mark) to echo the commands that will be executed.

Example: A-1 Using db2_all on a multiple partitions environment

```
$ db2_all "db2 update db cfg for sample using logprimary 10 logsecond 15"
```

```
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
UDBLN06: db2 update db cfg for sample using logprimary 10 logsecond 15
completed ok
```

```
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
UDBLN07: db2 update db cfg for sample using logprimary 10 logsecond 15
completed ok
```

```
$ db2_all "'>\"echo Machine Name: <> Partition Number: ##"
echo Machine Name: <> Partition Number: ##
```

```
Machine Name: UDBLN06 Partition Number: 0
UDBLN06: echo Machine Name: <> Partition Number: ## completed ok
```

```
Machine Name: UDBLN07 Partition Number: 1
```

UDBLNx07: echo Machine Name: <> Partition Number: ## completed ok

Please be aware of the prefix sequence “ (double quotation mark) is preceded by a back slash “\” in our example. Here the back slash is used as an escape character. In addition, DB2 instance owner’s login shell is bash in the example, not ksh that DB2 originally supports.

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.

For more information regarding the commands mentioned here, please refer to the “*Administration*” Guides (listed in “Other publications” on page 303), or refer to the online help provided by db2_all or rah.



B

DB2 Tools Catalog creation

This appendix provides a sample to create the DB2 Tools Catalog manually and information related to the Tool Catalog creation. In addition, the notification setup is also briefed here.

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. If using DB2 Control Center to do that, after invoking DB2 Control Center, then you can select **Tools** → **Tools Settings** → **Scheduler Settings**, and create one new Tools Catalog.

Here we provide a sample to create the Tools Catalog by using the DB2 CLP manually, also included are the steps to check DAS's configuration variations before and after creating Tools Catalog, and the new tablespaces and bufferpools 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@UDBLN08:~> 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) = UDBLN08

Java Development Kit Installation Path DAS (JDK_PATH) = /opt/IBMJava2-131

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                        =
```

Then we create the Tools Catalog by CLP as in Example B-2.

Example: B-2 Creating Tools Catalog by using CLP

```
db2inst1@UDBLN08:~> db2 "create database toolscat"
DB20000I The CREATE DATABASE command completed successfully.
db2inst1@UDBLN08:~> db2 "create tools catalog esecat0 use existing database
toolscat"
DB20000I The CREATE TOOLS CATALOG command completed successfully.
```

In the preceding example, if you want, you can create the new database (here it is TOOLSCAT) via CREATE TOOLS CATALOG with CREATE NEW DATABASE parameter directly; for details regarding the command syntax of CREATE TOOLS CATALOG, refer to *IBM DB2 UDB Command Reference V8, SC09-4828*.

Then you can check DAS configuration parameters again and you may find that the parameters associated with the Tools Catalog were changed automatically.

Example: B-3 DAS Tools Catalog parameters changed automatically

```
db2inst1@UDBLN08:~> 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) = UDBLN08

Java Development Kit Installation Path DAS (JDK_PATH) = /opt/IBMJava2-131

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                        =
```

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. And two new tablespaces with a

pagesize of 32K and a bufferpool also with pagesize 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 tablespaces and bufferpool created automatically

```
db2inst1@UDBLN08:~> db2 list tablespaces
```

Tablespaces for Current Database

```
Tablespace ID          = 0
Name                   = SYSCATSPACE
Type                   = System managed space
Contents               = Any data
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                   = System managed space
Contents               = Any data
State                  = 0x0000
```

```
Detailed explanation:
Normal
```

```
Tablespace ID          = 3
Name                   = TBSP32K0000
Type                   = System managed space
Contents               = Any data
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@UDBLNx08:~> db2 "select
substr(tbspace,1,20),tbspaceid,substr(dbpname,1,15)
DBPG,bufferpoolid,substr(ngname,1,15) NGName from syscat.tablespace where
tbspaceid>2"
```

1	TSPACEID	DBPG	BUFFERPOOLID	NGNAME
TBSP32K0000		3 IBMCATGROUP	2	IBMCATGROUP
TBSP32KTMP0000		4 IBMTEMPGROUP	2	IBMTEMPGROUP

2 record(s) selected.

