DB2® Information Management Software

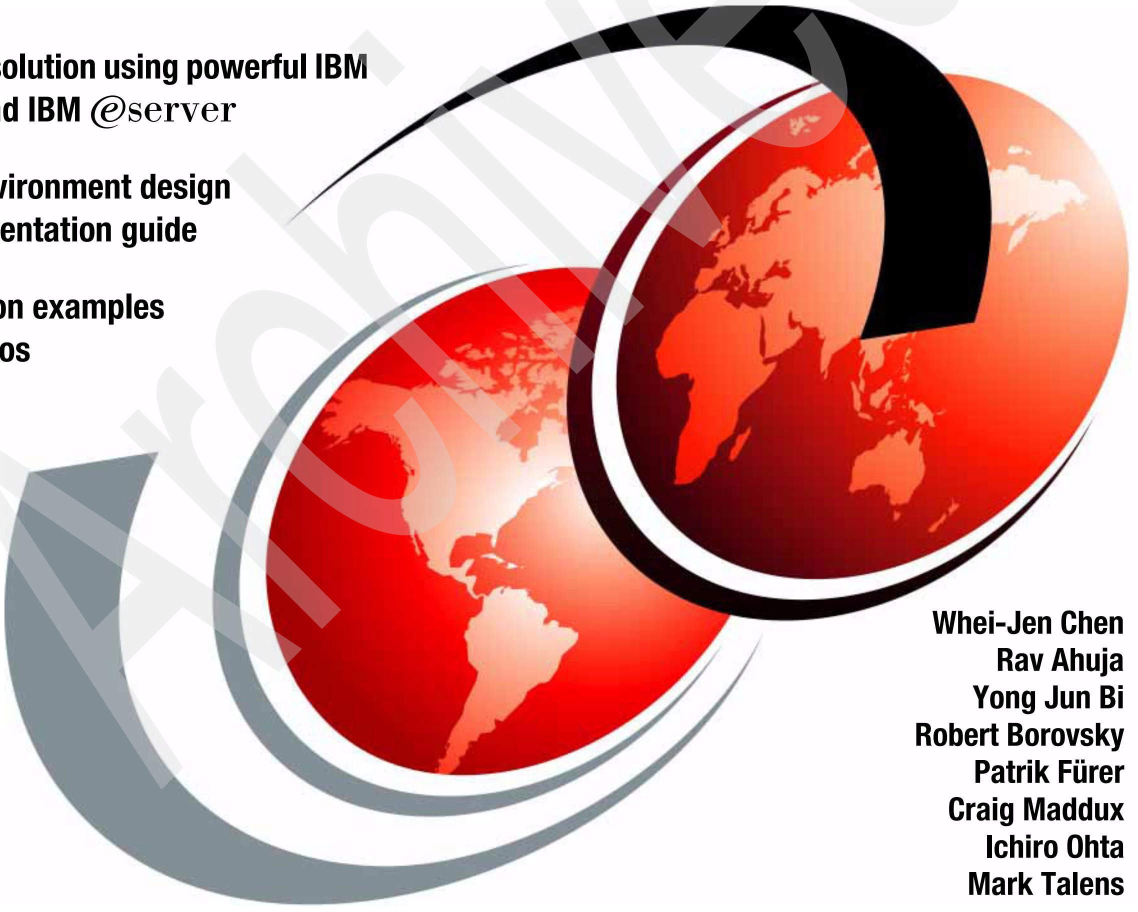
IBM

DB2 Integrated Cluster Environment Deployment Guide

Integrated solution using powerful IBM DB2 UDB and IBM @server

Detailed environment design and implementation guide

Configuration examples and scenarios



Whei-Jen Chen
Rav Ahuja
Yong Jun Bi
Robert Borovsky
Patrik Fürer
Craig Maddux
Ichiro Ohta
Mark Talens

Redbooks

ibm.com/redbooks

International Technical Support Organization

DB2 Integrated Cluster Environment Deployment Guide

October 2004

SG24-6376-00

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

First Edition (October 2004)

This edition applies to DB2 Universal Database Version 8.2 for use with SuSE Linux Enterprise Server (SLES) 8 and later and Red Hat Linux Enterprise Linux (RHEL) 3 and later operating system.

Contents

Figures

Tables

Notices

Trademarks

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

AIX®	IBM®	Redbooks (logo) ™
BladeCenter™	ibm.com®	Redbooks™
CICS®	IMS™	ServeRAID™
DB2 Connect™	Informix®	Tivoli®
DB2 Extenders™	iSeries™	TotalStorage®
DB2 Universal Database™	Lotus®	VisualAge®
DB2®	NetView®	WebSphere®
Enterprise Storage Server®	PowerPC®	X-Architecture™
eserver®	POWER™	xSeries®
eserver®	pSeries®	zSeries®

The following terms are trademarks of other companies:

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.

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

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

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

Preface

This IBM Redbook is an informative guide for deploying a DB2 Integrated Cluster Environment on Linux. This book is organized as follows:

- Chapter 1, “Introduction” on page 1, provides a general overview of DB2 Integrated Cluster Environment and discusses the fundamentals of DB2 in a Linux cluster environment.
- Chapter 2, “Selecting your cluster” on page 33, discusses options for cluster ingredients that constitute DB2 Integrated Cluster Environment and considerations for their selection.
- Chapter 3, “Planning and design” on page 63, goes over simple planning and design considerations to assist you with building your first DB2 partitioned database that is both flexible and scalable.
- Chapter 4, “Implementation” on page 125, describes our lab environment for DB2 Integrated Cluster Environment. This chapter guides you through implementing DB2 Linux clusters focusing on a multiple-partition environment.
- Chapter 5, “Operations and administration” on page 201, describes the new autonomic features of DB2 UDB V8.2 and some of the tools on the system that can be used for monitoring.
- Chapter 6, “High availability” on page 309, describes components and possible configurations of a high-availability solution. It also gives a detailed description of an implementation of a failover protection in a DB2 Integrated Cluster Environment.
- Chapter 7, “Scaling” on page 349, discusses scalability in the DB2 Integrated Cluster Environment. It includes some guidance in planning for the growth of a business and database. This chapter also examines a number of scaling strategies in more detail and provides the steps involved in scaling a database.

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.

Figure 1 Team (left to right): Yong Jun, Robert, Whei-Jen, Craig, Patrik, Rav, Ichiro, Mark

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 and is based out of the IBM Toronto Lab. He has been working with DB2 on distribution platforms since Version 1 days. He is a vocal advocate for Linux clusters and frequently contributes to articles and books related to databases on Linux. Rav can be reached at rsahuja@ca.ibm.com®.

Yong Jun Bi is an IT Specialist currently working in Technical Sales Support, IBM China, mainly focused on database administration and troubleshooting, as well as Business Intelligence solution implementation. He is an IBM Certified Advanced Database Administrator—DB2 Universal Database™ V8.1 for Linux, UNIX®, and Windows®.

Robert Borovsky joined IBM Czechoslovakia in 1991, as a Systems Engineer in a branch office. He was responsible for implementation of various IBM Software solutions: Lotus®, Tivoli®, and DB2. In 1998 he moved to the Software group in IBM Slovakia, with the technical responsibility for IBM Data Management software and Tivoli software. In the year 2000, he joined the IBM Canada Software group as an IT Specialist, and is responsible for technical sales of the IBM DB2 Information Management software. In this role his main focus is DB2 on Linux.

Patrik Fürer is an IT Specialist currently working in IBM Global Services, IBM Switzerland. He has been working for three years for IBM in the Database Management Sector. Before he got his Masters degree in Natural Sciences (biotechnology) at the Federal Institute of Technology in Zuerich, Switzerland. He is an IBM Certified Advanced Database Administrator - DB2 Universal Database V8.1 for Linux, UNIX, and Windows.

Craig Maddux is a Certified Senior IT Specialist for IBM's Pacific Northwest region with over 20 years of IT experience as a programmer and database administrator. For the past four years he has been with IBM as a DB2 Technical Specialist, assisting IBM customers design and implement solutions on DB2 UDB on Linux, Unix and Windows.

Ichiro Ohta is an IT Specialist, currently providing pre-sales support on BI solutions and Information Management products in Japan. He is also a published author, whose works include various technical articles and books on Java™ programming, XML applications, and DB2 administration, etc. He holds a Masters degree in Information and Computer Sciences from Osaka University, Osaka, Japan.

Mark Talens is a Certified Consulting IT Specialist for IBM's Pacific Southwest region. He has been working with DB2 since 1988. Prior to IBM, Mark was an independent consultant providing DBA services for many companies in Southern California. He has worked for IBM for seven years; three years in IBM Global Services and four years as a DB2 pre-sales technical specialist. He provides customers with technical information to assist in the purchase of DB2 for Linux, UNIX, and Windows platforms.

Acknowledgement

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

Amy C. Freeman
IBM WW Brand Manager - xSeries® Linux Marketing

Vicki Martin Petruch
IBM WW Brand Manager - DB2 on Linux

Grant Hutchison
DB2 Linux/Windows Solution Center manager, IBM Toronto Software Laborotory

Conor Malone
Advanced Micro Devices

Ben Eiref
TopSpin Communications, Inc.

Al Kliethermes
IBM Global Business Intelligence Solutions, West, Software Technical Sales

Stuart Alexander
IBM Linux Cluster Marketing Manager, Linux Cluster, IBM eServer xSeries

Scott Andrus, Kitman Cheung, John Keenleyside, Peter Robinson, Baris Naciterhan, Brian Olynyk, Steve Raspudic, Berni Schiefer
IBM Database Technology, IBM Toronto Laboratory

Monty Wright
IBM America Advanced Technical Support - Software, DB2, Content Management

Emma Jacobs
International Technical Support Organization, San Jose Center

Julie Czubik
International Technical Support Organization, Poughkeepsie 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

1

Introduction

Organizations today face the challenging job of integrating technologies to pull together a comprehensive solution that fits their business needs and IT budgets.

A complete data center solution consists of enterprise-class applications, databases, management tools, network infrastructure, and server hardware. Until recently, such solutions were based on proprietary systems and involved significant investment in capital, time, and resources to assemble, integrate, test, manage, and support. This scenario has changed dramatically with Linux clusters and best-of-breed software, hardware, and services from IBM and its partners.

In this chapter we provide a general overview of DB2 Integrated Cluster Environment and discuss the fundamentals of DB2 in a Linux cluster environment. We cover:

- Why DB2 on Linux
- What is a cluster
- The DB2 Integrated Cluster Environment solution
- Benefits of DB2 Integrated Cluster Environment
- DB2 Architecture

1.1 Why DB2 on Linux

IBM® DB2 Universal Database™ has long been known for its technology leadership. Therefore it was not surprising when IBM took the lead in bringing the proven performance, scalability, and ease-of-use features of DB2 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 continuously 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 systems. The reasons why major companies and governments around the globe choose to deploy DB2 for Linux in the enterprise setting are quite simple. The DB2 system's rich set of features has been running on Linux for almost six 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.

1.1.1 DB2 + Linux = Versatility

Today DB2 is the most versatile and powerful information management system on Linux. DB2 for Linux is 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:

- Backend for Web and application servers
- Business intelligence and data warehousing
- Transactional enterprise systems
- Enterprise applications such as ERP, CRM, and SCM
- Information integration and content management
- Gateway to mainframe and host data
- Financial, retail, public sector, and manufacturing applications
- Life sciences and bio-informatics solutions
- Store for spatial and geographical systems
- High-performance computing applications including:
  - Financial modeling
  - Oil and gas exploration
  - Research and scientific applications

1.1.2 Versatility + eServers = Value

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 Oracle[1]. 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 autonomic 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 24 x 7 support for both DB2 and Linux directly from IBM provides added piece of mind.

1.2 What is a cluster

In its simplest form, a cluster is two or more computers that work together to provide a solution. This should not be confused with a more common client/server model of computing, where an application may be logically divided such that one or more clients request services of one or more servers. The idea behind clusters is to join the computing powers of the servers involved to provide higher scalability, more combined computing power, or to build in redundancy to provide higher availability. So, rather than a simple client making requests of one or more servers, clusters utilize multiple machines to provide a more powerful computing environment through a single system image.

Clusters of computers must be somewhat self-aware, that is, the work being done on a specific server often must be coordinated with the work being done on other servers. This can result in complex connectivity configurations and sophisticated inter-process communications between the servers of a cluster. In addition, the sharing of data between the servers (or nodes) of a cluster through a common file system is almost always a requirement. There are many other complexities that are introduced by clusters, such as the operational considerations of dealing with a potentially large number of computers as a single resource.

For additional information, refer to the following Web site:

http://www.pc.ibm.com/ww/eserver/xseries/clustering/info.html

[1] For details refer to http://www.ibm.com/software/data/highlights/db2tco.html.

1.2.1 Cluster types

As just described, clusters may exist in many different forms. The most common cluster types are:

- High availability (HA)
- High performance computing (HPC)
- Horizontal scaling (HS)
- Database cluster (DBC)

It should be noted that the boundaries between these cluster types are somewhat indistinct and often an actual cluster may have properties or provide the function of one or more of these cluster types.

High availability (HA)

In its simplest form, a high-availability cluster consists of two servers with shared disk space between them and an extra connection between the two servers to provide a heartbeat between the two machines. One server (Server A in this example) is where we will run our database. In the event of a problem with Server A, the second server, Server B, will take over running the database, take over the IP address that the users are using to connect to server A, and resume database operations with the minimum of disruption to the users. After a brief failover, the length of which will depend on the nature of the failure and the size and state of the database at the time of failure, the user will once again be able to access the data from the same database at the same IP address. Server B will continuously monitor the state of Server A (through the LAN and, for redundancy, through a separate serial cable) in order to know when and if to take over the resources.

High-performance clusters (HPC)

High-performance computing clusters are designed to use parallel computing techniques to apply more processor power in order to develop a solution for a given problem. There are many examples of this in the scientific computing arena where multiple low-cost processors are used in parallel to perform a large number of operations. This is referred to as parallel computing or parallelism.

High-performance clusters are typically made up of a large number of computers. The design of high performance clusters is a challenging process that needs to be carefully examined throughout the entire life cycle of the solution.

The goal of high-performance clustering is to present what appears to be a single “virtual” system to any given process or task. When the cluster system is configured properly, the process or task has no idea that its work load is being divided up into smaller, more manageable pieces and then delegated for simultaneous execution by many or all of the compute nodes.

Horizontal scaling (HS)

Horizontal scaling clusters are used to provide a single interface to a set of resources that can arbitrarily grow (or shrink) in size over time. The most common example of this is a Web server farm. In this example, a single interface (URL) is provided, but requests coming in through that interface can be allocated across a large set of servers providing higher capacity and the ability to manage the end-user experience through functions such as load balancing.

Of course, this kind of cluster also provides significant redundancy. If one server out of a large farm fails, it will likely be transparent to the users. Therefore, this model also has many of the attributes of a high-availability cluster. Likewise, because of the work being shared among many nodes, it also is a form of high-performance computing.

Database clusters (DBC)

The database cluster may incorporate all three cluster types above. A database cluster may incorporate HA clustering to ensure that the database is available even after a hardware failure. Query workloads with demanding service level agreements may require the parallel processing of a High Performance Cluster. When the database grows to where the existing hardware can no longer handle the capacity, then you will need to horizontally scale your database to add additional computing resources.

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 a lot 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.

1.2.2 DB2 partitioned databases

DB2 exploits the power of Linux clusters by employing database partitioning. In a partitioned environment, a database is distributed across multiple partitions, capable of residing on different machines. Each partition, called a database partition, is responsible for a portion of a database's total data. Because data is divided across database partitions, you can use the power of multiple processors on multiple physical servers 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.

Prior to DB2 V8, a database partition was often referred to as a database node. Sometimes this introduced some confusion, because the computer hardware industry often refers to a server as a node. In DB2 V8.1 we use *database partition* instead of *database node*. You may find references to a database node in text and in some DDL, but this remains to make it easier for those transitioning from prior releases of DB2. In this book, and in practice, an individual DB2 partition is called a *database partition*. The term *node* should only be used in reference to hardware and servers.

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 partition for that user; see Figure 1-1 on page 7. Any database partition can be used as a coordinator partition. The database partition that a client or application connects to becomes the coordinator partition. You should consider spreading out users across database partition servers to distribute the coordinator function.

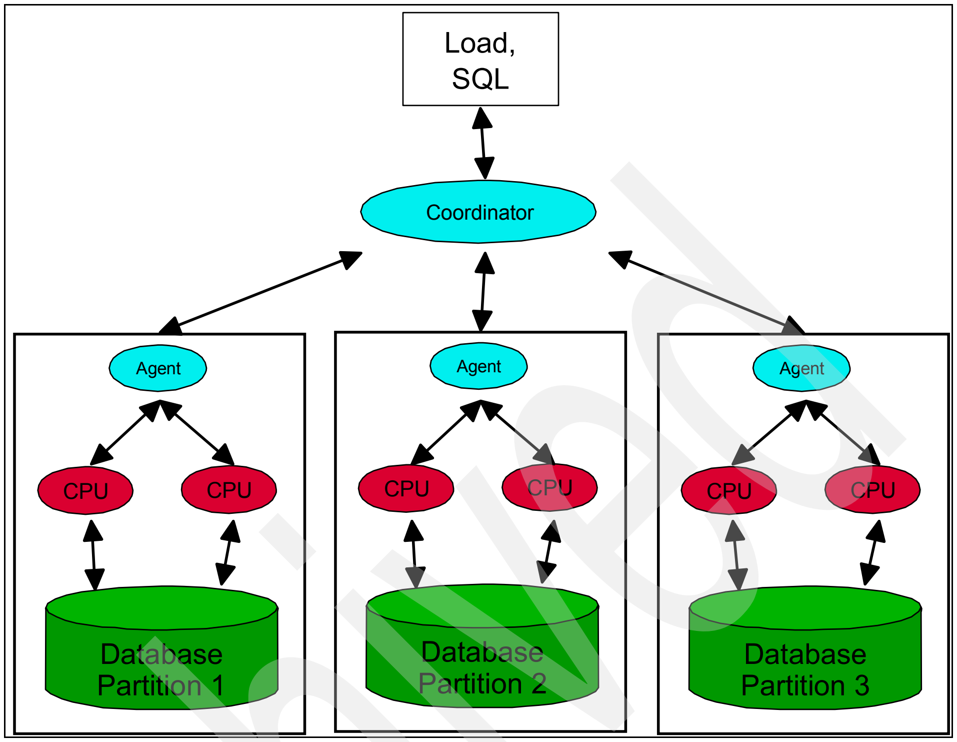


Figure 1-1 DB2 database partition

1.3 The DB2 Integrated Cluster Environment solution

The biggest problem facing customers today, who try to implement any Linux environment, is the integration of both hardware and software that are compatible with their Linux kernel. You may find yourself installing the latest Linux kernel only to find that your storage vendor has not written a driver for that kernel.

The IBM DB2 Integrated Cluster Environment for Linux is a completely integrated, high-performance, and pre-tested solution that incorporates best-of-breed software, hardware, and services. It provides you with a high-performance, reliable data management system that can scale from one to 1,000 nodes[2]. You can order a pre-configured solution or customize it to meet your needs.

[2] Scalability limits may vary by type of workload.

The core solution is based on IBM DB2 Universal Database and IBM Linux Cluster 1350 (xSeries, 325, and Blade Center). Figure 1-2 shows a sample DB2 Integrated Cluster Environment configuration. The other components in a DB2 Integrated Cluster Environment can include (but are not limited to):

- SuSE or Red Hat Linux
- IBM Tivoli System Automation software
- IBM WebSphere® Application Server
- IBM TotalStorage®
- IBM Global Services
- TopSpin InfiniBand

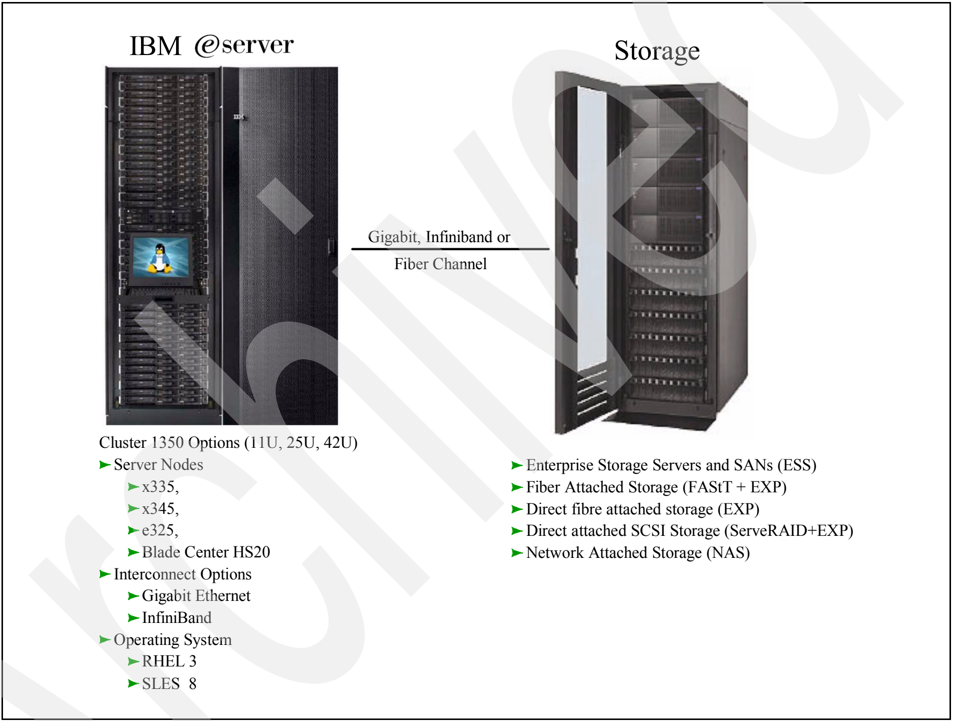

IBM @server
Storage
Gigabit, Infiniband or
Fiber Channel
Cluster 1350 Options (11U, 25U, 42U)
►Server Nodes
►x335,
►x345,
►e325,
►Blade Center HS20
►Interconnect Options
►Gigabit Ethernet
►InfiniBand
►Operating System
►RHEL 3
►SLES 8
►Enterprise Storage Servers and SANs (ESS)
►Fiber Attached Storage (FAStT + EXP)
►Direct fibre attached storage (EXP)
►Direct attached SCSI Storage (ServeRAID+EXP)
►Network Attached Storage (NAS)

Figure 1-2 Sample DB2 Integrated Cluster Environment configuration

1.3.1 The ingredients for success

The success of DB2 Integrated Cluster Environment contains several factors, discussed in this section.

Autonomic database

The benefits of Linux clusters, including superior performance at a low cost, are well known. When you add to the mix the built-in clustering capabilities of IBM

DB2 Universal Database (UDB) for Linux, Enterprise Server Edition (ESE), you get a rock-solid foundation for all your enterprise e-business data processing needs.

Powerful hardware

IBM has extensive and proven experience with clustered UNIX® computers. IBM xSeries® has applied that knowledge to produce servers that are armor-plated for Linux, optimized for database work loads, and deliver tremendous power at a fraction of the price.

Demanding applications

The combination of DB2 UDB and Linux on IBM eServer xSeries, with the latest Intel® and AMD® processors, is powerful enough to run highly demanding business applications, including SAP R/3, mySAP Customer Relationship Management (CRM), mySAP Business Intelligence (BI), and IBM WebSphere Application Server.

Reliable management

Managing a clustered environment is easy using autonomic capabilities built into DB2 software and xSeries hardware along with management tools from IBM® Tivoli® software. The availability of the solution is further enhanced using Tivoli System Automation for Linux.

High-speed network

Gigabit Ethernet is the standard network connection in most servers and would suffice in most DB2 Integrated Cluster Environment implementations. However, if high bandwidth, low latency, and low overhead is required, then infiniband can be used as both server and storage interconnect.

Platform support and services

You do not have to go at it alone. IBM Global Services Linux professionals help you install, migrate, configure, tune, and service your database solution. IBM's strong relationships with its Linux partners, supplemented by expertise from SuSE/Novell, Red Hat, and others, give you the confidence and support to deploy even the most critical solution.

1.4 Benefits of DB2 Integrated Cluster Environment

Managers and IT professionals tasked with selecting and implementing a data management solution that is robust enough for mission-critical enterprise needs, yet flexible enough to deliver exceptional value to small businesses, often evaluate technologies based on the following criteria:

- Scalability
- Price/performance
- Availability
- Manageability
- Interoperability

1.4.1 Scalability pays great returns

Many IT professionals are concerned about whether their relational database system on Linux will be able to scale as workloads increase. DB2 software and xSeries servers for Linux help you avoid large up-front costs as well as migration and training costs later. They can be deployed based on your initial requirements and then scale as large as your business demands.

The possibilities are indeed empowering. You can build powerful clusters using IBM xSeries servers running Linux. IBM DB2 Universal Database Enterprise Server Edition provides a single database that can be distributed across such a cluster. You can easily scale up by adding a new machine to the cluster, and trigger DB2 to automatically redistribute data to the new partition.

1.4.2 Proven performance at a low cost

Operating DB2 Universal Database on IBM xSeries servers for the Linux platform, the customer can expect the high performance your business needs to support its growth. The combination of IBM DB2 Universal Database and IBM xSeries servers is a proven leader in several performance benchmarks on Linux.

Numerous application and industry standard benchmarks have delivered performance not only in lab conditions but also at countless customer installations around the world. A key reason for this success is the close cooperation with IBM business partners such as SuSE, Red Hat, and TopSpin, who participated in these efforts to deliver a superior system. Host channel adaptors and switches based on Mellanox silicon featured in components from Voltaire can meet the most demanding needs for fast, low-latency communication between database partitions in the cluster. This solution combines leading technologies into a powerful supercomputer at a fraction of the costs of a special purpose machine with equivalent capabilities.

1.4.3 High levels of availability

DB2 Universal Database is setting the standard for quality and reliability in the database industry. As more mission-critical applications are implemented on

Linux, IBM's ability to bring mainframe-level reliability to this environment has become a major factor for customers choosing the DB2 family.

IBM eServer offers a reliable foundation for leading Linux-based computing. With the IBM X-Architecture™ advantage, you can feel confident about the availability of the hardware running your core business applications.

The Tivoli System Automation (TSA) for Linux increases the availability of DB2 operating environments on xSeries hardware by effectively monitoring system and application health. In the event of an outage, TSA allows applications to automatically fail over to other servers in the cluster, providing continuous access to DB2 and restored data. TSA is based on autonomic research inside IBM, long-standing experience with IBM cluster technology, and is specifically designed for the benefits of enterprise class solution stacks. The tight integration with DB2 for Linux offers smooth and seamless production operation with highest levels of availability.

1.4.4 Easily managed

DB2 greatly reduces the complexity of data management by eliminating, simplifying, and automating many tasks traditionally associated with maintaining an enterprise class database. These advances are the first implementation of the autonomic project and the first step towards making autonomic computing a reality for database implementations.

Tivoli management products simplify the management of distributed systems. Tivoli Monitoring provides monitoring for essential system resources, to detect bottlenecks and potential problems, and to automatically recover from critical situations. Tivoli NetView® extends traditional network management to ensure the availability of critical business systems and to provide rapid resolution of problems.

Tivoli Storage Manager (TSM) protects critical business information by simplifying, centralizing, and automating backup and recovery operations. Tivoli TSM support for DB2 provides online backup and quick, granular-level recovery, helping to reduce downtime and administrative costs.

1.4.5 Seamless interoperability

We recognize that today's business environment is heterogeneous and there is a great need for different technologies to integrate seamlessly. Starting with a vision of making Linux ready for business, coupled with strong relationships, the technology vendors featured here have created a highly capable enterprise class solution. Furthermore, DB2 and xSeries have introduced validation programs designed to set standards in interoperability on Linux.