```
db2inst1@UDBLNx08:~> db2 "select substr(bpname,1,15)
BPNAME,bufferpoolid,substr(dbpname,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 tablespaces TBSP32K0000 with tablespace ID 3 and TBSP32KTMP0000 with tablespace ID 4 were automatically created by the CREATE TOOLS CATALOG command, each with a pagesize of 32K. The first one is a regular tablespace that is used to store the tasks related tables, and the second one is a temporary tablespace. In addition, a new bufferpool named BP32K0000 is created with a pagesize of 32K. You need to recycle the database manager to have the new changes take effect. Before you stop the instance, if you use LIST APPLICATIONS to display current applications, you may find that some applications are connected to the Tools Catalog Database with the Application Name "db2dastm":

Example: B-5 Applications connecting to Tools Catalog Database

```
db2inst1@UDBLNx08:/db2home/dasusr1/das/bin> db2 list applications
```

Auth Id	Application Name	Appl. Handle	Application Id	DB Name	# of Agents
---------	------------------	--------------	----------------	---------	-------------

```

-----
-----
DB2INST1 db2dasstm      87          *N0.db2inst1.079EF4175435    TOOLSCAT 1
DB2INST1 db2dasstm      86          *N0.db2inst1.079EF4175434    TOOLSCAT 1
-----

```

So, to restart the database manager, you can use DB2STOP with FORCE option, or use the following steps (as the instance owner, here DB2INST1):

1. \$ /db2home/dasusr1/das/bin/db2admin stop
2. \$ db2stop [force]
 # force may be required if other applications are still connected.
3. db2start
4. /db2home/dasusr1/das/bin/db2admin start

At this point in time, the Tools Catalog should be ready to use.

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 304.

- ▶ *DB2 UDB Exploitation of the Windows Environment*, SG24-6893

Other publications

These publications are also relevant as further information sources:

- ▶ *IBM DB2 UDB Command Reference V8*, SC09-4828
- ▶ *IBM DB2 UDB What's New V8*, SC09-4848
- ▶ *IBM DB2 UDB Administration Guide: Planning V8*, SC09-4822
- ▶ *IBM DB2 UDB Administration Guide: Implementation V8*, SC09-4820
- ▶ *IBM DB2 UDB Administration Guide: Performance V8*, SC09-4821
- ▶ *IBM DB2 UDB Data Movement Utilities Guide and Reference V8*, SC09-4830
- ▶ *IBM DB2 UDB Data Recovery and High Availability Guide and Reference V8*, SC09-4831
- ▶ *Federated Systems PIC Guide Version 8 Release 1*, GC27-1224
- ▶ *IBM DB2 UDB Guide to GUI Tools for Administration and Development*, SC09-4851
- ▶ *IBM DB2 UDB SQL Reference, Volume 1, V8*, SC09-4844
- ▶ *IBM DB2 UDB SQL Reference, Volume 2, V8*, SC09-4845
- ▶ *IBM DB2 UDB System Monitor Guide and Reference V8*, SC09-4847
- ▶ *IBM DB2 UDB Application Development Guide: Building and Running Applications V8*, SC09-4825
- ▶ *IBM DB2 UDB Application Development Guide: Programming Client Applications V8*, SC09-4826

- ▶ *IBM DB2 UDB Application Development Guide: Programming Server Applications V8*, SC09-4827
- ▶ *IBM DB2 UDB Call Level Interface Guide and Reference, Volume 1, V8*, SC09-4849
- ▶ *IBM DB2 Universal Database Call Level Interface Guide and Reference, Volume 2, V8*, SC09-4850
- ▶ *Data Warehouse Center Application Integration Guide Version 8 Release 1*, SC27-1124
- ▶ *DB2 XML Extender Administration and Programming Guide Version 8 Release 1*, SC27-1234
- ▶ *IBM DB2 UDB Quick Beginnings for DB2 Clients V8*, GC09-4832
- ▶ *IBM DB2 UDB Quick Beginnings for DB2 Servers V8*, GC09-4836
- ▶ *IBM DB2 UDB Installation and Configuration Supplement V8*, GC09-4837

Online resources

These Web sites and URLs are also relevant as further information sources:

DB2

- ▶ Database and Data Management
<http://www.ibm.com/software/data/>
<http://www.ibm.com/software/data/highlights/db2tco.html>
- ▶ DB2 Developer Domain
<http://www7b.software.ibm.com/dmdd/>
- ▶ DB2 Universal Database
<http://www.ibm.com/software/data/db2/udb/>
<http://ibm.com/db2/v8>
- ▶ DB2 Universal Database for Linux
<http://www.ibm.com/software/data/db2/linux/>
<http://www.ibm.com/db2/linux/validate>
<http://ibm.com/db2/linux>
<http://www-3.ibm.com/software/data/db2/linux/validate>
- ▶ DB2 Universal Database V8 Application Development
<http://www.ibm.com/software/data/db2/udb/ad>
- ▶ DB2 Technical Support

<http://www-3.ibm.com/cgi-bin/db2www/data/db2/udb/winos2unix/support/index.d2w/report>

- ▶ DB2 Extenders
<http://www.ibm.com/software/data/db2/extenders/>
- ▶ IBM Manuals for Data Management Products
<http://www.ibm.com/software/data/db2/library/>
- ▶ DB2 NOW!
<http://www.ibm.com/db2/migration>

Linux

- ▶ IBM Software for Linux
<http://www.ibm.com/software/is/mp/linux/software/>
- ▶ SuSE home page
http://www.suse.com/index_us.html
- ▶ Red Hat home page
<http://www.redhat.com/>

Other

- ▶ SAP Standard Application Benchmarks
<http://www.sap.com/benchmark/pdf/cert5702.pdf>
- ▶ Apache Web Development with IBM DB2 for Linux
<http://www7b.boulder.ibm.com/dmdd/library/tutorials/db2linux/db2linux.html>
- ▶ 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
- ▶ Rapid e-business development for Linux
<http://www.borland.com/kylix/index.html>
- ▶ VisualAge for Java
<http://www-3.ibm.com/software/ad/vajava>
- ▶ Net.data
<http://www-3.ibm.com/software/data/net.data>

- ▶ WebSphere Developer Domain Products
<http://www7b.boulder.ibm.com/wsdd/products>
- ▶ Bunkerhill ei-O for DB2
http://www.bunkerhill.com/ei-0_files/WebHelp/2/index.htm
- ▶ Apache HTTP Server Project
<http://httpd.apache.org>
- ▶ Perl.apache.org
<http://perl.apache.org/docs/1.0/guide>
PHP scripting language
<http://php.apache.org>

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 243
.rhosts 82
/etc 245

Numerics

2-tier 11
32-bit 5
64-bit 3, 5

A

access path 143
access plan 258
action property 232
additional partition 76
Administration Client, 66
administration notification log 237
Administrator Development Client, 273
alert log 242
AMD 3
AOLServer 282
APIs 225
Application Client 11
Application Development Client 66
AS/400 11
availability 16

B

backup 38
benchmarks 3
Bio-informatics 3
block device 249
block IO 248
BMDEFAULTBP 270
buffer pool 14, 270
built-in functions 275
Business Intelligence 3

C

cache 17
Calculate window 262

Central Processing Unit (CPU) 250
centralized database 11
chown 245
CLI 4
CLP 64, 183
cluster 4
column function 275
columns 14
command

db2start 28
df 38
iostat 246
ipcs 29
ntsysv 44
raw 37
rpm 33
sar 246
showmount 46
sysctl 29
vmstat 246