The right products working in harmony enhance productivity, reduce costs of deployment, and simplify management of critical e-business applications. In other words, they empower you to focus on your business rather than the technology on which it runs.

1.5 DB2 Architecture

Figure 1-3 on page 13 shows the architecture overview of DB2 UDB. DB2 UDB implements a dedicated process architecture.

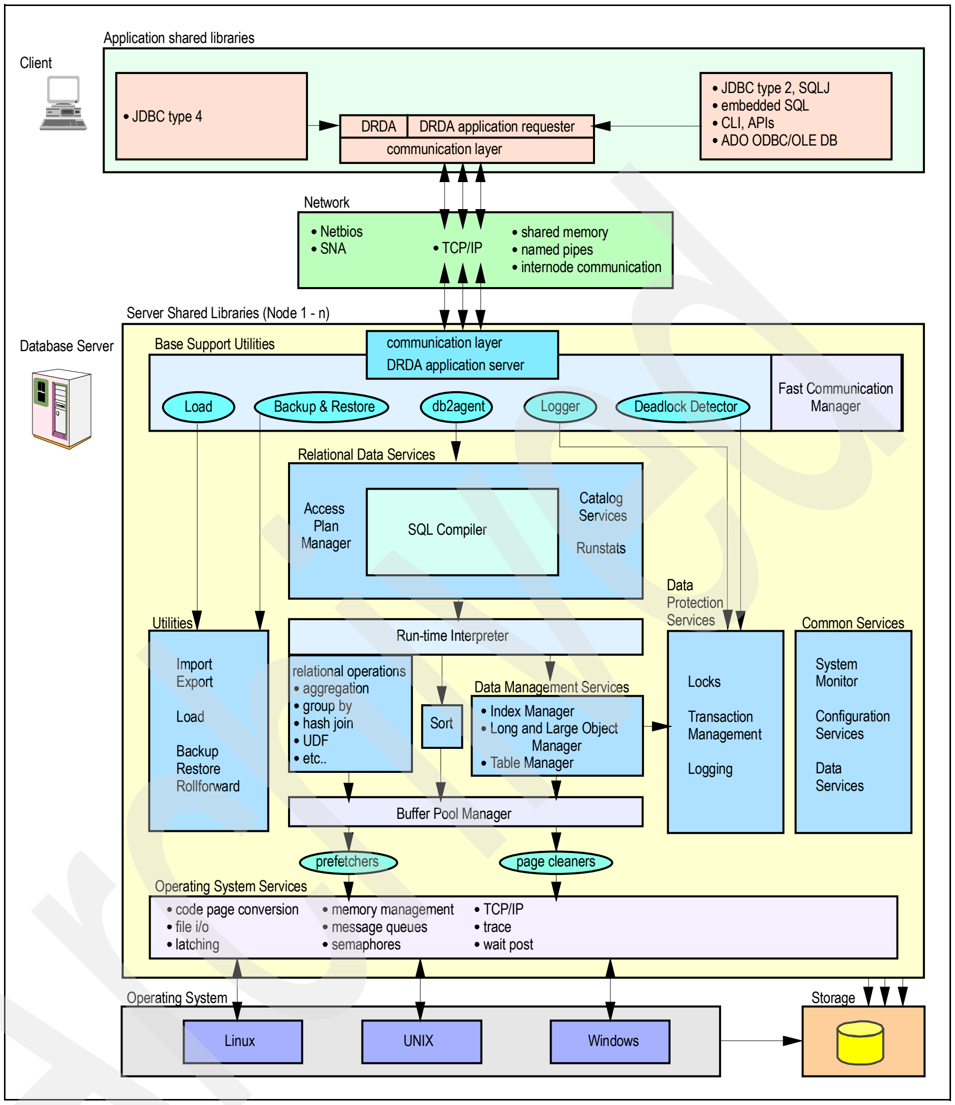


Figure 1-3 DB2 UDB architecture overview

From a client-server view, the client code and the server code are separated into different address spaces. The application code runs in the client process, while the server code runs in separate processes. The client process can run on the same machine as the database server or a different one, accessing the database server through a programming interface. The memory units are allocated for database managers, databases, and applications.

To enable access to a special database, the DB2 instance process responsible for the database must be running on the DB2 server. When an instance process is started, several processes are created and interact with each other, to maintain connected applications and the database. There are several background processes in DB2 that are pre-started; others start on a need-only basis. This section explains some of the important background processes.

DB2 UDB processes

The DB2 UDB server activities are performed by Engine Dispatch Units (EDUs) that are defined as background processes on Linux systems.

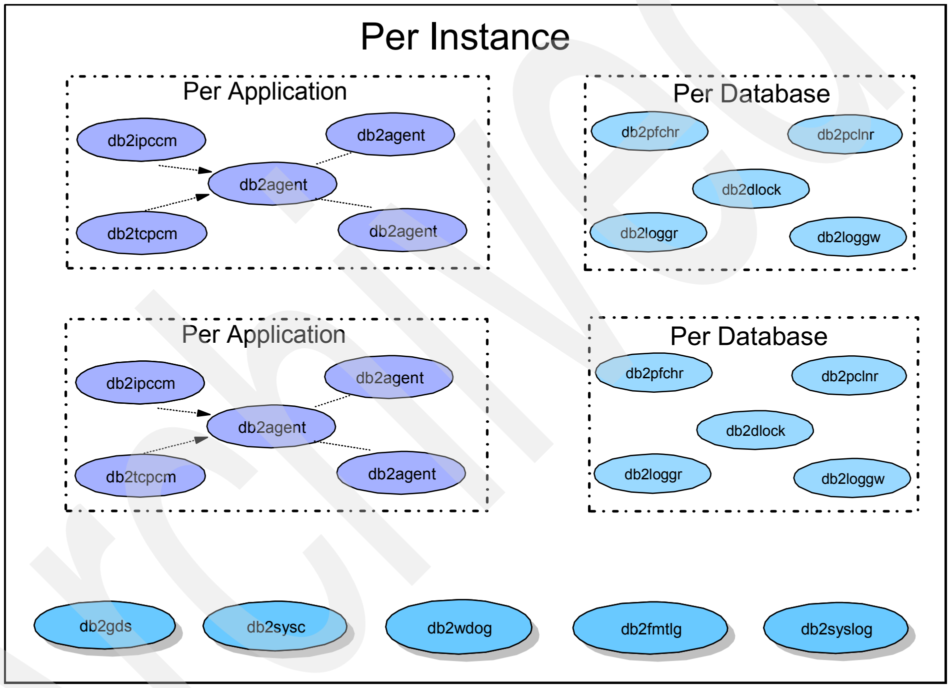


Figure 1-4 DB2 processes

Some DB2 background processes are started with the instance, and others are initialized when the database is activated by a connection. Figure 1-4 shows the necessary background processors of the DB2 UDB server at the instance, application, and database level. In the following sections, we discuss some of the important processes on the respective levels.

Instance-level processes

The following background processes will start as soon as the DB2 UDB server is started with the **`db2start`** command.

- DB2 daemon spawner (db2gds)

  This is a global daemon processor started for each instance. This process starts all the EDUs as UNIX processes.
- DB2 system controller (db2sysc)

  This is the system controller processor. Without this process the instance cannot function properly.
- DB2 watchdog (db2wdog)

  This process is required only in UNIX platforms. It is the parent process for all other processes.
- DB2 format log (db2fmtlg)

  Pre-allocates log files in the log path when the LOGRETAIN database configuration parameter is enabled, and the USEREXIT disabled.
- DB2 system logger (db2syslog)

  This is the system logger process responsible for the writing operating system error log.

Database-level processes

The following background processes are started when a connection activates the database:

- DB2 log reader (db2loggr)

  This process reads the log files during transaction rollback, restart recovery, and roll-forward operations.
- DB2 log writer (db2logw)

  This is the log writer process that flushes the database log from the log buffer to the transaction log files on disk.
- DB2 page cleaner (db2pclnr)

  An asynchronous process to make room in the buffer pool by writing changed pages to disk, before pre fetchers read pages on disk storage and move into the buffer pool.
- DB2 prefetcher (db2pfchr)

  This process retrieves data from disk asynchronously and moves it into the buffer pool before the application requested the data.

- DB2 deadlock detector (db2dlock)

  This is the database deadlock detector process. It scans the lock list (the lock information memory structure of DB2) and looks for deadlock situations.

Application-level processes

These processes are started for each application connecting to the database:

- DB2 communication manager (db2ipccm)

  This is the inter-process communication (IPC) process started for each application connecting locally. This process communicates with the coordinating agent to perform database tasks.

- DB2 TCP manager (db2tcpcm)

  This is the TCP communication manager process. This process is started when the remote client or applications connect to the database using TCP/IP communication. This process communicates with the coordinating agent to perform database tasks.

- DB2 coordinating agent (db2agent)

  This process handles requests from applications. It performs all database requests on behalf of the application. There will be one db2agent per application unless the connection concentrator is established. If intra-partition parallelism is enabled, the db2agent will call DB2 subagents to perform the work.

- DB2 subagent (db2agnta)

  Subagent, which works with the db2agent process when intra-partition parallelism is enabled.

- Active subagent (db2agntp)

  This is the active subagent that is currently performing work. This process is used when enabling SMP parallelism, which means having more processes achieving the same task. In order to enable this feature in DB2, we must set the intra-parallelism database parameter to *YES*.

DB2 database objects

In this section, DB2 objects and their relationships to each other are introduced. Figure 1-5 on page 17 shows the basic DB2 database objects.

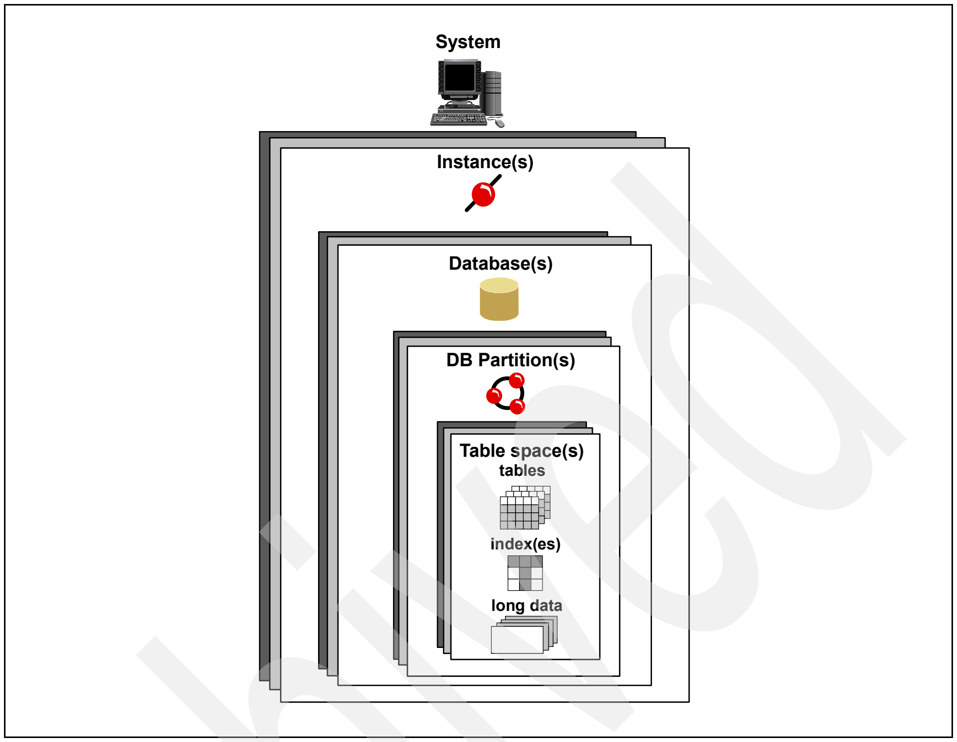


Figure 1-5 DB2 object relationships

- Instances

  An instance 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, independent 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.

- Databases

  A relational database presents data as a collection of tables. Each database includes a set of system catalog tables that describe the logical and physical structure of the object in the database, a configuration file containing the parameter values configured for the database, and a recovery log. An application or a user connects to a specified database to read or manipulate data in tables.

Figure 1-6 shows the relationship between instances, databases, and tables.

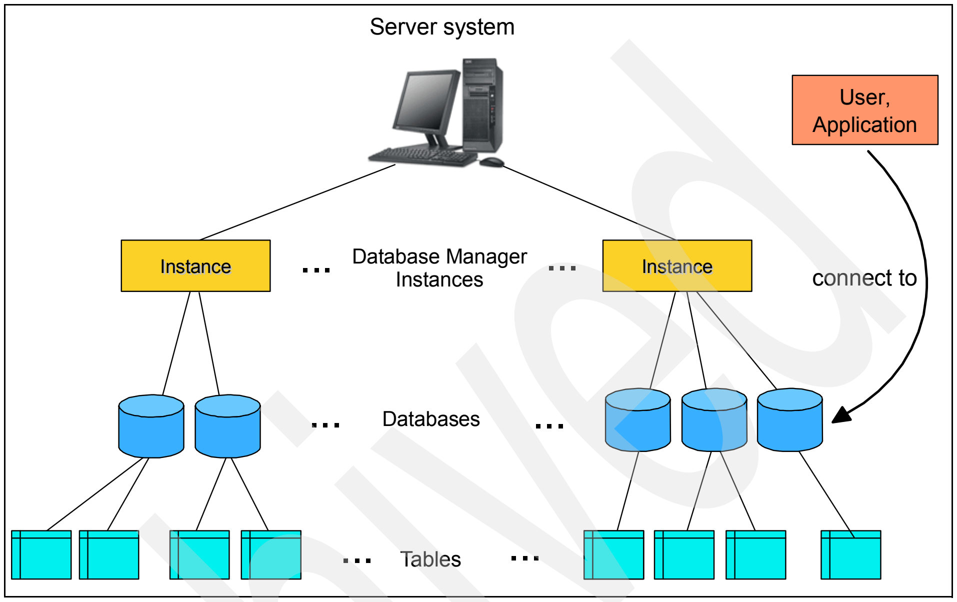


Figure 1-6 Relationship to DB2 instance, databases, and tables

- Database partition groups

  A database partition group is a set of one or more database partitions (Figure 1-7 on page 19). Before creating tables for the database, you first need to create the database partition group where the table spaces will be stored, and then create the table space where the tables will be stored. If a partition group is not specified, there is a default group where table spaces are allocated. In earlier versions of DB2, database partition groups were known as *nodegroups*. In a non-partitioned environment, all the data resides in a single partition; therefore it is not necessary to worry about partition groups.

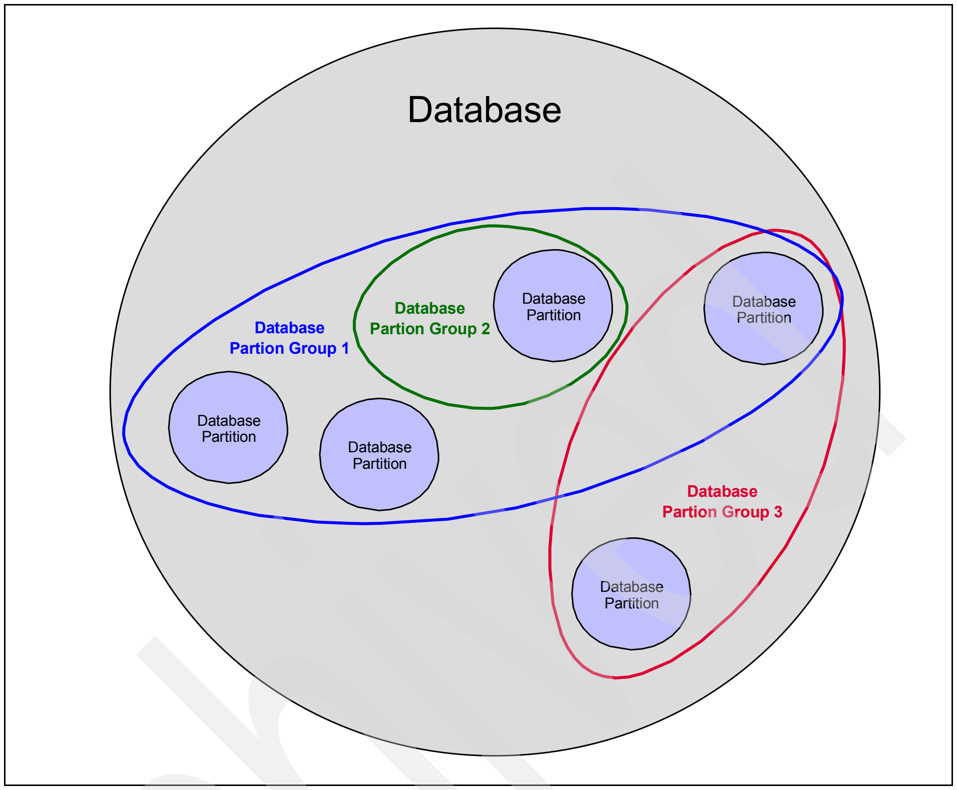


Figure 1-7 DB2 database partition groups

- System catalog tables

  Each database includes a set of system catalog tables, which describe the logical and physical structure of the data. DB2 UDB creates and maintains an extensive set of system catalog tables for each database. These tables contain information about the definitions of database objects such as user tables, views, and indexes, as well as security information about the privilege that users have on these objects. Catalog tables are created when the database is created, and are updated during the course of normal operation. You cannot explicitly create or drop them, but you can query and view their contents using the catalog views.

- Table spaces

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

Table spaces reside in database partition groups if they were created. Table space definitions and attributes are maintained 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 system 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.

- Schemas

  A *schema* is an identifier, by default the user ID, that qualifies tables and other database objects. A schema can be owned by an individual, and the owner can control access to the data and the objects within it. A schema name is used as the first part of a two-part object name. For example, a schema named Smith might qualify a table named *SMITH.PAYROLL*.

- Tables

  A relational database presents data as a collection of tables. Data in a table are arranged in columns and rows. The data in the table is logically related, and relationships can be defined between tables. 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>
  ```

- Views

  A view provides a different way of looking at data in one or more tables; it is a named specification of a result table. The specification is a SELECT statement that runs whenever the view is referenced in a SQL statement. A view has columns and rows just like a base table. All views can be used just like base tables for data retrieval. Figure 1-8 on page 21 shows the relationship between tables and views.

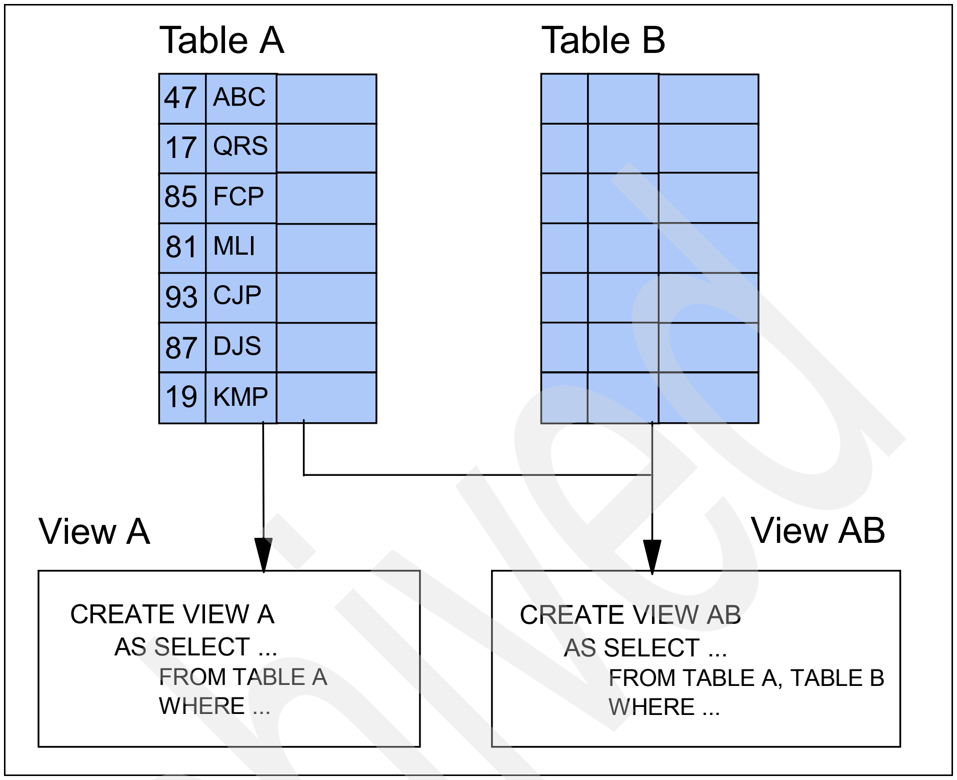


Figure 1-8 Relationship between tables and views

- Indexes

  An index is a set of keys, each pointing to rows in a table. For example, table A has an index based on the first column in the table (Figure 1-9 on page 22). This key value provides a pointer to the rows in the table: Value *19* points to record *KMP*. An index allows efficient access when selecting a subset of rows in a table by creating a direct path to the data through pointers.

  The DB2 SQL Optimizer chooses the most efficient way to access data in tables. The optimizer takes indexes into consideration when determining the fastest access path.

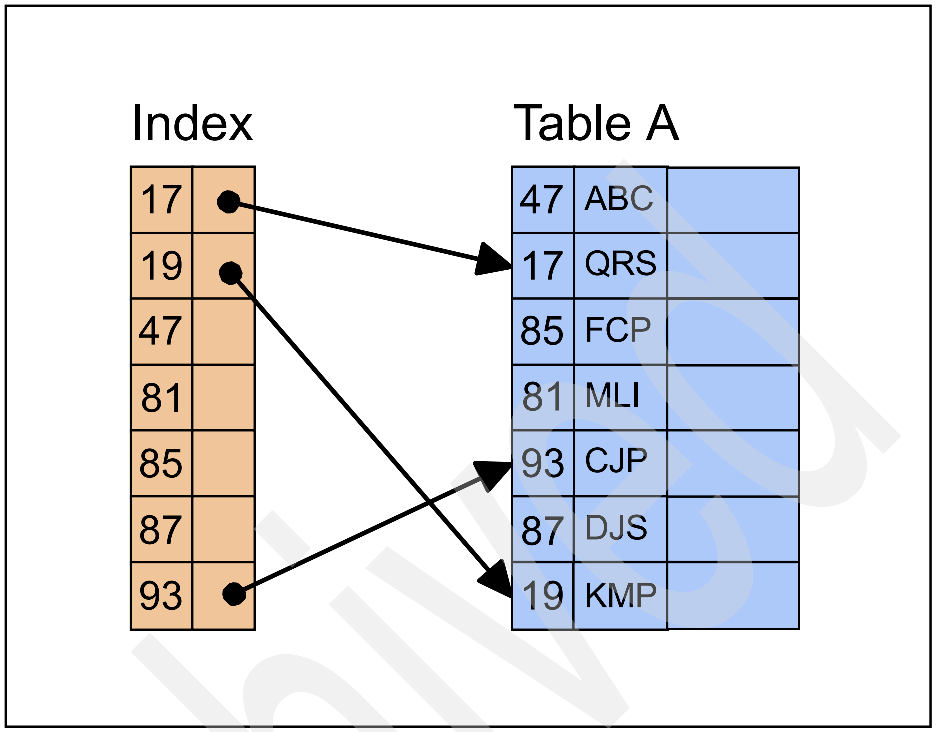


Figure 1-9 Relationship between indexes and tables

- Containers

  A container is a physical storage device. It can be identified by a directory name, a device name, or a file name. A container is assigned to a table space. A single table space can span many containers, but each container can belong to only one table space, as shown in Figure 1-10 on page 23.

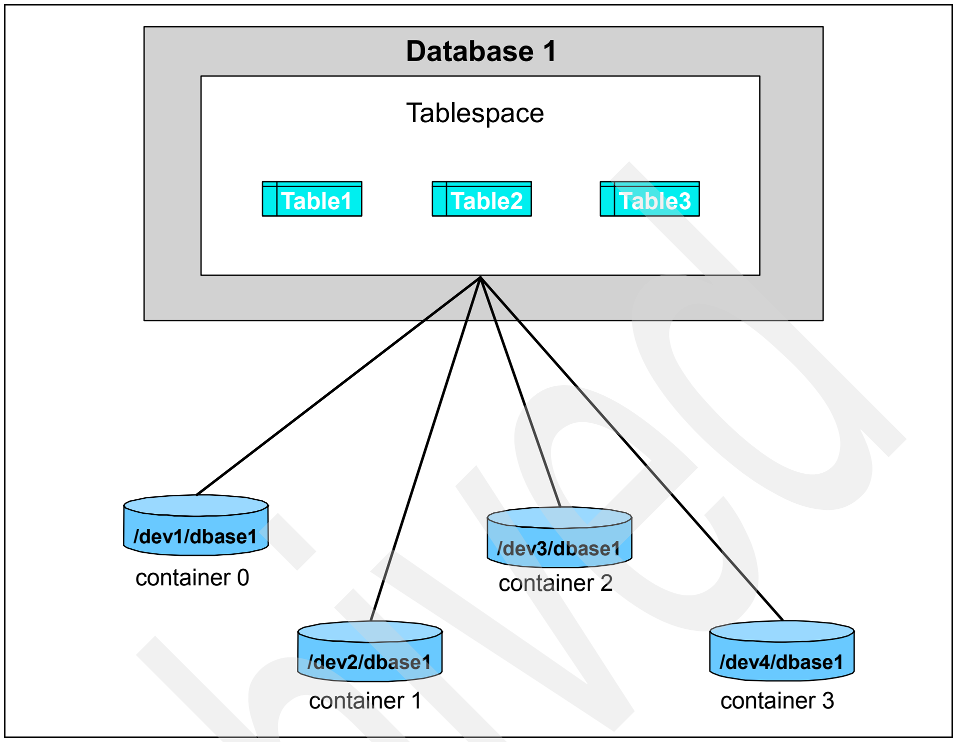


Figure 1-10 Relationship between table space and containers

- Buffer pools

  A buffer pool is the amount of memory allocated to cache table and index data pages. 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) synchronously, the better the performance of the application. The size of the buffer pool is the single most important performance tuning area, because you can reduce the delay caused by synchronous I/O.

DB2 directory structure

On Linux systems the default installation path for DB2UDB V8.2 is /opt/IBM/db2/V8.1. $DB2DIR is the environment variable for the DB2 installation directory. Figure 1-11 on page 24 shows the default directory structure for a simple CREATE DATABASE command with no table space options specified. By default DB2 creates SMS table space containers in the specified database directory. The three default table spaces created are system catalog, system temporary, and user table space. For the log files DB2 creates a directory called sqllogdir. On a Linux system, by default, a sqllib directory will be created under the instance home directory, which has a symbolic link to the DB2 installation directory.

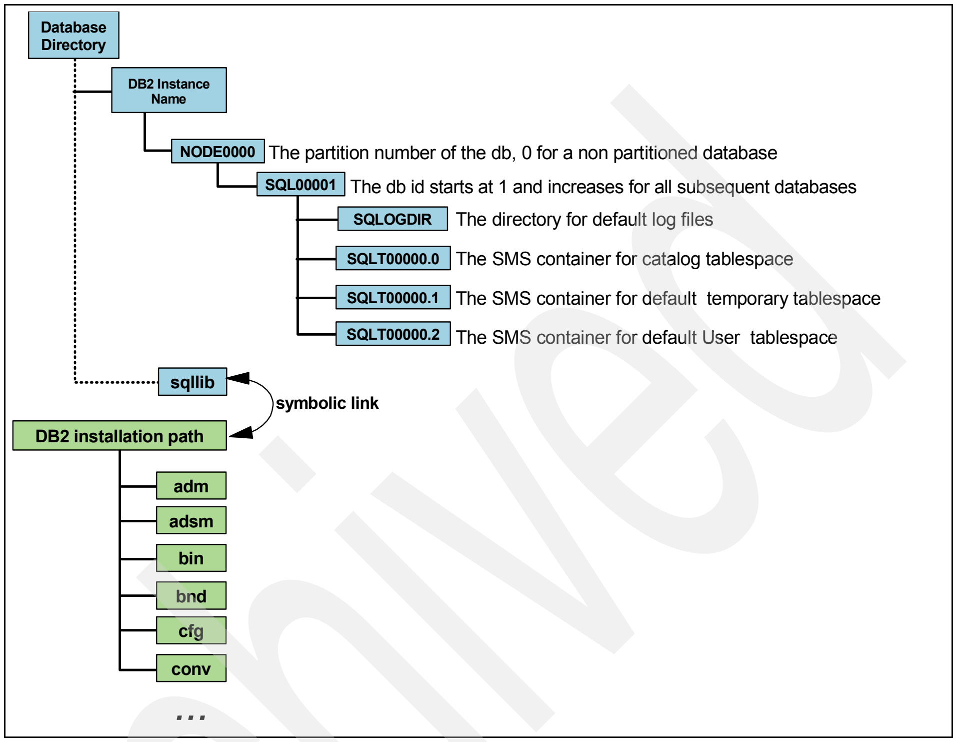

Database Directory
DB2 Instance Name
NODE0000
The partition number of the db, 0 for a non partitioned database
SQL00001
The db id starts at 1 and increases for all subsequent databases
SQLOGDIR
The directory for default log files
SQLT00000.0
The SMS container for catalog tablespace
SQLT00000.1
The SMS container for default temporary tablespace
SQLT00000.2
The SMS container for default User tablespace
sqllib
symbolic link
DB2 installation path
adm
adsm
bin
bnd
cfg
conv
. . .

Figure 1-11 DB2 directory structure

DB2 catalog

In DB2 UDB, the metadata is stored in a set of base tables and views called the *catalog*. The catalog contains information about the logical and physical structure of the database objects, object privileges, integrity information, etc.

The catalog is automatically created with the database. The base tables are owned by the *SYSIBM* schema and stored in the *SYSCATSPACE* table space. On top of the base tables, the *SYSCAT* and *SYSSTAT* views are created. *SYSCAT* views are the read-only views that contain the object information and are found in the *SYSCAT* schema. *SYSSTAT* views are updateable views containing statistical information that are found in the *SYSTAT* schema. The complete DB2 UDB catalog views can be found in *DB2 UDB SQL Reference* Volume 1 and 2, SC09-4484 and SC09-4485.

1.5.1 DB2 database access

In this section the following topics are discussed:

- DB2 clients
- Application access
- DB2 application programming interfaces

DB2 clients

To access a DB2 UDB database a DB2 client has to be installed on the client system. IBM offers four types of DB2 clients:

- Run-Time Client

  This client provides you access to DB2 UDB servers with application interfaces, such as JDBC, SQLJ, ODBC, CLI and OLE DB. This client can be used if no DB2 server administration has to be done from this client.

- Run Time Client Lite

  This is a new addition in DB2 UDB Version 8.2. Run-time Client Lite is a smaller footprint version of the DB2 Run-Time Client and is available only on Windows environments. It provides basic functions that allow your applications to access a DB2 UDB server. The DB2 Run-Time Client Lite also contains support necessary for JDBC, SQLJ, ODBC, CLI, OLE DB, and .NET, similar to the DB2 Run-Time client. With its reduced installation image size, the DB2 Run-Time Client Lite is ideal for mass deployment or for bundling with your applications.

- 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.

For client-server communication DB2 supports several communication protocols like TCP/IP, APPC, NPIPE, NetBIOS, etc. Most protocols are automatically detected and configured during an instance creation. The DB2COMM registry variable identifies the protocol detected in a server. To enable a specific protocol, the **`db2set DB2COMM`** command has to be executed. For TCP/IP, a unique port address has to be specified in the database manager configuration. This port is registered in the *services* file. To reserve port 50000 with the service name db2cidb2, for example, the entry in the services file would be:

```
db2icdb2 50000/tcp
```

To update this information in the database manager the following command is used:

```
db2 UPDATE DBM CFG USING SVCENAME db2icdb2
```

These tasks can also be performed using the DB2 configuration assistant utility. At the client, the database information is configured using either the CATALOG command or using the configuration assistant. The database is configured under a node that describes the host information like protocol and port, etc. To configure a remote TCP/IP node the following command is used:

```
db2 CATALOG TCPIP NODE node-name REMOTE host-name SERVER service-name
```

The service name registered in the server or the port number can be specified in the SERVER option. To catalog a database under this node the command used is:

```
db2 CATALOG DATABASE database-name AS alias-name AT NODE node-name
```

When using the Configuration Assistant GUI tool to add a database connection, a database discovery can be started to find the desired database.

Note: The DB2 Discovery method is enabled at the instance level using the DISCOVER_INST parameter, and at the database level using the DISCOVER_DB parameter.

Application access

When deploying applications with DB2 UDB, different methods can be used:

- 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 or batch applications.

- 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 configuration scenarios are illustrated in 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	zSeries®, iSeries™
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 Table 1-1, IBM offers the most comprehensive business integration solution by allowing federated access to a variety of distributed data sources.

DB2 application programming interfaces (APIs)

In order to access or manage DB2 objects, several different programming interfaces can be used, as seen in Figure 1-12 on page 28.

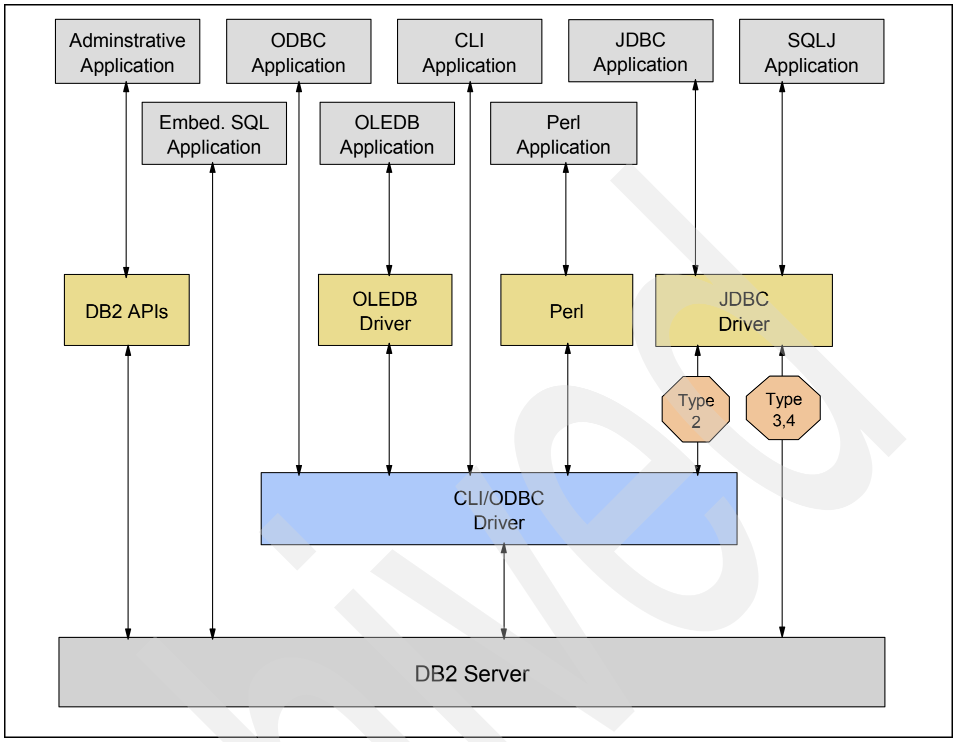


Figure 1-12   DB2 application connections

DB2 administrative API

DB2 provides numerous administrative APIs, which allow applications to perform database administration tasks available in the DB2 UDB Control Center, for example, importing and exporting data, creating, activating, backing up, or restoring a database. These calls can be included within embedded SQL and DB2 CLI applications. Examples of API programs can be found in the DB2 home directory *sqllib/sample/* for different programming languages. For additional information refer to DB2 *Administrative API Reference*, SC09-4824.

Embedded SQL statements in applications

Two different kind of SQL statements have to be distinguished:

- Static SQL statements

  With static SQL statements, you know before compile time the SQL statement type and the table and column names. The only unknowns are the specific data values the statement is searching for or updating. These values can be represented in host language variables.

Before compiling and linking the program, precompiling and binding of the embedded SQL statements has to be done. Precompiling converts embedded SQL statements into DB2 run-time API calls that a host compiler can process, and creates a bind file. The **bind** command creates a package in the database. This package then contains the SQL operation and the access plan that DB2 will use to perform the operation.

- Dynamic SQL

  Dynamic SQL statements in an application are built and executed at run time. For a dynamically prepared SQL statement, the syntax has to be checked and an access plan has to be generated during the program execution.

  Examples of embedded static and dynamic SQL can be found in the DB2 home directory, *sqllib/samples/*.

DB2 Call Level Interface (DB2 CLI)

DB2 CLI is a programming interface that can be used from C and C++ applications to access DB2 databases. DB2 CLI is based on the Microsoft Open Database Connectivity (ODBC) specification, and the ISO CLI standard. The DB2 CLI library can be loaded as an ODBC driver by an ODBC driver manager. DB2 CLI includes support for many ODBC and ISO SQL/CLI functions, as well as DB2-specific functions.

When using DB2 CLI, the application passes dynamic SQL statements as function arguments to the database manager for processing. Because these applications use common access packages provided with DB2, DB2 CLI applications do not need to be precompiled or bound. Only compiling and linking of the application is needed. Before DB2 CLI or ODBC applications can access DB2 databases, the DB2 CLI binds files that come with the DB2 Application Development Client to each DB2 database that will be accessed. This occurs automatically with the execution of the first statement.

Typically when building an ODBC application an ODBC driver manager is needed, which is normally provided by platform vendors like Microsoft or others. There is also an ODBC driver manager for Linux available, which can be found at `http://www.unixodbc.org/`. However, in environments without an ODBC driver manager, DB2 CLI is a self-sufficient driver that supports a subset of the functions provided by the ODBC driver. Examples of C programs using CLI calls can be found in the DB2 home directory, *sqllib/samples/cli.* For additional information regarding CLI refer to *Call Level Interface Guide and Reference*, Volume 1 and Volume 2, SC09-4849 and SC09-4850.

Java Database Connectivity application (JDBC)

DB2's Java support includes JDBC, a vendor-neutral dynamic SQL interface that provides data access to applications through standardized Java methods.

Similar to DB2 CLI, you do not have to precompile or bind a JDBC program. As a vendor-neutral standard, JDBC applications offer increased portability. The JDBC API, which is similar to the CLI/ODBC API, provides a standard way to access databases from Java code. The Java code passes SQL statements to the DB2 JDBC driver, which handles the JDBC API calls. Java's portability enables you to deliver DB2 access to clients on multiple platforms, requiring only a Java-enabled Web browser or a Java runtime environment.

DB2 Version 8 offers different ways of creating Java applications, either using a Type 2, Type 3, or Type 4 JDBC driver:

- Type 2 driver

  With a Type 2 driver, calls to the JDBC application driver are translated to Java native methods. The Java applications that use this driver must run on a DB2 client, through which JDBC requests flow to the DB2 server. This is typically how DB2 is accessed by WebSphere Application Server.

  Tip: If you want to prototype CLI calls before placing them in a program, you can use the db2cli.exe (Windows) or db2cli (Linux) file in the *sqllib/samples/cli* directory. There is also a document, *INTCLI.DOC*, which advises you on how to use the utility.

- Type 3 driver

  The DB2 JDBC Type 3 driver, also known as the applet or net driver, consists of a JDBC client and a JDBC server. The DB2 JDBC applet driver can be loaded by the Web browser along with the applet. Another way is to use the applet driver in standalone Java applications. When the applet requests a connection to a DB2 database, the applet driver opens a TCP/IP socket to the DB2 JDBC applet server that is the machine where the Web server resides. After a connection is set up, the applet driver sends each of the subsequent database access requests from the applet to the JDBC server through the TCP/IP connection. The JDBC server then makes corresponding DB2 calls to perform the task. Upon completion, the JDBC server sends the results back to the JDBC client through the connection. The use of the Type 3 driver is being deprecated with DB2 Version 8 because of the new Type 4 driver.

- Type 4 driver

  The JDBC Type 4 driver, which is new for Version 8, can be used to create both Java applications and applets. To run an applet that is based on the Type 4 driver, only a Java-enabled browser is required, which downloads the applet and the JDBC driver (db2jcc.jar). To run a DB2 application with a Type 4 driver, only an entry for the JDBC driver in the *class path,* and no DB2 client, is required. This Type 4 driver provides the initial implementation of the new JDBC driver architecture known as the IBM DB2 JDBC Universal Driver. The

Universal Driver is architected as an abstract JDBC processor that is independent of driver-type connectivity or a target platform. Examples of JDBC calls can be found in *sqllib/samplesjava/jdbc*. For detailed information on the Java support provided by DB2 Version 8, we strongly recommend the manual *Developing Enterprise Java Applications Using DB2 Version 8*.

Embedded SQL for Java (SQLj)

DB2 Java embedded SQL (SQLj) support is provided by the DB2 AD Client. With DB2 SQLj support (in addition to DB2 JDBC support), SQLj applets, applications and stored procedures can be built, which contain static SQL and use embedded SQL statements that are bound to a DB2 database.

SQLj applications use JDBC as a foundation for tasks, such as connecting to databases and handling SQL errors, but also contain embedded static SQL statements in separate SQLj source files. Unlike the other languages that can contain embedded SQL (COBOL, C, C++), the Java code is not precompiled; instead, the SQLj translator converts SQLj clauses into JDBC statements. As SQLj shares its underlying connection with that of JDBC applications, it can connect to DB2 using either Type 2, Type 3 or Type 4 drivers.

Examples of SQLj calls can be found in sqllib/samplesjava/sqlj. More detailed information can also be found in the article "Developing Enterprise Java Applications Using DB2 Version" at:

http://www.ibm.com/developerworks/db2/library/techarticle/0209hutchison/0209hutchison.html

ActiveX Data Objects and Remote Data Objects (Windows only)

DB2 supports ActiveX Data Object (ADO) applications that use the Microsoft OLE DB to ODBC bridge. ActiveX Data Objects (ADOs) allow you to write applications to access and manipulate data in a database server through an OLEDB provider.

When installing the client version of DB2 Version 8.1 for Windows, optionally IBMDADB2, the IBM OLE DB 2.0 compliant provider for DB2 can also be installed. With this driver the DB2 database does not have to be cataloged as an ODBC data source.

Remote Data Objects (RDOs) provide an information model for accessing remote data sources through ODBC. RDO offers a set of objects that make it easy to connect to a database, execute queries and stored procedures, manipulate results, and commit changes to the server. As RDO implements a thin code layer over the ODBC API, it requires an ODBC data source to be created for the DB2 database you are connecting to.

ADO.NET

DB2 UDB supports Microsoft's ADO.NET programming interface via a native managed provider. High-performing WinForm, WebForm, and mobile WebForm applications can be developed using the ADO.NET API. When used in conjunction with stored procedures and the federated database capabilities of DB2 UDB and DB2 Connect™ servers, this data access can be extended to include a wide variety of other data sources, including non-DB2 mainframe data (such as VSAM, CICS®, IMS™), Informix® Dynamic Server (IDS), Microsoft SQL Server, Sybase and Oracle databases, as well as any data source that has an OLE DB Provider available. The IBM DB2 .NET Data Provider is a native provider written in managed C# code to deliver high-performing, secure access to DB2 data.

Perl DBI

DB2 supports the Perl Database Interface (DBI) specification for data access through the DBD::DB2 driver. The Perl DBI module uses an interface that is similar to the CLI and JDBC interfaces, which makes it easy to port Perl prototypes to CLI and JDBC. More information about Perl DBI can be found at:

`http://www-306.ibm.com/software/data/db2/perl/`

1.6 Summary

This solution is a proof point for implementing total enterprise solutions on Linux using scalable data management software, running on a powerful server cluster and an extremely fast network fabric, complemented with tools that provide extreme reliability and intelligent management.

DB2 Integrated Cluster Environment provides all the power and reliability you would expect from any large database solution on an attractive cost effective platform. DB2 Integrated Cluster Environment integrates hardware, software and services to provide you with an intelligent solution that will reduce your implementation time.

2

Selecting your cluster

There are many infrastructure options for deploying DB2 Universal Database on multi-node Linux clusters. DB2 Integrated Cluster Environment simplifies the cluster options to best-of-breed hardware configurations with an offering called IBM eServer Cluster 1350.

The Cluster 1350 is an integrated offering that is designed to be easy to configure and lower deployment effort. By reducing the time and resources needed for researching, assembling, integrating, testing, and tuning a Linux cluster, the Cluster 1350 can help an organization speed time-to-production. Even within DB2 Integrated Cluster Environment, a Cluster 1350 choice needs to be made for components that constitute the cluster in order to best meet the needs of your database solution. In this chapter we discuss options for cluster ingredients that constitute DB2 Integrated Cluster Environment and considerations for their selection. We cover:

- Platforms and processor architectures
- Server types and models
- Storage infrastructure and connectivity
- Network interconnects
- Linux distributions
- Sample configurations

2.1 Platform

DB2 UDB is available for a variety of Linux platforms; however, at the time of writing, most DB2 Integrated Cluster Environment configurations have been mainly tested and deployed on two platforms: IA32 and AMD64. By IA32 we are referring to 32-bit Intel processor architecture, also generically known as x86. And AMD64 refers to AMD's technology that extends the x86 platform to 64-bits, hence generically called x86-64.

We limit discussion to these two architectures (x86 and x86-64), as they are the most commonly used for Linux clusters, primarily due to their compelling “price/performance” characteristics. With the growing appeal of Linux on IBM POWER™ technology, future DB2 Integrated Cluster Environment offerings based on IBM servers with PowerPC® chips may also be introduced. It is possible to extend and apply many of the clustering concepts outlined in this document to other Linux platforms.

It should be noted that the overall solution could be a hybrid one involving both 32-bit and 64-bit servers and clients; however, a single DB2 database can only be partitioned across servers belonging to the same platform. In fact, all of the servers in a database cluster should be running the same levels of the Linux distribution and DB2 UDB.

2.1.1 IA32 (and x86)

When most people think about Linux, the 32-bit Intel environment is the default and the most popular choice for deployment today. Among the available Linux platforms, this platform is deemed to be the most mature, offers the largest range of industry-standard (“commodity”) servers, and has the widest availability of applications and utilities. DB2 UDB has been ported to this platform since 1998. In addition to DB2 UDB, a large number of other add-on DB2 components and extenders are available for this platform.

However, the drawback of this platform is the address limitation inherent in 32-bit architectures. Processes in this environment can only directly address four gigabytes of memory, which limits database shared memory area to a default of about 1.7 gigabytes (and up to 2.5 gigabytes on certain Linux distributions). The shared memory area is where database buffer pools and sort heaps are kept and are critical to database performance. While it is possible to address more than four gigabytes on 32-bit operating systems using memory extensions such as PAE and AWE or though in-memory file systems, DB2 UDB has chosen not to utilize these on Linux due to the performance overhead of these methods and availability of several 64-bit Linux platforms.

2.1.2 AMD64 (and x86-64)

A 64-bit architecture provides DB2 servers with access to vast amounts of memory, well beyond the maximum of two to three gigabytes of shared memory available on 32-bit systems. Until recently, 64-bit systems came at a substantial cost premium. AMD changed the 64-bit landscape by releasing AMD64 technology that added 64-bit extensions to the x86 architecture. Systems with 64-bit AMD Opteron processors do not cost too much more than similarly configured 32-bit Intel Xeon systems.

This platform allows running both the existing 32-bit x86 applications as well as native 64-bit ones, thereby providing flexibility and making the transition to 64-bit easier. Applications such as database managers that benefit the most from the 64-bit architecture can be among the first to be deployed to this platform, while client applications can be kept as 32-bit or moved to 64-bit at their pace. Given the benefits and success of this platform, it is not surprising that Intel has also released its 64-bit extended technology known as EM64T.

DB2 UDB was the first major commercial database for this platform. Whether you want to deploy on AMD64 or Intel EM64T, there is a common version of DB2 UDB that runs on both of these x86-64 architectures. The x86-64 version of DB2 UDB is a hybrid one that allows you to create both 32-bit and 64-bit DB2 instances on the same system; see Figure 2-1.

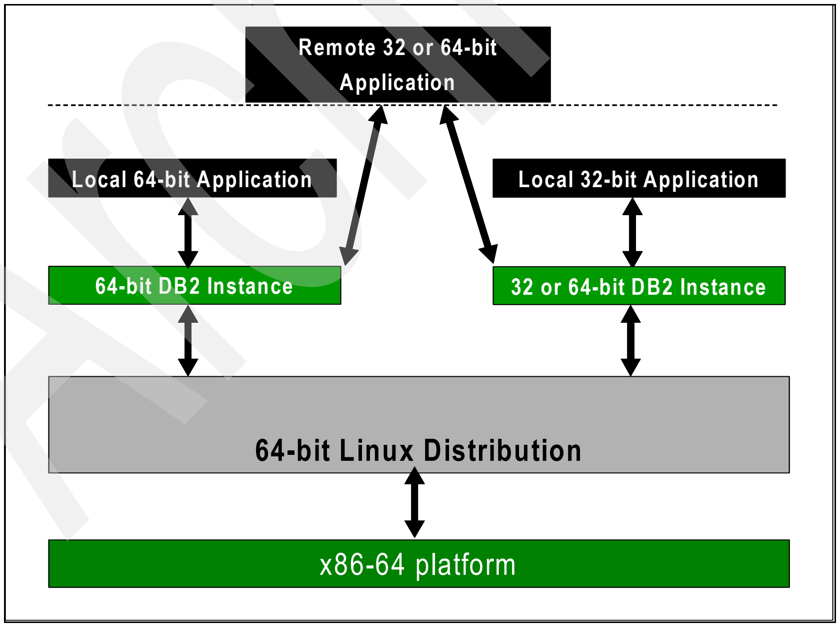


Figure 2-1 Application execution scenarios against a database on x86-64

The ability to exploit large amounts of memory, buffer pools, etc. only exists on 64-bit instances; however, you can execute either 32-bit or 64-bit local applications (running on the same system) against it. A 32-bit instance, on the other hand, allows the direct execution of only 32-bit applications. Remote client applications (running on different systems or on the same system using TCP/IP loopback) can connect to either a 32- or 64-bit instance on the database server.

Although the platform is a hybrid one, and theoretically you can run any 32-bit x86-based Linux application on it, not all applications and tools have been certified or validated for running on a 64-bit Linux distribution. This is also the case for certain DB2 extenders and add-on products that can only operate on a 32-bit distribution.

2.1.3 Platform selection

Below are some criteria to help you with the selection of a Linux platform for running DB2 UDB clusters.

If your solution:

- Requires the use of 32-bit server-side DB2 add-ons or extenders, consider x86
- Involves running 32-bit ISV applications that are not certified against a 64-bit environment, consider x86
- Will benefit from having more memory in the server or requires large buffer pools or heaps, consider x86-64

Note: With the availability and replacement of many x86 server models with x86-64 capable versions, the usage of this platform may become quite pervasive over the coming years. Even on x86-64 hardware you can run 32-bit operating systems. The references to x86 and x86-64 in the platform selection section imply running x86 versions of DB2 UDB on 32-bit Linux distributions and x86-64 versions of DB2 UDB on 64-bit Linux distributions.

2.2 Server types and models

There are plenty of choices available when it comes to servers. For DB2 Integrated Cluster Environment we have limited the selection to the following shortlist of rack-mountable systems that are orderable as IBM eServer Cluster 1350 configurations[1]:

- IBM eServer xSeries Model 335 (and x336)
- IBM eServer Model 325 (and e326)
- IBM eServer xSeries Model 345 (and x346)

- IBM eServer xSeries Model 365 (x365)
- IBM eServer BladeCenter™ with HS20 Blade Servers (HS20)

Note that all of the servers in the list above support at least two processors. There are also choices for uniprocessor systems that are less expensive; however, we feel that at a very minimum, a dual processor system provides the parallelism and scalability that is important for database workloads. You could even start with a single processor in each system; however, you have the capability of adding at least another one if needed in the future.

For our shortlist we have also omitted servers that can accommodate more than four processors. It is quite possible to carve a virtual cluster out of a large SMP system, and some people may prefer this to simplify management. However, systems with up to four processors tend to be the most common for building Linux clusters because of their price points. The price for systems with more than four processors increases significantly. In many cases the price of an 8-way system substantially exceeds the price of two 4-way systems.

Here are some of the factors to consider when selecting servers for the database cluster:

- Server density and form factor
- Memory, disk, and PCI slot expandability
- Component redundancy
- Number of CPUs per server
- Processor type, speed, cache
- Bus speeds

The following sub-sections highlight some of these factors as they relate to selected server models.

2.2.1 1U "Pizza-box" - x335

A 1U server is the quintessential building block in a Linux cluster. This type of server is one unit or 1.75 inches high, allowing 42 such servers to be stacked in a six foot high rack, and making it very popular in space-constrained data centers. Because of its dimensions, 1U servers are also commonly known as "pizza-boxes", as they are just the right size to support a pizza.

1U 2P (2P refers to a system with two processors) are highly popular for Linux clusters because their price/performance ratios appeal to many people. Among their drawbacks for database environments is that they lack redundant components such as power supplies, and the number of expansion slots (for example, PCI, disks, memory) is often limited. Not being able to add adaptors

[1] The 4-way x365 is not as available as a standard Cluster 1350 configuration, but it may be ordered as a "special" configuration.

(say more than two PCI) also limits the redundancy of the system. However they are generally economical enough that they do not need to provide five 9's of uptime; in case of a server failure, they can be failed over to an identical system, replaced, and the defective system can be repaired or removed from service.

In a DB2 UDB cluster, such 1U 2P are generally fitted with 4 GB of RAM when used in a 32-bit environment. Some servers allow 2GB RAM modules to be used to double the memory capacity of the server. At the time of writing, 2-GB memory modules cost several times more than their 1-GB counterparts, but their prices are expected to come down with time, making their usage more justifiable. Furthermore, without a 64-bit database running on the system, the extra memory may not be efficiently utilized or it may even remain unused.

The Model 335 is a 1U server with one or two Intel Xeon processors. The system has a 266 MHz front-side bus, integrated Ultra 320 SCSI interface, two integrated 10/100/1000 Ethernet adapters, and two 64-bit/100 MHz PCI-X slots (one full-length and one half-length). The Model 335 is designed for space efficiency and usability in a rack environment, as shown in Figure 2-2. Model 336 is a replacement for the x335 and offers other improvements over its predecessor, including Intel EM64T processors, Eight DIMM slots, light path diagnostics, and redundant hot-swap power supplies.

Figure 2-2 x335

2.2.2 AMD64 “Pizza-box” - e325

The e325, as shown in Figure 2-3 on page 39, is also a 1U 2P rack-optimized server. It takes AMD Opteron (64-bit) processors, has two hot swap bays for IDE or Ultra320 SCSI drives, and two 64-bit PCI-X slots. The e325 is a 64-bit proven server that has been used in several DB2 Integrated Cluster Environment benchmarks and customer configurations. When running a 64-bit database manager on the system, it can make use of all of 12 GB of RAM that can be added to the system.

The IBM eServer e326 is a follow-on model to the e325 and supports eight memory slots.

Figure 2-3 e325

2.2.3 2U 2P - x345

The Model 345 (Figure 2-4) is a 2U server featuring one or two Intel Xeon processors. The system has a 533 MHz front-side bus and utilizes PC2100 ECC RAM. It has an integrated Ultra320 SCSI interface that supports mirroring, two integrated 10/100/1000 Ethernet adapters, and five PCI slots (two 64-bit/133 MHz PCI-X, two 64-bit/100 MHz low profile PCI-X, and one 32-bit/33 MHz half length PCI). The five PCI slots may optionally be populated with a combination of SCSI RAID cards, Fibre Channel cards, and network adapters.

Figure 2-4 x345

The Model 345 supports up to 8 GB of memory per node (maximum of 4 GB until 2-GB DIMMs become available). Up to six internal hot-swap Ultra320 SCSI disks can be configured in sizes of 18, 36, or 72 GB, giving a total maximum of 440 GB. These can optionally be mirrored using the on-board LSI-Logic SCSI controller, which supports hardware level mirroring (RAID1). If increased throughput and/or capacity is desired, an optional RAID5i adapter can be installed that works in conjunction with the on-board controller.

Although more expensive than the x335, this server offers better redundancy, and with five PCI slots you can add more adaptors to drive a lot more I/O between the server and external storage.

Model 346 is a replacement for x345 and supports Intel EM64T processors and eight memory DIMM slots.

2.2.4 4P - x365

Model 365 (Figure 2-5) is a 3U rackable server and can take up to four Xeon MP processors. It has 16 memory slots and can support up to 32 GB of RAM using 2-GB modules. The server offers built-in redundancy and has hot-swappable redundant power supplies. It can take up to six hot-swappable hard drives and has a on-board Ultra320 SCSI controller. It comes standard with six PCI slots (five active), and if you need additional expansion capacity you can add a RXE-100 Remote Expansion Enclosure for 12 PCI-X slots.