Command Line Processor 64, 183, 296
command option 246
communication 42
 requirements 27
Compact 26, 53, 62
Configuration Advisor 223
contact group 229
contact list 72
Contact Management 228
Content Management 3
Control Center 72, 226, 296
create function 275
create procedure 276
create trigger 274
Custom 26, 53, 62

D

dasmigr 138
Database Administration Server (DAS) 35, 48
database managed space (DMS) 14
database managed storage (DMS) 36
Database Manager Configuration Parameter
 DIAGLEVEL 244

DIAGPATH 243–244
 health_mon 227
 KEEPDAI 278
 KEEPFENCED 278
 NOTIFYLEVEL 243
 Database Object Health Indicator setting 231
 database partition group 13
 DB2 Administration Notification Log 242
 DB2 Administration Server (DAS) 70
 DB2 Database Partitioning Feature (DPF) 8
 DB2 Enterprise Server Edition (ESE) 8
 DB2 Enterprise-Extended Edition (EEE) 8
 DB2 Everyplace 10
 DB2 Intelligent Miner Products 9
 DB2 Net Search Extender (NSE) 9
 DB2 Notify Log 242
 DB2 object 273
 DB2 Personal Developer's Edition (PDE) 10
 DB2 Personal Edition (PE) 7
 DB2 Setup 76
 DB2 Setup wizard 34, 49
 DB2 Spatial Extender (SPE) 9
 DB2 Tools Catalog 295
 DB2 Universal Developers Edition (UDE) 10
 DB2 Workgroup Server Edition (WSE) 7
 DB2 Workgroup Server Unlimited Edition (WSUE) 7
 db2_all 292
 db2_call_stack 291
 db2_install 138
 db2_ps 294
 db2admin start 80
 db2advis 258
 db2diag.log 242
 db2exmig 143
 db2expln 258
 db2imigr 140
 db2mtrk 269
 db2rbind 143
 db2set parameter
 DB2_ALERT_LOG 242
 DB2_SQLROUTINE_COMPILE_COMMAND
 288
 DB2_SQLROUTINE_COMPILER_PATH 288
 DCS_ENCRYPT 68
 Design Advisor 258
 Development Center 284
 diagnostic log file 242
 Disable Evaluation options 238
 disk capacity 38
 disk I/O 14, 245
 disk requirements 26
 disk usage 261
 DMS 231
 DMS tablespace 230
 DNS server 45
 Domain name server 27
 DRDA 4
 dynexpln 258

E
 engine processes 294
 enterprise applications 3
 errno.h 244
 evaluation options 238
 explain 143
 Explain SQL window 266
 Ext3 format 38
 external procedure 276

F
 fail-over 5
 fenced use 278
 file systems 36, 46
 flexibility 3

G
 Gateway 3, 11, 27
 gnome-system-monitor 254

H
 Health Center 223, 226
 Health Indicator 225, 230–231
 Health Monitor 223, 225
 health_mon 225–226
 heterogeneous environment 11
 high availability 5
 High Performance Computing applications 4
 high-SMP systems 21
 Host IP address 27
 HTML 72
 httpd 282

I
 include file 244
 indexes 14
 Information Center 72

Information Integration applications 3
initial cost 4
instance 12
instance-owning machine 51
iostat 250
IP addresses 45
ipcs 29
iSeries 5

J

Java Developer's Kit (JDK) 31
JDBC 4
journal filesystem 38

K

kernel 24, 27, 36
kilobytes 250
Korn Shell 32
ksysguard 254

L

large object (LOB) 14
Life Sciences 3
Linux distribution 24, 37
Linux Distribution Partners 6
Load Wizard 257
log mirroring 41
log path 40
logfilesiz 135
logical database partitions 20
logical partitions 294
logical_port 83
logprimary 135
logsecond 135

M

memory 24, 249
memory component 267–268
memory consumption 245
memory requirements 26
Memory Tracker 269
Memory Tracker tool 270
memory usage 269
Memory Visualizer 267
message queue limits 29
migrate database 143
migration 133–134

mobile devices 10
monitoring memory 267
monitoring tool 267
msgmni 27
multiple partition environment (MPE) 291
multiple-partition 35
multi-tier 11

N

netstat 254
network 245
Network File System (NFS) 46–47
Network Information System (NIS) 76
NFS 45, 254
NFS server machine 39
nfsstat 254
nfs-utils 32
No cascade before 273
nodegroups 13
notification 35, 226, 228, 230, 236

O

ODBC 4
OLAP 4
OOLSCAT_SCHEM 296
optimization 264
Oracle RAC 17
OS/390 11

P

paging 248
parallel processing 14
parallelism 4
parameter 27
partition groups 13
partitioned database environment 41
pdksh 32
performance 3, 38
Performance Expert 223
Perl 5, 278
personal digital assistants (PDAs) 10
pgrep 254
PHP 5
physical device 36
physical storage devices 14
pkill 254
pop-up 259

- pop-up window 230
- PowerPC 5
- prefix sequences 292
- processor time 246
- processors 17
- procs 248
- Procs 249
- protocol 69
- pSeries 5
- pstree 254
- Python 5

R

- rah 292
- raw devices 36
- rebind packages 143
- reboot 249
- recommendations 134
- Recovery Expert 223
- recovery log 13
- Redbooks Web site 304
 - Contact us xii
- relational database 13
- relations 14
- response file 35, 49
- rollforward recovery 147
- row-function 275
- rows 14
- RPM (Red Hat Package Manager) 27
- rsh-server 32

S

- scalability 3, 16
- scalar function 275
- scaling out 15
- scaling up 15
- scheduler 71
- Script Center 284
- search domain 45
- secondary logs 40
- Self-Managing and Resource Tuning (SMART) 4, 223
- semaphore array 29
- SERVER_ENCRYPT 68
- several ways to install DB2 61
- shared 17
- shared memory segment 29
- shared-disk 17

- shared-nothing 17
- single partition 52
- single processor 14
- single-tier 10
- SMP architectures 15
- SMTP server 71
- SMTP_SERVER 230
- SQL 4
- SQL Assist 284
- SQL procedure 276
- SQL statements 258
- statistics 265
- stored procedures 5, 273, 276
- Structured Query Language (SQL) 14
- Subnet mask 27
- swap 249
- swap space 26, 251
- symmetric multi-processor 15
- syntax 292
- SYSCATSPACE 134
- syslog 294
- syslog.conf 245
- syslogd 245
- sysstat package 250
- System Activity Report (sar) 251
- system managed space (SMS) 14
- system managed storage (SMS) 36
- system-based I/O 36

T

- table 14
- table function 275
- tablespaces 14, 231
- Task Center 71, 285
- TCP/IP 69
- temporary tablespaces 41
- TEMPSPACE1 134
- threshold 226, 232
- tools catalog 35, 71, 145
- TOOLSCAT_DB 296
- TOOLSCAT_INST 296
- Top 246
- traceback 294
- transactional enterprise systems 3
- transfer rate 251
- trigger 273
- triggering action 273
- troubleshooting 242

Typical 26, 53, 62

U

UDFs 5

unique index 231

UNIX 294

user defined functions (UDFs) 273

userexit 150

V

virtual indexes 258

virtual memory 248

Visual Explain 265

VM/VSE 11

W

Web-browser 11

workload 258



Up and Running with DB2 for Linux

(0.5" spine)
0.475" <-> 0.875"
250 <-> 459 pages



Up and Running with DB2 for Linux



Experience the power of the integration of DB2 UDB Version 8 and Linux

Make it easier to get DB2 UDB 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 Universal Database has long been known for its technology leadership. This IBM Redbook is an informative guide that describes how to effectively integrate DB2 Universal Database (UDB) with SuSE and Red Hat Linux operating systems. This book provides both introductory and detailed information on installing, configuring, managing, and monitoring DB2 UDB in a Linux environment.

We describe the DB2 UDB 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 8, 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