Despite generally being more than double the price of two 2-way systems, some customers prefer a 4-way server instead of a 1U 2P system to build a database cluster. This is not only due to better redundancy, availability, and expandability characteristics of a 4-way server, such as the x365, but also because this option offers better scalability and growth manageability. That is, you could start out with two processors in the system, and as the database grows, more CPUs, memory, and expansion cards can be added without provisioning another server in the cluster. Fewer systems to manage and administer can sometimes reduce ongoing ownership costs.

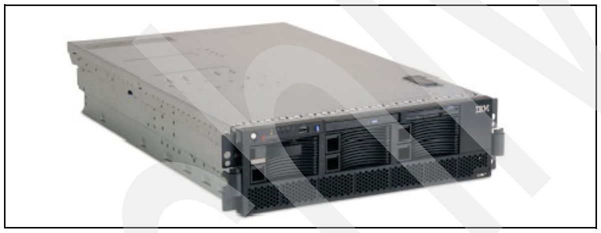


Figure 2-5 x365

2.2.5 HS20 (2-way; 14 Blades in a 7U unit)

Blade servers enable higher density with far greater ease. With switches and power units shared, precious space is freed up and cable tangle reduced. Blades servers can also help reduce total cost of ownership and improve availability by allowing blades to be quickly replaced. And additional capacity can also be provisioned rapidly.

The Model HS20 is the blade server (displayed in the right-hand side portion of Figure 2-6 on page 41) that is housed inside of the BladeCenter 7U chassis (displayed in the left-hand portion of Figure 2-6 on page 41). The HS20 is a 1U vertically mounted blade server and comes with either one or two Intel Xeon DP processors. The HS20 has room for 4 GB of (PC2100 ECC DDR) memory and comes with a 533 MHz front-side bus. The HS20 can house an Ultra 320 SCSI card, and two internal drives. The HS20 has two integrated 10/100/1000

Ethernet adapters, and daughter board based expansion slot. The Model HS20 is so compact that 14 blades can easily fit into the BladeCenter chassis.

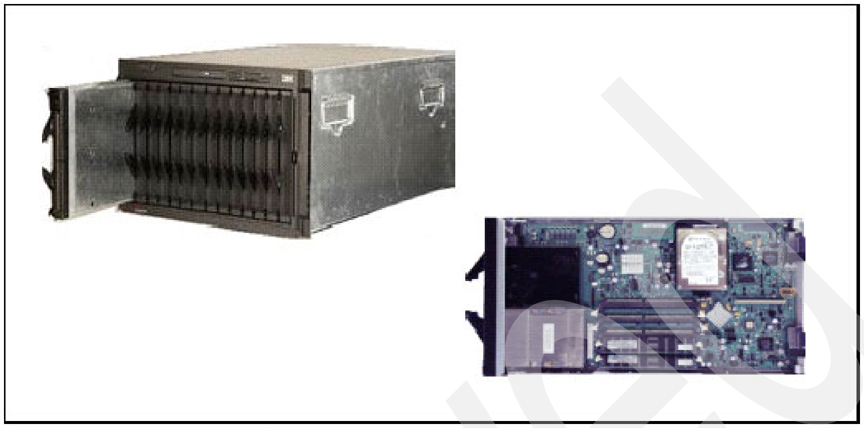


Figure 2-6 IBM BladeCenter chassis (left) and Blade (right)

One of the drawbacks of the BladeCenter for a database cluster is the number of expansion slots that are available, which in turn limit the amount of I/O that can be driven by each Blade. You can add only one dual port Fibre Channel expansion card to each blade. Although you can add a Fibre Channel switch module to the BladeCenter chassis, it is not recommended for I/O heavy database workloads since the switch module contains only two uplink ports. Instead, the optional Optical Pass-thru module (OPM) that provides unblocked, unswicthed connection to each Blade is recommended. Assuming 155 MBps of effective throughput per Fibre Channel port, using a conservative sizing, this could theoretically allow up to 300 MBps of sustained storage communication per blade.

2.2.6 Server selection

The following may help with choosing the database server nodes in the cluster.

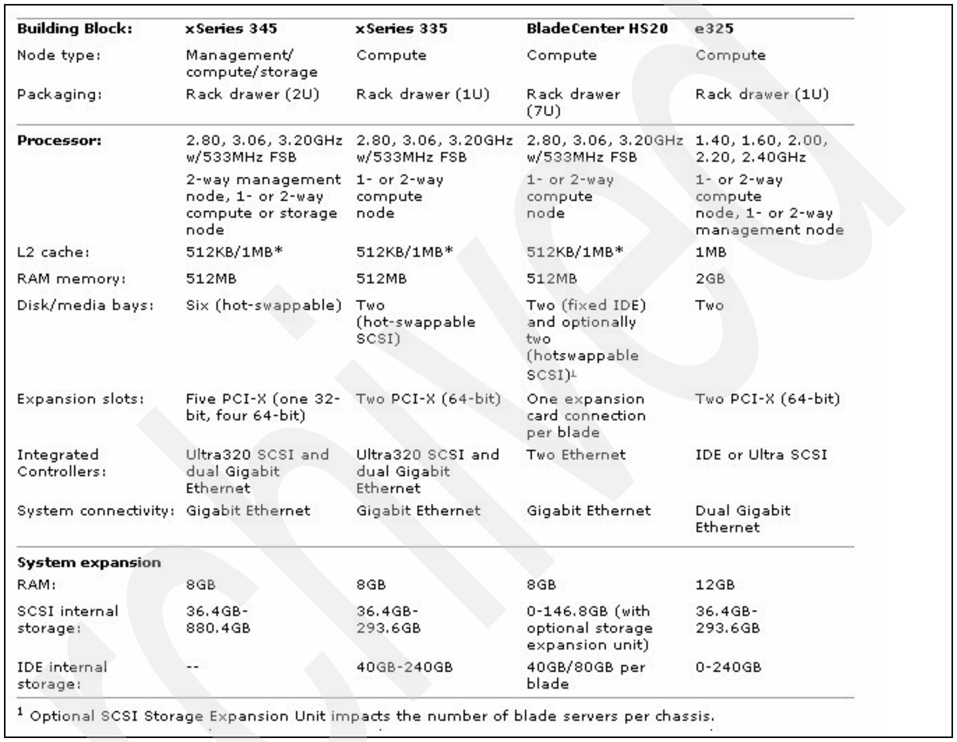
If you:

- Need a low cost “pizza-box” style server with minimal internal expandability or component redundancy, consider x335
- Want to go with a “tried and proven” 64-bit “pizza-box” server, consider e325
- Are looking for a 2-way server that has some built-in redundancy and room for expansion, consider x345
- Prefer a server with redundant components and ample room for growth, consider x365 with capacity for up to four CPUs

- Require a space-saving solution that does not demand very high I/O throughput and offers ease of maintenance through flexible node addition/substitution capabilities, consider BladeCenter with HS20

Example 2-7 is a summary of Cluster 1350 configurations. For the most current information, refer to the Web site:

http://www.ibm.com/servers/eserver/clusters/hardware/1350_specs.html



Building Block:	**xSeries 345**	**xSeries 335**	**BladeCenter HS20**	**e325**
Node type:	Management/ compute/storage	Compute	Compute	Compute
Packaging:	Rack drawer (2U)	Rack drawer (1U)	Rack drawer (7U)	Rack drawer (1U)
Processor:	2.80, 3.06, 3.20GHz w/533MHz FSB 2-way management node, 1- or 2-way compute or storage node	2.80, 3.06, 3.20GHz w/533MHz FSB 1- or 2-way compute node	2.80, 3.06, 3.20GHz w/533MHz FSB 1- or 2-way compute node	1.40, 1.60, 2.00, 2.20, 2.40GHz 1- or 2-way compute node, 1- or 2-way management node
L2 cache:	512KB/1MB*	512KB/1MB*	512KB/1MB*	1MB
RAM memory:	512MB	512MB	512MB	2GB
Disk/media bays:	Six (hot-swappable)	Two (hot-swappable SCSI)	Two (fixed IDE) and optionally two (hotswappable SCSI)[1]	Two
Expansion slots:	Five PCI-X (one 32-bit, four 64-bit)	Two PCI-X (64-bit)	One expansion card connection per blade	Two PCI-X (64-bit)
Integrated Controllers:	Ultra320 SCSI and dual Gigabit Ethernet	Ultra320 SCSI and dual Gigabit Ethernet	Two Ethernet	IDE or Ultra SCSI
System connectivity:	Gigabit Ethernet	Gigabit Ethernet	Gigabit Ethernet	Dual Gigabit Ethernet
System expansion				
RAM:	8GB	8GB	8GB	12GB
SCSI internal storage:	36.4GB-880.4GB	36.4GB-293.6GB	0-146.8GB (with optional storage expansion unit)	36.4GB-293.6GB
IDE internal storage:	--	40GB-240GB	40GB/80GB per blade	0-240GB

[1] Optional SCSI Storage Expansion Unit impacts the number of blade servers per chassis.

Figure 2-7 IBM eServer Cluster 1350 configuration summary

2.3 Storage

Storage is probably one of the single most critical hardware components inside a database cluster, as it houses the most valuable aspect of the database—the data—and tends to be the most expensive part of the cluster. Therefore storage architecture and selection merits careful consideration. In this section we discuss the two extremes for storage topologies: Local vs. central, and the various types of storage options from IBM TotalStorage:

- Enterprise storage servers and SANs (ESS)
- Ds4000 Series (FAStT)

- DS400 and DS300 (EXP)
- Direct attached SCSI stoarage (ServeRaid+EXP)
- Network attached storage (NAS)

2.3.1 Localized vs. centralized storage topologies

DB2 UDB on Linux does not employ “shared-disk” architecture. That is, different partitions and nodes in the database cluster do not require a storage infrastructure that is shared or accessible from all nodes. Storage for each database partition can be independent; therefore it is possible to have storage that is local to each node in the cluster, as shown in Figure 2-8.

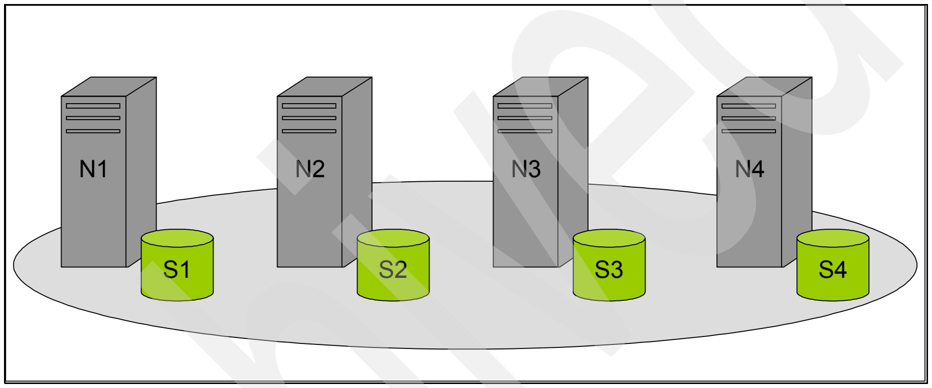


Figure 2-8 Each node with its own local storage

In its simplest form, local storage at each node could just be internal disks. With individual disk drive sizes reaching 250 GB, it is theoretically possible to have just four internal disks to house a terabyte of data at each node. In practice, however, for performance and availability reasons it makes more sense to spread your data across a large number of smaller capacity fast disks rather than a few large capacity disks. This is because accessing data from disk is typically the slowest part of a database operation, since hard drives on which data resides are several orders of magnitudes slower than memory. By having data across more disks and initiating parallel access against the disks, the time to retrieve data can be significantly released. In the most performance demanding configurations and for database benchmarks, it is not uncommon to see disk drives to processor ratios of 20:1 or higher. In less demanding customer environments 6–14 disk drives for every processor or database partition are quite common.

Most servers do not have the capacity to house enough disks internally. And for availability reasons, some level of RAID is often used, which further increases disk drive requirements. For this reason, external storage arrays are almost always needed in database clusters. The most inexpensive type of disk arrays tend to be the ones that can be locally or directly attached to each individual server. Hence, locally attached storage options are popular where cost of the storage hardware is the most important criteria.

For improving availability of the solution it may be desirable to connect more than one database server to the same disk array. In case of server failure, another database server connected to shared storage can then take over the operations of the failed server.

At the other end of the spectrum are storage area networks and storage farms where storage for a large number of servers and applications is consolidated; see Figure 2-9. By consolidating storage across multiple servers or centralizing across an enterprise (or its parts), synergies and cost efficiencies can be derived through common infrastructure management and capacity balancing. However, the cost of hardware components in a shared storage environment tend to be more complex, need to have advanced features, and cost more. For example, bandwidth requirements are higher and there may be the need for fibre switches for storage servers to which multiple database servers are connected.

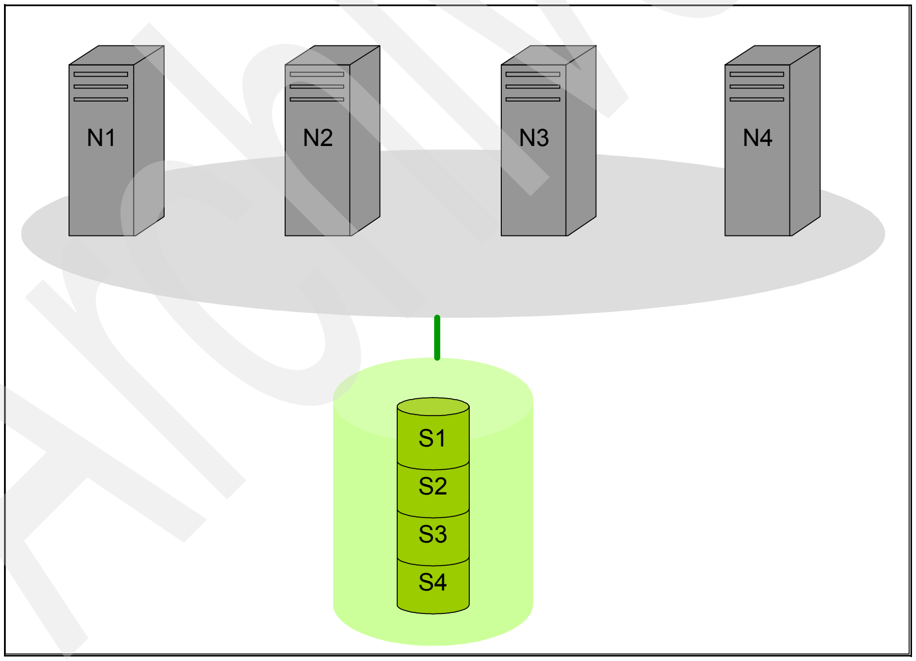


Figure 2-9 Storage for all nodes inside a consolidated infrastructure

It is possible to implement solutions in-between the two extremes of centrally vs. locally connected storage topologies. Figure 2-10 on page 45 illustrates storage options discussed in the following sections, and their suitability under the two storage topologies.

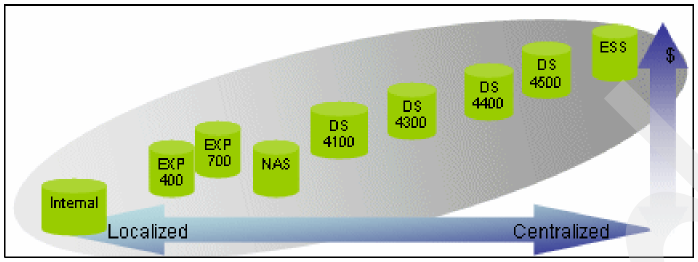


Figure 2-10 Storage options

2.3.2 Enterprise Storage Servers (ESSs)

Enterprise-class storage has been the traditional option of choice for large databases and mission-critical data. IBM has an offering called Enterprise Storage Server®, also commonly referred to as “Shark”. It offers the highest levels of availability, performance, manageability, and scalability among the storage options listed in this document. It is also the most expensive. ESS enables attachment of multiple heterogeneous servers, helps address unpredictable growth requirements, and features state-of-the-art copy services for rapid backup and disaster recovery.

While it is quite possible to connect a single database server to a single ESS system, it is more cost-effective and beneficial to deploy it in a consolidated storage configuration, that is, use a Fibre Channel SAN fabric to connect several servers and clusters to ESS racks.

Some companies have defined storage policies that require the use of an enterprise SAN. The choice of your storage solution under such situations is quite straightforward. However, one of the motivations for deploying Linux clusters is to reduce hardware costs by using mass-produced industry-standard components. Therefore some may question the use of an expensive enterprise SAN. Your company may already have an ESS-based SAN with excess storage capacity and sunk costs, making it an affordable storage option for your database cluster on Linux.

Another advantage of connecting your Linux-based database cluster to the enterprise SAN is the ability to tap into existing data life-cycle policies and processes. For example a tape backup or cross-site disaster recovery strategy may already be in place for your enterprise SAN environment. Whereas if you have storage arrays that are local to each server, you may need to devise and manage backup and disaster recovery policies for multiple storage systems separately. This may be acceptable when there are a small number of servers and local storage systems. For a database cluster with lots of nodes, an enterprise storage solution may be a more manageable option despite being more expensive to deploy initially. Figure 2-11 shows a sample configuration using ESS-based SAN with remote site connectivity.

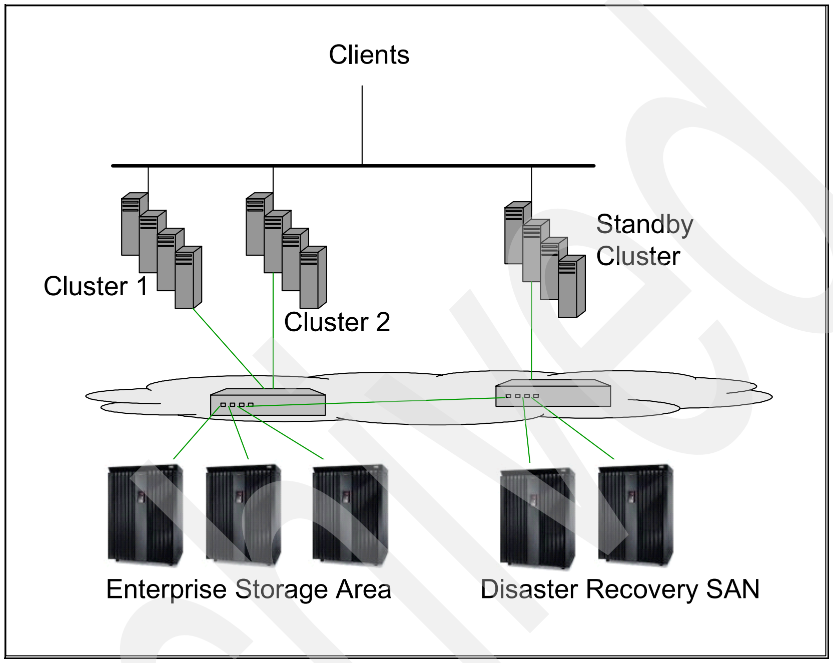


Figure 2-11 Sample layout for ESS-based SAN with remote site connectivity

2.3.3 DS4000 series

IBM DS4000 series (formerly FAStT) uses a family of storage servers, which offers more flexible fibre-attached solutions that cost less than many traditional enterprise storage solutions. Fibre Channel host bus adapters (HBAs) inside database servers provide connectivity to the DS4000 servers. Some DS4000 servers allow a certain number of disks to be inserted directly into the storage servers. Others require separate expansion (EXP) units for disks. In either case, additional disks can be added to the storage servers using more expansion units.

There is a range of storage servers in theDS4000 series:

- DS4500 (formerly FAStT900)
- DS4400 (formerly FAStT700)
- DS4300 (formerly FAStT600)
- DS4100 (formerly FAStT100)

Table 2-1 is a summary of features and capabilities of DS4000 servers. For the most current information refer to the Web site:

http://www.storage.ibm.com/disk/fastt/index.html

Table 2-1 IBM TotalStorage DS4000 Series

The DS4000 Series: A variety of options to help meet diverse storage needs

Product	Performance requirements	Max physical capacity	Max drives	Redundant drive channels	Applications	Environment	Bottom line
DS4100	Minimal	14.0TB	56 SATA	Two 2Gbps	File storage and cache online storage for quicker backups to tape	Workgroup	Affordable storage for near-line applications and quick access compared to
DS4300 Single	Minimal	2.0TB	14 FC	One 2Gbps	Specialized file storage	Workgroup	Affordable performance and bandwidth, and upgradable to dual and turbo
DS4300 Dual	Minimal	8.2TB	56 FC/112 SATA	Two 2Gbps	Specialized file storage	Workgroup	High IOpS performance for value (small-scale configuration)
DS4300 Turbo	Consistent	16.4TB	112 FC/112 SATA	Two 2Gbps Copy Services Options	Specialized file storage	Department al	High IOpS performance for value (small-scale configuration)
DS4400	Varying	32.9TB	224	Four 2Gbps	"Tier 2" specialized file storage	Department al	Excellent price-conscious IOPS performance and high availability for mid-range
DS4500	Varying	32.9TB	224	Four 2Gbps	Production, OLTP and replication	Enterprise	Highest throughput of DS4000 series and high IOPS performance for large configuration

DS4500

DS4500 delivers breakthrough disk performance and outstanding reliability. It has been the storage server of choice for many database solutions on AIX® and is suitable for storage consolidation. Because it is the most expensive in the DS4000 series, those looking to implement low-cost Linux clusters prefer some of the lower-end DS4000 servers. Figure 2-12 on page 48 shows a rack with DS4500 servers and disk expansion units.

Figure 2-12 A rack with FD4500 servers and disk expansion units

DS4400

DS4400 is a popular option for performance-intensive database clusters. It has a standard of four 2-Gbps fibre ports that can be expanded to eight via additional mini-hubs, allowing several severs to easily attach to it. It also provides ample scalability for multiple servers by allowing up to 224 drives to connected to it using 16 DS4000 EXP700s. If maximum throughput or redundancy is desired, you can chose to connect it to just one or two database servers with multiple HBAs or fibre ports. Figure 2-13 on page 49 shows a server attached to a DS4400 and several DS4000 EXP700s.

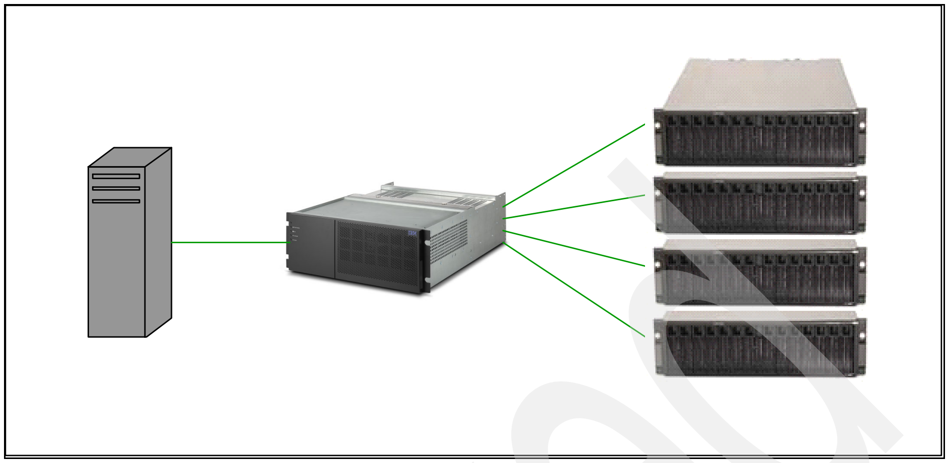


Figure 2-13 Server attached to a DS4400 and several DS4000 EXP700s

DS4300

DS4300 is a mid-level storage server and can accommodate 14 drives directly in the storage server. You can connect three additional DS4000 EXP700s to the DS4300. The Turbo feature allows up to using seven DS4000 EXP700s to be attached, making the total capacity 112 drives. The storage server can also be expanded using DS4000 EXP100s with less expensive SATA drives.

DS4100

DS4100 provides four 2-Gb Fibre Channel host ports and can take up to 14 SATA drives inside the controller. You can expand DS4100 capacity to 56 SATA disk drives with DS4000 EXP100 SATA disk drive units.

2.3.4 Direct Attached Fibre Storage (EXP)

The typical fibre-attached setups involve connecting database servers to storage servers, which in turn are connected to disk drive expansion units (as in Figure 2-14 on page 50). It is also possible to connect database servers directly to expansion units, as shown in Figure 2-14 on page 50.

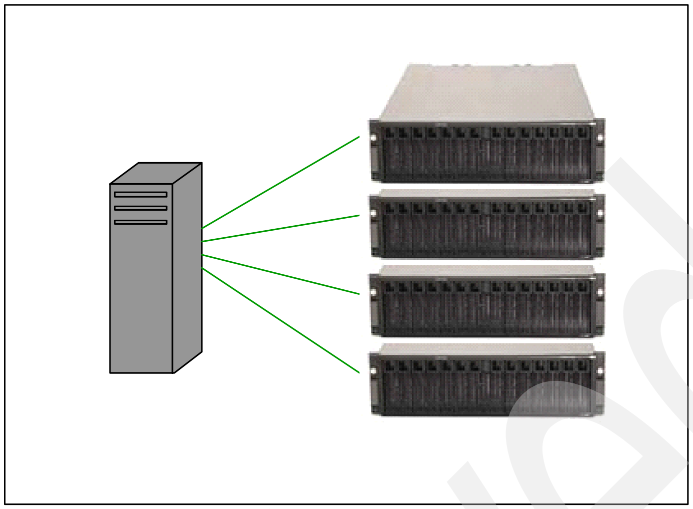


Figure 2-14 A database server attached directly to DS4000 EXP700 units

> **Note:** This storage configuration may not be supported with all servers.

An EXP700 unit, with a capacity of 14 hot-swap drives, has four 2-Gbps Fibre Channel ports, allowing a maximum of four database servers to connect to it. In practice it would be quite performance and capacity limiting to attach four servers to the expansion unit, and you probably would not attach more than two servers, and that to connect the second server only for failover purposes.

You could also use other Fibre Channel attached expansion units like EXP500 (with up to 10 FC disks) and EXP100 (with up to 10 SATA drives), but they offer even fewer ports.

In order to connect the servers directly to Fibre Channel disk expansion units, you still need host bus adapters inside the database servers. IBM DS4000 FC2-133 Host Bus Adapter is a 64-bit, low-profile (PCI-X/133MHz) single channel adapter that supports auto-sensing for 1-GBps or 2-GBps operations. For higher bandwidth or redundancy you could use QLA2342, which has two 2-Gbps channels.

Attaching servers directly to EXP units leads to a less expensive configuration. At the same time, you lose many of the benefits that DS4000 servers provide, including consolidation functions and storage management. Database servers also incur a processing overhead for having to control the storage units directly. The scalability (the number of units that can be added directly to a database server) and availability (that comes through sharing storage between servers for failover purposes) is also diminished.

2.3.5 Direct-attached SCSI storage (ServeRAID™)

SCSI-based products are a mainstream standard, produced in large quantities in the industry, and somewhat “commoditized”. They are therefore less expensive than Fibre Channel products. They also generally offer lower throughput than fibre-attached disk products.

Directly attaching external SCSI storage enclosures provides the ability to expand the storage capacity of your system beyond the physical limitations of the server. IBM provides EXP400 and DS300 storage expansion units that are SCSI attachable. EXP400 can support up to 14 Ultra320 hot-swappable SCSI drives on a single logical bus, and with the optional twin tailing/split bus feature, it allows two servers to access the data. IBM also has an older external SCSI storage enclosure called EXP300.

Connection to the SCSI disk enclosures is facilitated by adding SCSI controllers to the servers. ServeRAID-6M from IBM is a dual-channel Ultra320 SCSI PCI-X controller that delivers a robust Redundant Array of Independent Disks (RAID) solution. You could also consider using the less expensive ServeRAID-4 family of Ultra160 SCSI controllers if your I/O throughput requirements or measured results are lower than 160 MBps.

Since you can only attach a maximum of two servers per expansion array, this storage attachment option is suitable for storage of data local to a single server only, and attaching a second server for failover purposes. Unlike fibre cables, which can go long distances, SCSI-attached storage expansion units need to be kept close to the servers due to length limitations of SCSI cables.

2.3.6 Network Attached Storage (NAS)

There has been a lot of debate around NAS vs. SAN, and their use for database workloads. Fibre Channel SANs offer numerous benefits. Yet there are customers that deploy inexpensive NAS solutions for critical database applications and prefer their simplicity for management.

NAS servers, popularly known as filers, have a lot of disks attached to them, and are connected to a Gigabit Ethernet network where other servers on the network can access files using NFS protocol. More recently iSCSI protocol is also being used for improved efficiency.

While it is possible to attach NAS filers to the general or a cluster-specific network with many servers connected to the network, it is recommended that for performance-sensitive database workloads, the filers be placed on dedicated networks with very few database servers attached to that network, as illustrated in Figure 2-15.

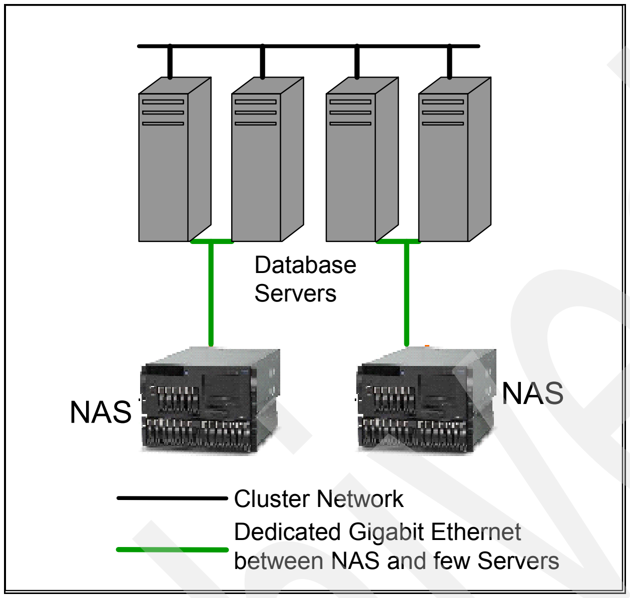


Figure 2-15 Database servers attached to NAS filers

In general, DB2 UDB does not support network-attached storage solutions unless they have been explicitly tested and validated[2] for DB2. For DB2 UDB Version 8.x for Linux, IBM TotalStorage NAS 200 and 300 series have been validated using NFS protocol, and IBM TotalStorage IP Storage 100i and 200i have been validated over iSCSI protocol. Some of these are no longer sold by IBM.

2.3.7 Storage selection

The following pointers may aid in the selection of your storage infrastructure for use with DB2 database clusters on Linux.

If you:

- Have sufficient spare capacity in your enterprise storage infrastructure or have extremely high availability and performance requirements, consider using Enterprise SAN

[2] To check for DB2-supported NAS solutions, go to the DB2 UDB technical support Web site: `http://www.ibm.com/software/data/db2/udb/support.html`
Click **Technotes** (FAQ) in the Self help section, and then search for NAS.

- Have high-performance requirements with advanced features such as volume copy or need to connect to a large number of storage expansion units, consider DS4400 (FAStT700) or DS4300 (FAStT600)
- Are looking for a more affordable fibre SAN solution with base features such as flash copy and hardware RAID, consider Serial ATA drive-based DS4100 (FAStT100)
- Want to reduce expense of fibre-attached storage and are willing to forgo the features of SAN servers by off loading storage serving and RAID controlling operations to the database servers, consider connecting storage expansion units such as the EXP700 or EXP500 directly to the database servers using host bus adaptors such as FC2-133 or QLA2342
- Want a more cost-effective SCSI-based solution, consider connecting the database servers directly to EXP400 expansion units (maximum of two servers per EXP unit) using SCSI controllers like the ServeRAID-6M or ServeRAID-4
- Prefer to do away with the complexity of managing a SAN, desire a highly economical storage infrastructure, and do not require raw volumes, consider using a DB2 UDB validated NAS filer on a dedicated network

Refer to Table 2-1 on page 47 for a summary of IBM DS4000 storage.

2.4 Network interconnect

There are mainly three types of dedicated networks involved in the database cluster:

- Application network - Connects application servers and client systems to the database servers
- Cluster interconnect - Connects the database servers in the cluster to each other
- Storage fabric - Connects the database servers to the storage infrastructure

Figure 2-16 on page 54 shows a database cluster with all these three types of network.

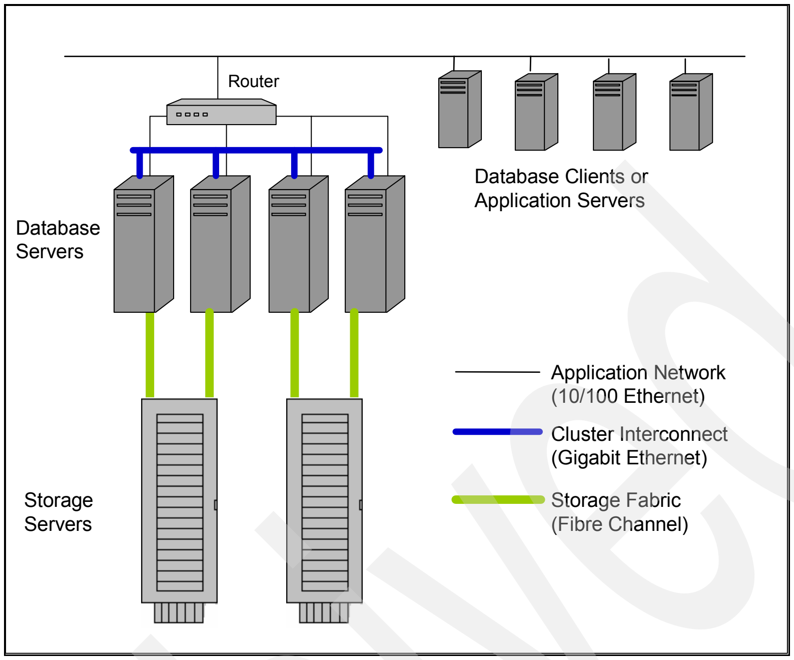

Router
Database Clients or Application Servers
Database Servers
Storage Servers
Application Network (10/100 Ethernet)
Cluster Interconnect (Gigabit Ethernet)
Storage Fabric (Fibre Channel)

Figure 2-16 Network connectivity for a database cluster

The application or the general network may not need to be a very fast one if a lot of data is not being exchanged between the database cluster and client/application systems. In some cases even a 10/100 Mbit Ethernet network can suffice. For transaction heavy environments, a Gigabit Ethernet network can be considered.

In a large SMP machine, the system bus provides the high-speed connectivity between processors being used for database workloads. In a cluster, therefore, a fast interconnect is needed between the nodes to allow quick message and data passing between database server nodes. Gigabit Ethernet ports come standard on most servers these days and can satisfy the needs of many workloads on database clusters. Being the industry standard, Gigabit Ethernet is quite affordable, too. Where the network is the bottleneck, a specialized interconnect can be considered. As a general rule, specialized network interconnects are not supported for DB2 UDB unless they have been specifically validated by IBM.

It is strongly recommended that the interconnect between database nodes not be shared with the general application network. That is, either separate network adaptors be used in each server for the purposes of connecting the nodes in the database cluster to each other and connecting the database servers to the rest of the company network, or a gateway/router be used between the two networks to isolate their traffic, or preferably both approaches be combined. This prevents unnecessary network traffic from disrupting high-bandwidth requirements between database servers in the cluster. The database has the capability (through configuration of db2nodes.cfg) to utilize a specific network interface for inter-node (FCM) database cluster connectivity.

The storage fabric is typically fibre-based, and thus tends to be independent from the other two Ethernet-based networks. Where Ethernet-connected NAS is used for storage, a separate network for attaching the storage is recommended.

2.4.1 InfiniBand option

InfiniBand is a high-bandwidth, low-latency, and low-overhead interconnect that can offer many benefits for database clusters. DB2 UDB was the first database to run on InfiniBand networks. IBM works closely with a number of InfiniBand vendors and products; several of them have been validated to run in DB2 UDB environments. Cluster 1350 has the option to be ordered with InfiniBand connectivity.

You can use InfiniBand as the interconnect between the database servers in the cluster, or as the unified fabric to replace the different kinds of networks used for the cluster. Use of InfiniBand as the cluster interconnect makes sense for workloads where a lot of simultaneous data or message passing takes place between the database servers and is constrained by the Gigabit Ethernet network. Where Gigabit Ethernet provides a line speed of 1 Gbps, InfiniBand can provide 10 Gbps at 4X, and 30 Gbps at 12X.

If you use InfiniBand as the cluster interconnect it is quite possible that you may not fully saturate its capacity of the network. This provides the opportunity to use InfiniBand as the unified fabric, as shown in Figure 2-17 on page 56. That is, a single InfiniBand network is connected to the database cluster instead of connecting each database server to the storage fabric and application network. This reduces the number of adapters in the database servers (which can be quite useful in 1U servers and blades), simplifies management, and can reduce costs.

Products such as the TopSpin 90 InfiniBand switch have the provision to act as gateways to Fibre Channel and traditional Ethernet networks. If you are using InfiniBand as the unified fabric, it is important to choose a switch according to the needs of your workload and ensure that sufficient bandwidth to storage is available. For example, the entry-level TopSpin 90 switches can only have two 2-Gbps Fibre Channel ports.

At the time of publication, DB2 UDB is supported over InfiniBand using IP over IB or SDP. In the future, more efficient transports may be supported.

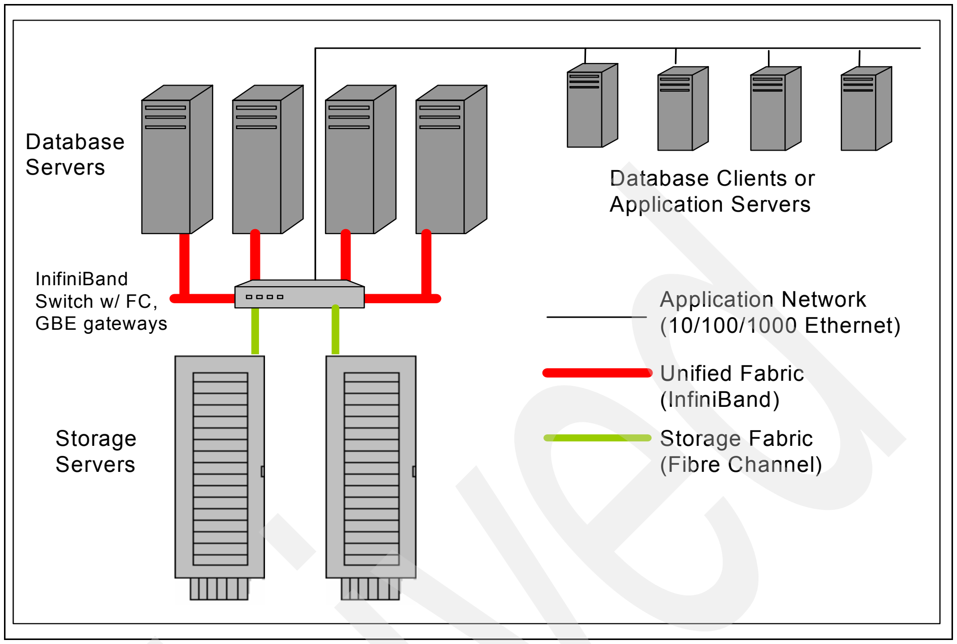

Database Servers
Database Clients or Application Servers
InifiniBand Switch w/ FC, GBE gateways
Application Network (10/100/1000 Ethernet)
Unified Fabric (InfiniBand)
Storage Servers
Storage Fabric (Fibre Channel)

Figure 2-17 InfiniBand being used as a unified cluster fabric

2.4.2 Network selection

The following information may help with your network architecture:

- Use a dedicated cluster network (Gigabit Ethernet as a minimum) to connect the database servers to each other. Keep it separate from the application network.
- For 32-bit or memory-constrained environments, use the network instead of shared memory for communication between logical database partitions on the same server.
- When the network is a bottleneck for your workload, consider using a higher-speed interconnect such as InfiniBand.
- InfiniBand may also be used as the unified fabric for the cluster to simplify management and reduce costs.
- Gigabit Ethernet, or a second InfiniBand network, may be used for redundancy or backup purposes.
- If using InfiniBand as the unified cluster fabric, ensure that sufficient bandwidth and Fibre Channel ports are available between the InfiniBand-FC gateway and the storage servers.

2.5 Linux distribution

There are several hundred Linux distributions (distros) available as free downloads on the Web or as purchasable offerings in the market. They can be broadly classified into four main categories:

- Community projects, for example, Debian, Fedora
- Consumer distributions, for example, SUSE Linux Pro
- Special-purpose distributions, for example, for embedded devices
- Enterprise-class distributions

Each of these types of distributions is suitable for different types of users and have varying release schedules, support policies, and costs. Previous versions of DB2 UDB were supported on any Linux distribution, as long as minimum kernel and library-level criteria were met. This could have led to situations where untested combinations could deliver unpredictable results. And in some cases, the Linux distribution provider may have offered limited or no maintenance and support for the distribution, which could have been problematic if a critical bug fix to the kernel was required for proper operation of the database on that distribution.

2.5.1 Validated enterprise-class Linux distributions

In order to ensure better interoperability of DB2 UDB with Linux platforms and enhance customer experience, a Linux Validation program was introduced with DB2 UDB V8.1. As a part of this program IBM focuses testing efforts on a few major Linux distributions and supports running DB2 products for the rigorous validation criteria. Starting with DB2 UDB V8.2, the strategy for production environments is to focus on the most popular enterprise-class distributions from Red Hat and SuSE/NOVELL.

Limiting testing and support to these two enterprise-class distribution vendors results in many benefits for users of DB2 UDB. IBM has strong relationships with both of the Linux distribution vendors and works closely with them during development and test phases of new DB2 UDB releases and new releases of their Linux distributions. Focusing on a few major distributions results in better integration and robustness of the solution.

Customers further benefit from alignment of release and support cycles of these enterprise-class distributions with DB2 UDB. Major DB2 UDB releases are typically made available every 12 to 24 months and are often supported for two to five years. It is preferable that the Linux distribution that you deploy on also be maintained and supported for just as long. Most consumer or professional distributions are released every three to six months, and their support ends once the new version is out. Enterprise-class distributions, on the other hand, have much longer release and support cycles and offer enhanced support options, such as 24x7 support, that are generally not available for other types of distributions.

At the time of writing the following Linux distributions were recommended for use with DB2 Integrated Cluster Environment:

- Red Hat Enterprise Linux (RHEL) 3 (min. Update 2)
- Red Hat Enterprise Linux (RHEL) 2.1 (min. Update 4)
- SuSE Linux Enterprise Server (SLES) 9
- SuSE Linux Enterprise Server (SLES) 8 (min. SP3) and other distributions powered by United Linux 1.0

It is important to note that only the officially supported "stock" kernels shipped with validated distributions (or their service releases) are supported for running DB2 UDB. Custom patches and kernel recompiles often render your Linux environment as unsupported with your Linux distributor, and are not supported for running DB2 UDB. The DB2 Linux validation Web site has further details about support requirements and limitations. Information about newly validated distributions is also posted to the validation Web site on a frequent basis:

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

2.5.2 Linux distribution selection

The following information can assist with the selection of your Linux distribution:

- Has your company standardized on a Linux distribution that is also supported for DB2 UDB? Take cost and support considerations for your region into account.
- Are supported drivers for all components in the cluster (for example, network interconnect, storage interface, etc.) available for your preferred distribution?
- Is your preferred distribution certified for the servers in your cluster?
- Are you planning to use DB2 UDB V8.1 with JDK 1.3.1? Consider using non-NPTL enabled distribution such as SLES 8 or RHEL 3.
- Do you want to run Java workloads on an NPTL distribution such as RHEL 3 or SLES 9? Upgrade to DB2 UDB V8.2 with JDK 1.4.x is strongly recommended.
- Are you planning to deploy on an Intel EM64T-based server? 64-bit versions of SLES 8 and RHEL 2.1 are not supported for this platform (64-bit SLES 8 is supported on AMD64).

- Do you want 2.6 kernel features? Many of the features have been back-ported to 2.4 kernel-based enterprise distributions. Use of DIRECT I/O with DB2 UDB requires a 2.6 kernel. SLES 9 is the first 2.6 kernel-based enterprise distribution to be supported for DB2 UDB. IBM also intends to support DB2 UDB on future 2.6 kernel-based enterprise distributions from Red Hat and SuSE/Novell.

2.6 Sample configurations

In this section, we provide a few hardware configurations for the DB2 Integrated Cluster Environment.

2.6.1 A 32-bit configuration

Figure 2-18 shows a 32-bit configuration:

- Size: 4-node building block
- Platform: IA32 (Intel Xeon processors)
- Servers: IBM xSeries 335 with two CPUs each
- Memory: 4 GB per server
- Cluster Inter-connect: Gigabit Ethernet
- Storage: 1 DS4400 with 8 DS4000 Exp700s per 4 servers

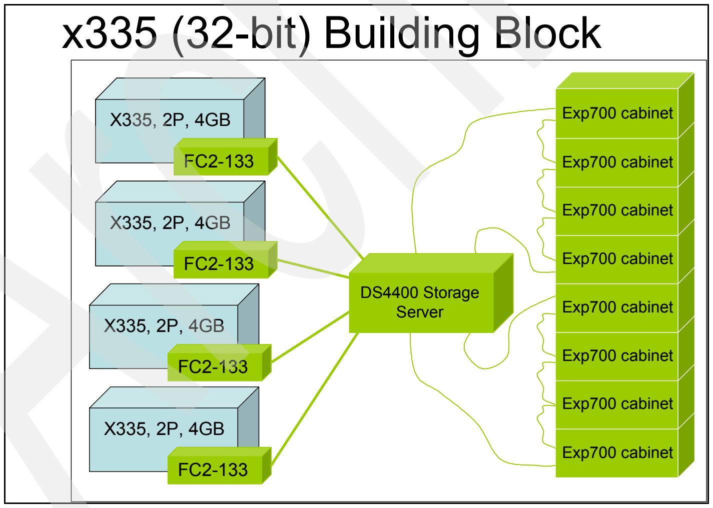


Figure 2-18 32-bit configuration

2.6.2 A Scaled-out 32-bit configuration

The configuration in Figure 2-19 is 16 nodes connected together using 4-node building blocks from Figure 2-18 on page 59.

- Size: 16-nodes using 4-node building blocks
- Platform: IA32 (Intel Xeon processors)
- Servers: IBM xSeries 335 with two CPUs each
- Memory: 4 GB per server
- Cluster Inter-connect: Gigabit Ethernet
- Storage: 1 DS4400 with 8 DS4000 Exp700s per 4 servers

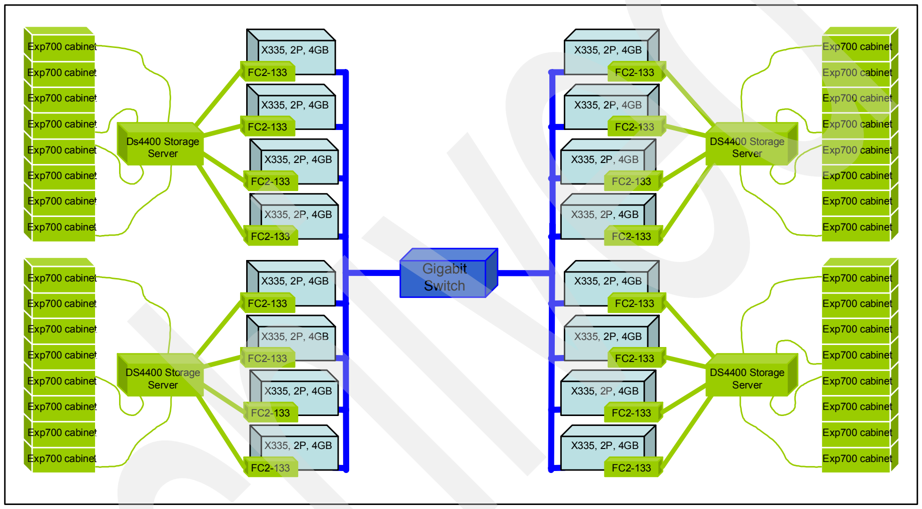


Figure 2-19 Scaled-out 32-bit configuration

2.6.3 A 64-bit benchmark configuration

The following configuration, shown in Figure 2-20 on page 61, was used for record setting TPC-H benchmarks:

- Size: 8-node cluster
- Platform: AMD64 (64-bit AMD Opteron processors)
- Servers: IBM eServer 325 with two CPUs each
- Memory: 6 GB per server
- Cluster Inter-connect: Gigabit Ethernet
- Storage: Direct Fibre Attached - 3 x EXP700 per server
- Linux Distribution: SUSE Linux Enterprise Server 8

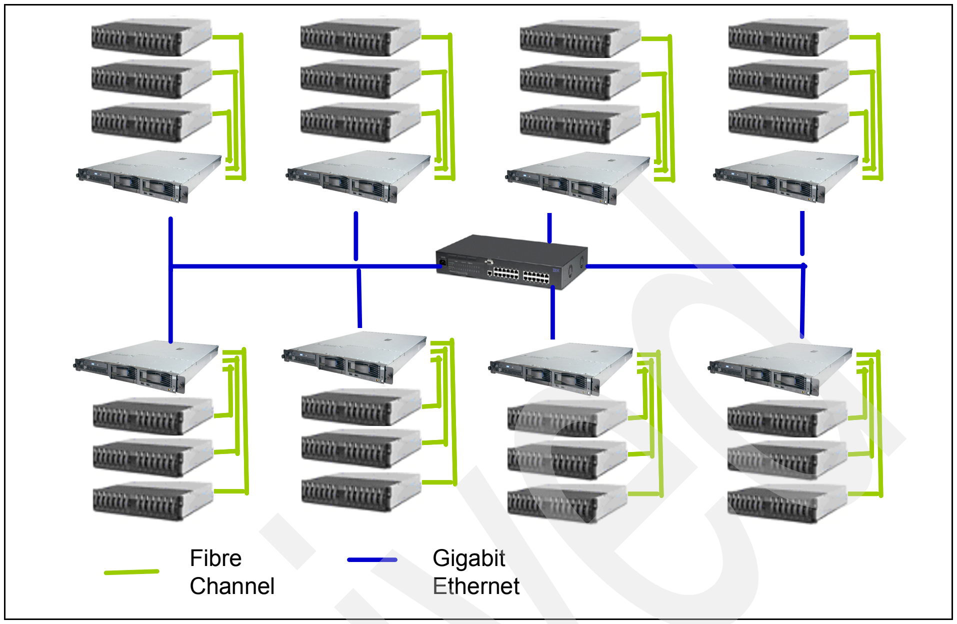


Figure 2-20 64-bit benchmark configuration

2.6.4 A 4-way building block

Four-way building blocks from Figure 2-21 on page 62.

- Size: Two-node building blocks
- Platform: IA32 (Intel Xeon processors)
- Servers: IBM xSeries 365 with four CPUs each
- Memory: 16 GB per server
- Cluster Inter-connect: InfiniBand
- Storage: One DS4500 with 8 DS4000 Exp700s per two servers

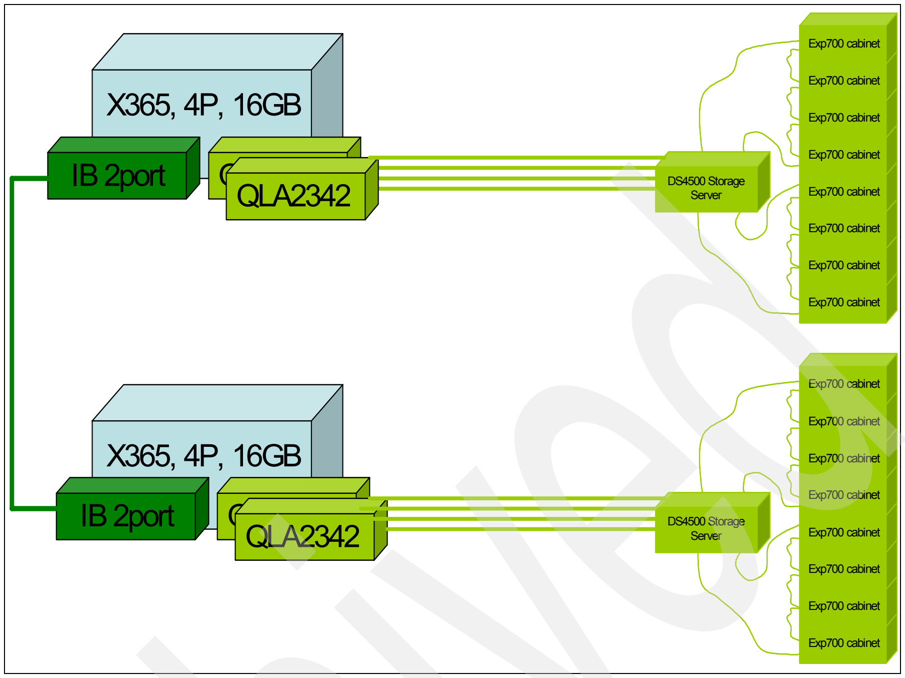


Figure 2-21 Four-way building block

2.7 Summary

In this chapter we discussed cluster selection for DB2 Integrated Cluster Environment including platform, server, storage and network options. Several choices and combinations are available, and selecting the optimal cluster for your DB2 Integrated Cluster Environment solution involves careful consideration of many factors. Once your initial cluster selection has been made, it is useful to validate the cluster for your environment. This validation can take the form of a mini-benchmark or a proof of concept to ensure that your selection will indeed be right for your business needs and technical criteria.

Planning and design

Say that you show up one morning and you have an urgent message from the mail room. You go downstairs and the mail room is packed with all of the new hardware you just selected from Chapter 2, “Selecting your cluster” on page 33. It feels like Christmas. You plug it all together and you are ready to go. “Uhhh... Anybody know how to build a DB2 partitioned database on a Linux Cluster?”

Building a DB2 partitioned database can seem like a very daunting task. You have probably heard that if you do it wrong your new hardware will implode, creating a black hole that will swallow your whole IT budget. In this chapter, we go over simple planning and design considerations to assist you with building your first DB2 partitioned database that is both flexible and scalable. This chapter covers:

- Designing a partitioned database
- Planning rules of thumb
- Naming conventions

This chapter contains the following sections:

- Steps for building a large partitioned database
- Disk considerations
- Database design considerations

3.1 Steps for building a large partitioned database

In this section we provide the necessary steps to build a large partitioned database with the main tasks, and the related sections, as shown in Table 3-1.

Table 3-1 Steps for building a large partitioned database

Steps	Main tasks involved	Related sections
1. Understand your environment. This first step is to understand your whole environment and have basic information to start planning your database building.	► Information of machine configuration. ► Information of external storage type. ► Define type of workload. ► Information of raw data size.	3.2, “Disk considerations” on page 65
2. Determine disks layout and placement. This step is related to the disk definition layout based on the placement plan. You will actually allocate the data, log, DIAGPATH and temp space when you create your database and table spaces.	► Have enough space for the database. ► Define RAID level. ► Plan data, log, DIAGPATH, temp space placement. ► Consider space for working and staging area.	3.2, “Disk considerations” on page 65 3.3.10, “Configure DB2 UDB” on page 104
3. Define tables sizes. The tables size information will be needed to define the number of database partitions, the next step.	► Define tables sizes. ► Consider MQT and MDC sizing.	3.3.7, “Size the tables” on page 103 3.3.8, “Size for MDC utilization” on page 104 3.3.9, “Size for MQT utilization” on page 104
4. Define database partitions.	► Define number of database partitions. ► Define database partitions group.	3.3.2, “Define the number of database partitions” on page 72 3.3.3, “Define database partition groups” on page 78

Steps	Main tasks involved	Related sections
5. Set configuration parameters. Set the parameters that are applicable to your environment.	Set network and DB2 UDB parameters.	3.3.11, “Recommended parameters for performance” on page 110
6. Design table spaces and buffer pools. The focus of this step is to define all table spaces and mainly the container’s layout. Buffer pools have to be defined since they are directly related to the table spaces.	Define container’s layout. Decide SMS or DMS. Define table space definition parameters. Define catalog table space placement. Define tempspace table space. Define bufferpools.	3.3.4, “Design the table spaces” on page 81
7. Choose the right partitioning key.	Choose the partitioning key.	3.3.6, “Choose the partitioning key” on page 97

3.2 Disk considerations

Deciding how to manage disk storage is one of the most important decisions a database administrator has to make.

How the data is physically laid out across the disks affects performance, as we explain in the following sections:

- Summary of the most popular RAID levels
- Data placement
- Data availability and performance

3.2.1 Summary of the most popular RAID levels

RAID can be implemented by both the storage system (ESS, DS4000 Series, EMC, etc.) and the Linux Operating System. The storage system provides faster RAID performance than software RAID. Each of the RAID levels supported by disk arrays uses a different method of writing data and, hence, provides different benefits, as discussed below.

RAID 0

Also known as striping, RAID 0 does not bring any additional data redundancy to a disk subsystem. The primary goal of RAID 0 is to improve I/O performance. When data is written, using a striped logical volume, the data is divided into small pieces known as chunks or stripe units. These units are then written to the disks in parallel, in a round-robin fashion. During a read operation, the same units are read back in parallel, then written directly to the caller's memory area where they are re-assembled into the original data.

The main problem with RAID 0 is obviously that any disk failure in the array will render the data on the remaining disks unusable. For that reason, only consider RAID 0 if you require good performance, but are not concerned about availability.

> **Tip:** DB2 UDB has its own concept of striping, using multiple containers for each table space. So consider RAID 1 or RAID 5 to safeguard your data.

RAID 1

Also known as mirroring, RAID 1 is simply keeping a copy of the data in a different location to protect against data loss in the event of a disk failure. So, from the availability perspective, the copies should be kept on a separate physical volume, which ideally should be attached to a separate I/O adapter. To take this one stage further, another possibility would be to have the second disk in a separate disk cabinet to protect against events such as a power loss to one of the disk cabinets.

In the event of a disk failure, read and write operations will be directed to the mirrored copy of the data. In performance terms, mirroring has some advantages over striping. If a read request is received and the primary data copy is busy, then the Linux LVM (Logical Volume Manager) will read the data from the mirrored copy.

Although mirroring can improve the read performance, the penalty is write performance.

> **Tip:** For SSA disks, you should use mirroring for maximum performance (for data, logs, temporary space) and also use the inherent striping of DB2 UDB, since it has its own concept of striping when using multiple containers for each table space.

RAID 5

In a RAID 5 array, data and parity are spread across all disks. For example, in a 5+P RAID 5 array, six disks are used for both parity and data. In this example, five-sixth of the available space is used for data and one-sixth is used for parity.

Because of the parity data used by RAID 5, each write to the array will result in four I/O operations; this is known as the RAID 5 write penalty:

- Read old data.
- Read old parity.
- Write new data.
- Write new parity.

Fast Write Cache (FWC), which exists on some RAID adapters, can reduce the effects of this write penalty.

RAID 5 is commonly chosen because it provides an extremely good price/performance ratio combined with good availability.

Using the Enterprise Storage Server (ESS), which has a large cache, can significantly decrease the effects of the write penalty.

Tip: For ESS, use the built-in RAID5, which has a large cache, and can significantly decrease the effects of the write penalty, but make sure you spread the logical disks evenly across all of ESS internal devices.

RAID 10

Sometimes called RAID 0+1, this RAID level provides better data availability at the cost of extra disks. Consider a striped logical volume, where failure of one disk renders the entire logical volume unusable. RAID 10 provides the ability to mirror the striped logical volumes to prevent this. When data is written to the disks, the first data stripe is data and the subsequent stripe copies (maximum of three copies, including first copy) are the mirrors, and are written to different physical volumes. RAID 10 is useful for improving DB2 log performance.

Tip: RAID 10 must be implemented with the hardware. Linux does not provide RAID 10.

3.2.2 Data placement

To choose a physical disk layout, some factors should be considered:

- Workload considerations
- Available disk space and number of disks

Workload considerations

The types of workloads of your database have to be considered when you define the physical database layout. For example, if the database is part of a Web application, then typically its behavior will be similar to Online Transaction

Processing (OLTP), as it can be in an Operational Data Store. There will be a great deal of concurrent activity, such as large numbers of sessions active, simple SQL statements to process, and single row updates. Also, there will be a requirement to perform these tasks in seconds.

If the database is part of a Decision Support System (DSS), there will be small numbers of large and complex query statements, fewer transactions updating records as compared to transactions reading records, and many sequential I/O operations.

This information gives the database administrator some idea of which can be used to aid in the physical design of a database that will be well-suited to that type of workload.

Available disk space and number of disks

The basic rule of thumb is that three to four times the amount of raw data is needed to build a database, including indexes and temp space without factoring in any data protection such as mirroring or RAID 5. With mirroring, it will be closer to eight times the raw data. For details about RAID levels, see 3.2.1, "Summary of the most popular RAID levels" on page 65.

Consider space for the staging and working area.

Take into account the number of disks required to achieve a balanced system: Too many small disks can cause system bus contention. Such a system may also produce increased system administration overhead; but too few large disks can cause I/O bottlenecks.

In an average system, a good balance would be a a minimum of six to 10 disks per CPU. In summary, the basic goal is to design a system so that no one disk or set of disks becomes a performance bottleneck.

3.2.3 Log placement

Logging performance such as log file, number of logs, and the type of storage for the logs is discussed in "Database logs" on page 104.

Before any log files even exist, the database administrator needs to decide on two main factors:

- On which disks are the logs to be placed
- Availability

On which disks are the logs to be placed

Due to the importance of the log files, it is recommended that the log files should always reside on their own physical disk(s), separate from the rest of the database objects. The disks ideally should be dedicated to DB2 logging to avoid the possibility of any other processes accessing or writing to these disks.

Availability

Irrespective of the type of logging you choose, whether it be circular logging or archival logging, the availability of the physical disks is crucial to the database. For this reason, it is strongly recommended that you protect the log against single disk failures by using RAID 1 (mirroring the log devices), recommended for SSA disks; or by storing log files on a RAID 5 array, recommended for ESS disks, since it has a large cache, and the write penalty effects can be significantly decreased.

3.2.4 Data availability and performance

For an e-BI environment, with DB2 UDB databases commonly being used to support Web applications, 24x7 availability is an important consideration; for example, for any distributor company that sells the products using Web applications, and needs to operate 24x7 to manage its products inventories and ship the products.

It is worth remembering that in most cases increasing availability is a trade-off against a potential reduction in performance. The most common solutions implemented today are RAID 5 and data mirroring.

3.2.5 General storage performance recommendations

The following are general performance considerations to bear in mind when creating logical volumes on Linux:

- Try to spread heavily accessed logical volumes across as many physical volumes as possible to enable parallel access.
- Use the inherent striping of DB2 UDB, placing containers for a table space on separate logical disks, which reside on separate RAID arrays. This will eliminate the need for using underlying operating system or logical volume manager striping.
- Try to create your logical volumes with sufficient size for your data. This will avoid fragmentation if the logical volumes have to be increased later.

3.3 Database design considerations

Deciding on the database design is one of the most important decisions, since it directly affects the database performance.

In this section, we provide considerations and recommendations about database design. The topics included are:

- Understand data partitioning.
- Define the number of database partitions.
- Define database partition groups.
- Design the table spaces.
- Understand partitioning map.
- Choose the partitioning key.
- Size the tables.
- Size for MDC utilization.
- Size for MQT utilization.
- Configure DB2 UDB.
- Recommended parameters for performance.
- Naming conventions.

3.3.1 Understand data partitioning

DB2 UDB ESE can be configured with multiple database partitions, and utilizes a *shared-nothing* architecture (Figure 3-1). This means that no data or memory is shared between database partitions, which gives good scalability.

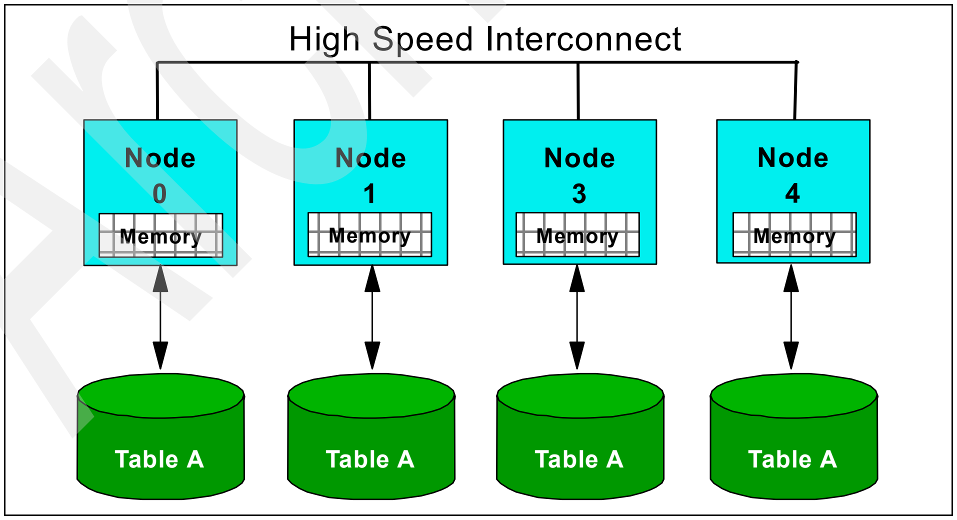


Figure 3-1 DB2 shared-nothing architecture

A *partitioned database* environment allows a database to be transparent to most users, despite being physically divided across more than one partition. Work can be divided among the database managers; each database manager in each partition works against its own part of the database.

In a partitioned database environment there are two terms: Physical partition and database partition (called a “logical node” before Version 8). The *physical partition* is the server where one or more database partitions of the database reside. Although the server has shared resources, database partitions do not share the resources.

Because data is divided across database partitions, you can use the power of multiple processors on multiple physical partitions 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. The fact that a database is split into database partitions is transparent to users issuing SQL statements.

The data is physically stored across more than one database partition, and yet can be accessed as though it were located in the same place. Applications and users accessing data in a partitioned database do not need to be aware of the physical location of the data.

User interaction occurs through one database partition, known as the coordinator partition for that user/application. In this way, coordinator partitions (and the associated workload) can be spread across the partitioned database.

When processing large volumes of data, multiple database partitions can be configured within a single multi-processor physical server, or if a single server is not sufficient for workload needs, across multiple physical servers. Figure 3-2 on page 72 illustrates the partitioned database environment.

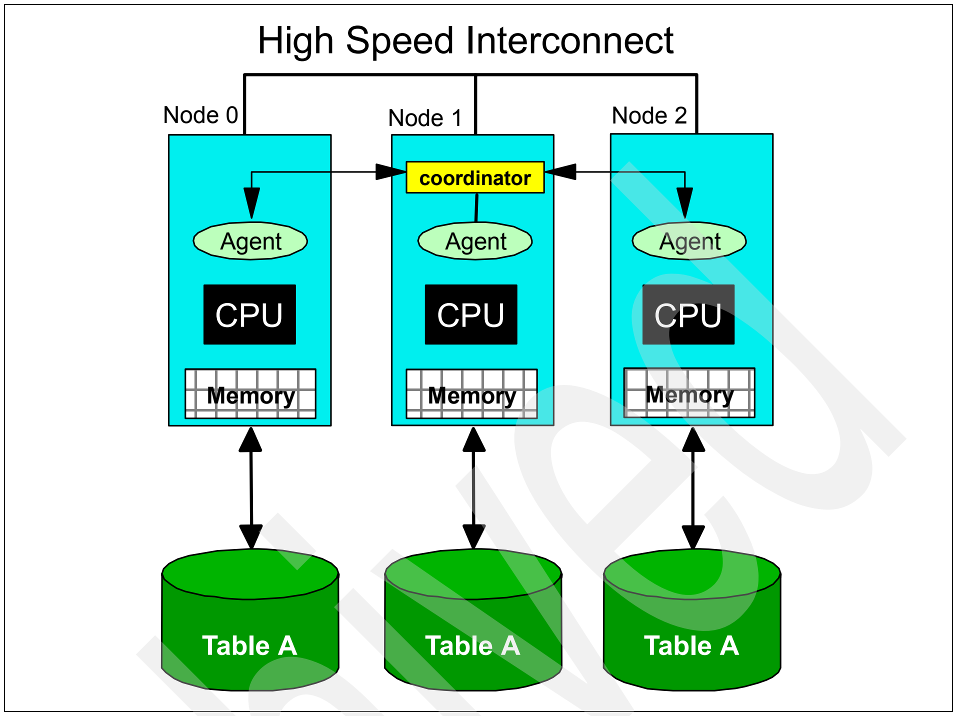


Figure 3-2 Physical partitions and database partitions

3.3.2 Define the number of database partitions

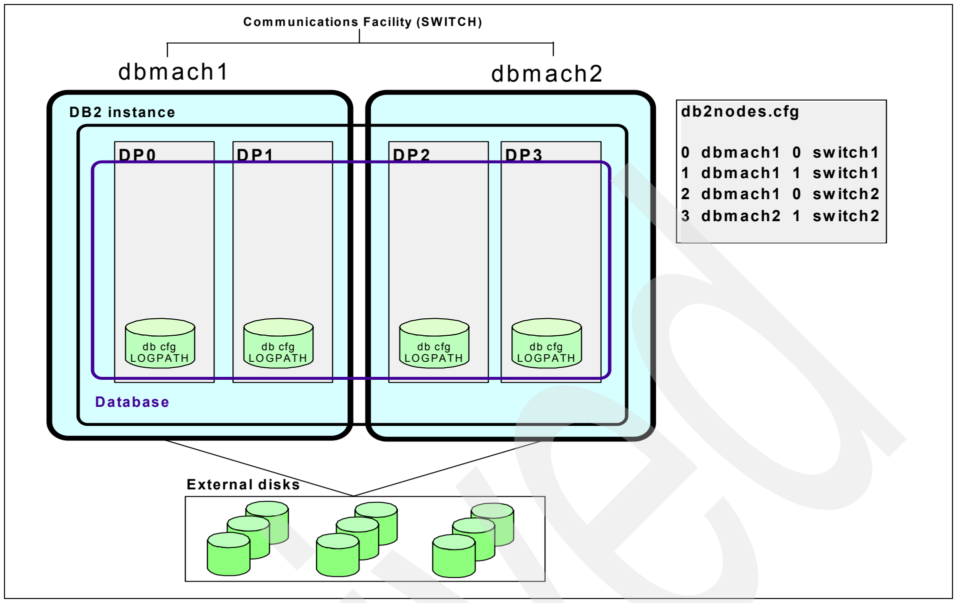
The number of database partitions is defined at the instance level in the file db2nodes.cfg, which is located in the /$INSTHOME/sqllib directory in Linux.

Important: Each *database partition* has its own:

- Logs
- Database configuration file (DB CFG)
- Buffer pools

A *partition group* contains one or more database partitions.

In Figure 3-3 on page 73, there are two physical machines with two database partitions on each machine. The database partitions are defined in the db2nodes.cfg file at the instance level.



```
db2nodes.cfg

0 dbmach1 0 switch1
1 dbmach1 1 switch1
2 dbmach1 0 switch2
3 dbmach2 1 switch2
```

Figure 3-3 Number of logical partitions example

Think at the DB2 partition level

The Balanced Partition Unit (BPU) is the “building block” from which you build your whole database. It is designed to have balanced components assembled to meet the requirements for performance, failover and support. Multiple identical building bocks are combined to form the entire database solution. Scalability is achieved by partitioning the data across multiple DB2 data partitions. Those data partitions reside on a “stack” of technology, which includes operating system, processors, memory, I/O backplane, I/O paths, disk controllers and individual disk spindles.

It is important to note that in a DB2 shared-nothing environment the best performance will be attained when the shared-nothing concept is carried out from the data perspective from “top to bottom.” This means from the partition level all the way down to the disk (spindles or LUNS). In other words, each DB2 partition will “own” its own stack of resources all the way down to the individual disks. *Think at the partition level.* A partition should own one vertical slice of this hierarchy. Figure 3-4 on page 74 shows a graphic depiction of the DB2 “building block” and its resources.

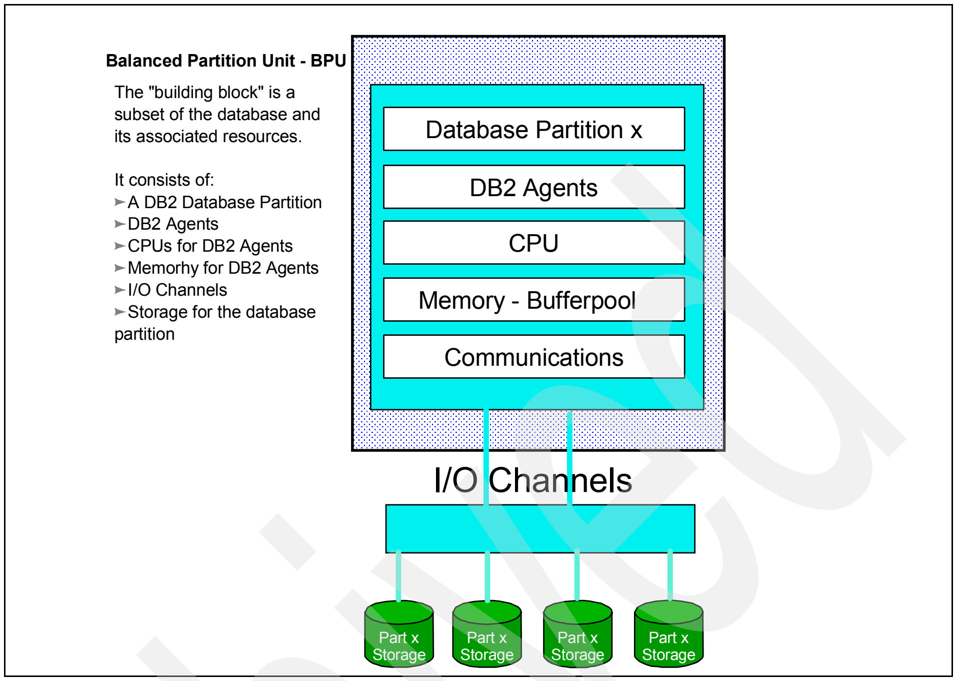
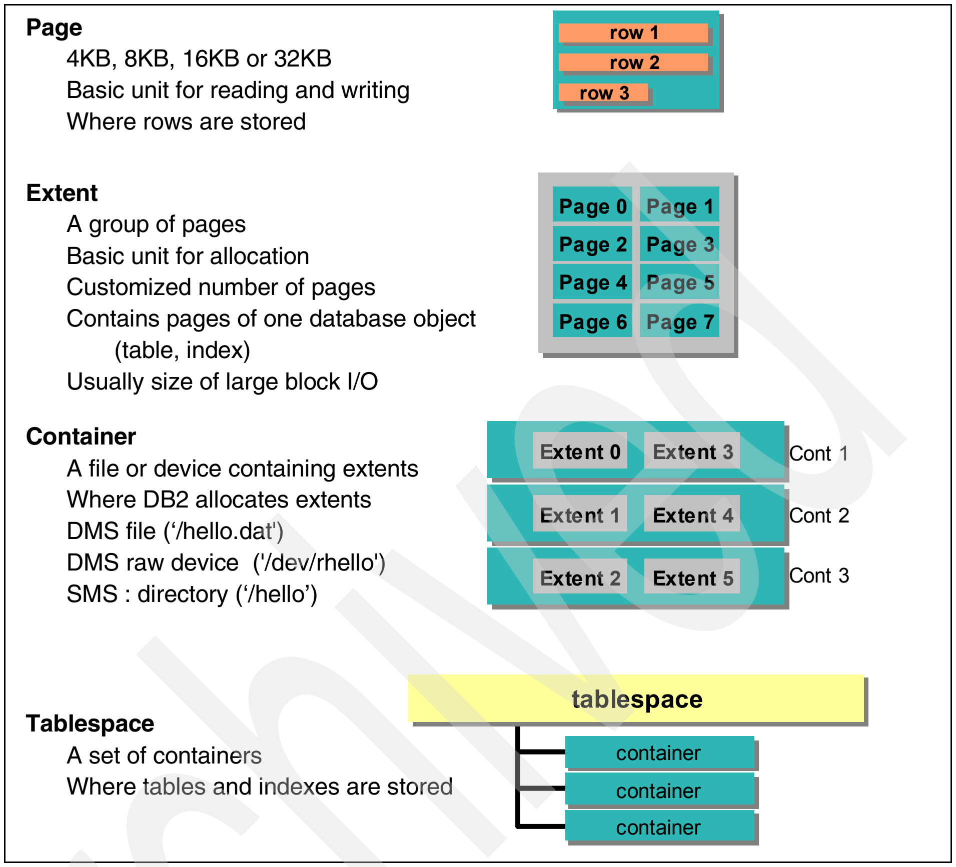


Figure 3-4 Balanced Partition Unit as the building block to a partitioned database

Keep in mind DB2's I/O hierarchy (shown in Figure 3-5 on page 75) and maintain partition-level thinking all the way through the hierarchy.



Page
- 4KB, 8KB, 16KB or 32KB
- Basic unit for reading and writing
- Where rows are stored

Extent
- A group of pages
- Basic unit for allocation
- Customized number of pages
- Contains pages of one database object (table, index)
- Usually size of large block I/O

Container
- A file or device containing extents
- Where DB2 allocates extents
- DMS file (‘/hello.dat')
- DMS raw device ('/dev/rhello')
- SMS : directory (‘/hello’)

Tablespace
- A set of containers
- Where tables and indexes are stored

Figure 3-5 DB2 I/O hierarchy

Building block rules of thumb

Keep in mind that “rules of thumb” provide only a first cut for sizing. They will not provide optimal performance, but will serve as a starting point to help avoid gross errors in undersizing or mismatched performance capabilities of system components.

Rules of thumb must *always* be “reality checked” against your business requirements, workload characteristics, and budget. For example, small databases may not require being spread across six disk spindles. In general:

- Per CPU
  - 50–250 GB raw data, depending on how much data is “active”
  - 3–4 GB memory
  - Six disks at 15000 RPM (minimum) - more is better

    - 900 I/Os per second (assuming 150 I/Os per second/disk max)
    - 50–125 MB per second I/O capacity
- One 2 Gb Fibre Channel Adapter per 2 CPUs
- 1 GB Ethernet Adapter per 4 CPUs
- 1–2 CPUs per partition

Disk size

When determining the amount of data and disks to size, a good rule of thumb is to plan for four times the amount of raw data the system will hold. This is a conservative estimate and works well in most cases. Larger systems (multi-terabyte) may not require such a high ratio.

Memory size

In general, start with 2 GB of memory per CPU and expand towards 4 GB for some workloads. A larger number of users and high concurrency will push towards more memory per CPU. Also, very complex queries will push you towards more memory.

I/O

On the I/O side, there are three factors. The number of disk spindles per CPU, the number of I/Os per second, and the I/O throughput (MB per second) needed. With regard to spindles, six spindles per CPU is a good starting point. You will need to make adjustments based on the I/O demands of your system and the performance characteristics of your disk drives.

Disk subsystem

The most often ignored piece of the technology stack is the disk subsystem. In many enterprises storage, it is managed by a systems group and “handed off” to the database group after the file systems have been built. If you do not manage your own storage it will be important that you work with your systems group to understand the performance characteristics of the underlying storage subsystem.

With today’s larger disks, it is tempting to cut costs by purchasing fewer larger disk drives. Many of these disks are offered in a number of capacities with the same performance characteristics. Attention must be paid to the I/O rate and throughput capabilities of the disks. For example, assume we need to build a system with 1 TB of storage. If we use 14 72GB drives, we will have only half of the MB/sec and IO/sec performance off of the disk than if we use 28 36GB drives.

Strive for symmetry

Strive for symmetry in your building blocks. Your building blocks should be physically and logically identical as much as possible. Doing so will provide the following advantages:

- Easier to configure
- Easier to manage
- Easier to scale
- Easier to operate
- Easier to troubleshoot
- Easier to upgrade
- Easier failover
- Easier to predict capacity costs
- Allows DB2 to hash partition evenly
- Minimizes data skew and eliminate hot spots

Considerations

Following are some considerations for database partition definition:

- When determining the number of database partitions, a number of considerations need to be taken into account:
  - The number of CPUs/memory available
  - The disk layout
  - The total amount of data residing under each partition
  - The largest table size under each partition
  - Future growth plans
  - The number of insert/update/delete operations
  - Amount of logging required per partition
  - Time required for batch/maintenance operations
- DB2 UDB ESE partitioned database can support up to 1000 partitions.
- Any database partition can be used as a coordinator partition to balance the workload.
- You are not restricted to having all tables divided across all database partitions in the database. DB2 UDB supports partial declustering, which means that you can divide tables and their table spaces across a subset of database partitions in the system. For example, you can create a database partition group containing a single database partition to store the dimension tables, which are always small.
- Balanced disk layout between partitions. It is important to have data from each database partition distributed over sufficient physical disks, potentially with multiple containers; at least four disks per table space.

- The total amount of data residing under each database partition should not be too high to improve the parallelism of partition-wide operations such as backups. Many factors come into play when deciding how much data there should be per database partition, such as the complexity of the queries, response time expectations, and other application characteristics. For example, a high percentage of non-queried data allows more data per database partition, while a low percentage suggests less data per database partition.

  Tip: Rules of thumb based on CPU vary, but may be in the region of 50–200 GB raw data per partition, with the real requirement to keep the individual table size about 100 GB/partition for utility performance and table scans.

- The largest table size under each partition should not be too high. There is an architectural limit per partition:
  - 64 Gb for 4 k page size
  - 128 Gb for 8 k page size
  - 256 Gb for 16 k page size
  - 512 Gb for 32 k page size

  Beyond this, the maximum table size per partition affects the performance of processes such as **`create index`** or **`runstats`**.
- The future growth plans affect the number of partitions since it is desirable not to have to repartition within a period of two years, for example. It is sometimes desirable, therefore, to create a partitioning strategy based on a two-year sizing, rather than the current position. It is much more straightforward to add processors/memory or even migrate to multiple physical machines than to repartition a large database.

3.3.3 Define database partition groups

A *database partition group* is a set of one or more database partitions that determine over which partitions each table space is distributed.

"Database partition groups" on page 79 shows the database partition groups example.

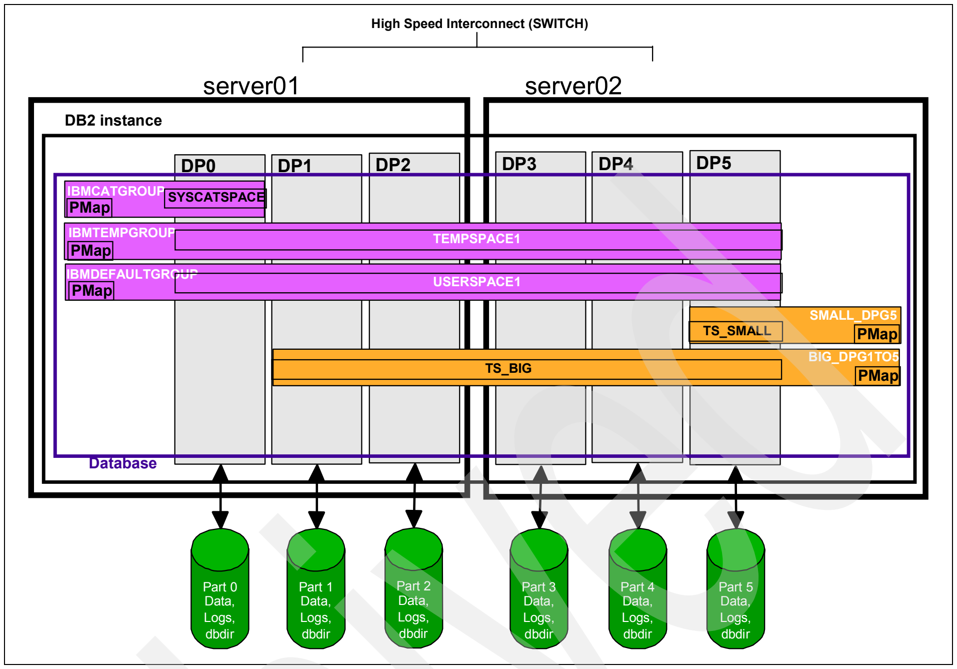


Figure 3-6 Database partition groups

Three default database partition groups are created when the database is created:

- IBMCATGROUP is the default database partition group for the table space containing the system catalogs. This database partition groups only contain the catalog partition.
- IBMTEMPGROUP is the default database partition group for system temporary table spaces, which contain all partitions defined in the db2nodes.cfg file.
- IBMDEFAULTGROUP is the default database partition group for the table spaces containing the user-defined tables that you may choose to put there. This partition group contains all data partitions defined in db2nodes.cfg. The data partitions in IBMDEFAULTGROUP can be changed.

You can create new database partition groups (shown in “Database partition groups” on page 79) using the CREATE DATABASE PARTITION GROUP command.

The commands described in Example 3-1 will create two database partition groups:

- The database partition group “BIG_DPG1TO5” on partitions 1, 2, 3, 4 and 5 for the table space TS_BIG, which contains the big tables
- And another database partitions group “SMALL_DPG5”, which contains only partition 5 for the table space TS_SMALL, which contains the small tables

Example 3-1 Create database partition group command

```
CREATE DATABASE PARTITION GROUP big_dpg1to5 ON DBPARTITIONNUMS (1 TO 5);

CREATE DATABASE PARTITION GROUP small_dpg5 ON DBPARTITIONNUM (5);
```

Design points

You have to consider the following design points:

- In a multipartition database partition group, you can only create a unique index if it is a superset of the partitioning key.
- Depending on the number of database partitions in the database, you may have one or more single-partition database partition groups, and one or more multipartition database partition groups present.
- Each database partition must be assigned a unique partition number. The same database partition may be found in one or more database partition groups.

Design recommendations

You should consider the following design recommendations:

- It is suggested that a simple design of single multipartition group across all data partitions be adopted for large tables (fact and the large dimension tables).
- For small tables, you should place them in a single-partition database partition groups, except when you want to take advantage of collocation with a larger table since the tables must be in same database partition group.

> **Tip:** A single partition group can be created for the remainder of the dimension tables. It may be desirable to replicate the majority of the dimension tables across all partitions using DB2 UDB replicated summary tables, which would effectively *pre-broadcast* the smaller tables once a day, for example, rather than having it take place for every query.

- You should avoid extending medium-sized tables across too many database partitions. For example, a 100-MB table may perform better on a 1- or

2-partition database partition group rather than on a 32-partition database partition group.

- You can use database partition groups to separate Operational Data Store OLTP-like tables from decision support (DSS) tables, to ensure that the performance of ODS transactions is not adversely affected.

3.3.4 Design the table spaces

How you set up your table spaces has a large impact on your database performance. In this section we provide some guidelines to define the table spaces placements getting the benefits from DB2 UDB and storage resources.

The topics included in this section are:

- Table space container's layout
- Table space SMS or DMS
- Table space definition parameters
- Tables per table space
- Catalog table space considerations
- Temporary table space considerations

Attention: Before creating the table spaces, check if there are any DB2 UDB parameters that apply to your environment (these must be set up first). See 3.3.11, "Recommended parameters for performance" on page 110.

Table space container layout

Before you create your table spaces, you need to think about how the container configuration will affect the overall performance of the table spaces.

In this section we provide some guidelines about the table space container's layout definition.

The topics included are:

- Prerequisites
- Considerations
- Recommendations

Prerequisites

Ensure that you read 3.2, "Disk considerations" on page 65.

Considerations

The following general principles can be used to your advantage:

- For a partitioned database environment, establish a policy that allows partitions and containers within partitions to be spread evenly across storage resources.
- DB2 query parallelism allows workload to be balanced across CPUs, and if DB2 UDB ESE with the partitioned database option is installed, across data partitions.
- DB2 UDB I/O parallelism allows workload to be balanced across containers.
- You may intermix data, indexes, and temporary spaces on RAID arrays. Your I/O activity will be more evenly spread and avoids the skew, which you would see if the components were isolated. This has the added advantage of making available more disk arms for each of these objects.

Recommendations

When determining the table space storage:

- If you have lots of disks, define one table space container per disk; otherwise, spread everything across all disks.
- At least four disks per table space (the more the better).

For example, for the big single table space “TS” (defined on the database partition group with eight database partitions) we can define one container per disk for each database partition, as shown in Figure 3-7 on page 83, following the *spread and balance* approach.

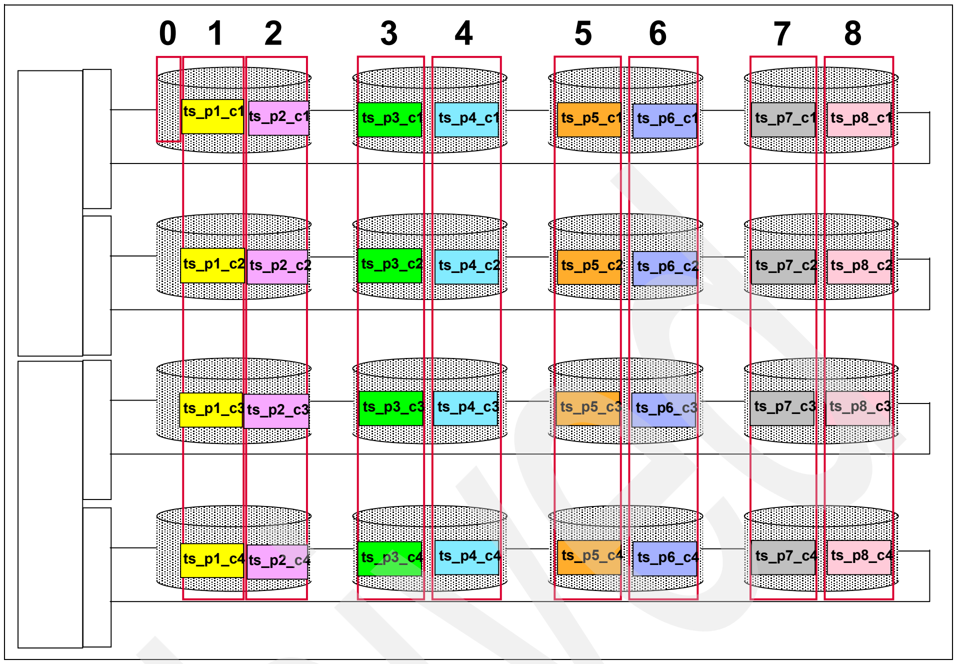


Figure 3-7 Table space containers layout example

Table space SMS or DMS

Once the table space and container design is complete, the next step is to decide between System Managed Storage (SMS) and Database Managed Storage (DMS).

Attention: Separate decisions need to be made for the base table data and temporary space.

The topics included are:

- SMS table spaces
- DMS raw table spaces
- DMS file table spaces
- Considerations and recommendations

SMS table spaces

SMS stands for system-managed space, and you can only define a directory as a container type. DB2 UDB does nothing to manage the space assigned to the table space. The operating system owns the directory; creates, grows, shrinks, and removes the files within it; and controls all of the I/O.

On the negative side, when SMS table spaces grow dynamically, they compete for system resources at a time when those resources are in high demand. Also, you cannot add containers to an SMS database dynamically like you can with DMS table spaces. The only way to add containers is through a backup and redirected restore.

There is also a performance penalty to be paid for using SMS table spaces. First, system I/O buffers are used in addition to the DB2 UDB buffers already allocated. Not only is this an additional layer in the I/O path, but DB2 UDB must compete dynamically with other applications for system I/O buffer resources. The penalty may be heaviest with a write-intensive workload. The Linux 2.6 kernel allows for Direct I/O (DIO) between disk and the buffer pools. DIO eliminates extra copy between application buffer and disk (bypass file system cache), thereby eliminating overhead. Specify the NO FILE SYSTEM CACHING parameter in the CREATE or ALTER TABLESPACE to enable DIO.

Another performance penalty is paid by the system journaling of I/O activities and subsequent flushing of that journal to disk. Journaled file systems have come a long way in the past few years, and that penalty is not as severe as it used to be, but it is still present. More importantly, this journaling is more or less duplicated by DB2 logs.

All in all, the conventional wisdom is that you will pay some performance penalty for choosing SMS table spaces, but many people are willing to pay that penalty for the convenience.

DMS raw table spaces

DMS stands for database-managed storage. Although the operating system knows that the storage devices exist, that storage is not made available to the system. It is given over to DB2 *raw* and unsullied by any operating system intervention.

In table spaces composed of raw containers, there is no file system overhead. DB2 UDB creates, grows, shrinks, and deletes tables; allocating and freeing space within the container. Table objects, such as Indexes and LOBs, can reside in separate DMS table spaces, and containers can be added to them or extended.

Furthermore, no system overhead is expended in preparing and launching an I/O in addition to that already done by DB2 UDB. The data remains grouped in extents; and it is buffered in memory reserved exclusively for DB2 UDB. The system I/O layer is bypassed.

DMS raw table spaces are the best performing for user data, especially for OLTP workloads and large DSS reads, such as applications that insert into append tables.

DB2 raw table spaces are limited to the total size of the containers assigned to them; unused space in those containers cannot be shared.

DMS file table spaces

DMS file table spaces are a compromise between SMS and DMS raw table spaces. In this case, file system files are created and then given to DB2 UDB to manage. DB2 UDB handles the files as if they were raw devices.

The drawbacks are that there is a little system overhead in maintaining the container files, the table spaces are limited to the total size of the files assigned, and unused space cannot be shared. The benefits are improved performance over SMS table spaces and the ability to perform conventional file system backups.

Considerations and recommendations

Some basic considerations between SMS and DMS table spaces are shown in Table 3-2.

Table 3-2 Basic considerations between SMS and DMS table spaces

Considerations	**SMS**	**DMS**
Managing containers (drop/add, reduce size)	No	Yes
Separating indexes and log from data	No	Yes
No tuning of operating system parameters	No	Yes
Space allocated as needed	Yes	No
Easy administration	Yes	No

We describe some performance considerations for SMS and DMS table spaces as follow:

- SMS
  - You might want to pay a little more attention to container layout for SMS table spaces, particularly if those table spaces are not spread across multiple containers on separate arrays.
  - Because file managers dynamically allocate disk storage, storage can become fragmented, and therefore it can become more difficult to maintain sequential access on disk.
  - Using SMS, be sure to allocate a sufficient number of containers on a single RAID array, particularly if you have few arrays, and if database load performance is especially important. We recommend four to eight SMS containers for every RAID array, depending on how many arrays will be used. (For example, if a single array is to be used, then we recommend 8

containers; however, if you are using more than three arrays, four containers per array may be sufficient.)

- DMS

  DMS raw table spaces generally provide the best performance for several reasons:

  – The layers of file management are removed from the I/O path.

  – Data generally maps fairly simply to underlying storage. Data that appears to DB2 UDB as sequential access also appears to ESS as sequential access, making best use of caching algorithms.

  – When using DMS raw devices, DB2 UDB will ensure that pages are placed contiguously on the disk drives; with SMS containers this is not the case, as the file system decides where to locate the pages (this can also apply to DMS file containers). Contiguous placement of pages is important, as pages make up DB2 extents. Contiguous pages will speed up operations like table scans. Note that DMS file table spaces require tuning of system parameters.

  Tip: Some tests indicated much better performance using DMS raw table spaces on DB2 UDB ESE V8.1.

Table space definition parameters

In this topic we describe some considerations about page size, extent size, and prefetch size settings that primarily affect the movement of data to and from the disk subsystem and how they work together. We also provide information about overhead and transfer rate parameters that are taken into account when making optimization decisions, and help to determine the relative cost of random versus sequential accesses:

- Container size
- Page size
- Extent size
- Prefetch size
- Overhead
- Transfer rate

Attention: Pay attention to defining the page size and extent size, since you cannot change them after the table space creation, unless you recreate the table space.

Container size

One of the factors that will affect the parallel I/O performance is the size of the containers used. If you use multiple containers of different sizes in a table space, then the effectiveness of any parallel prefetches performed against that table space will be reduced. For example:

- You created a new table space containing one small container and one large container. Eventually, the small container will become full and any additional data will be written to the large container. The two containers are now unbalanced. For example, in a BI environment, when you have queries that have large amount of rows, having unbalanced containers reduces the performance of parallel prefetching to nothing, as a SELECT statement may result in some data being required from the small container, but a large amount may be required from the large container.
- You have a table space that contains a number of large containers; these start to fill up, so you add a small container. The table space will then be rebalanced and the small container will contain far less data than will now reside in the larger containers, and once again the database manager will not be able to optimize parallel prefetching.

Tip: Keep the containers with the same size for best performance.

Note: Introduced in DB2 UDB V8.1 is the ability to drop a container from a table space, reduce the size of existing containers, and add new containers to a table space such that a rebalance does not occur using DROP, RESIZE and BEGIN NEW STRIPE SET clauses of the ALTER TABLESPACE command. For more detailed information, see *SQL Reference Volume 2 Version 8,* SC09-4845-01.

Page size

The overall performance of your table space will be affected by the page size you choose. Page sizes are defined for each table space during the creation. There are four supported page sizes: 4 K, 8 K,16 K, and 32 K. Some factors that affect the choice of page size include:

- The maximum number of records per page is 255. To avoid wasting space on a page, do not make page size greater than 255 times the row size plus the page overhead.
- For a dimensional model, it is recommended that a small page size should be used for the dimension tables, because dimension tables normally are small, and using a small page size, it will not waste disk space and space in the buffer pools.
- Because of the fact that tables that are normally large tables and access rows are sequential, larger page sizes will provide better performance. Remember the limit of 255 rows per page to avoid wasting disk and buffer pool space.

For example, if your row length is 39 bytes, in one page you can have at the maximum (39 bytes*255 rows)=10 KB. So, if you allocate the page size of 32 KB, you will waste 22 KB per page.

- The maximum size of a table space is proportional to the page size of its table space, as shown in Table 3-3. These limits apply at the table space level per database partition. So, if you have a multi-partitioned table space, the maximum size limit would be the correspondent size limit described in Table 3-3 times the number of database partitions where the table space was defined.

Table 3-3 Page size relative to table space size

Page size	Max data/index size per DP	Max row length	Max columns
4 KB	64 Gb	4,005 bytes	500
8 KB	128 Gb	8,101 bytes	1012
16 KB	256 Gb	16,293 bytes	1012
32 KB	512 Gb	32,677 bytes	1012

Tip: It is suggested that all table spaces be standardized to one page size because there are significant advantages, such as:

- Reduce wasting memory by requiring multiple bufferpools for each size.
- Reduce wasting temporary space by either having multiple temporary table spaces or forcing smaller page size table spaces to use the temporary space of the largest page size table space.
- Reduce the factor of determining if multiple page sizes are relevant when doing problem determination.

Important: DB2 catalog table space (SYSCATSPACE) will always use 4-Kb page size, and the 4-Kb buffer pool can be of a small size to satisfy this.

Extent size

The extent size you choose for a table space determines the number of table data pages written to one container before DB2 UDB writes to the next container; the type of table space used is a major factor.

When using DMS table spaces, we allocate space one extent at a time; when one extent is full, a whole new extent is allocated. Data is striped across the container(s) by extent, as in Figure 3-8 on page 89. Note that extents are mapped in a round-robin fashion, and extents for different objects (table 1 and table 2 in Figure 3-8) need not be contiguous.

This differs from SMS table spaces, which allocate space one page at a time.

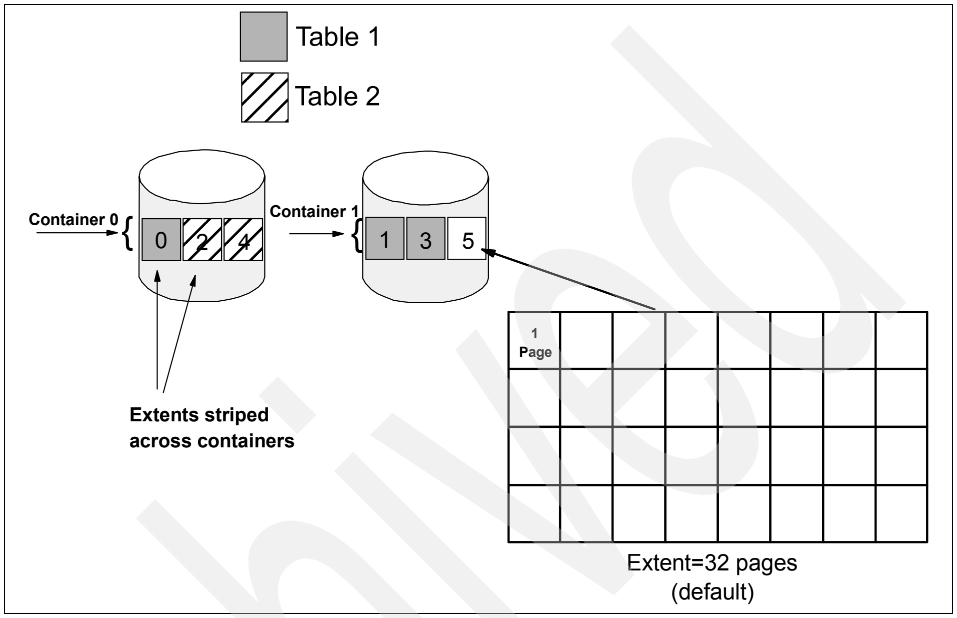


Figure 3-8 DMS table space structure

The optimum extent size value will be related to the actual disk layout used.

Use larger values for fact tables that are mostly scanned or have a very high growth rate. If your DMS table space does contain large tables, and you do not increase the value for EXTENTSIZE parameter, but instead use a small extent size, then you will incur an overhead when the database has to continually allocate these small extents (either when a table is expanding or new tables are being created).

When using striped containers (like RAID), extent size should be set to a multiple of the stripe size (such as RAID strip size).

Tip: In ESS, for example, a good starting point would be to set extent size equal to one complete stripe of disks in the LUN (for example, for a LUN striped across a 6+P array, set it to 6*32 K = 196 K.

Prefetch size

Setting the prefetch size correctly will improve the database manager's ability to read pages in advance and reduce execution times. You can change the prefetch size value using the ALTER TABLESPACE command.

If the database's main workload requires good sequential I/O performance, such as a DSS workload, then the PREFETCHSIZE becomes even more important.

Follow some considerations and recommendations:

- For most environments, the recommendation would be to set the PREFETCHSIZE equal to (EXTENTSIZE * number of containers). For optimal parallel prefetching, the containers in your table space should reside on separate physical devices. This is important when using a DMS raw container, as there will be no prefetching or caching performed by the operating system.
- For PREFETCHSIZE with RAID devices, we recommend that the PREFETCHSIZE is the RAID stripe size multiplied by the number of parallel disk drives (do not include parity disk) and also a multiple of the EXTENTSIZE; for example, if you have the table space shown in Example 3-2.

Example 3-2 Table space example for PREFETCHSIZE with RAID

```
CREATE TABLESPACE TBS1
MANAGED BY DATABASE USING
(DEVICE '/dev/rtbs1lv' 2000) PAGESIZE 8K
EXTENTSIZE 8
PREFETCHSIZE 24
```

A relatively small RAID array configuration is used to illustrate the following points.

Using an array = 3 + P (3 parallel plus one parity) and a strip size = 64 k, we can say that the above CREATE TABLESPACE command will give us the following results:

- Using an EXTENTSIZE of eight pages and a PAGESIZE of 8 K, each extent will span one drive. The value used for EXTENTSIZE should always be equal to or a multiple of the RAID stripe size.
- Using a PREFETCHSIZE of 24 pages, each prefetch done will utilize the three parallel disks. The calculation is strip size * N / PAGESIZE, which equates, in this example, to 64 KB * 3 / 8 = 24. We recommend that the PREFETCHSIZE is the RAID stripe size multiplied by the number of parallel disk drives (do not include parity disk) and also a multiple of the EXTENTSIZE.

Tip: The optimum prefetch size would retrieve data from all physical disks in one I/O operation.

Tip: The prefetch size should be set such that as many arrays as desired can be working on behalf of the prefetch request. In non-ESS storage, the general recommendation is to calculate prefetch size to be equal to a multiple of the extent size times the number of containers in your table space. For ESS, you may work with a multiple of the extent size times the number of arrays underlying your table space.

Note: To help you to tune PREFETCHSIZE for your table spaces, use the database system monitor and other monitor tools. You might gather information about whether:

- There are I/O waits for your query, using monitoring tools available for your operating system
- Prefetching is occurring, by looking at the pool_async_data_reads (buffer pool asynchronous data reads) data element in database monitor output (snapshot output, for example)

If there are I/O waits and the query is prefetching data, you might increase the value of PREFETCHSIZE. If the prefetcher is not the cause of the I/O wait, increasing the PREFETCHSIZE value will not improve the performance of your query.

If you wish to increase the PREFETCHSIZE, then you also need to consider increasing the NUM_IOSERVERS database configuration parameter to at least PREFETCHSIZE/EXTENTSIZE.

The NUM_IOSERVERS database configuration parameter specifies the number of the I/O servers that perform prefetch I/O for the database. You should normally set the same number for the NUM_IOSERVERS parameter as the number of physical disks. If PREFETCHSIZE is increased above (the suggestions outlined so far for *additional* prefetching), then the NUM_IOSERVERS database configuration parameter should be set to at least PREFETCHSIZE/EXTENTSIZE.

Note: To help you to tune NUM_IOSERVERS, use the snapshot or event-monitoring information.

Check the data element prefetch_wait_time. It shows the time an application spent waiting for an I/O server (prefetcher) to finish loading pages into the buffer pool. If the prefetch wait time is considerable, you might increase the NUM_IOSERVERS.

Although larger prefetch values might enhance throughput of individual queries, mixed applications would generally operate best with moderate-sized prefetch and extent parameters. You will want to engage as many arrays as possible in your prefetch, to maximize throughput.

It is worthwhile to note that prefetch size is tunable. By this we mean that prefetch size can be altered after the table space has been defined and data loaded. This is not true for extent and page size, which are set at table space creation time, and cannot be altered without re-defining the table space and re-loading the data.

Overhead and transfer rates

DB2 UDB defines two parameters related to I/O operational performance: Overhead and transfer rate. They are defined during the table space creation, but they can be altered after using ALTER TABLESPACE command.

These parameters are taken into account when making optimization decisions, and help determine the relative cost of random versus sequential accesses.

- Overhead

  This provides an estimate (in milliseconds) of the time required by the container before any data is read into memory. This overhead activity includes the container's I/O controller overhead as well as the disk latency time, which includes the disk seek time.

  The following formula can be used to estimate overhead cost (I/O controller, latency, seek time). A detailed explanation of this formula can be obtained in the *DB2 UDB Administration Guide: Performance,* SC09-4821.

  ```
  OVERHEAD = average seek time in milliseconds + ( 0.5 * rotational
  latency )
  ```

  In this formula, 0.5 represents an average overhead of one half rotation, and rotational latency (milliseconds/full rotation) is given as:

  ```
  (1 /RPM)*60 *1000
  ```

  So, for a disk with an RPM = 10000, we can obtain:

  ```
  ( 1 / 10000 ) * 60 * 1000 = 6.0 milliseconds
  ```

So, assume an average seek time of 5.4 milliseconds for this disk type; this gives the following estimated *overhead*:

```
5.4 +(0.5 *6.0 )=8.4 milliseconds
```

- Transfer rate

  This provides an estimate (in milliseconds) of the time required to read one page of data into memory.

  When calculating the TRANSFERRATE you need to take into account the following considerations if your table space containers are made up of more than one single physical disk (for example, an array such as RAID):

  – If the spec_rate value is small, then multiply by the number of physical disks on which the container resides, assuming bottleneck at disk level.

  – If the number of disks is large, then I/O bottleneck is not likely to be due to disks, but more likely due to disk controller or I/O buses (system). If this is the case, then you will need to test the actual I/O rate. Refer to the *DB2 UDB Administration Guide: Performance,* SC09-4821, for further suggestions on how to do this.

We recommend that if your container spans multiple physical disks, then these disks should ideally have the same OVERHEAD and TRANSFERRATE characteristics.

If they are not the same, or you do not have the hardware documents that contain the information you require for the formulas given, then set the OVERHEAD value to the average of all the disks in the table space. As for TRANSFERRATE, for a DSS environment, set TRANSFERRATE to the sum of all the disks.

The values described in Table 3-4 are recommended for use with ESS.

Table 3-4 Suggested overhead and transfer rates for ESS

Overhead	12.5
Transfer rate (4 KB)	0.1
Transfer rate (8 KB)	0.3
Transfer rate (16 KB)	0.7
Transfer rate (32 KB)	1.4

Tables per table space

Here are some considerations for placing tables, which might help you decide which tables should go:

- For a dimensional model, it is generally best to place dimension tables in the same table space to conserve space, and for the fact tables to be in their own table space. Snapshot monitoring occurs at the table space level, for buffer pool analysis, so separating tables as much as possible is advantageous. The disadvantage of having too many small table spaces is that considerable disk space may be wasted from over allocation and table space overhead.
- Consider that the rules of thumb vary but may be in the region of 50–200Gb raw data per partition.
- Keeping important tables in their own table space will simplify administration of this table space and will speed recovery should the table space need to be restored (because you will only have one table to restore), for example, for the fact tables.
- Consider table space capacity limits: 64 GB (4-K pages), 128 GB (8-K pages), 512 GB (32-K pages), 2 TB for long table spaces. In a partitioned database, they are the limits per partition.
- Place tables that need to be monitored more in their own table spaces, for example, MQTs, since the LIST TABLESPACE SHOW DETAIL command gives the number of used pages. If the table space is a DMS table space, the command also reports the number of free pages and the high water mark information. For example, you can monitor the growth of a fact table of a star schema by putting it in its own table space.

Catalog table space considerations

In this topic we describe some considerations about the catalog table space:

- Catalog table spaces are restricted to using the 4-KB page size.
- An SMS table space is recommended for database catalogs, for the following reasons:
  - The database catalog consists of many tables of varying sizes. When using a DMS table space, a minimum of two extents are allocated for each table object. Depending on the extent size chosen, a significant amount of allocated and unused space may result. When using a DMS table space, a small extent size (two to four pages) should be chosen; otherwise, an SMS table space should be used.
  - There are large object (LOB) columns in the catalog tables. LOB data is not kept in the buffer pool with other data, but is read from disk each time it is needed. Reading LOBs from disk reduces performance. Since a file system usually has its own cache, using an SMS table space, or a DMS

table space built on file containers, makes avoidance of I/O possible if the LOB has previously been referenced.

Important: Ensure that the catalog partition has been set correctly when the database is first created, since it cannot be changed at a later date. The catalog partition is determined by being the partition connected to when the database is first created. For example:

```
export DB2NODE=0
db2 terminate
db2 create database <dbname>.....
```

Or:

```
db2_all "<<+0< db2 create database <dbname> ..."
```

Temporary table space considerations

Temporary table spaces are used during SQL operations for internal temporary tables, for sorts, to store intermediate results, table reorganizations, and other transient data. So, in a BI environment, which normally has heavy SQL operations and sorts, it is very important that the temporary table space be correctly defined.

Follow some recommendations and considerations for the temporary table space:

- SMS is almost always a better choice than DMS for temporary table spaces because:
  - There is more overhead in the creation of a temporary table when using DMS versus SMS.
  - Disk space is allocated on demand in SMS, whereas it must be pre-allocated in DMS. Pre-allocation can be difficult: Temporary table spaces hold transient data that can have a very large peak storage requirement, and a much smaller average storage requirement. With DMS, the peak storage requirement must be pre-allocated, whereas with SMS, the extra disk space can be used for other purposes during off-peak hours.
  - The database manager attempts to keep temporary table pages in memory, rather than writing them out to disk. As a result, the performance advantages of DMS are less significant.
- If you have table spaces utilizing different page sizes, then our suggestion would be to create one temporary table space with a page size equal to that of the majority of your data tables. When selecting temporary table spaces, DB2 UDB ignores page sizes that are too small to accommodate an operation

that requires temporary space. For this reason you will need to ensure that the page size of your temporary table space(s) can accommodate the row lengths and column used in your database; see Table 3-3 on page 88 for a list of page size row and column limits.

- The reason we suggest only having one temporary table space for a given page size is because when you have multiple temporary table spaces using the same page sizes, the DB2 optimizer will generally choose a temporary table space size by selecting the temporary table space with the largest buffer pool.

  For example, you had two 16-KB temporary table spaces defined called TEMP1 and TEMP2. Now assume TEMP1 has a large buffer pool and TEMP2 has only a small buffer pool; UDB will select TEMP1, as it has largest buffer pool. Once the selection is made, UDB will then alternate between *all* temporary table spaces, which have the chosen page size (16 KB).

  This means that when performing an operation such as a sort, we alternate between TEMP1 and TEMP2 because they both use a 16-KB page size. The disadvantage here is that we only use TEMP1's large buffer pool half the time, because we are alternating between the two table spaces. A better solution in this example would be to have a single 16-KB temporary table space with one buffer pool.

Note: If you reorganize table spaces in the temporary table space, then you will need a temporary table space that has the same page size as the table space being reorganized.

Note: Catalog table spaces are restricted to using the 4-KB page size. As such, the database manager always enforces the existence of a 4-KB system temporary table space to enable catalog table reorganizations.

3.3.5 Understand partitioning map

In a partitioned database environment, the database manager must have a way of knowing which table rows are stored on which database partition. The database manager must know where to find the data it needs, and uses a pmap called a partitioning map, to find the data.

A partitioning map is an internally generated array containing either 4,096 entries for multiple-partition database partition groups. The partition numbers of the database partition group are specified in a round-robin fashion.

For example, in Figure 3-9 on page 97 we provide a partitioning map example for a database partition group that contains the partitions 1, 2, 3, and 4. The 4096 hash buckets with the partition numbers represent the partitioning map (pmap), and the partition numbers are specified in a round-robin manner. The column EMPNO had previously been defined as the partitioning key. The value of EMPNO, 000120, undergoes a hashing algorithm and returns the value of ‘9’. The Partitioning Map is then referenced to see what partition matches the hashed value of 9; in this example its partition 2. So Employ Number ‘000120 will be stored in partition 2.

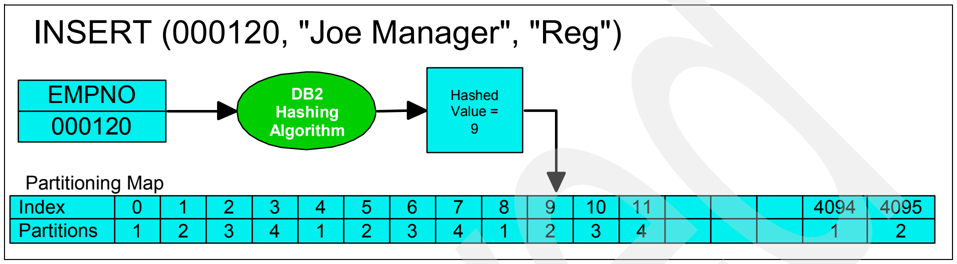

INSERT (000120, "Joe Manager", "Reg")
EMPNO
000120
DB2 Hashing Algorithm
Hashed Value = 9
Partitioning Map
Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | | | | 4094 | 4095
Partitions | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | | | | 1 | 2

Figure 3-9 Partitioning map

3.3.6 Choose the partitioning key

A partitioning key is a column (or group of columns) that the database manager uses to determine how data is distributed between the partitions; that is, the location (the database partition) where the data is stored. A partitioning key is defined on a table using the CREATE TABLE statement.

Hash partitioning is the method by which the placement of each row in the partitioned table is determined. The method works as follows:

1. The hashing algorithm is applied to the value of the partitioning key, and generates a partition number between zero and 4095.
2. The partitioning map is created when a database partition group is created. Each of the partition numbers is sequentially repeated in a round-robin fashion to fill the partitioning map.
3. The partition number is used as an index into the partitioning map. The number at that location in the partitioning map is the number of the database partition where the row is stored.

Why choose a good partitioning key

Choosing a good partitioning key is important for the following main reasons:

- An even distribution of data across the partitions
- Maximum use of collocated joins, as this avoids the database manager having to ship data between partitions

Note: Collocation between two joining tables means having to have the matching rows of the two tables always in the same database partition.

Why an inappropriate partitioning key should not be chosen

An inappropriate partitioning key can cause uneven data distribution. Columns with unevenly distributed data, and columns with a small number of distinct values, should not be chosen as a partitioning key. The number of distinct values must be great enough to ensure an even distribution of rows across all database partitions in the database partition group. Example 3-3 shows the ORDERS table creation command.

Example 3-3 ORDERS table creation command

```
CREATE TABLE ORDERS
( O_ORDERKEY integer not null,
  O_CUSTKEY integer not null,
  O_ORDERSTATUS char(1) not null,
  O_TOTALPRICE float not null,
  O_ORDERDATE date not null,
  O_ORDERPRIORITY char(15) not null,
  O_CLERK char(15) not null,
  O_SHIPPRIORITY integer not null,
  O_COMMENT varchar(79) not null)
```

If you choose the O_ORDERSTATUS as a partitioning key, you probably will have uneven data distribution, since this column certainly has a small number of distinct values. You should not choose this column as a partitioning key.

A good candidate for the partitioning key would be O_CUSTKEY or O_ORDERKEY, because of a high number of distinct values. Also, the queries should be considered for the table collocation to define a good partitioning key candidate, as described in "How to choose a good partitioning key" on page 99.

Another consideration about an inappropriate partitioning key is the cost of applying the partitioning hash algorithm that is proportional to the size of the partitioning key. The partitioning key cannot be more than 16 columns, but fewer columns result in better performance. Unnecessary columns should not be included in the partitioning key.

Tip: To check if you have uneven data distribution, you can run the following statement. This statement will show you the number of rows per partition you have in a table:

```
db2 "select dbpartitionnum(partkey), count(*) from <schema.table> * group by
dbpartitionnum(partkey) order by dbpartitionnum(partkey)"
```

Substitute the appropriate values for schema, table, and partitioning key column.

Follow an example result with number of rows for each partition where the table resides:

```
1           2
----------- -----------
          1      281645
          2      281417
          3      282250
          4      281854
          5      281444
          6      280939
          7      281515
          8      281141
          9      281684
         10      280886
         11      280267
         12      279949

12 record(s) selected.
```

How to choose a good partitioning key

In this section we provide the main considerations for choosing a good partitioning key.

The topics covered are:

- Rules for good partitioning key candidate
- Rules govern the use of partitioning keys
- Partition compatibility
- Table collocation
- Replicated Materialized Query Table

Rules for good partitioning key candidate

To choose a good partitioning keys candidate, you should consider the following rules:

- Frequently joined columns.

- Columns that have a high proportion of different values to ensure an even distribution of rows across all database partitions in the database partition group.
- Integer columns are more efficient than character columns, which are more efficient than decimal.
- Equijoin columns.
- Use the smallest number of columns possible.

Rules for the use of partitioning keys

The following rules are for the use of partitioning keys:

- Creation of a multiple partition table that contains only long data types (LONG VARCHAR, LONG VARGRAPHIC, BLOB, CLOB, or DBCLOB) is not supported.
- The partitioning key definition cannot be altered.
- The partitioning key should include the most frequently joined columns.
- The partitioning key should be made up of columns that often participate in a GROUP BY clause.
- Any unique key or primary key must contain all of the partitioning key columns.
- Tables containing only long fields cannot have partitioning keys and can only be placed into single-partition database partition groups.
- If a partitioning key is not defined, one is created by default from the first column of the primary key. If no primary key is specified, the default partitioning key is the first non-long field column defined on that table. (Long includes all long data types and all large object (LOB) data types.)
- If you are creating a table in a table space associated with a single-partition database partition group, and you want to have a partitioning key, you must define the partitioning key explicitly. One is not created by default. If no columns satisfy the requirement for a default partitioning key, the table is created without one. Tables without a partitioning key are only allowed in single-partition database partition groups.

Restriction: To *add* a partitioning key at a later time, the table must be defined in a table space on a single-partition database partition group and must not already have a partitioning key. If the partitioning key already exists for the table, the existing key must be dropped before adding the new partitioning key.

A partitioning key cannot be added to a table that is a sub-table.

Partition compatibility

The base data types of corresponding columns of partitioning keys are compared and can be declared partition compatible. Partition-compatible data types have the property that two variables, one of each type, with the same value, are mapped to the same partition number by the same partitioning algorithm.

Partition compatibility has the following characteristics:

- A base data type is compatible with another of the same base data type.
- Internal formats are used for DATE, TIME, and TIMESTAMP data types. They are not compatible with each other, and none are compatible with CHAR.
- Partition compatibility is not affected by columns with NOT NULL or FOR BIT DATA definitions.
- NULL values of compatible data types are treated identically; those of non-compatible data types may not be.
- Base data types of a user-defined type are used to analyze partition compatibility.
- Decimals of the same value in the partitioning key are treated identically, even if their scale and precision differ.
- Trailing blanks in character strings (CHAR, VARCHAR, GRAPHIC, or VARGRAPHIC) are ignored by the hashing algorithm.
- BIGINT, SMALLINT, and INTEGER are compatible data types.
- REAL and FLOAT are compatible data types.
- CHAR and VARCHAR of different lengths are compatible data types.
- GRAPHIC and VARGRAPHIC are compatible data types.
- Partition compatibility does not apply to LONG VARCHAR, LONG VARGRAPHIC, CLOB, DBCLOB, and BLOB data types, because they are not supported as partitioning keys.

Table collocation

In a partitioned database, the performance of join queries can be greatly enhanced through collocation of rows of the different tables involved in the join.

Collocation between two joining tables means having to have the matching rows of the two tables always in the same database partition.

Note: If tables are in the same database partition group, they have the same pmap or partitioning map. For collocation, there must be equijoin predicates between all corresponding partitioning key columns.

Figure 3-10 describes a collocated join example, where the CUST and ORDERS tables have been partitioned on the *cust_id* column. An SQL query that requires a join on the *cust_id* column will see significant performance benefits from this partitioning scheme because of the greater parallelism achieved through collocated joins.

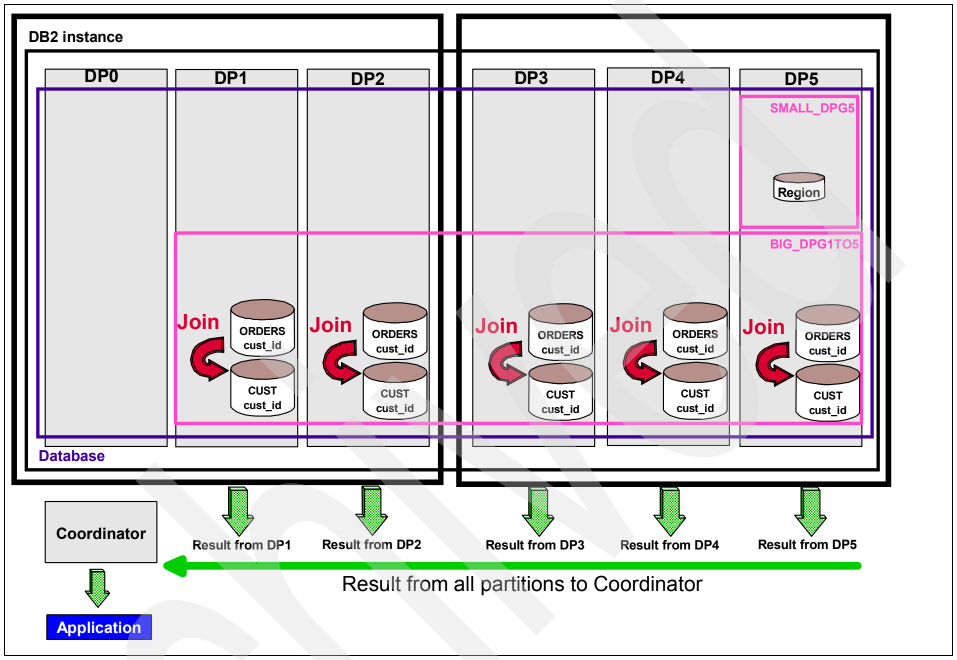


Figure 3-10 Table collocation

A collocated join will occur if two collocated tables are joined using all of the columns in the partitioning key.

Tables are collocated when they satisfy the following requirements:

- The tables must be in same database partition group.
- The database partition group of both the tables must have the same partitioning map.
- The partitioning keys from both tables must have the same number of columns.
- Corresponding partitioning key columns must be partition compatible (the same or similar).

We always have to use the maximum of collocated joins, as this avoids the database manager having to ship data between partitions.

You can now use MQTs to replicate tables to other database partitions to enable collocated joins to occur even though the all tables are not joined on the partitioned key.

3.3.7 Size the tables

The environment example we used to calculate the table sizes is:

- Eight x335 (2-way/4-GB memory)
- Two database partitions on each machine (total = 16 database partitions)

Figure 3-11 details the estimated table sizes used in our test.

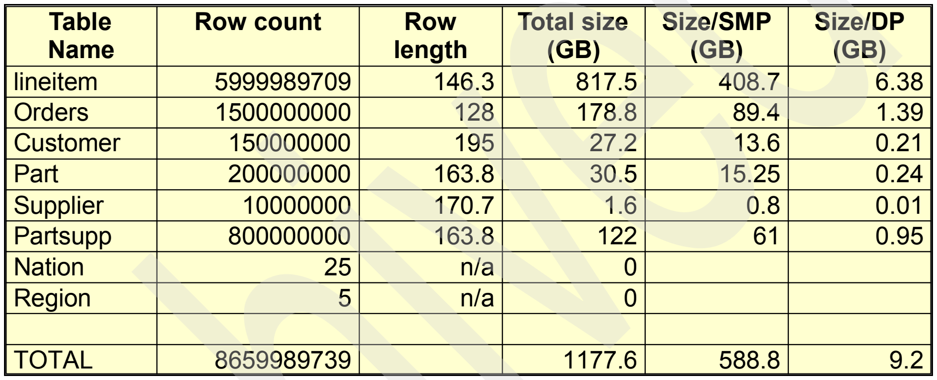


Table Name	Row count	Row length	Total size (GB)	Size/SMP (GB)	Size/DP (GB)
lineitem	5999989709	146.3	817.5	408.7	6.38
Orders	1500000000	128	178.8	89.4	1.39
Customer	150000000	195	27.2	13.6	0.21
Part	200000000	163.8	30.5	15.25	0.24
Supplier	10000000	170.7	1.6	0.8	0.01
Partsupp	800000000	163.8	122	61	0.95
Nation	25	n/a	0		
Region	5	n/a	0		
TOTAL	8659989739		1177.6	588.8	9.2

Figure 3-11 Table sizes

Follow the meaning of each of the columns described in Figure 3-11:

- Row count

  The number of rows in the table.

- Row length

  The byte counts sum of all columns in the table.

- Total size

  `Total size = Row count * Row length`

- Size/SMP

  `Size / SMP = Total size / number of physical machines`

  In this example, we used two p690 machines.

- Size/DP

  `Size / DP = (Size / SMP) / number of database partitions per machine`

  In this example, we used 32 partitions per machine.

3.3.8 Size for MDC utilization

You will get significantly improved performance and maintenance advantage when using MDC, if you choose the right set of dimensions for clustering a table and the properly extent sizes. On the other hand, if incorrect choices are made, utilization of space might be poor and performance could degrade. To design good MDC tables you must first identify potential dimension candidates, and then evaluate the storage space needed for an MDC table using some variation of those dimensions.

3.3.9 Size for MQT utilization

A materialized query table (MQT) is a table whose definition is based on the result of a query. The data of MQT is in the form of precomputed results that are taken from one or more tables on which the MQT definition is based. It can be used for applications that have a pattern of analysis using more or less similar queries with minor variations in a query's predicates, and are often issued repetitively against large volumes of data.

Based on this, the size for a MQT is estimated using the same calculation as for a base table. For more details about table sizing see 3.3.7, "Size the tables" on page 103. But, if your MQT is also defined as MDC, you also have to consider the MDC sizes.

3.3.10 Configure DB2 UDB

In this section we provide some important DB2 UDB configurations and recommendations for your database.

The topics included are:

- Database logs
- Buffer pool considerations
- DIAGPATH directory path placement

Database logs

In this section we describe management and performance considerations relating to database logging. If set up inappropriately for your workload, database logging could become a significant overhead.

The topics covered are:

- On which disks are the logs to be placed?
- Circular or archival log?
- Logging performance.
- File system or raw logical volume.

- Mirroring.
- Number of log files.
- Size of logs.

On which disks are the logs to be placed

Due to the importance of the log files, it is recommended that the log files should always reside on their own physical disk(s), separate from the rest of the database objects. The disks ideally should be dedicated to DB2 logging to avoid the possibility of any other processes accessing or writing to these disks.

Tip: It is highly recommended to place the database logs on separate disks from the data.

Availability

Irrespective of the type of logging you choose, whether it be circular logging or archival logging, the availability of the physical disks is crucial to the database. For this reason, it is strongly recommended that you protect the log against single disk failures:

- By using RAID 1 (mirroring the log devices) recommended for SSA disks.
- By storing log files on a RAID 5 array recommended for ESS disks. Since a RAID 5 array has a large cache, the write penalty effects can be significantly decreased.

For details about RAID levels, see 3.2.1, “Summary of the most popular RAID levels” on page 65.

Circular or archival log

DB2 UDB can operate in one of two modes relating to transaction logging:

- Circular logging

  This means that writes to the database are logged to enable rollback of outstanding units of work. However, after a commit, log records relating to that unit of work are not retained, and the log files will be reused in a circular manner. This means that it is not possible to replay transactions to recover data when operating in this mode. Online backups, table space backups, and roll forward are not allowed when using circular logging.

- Archival logging

  In this mode writes are logged in the same way, but log records are retained after a commit. This means that following a restore of the database, it is possible to replay transactions or roll forward through log records, up to a point in time or the end of the log records.

There are some other advantages, for example, a database in archival logging mode can be backed up using an online backup (reads and writes to the database are allowed at the same time), and also can be backed up and restored at the table space level.

The choice of logging mode is generally determined by the type of data being written and the availability requirements. If the data is *first-time data*, cannot be reproduced elsewhere, and has a significant cost associated with any loss, then generally archival logging is required. Also, if the database is required for read/write access 24x7, then online backup may also be a requirement.

If this is not the case, then for ease of management and simplicity, circular logging is generally preferred.

If you have sufficient time available to take offline backups, perhaps you may plan a cycle of full backups followed by one or more incremental backups.

Note: In your BI environment, if your data load is scheduled (for example, every night) it is recommended that you use circular logging if you have sufficient time available to take offline backups.

But, if your BI environment has several updates during the day, and if the database is required for read/write access 24x7, then archival logging is recommended, since online backups have to be taken. In this case, it is important to have good log administration (for example, using TSM, because logs are required for the database restore).

Logging performance

The log files exist to provide the ability to be able to recover your environment to a consistent state, and preserve the integrity of your data, so logging performance is very important.

Placement of these files needs to be optimized, not only for write performance, but also for read performance, because the database manager will need to read the log files during database recovery.

Note: In DB2 UDB V7.1, the total log files may be up to 32 GB. In DB2 UDB V8.1 and later, total log files can now be up to 256 GB in size, and you can perform extremely large amounts of work within a single transaction.

File system or raw logical volume

We now have two options when deciding how to store the DB2's transaction log files: Either in the operating system's file system or in a raw logical volume. The most familiar method here is to use the file system, as this is all we have supported in the past for logs. Using raw logical volumes for logging will bring performance improvements. However, it is worth pointing out that in most situations there are advantages and disadvantages to using raw logical volumes for your log devices:

- Advantages
  - The I/O path is shorter, since you are now bypassing the operating system's file system.
  - Raw device striping may provide faster I/O throughput.
  - No restrictions are imposed by a file system.
  - A raw device can span multiple disks.
- Disadvantages
  - The device (logical volume) created for the logs must be dedicated to DB2 UDB.
  - The device (logical volume) created for the logs cannot be operated upon by any operating system utility or third-party tools, which would back up or copy from the device.
  - Currently, if you are using IBM data replication products, the read log API will not call the user exit if you use the raw log device. The recommendation is that IBM data replication products should not be used when using raw log devices. Similarly, do not use raw log devices if you use the sqlurlog API.

Mirroring

As the log files are crucial to your database, you can either mirror your log devices to ensure that the disks they reside on are mirrored, or use the database configuration parameter *mirrorlogpath*. It is recommended that the log disks be entirely dedicated to DB2 UDB, and that no other processes write to these disks. If possible, place the disks on separate disk controllers to maximize availability.

Number of log files

Your active log space will be determined by the number of log files you define. As we have seen already, the active log space is defined by the parameters (LOGPRIMARY + LOGSECOND)*LOGFILSIZ. You need to take into account your database workload, and therefore the potential logging activity.

Most DSS workloads with little update, insert, or delete activity may benefit from a smaller number of log files. However, in this environment, where logging activity is low, it may be worthwhile to use a larger log file size to reduce administration overhead. For example, if you are using archival logging, the log files must be well administrated, since they are required for database restoration, so having larger log files and a smaller number of log files is easier to administrate than having smaller log files and a larger number of log files.

Size of logs

Remember that the total active log space can now be up to 256 GB. For a partitioned database, this limit is per partition. So the calculation would be:

```
( LOGPRIMARY + LOGSECONDARY ) * LOGFILSZ * 4k < 256 GB
```

In this formula, LOGFILSIZ represents the Log File Size database configuration parameter.

If you are using raw devices for logging, then remember that log records are still grouped into log extents; the size of each is LOGFILSIZ (4-KB pages). DB2 UDB places the log extents consecutively in the log device, and every log extent carries with it a two-page overhead for extent header information. This affects the number of log extents that can be written to your raw log device. The total number can be calculated by the formula:

```
raw-device-size / (logfilsz + 2)
```

The device must be large enough to support the active log space that you are going to allocate. By this we mean that the number of log extents that can be contained on your raw device must be greater than (or equal to) the value of LOGPRIMARY.

Buffer pools considerations

In a DSS environment, the main memory allocation is for buffer pool and sorting. For information about sort parameters, see "DB2 UDB parameters" on page 110.

The general rule of thumb for a DSS environment is to start with about 50 percent of your system's main memory devoted to buffer pool(s), but this rule has to be considered *very* carefully before being followed. The biggest factor for consideration is if this is a dedicated database server.

At least one buffer pool per table space page size is required. In a DSS environment we would suggest starting with as few buffer pools as possible (one per page size) with one exception, which is to use a small buffer pool for the fact table (since you are never going to get much of the table in memory, the hit rate will not be good), but big enough of course to get a few prefetches running concurrently. Then you can allocate a bigger buffer pool to the dimension tables, so they become memory resident without having pages flushed every time the fact table is scanned. In general, buffer pools do not need to be as large as in a OLTP environment, and memory may be better utilized for sorting (SORTHEAP and SHEAPTHRES).

To help you to tune buffer pool size, use the database system monitor (snapshot or event monitor) to gather information to get the *buffer pool hit ratio* information.

The buffer pool hit ratio indicates the percentage of time that the database manager did not need to load a page from disk in order to service a page request. That is, the page was already in the buffer pool. The greater the buffer pool hit ratio, the lower the frequency of disk I/O. The buffer pool hit ratio can be calculated as follows:

```
(1 -((pool_data_p_reads + pool_index_p_reads)/
(pool_data_l_reads + pool_index_l_reads)))*100%
```

This calculation takes into account all of the pages (index and data) that are cached by the buffer pool. You can find the values for calculation using the database system monitor (snapshot or event monitor).

Note: The buffer pool size specified in the CREATE BUFFERPOOL command is per partition. So, DB2 UDB will allocate this amount for each database partition, which belongs to the database partition group you specified in the CREATE BUFFERPOOL command. For example, if you have the database partition group “DPG_ALL”, which contains five database partitions, and then you execute the following command:

```
create bufferpool BP_16KB database partition group DPG_ALL size 32000
pagesize 16k
```

The calculation is:

```
BPTotal Size = (size in pages * page size) * number of partitions in DP
group
```

So, in the example there will be (32000*16 kb) * 5 = 2.4 Gb for BP total size.

Note 1: For a 64-bit database, remember that you can define larger buffer pools, SORTHEAP, package caches, and the other resources that can consume large amounts of memory.

Note 2: Each buffer pool page allocated uses space in the memory allocated for the database heap (DBHEAP), for internal control structures. To reduce the need for increasing the DBHEAP when buffer pools increase in size, nearly all buffer pool memory comes out of the database shared-memory set and is sized *online*, *automatically*.

DIAGPATH directory placement

DIAGPATH is a database manager parameter that allows you to specify the fully qualified path for DB2 UDB diagnostic information.

In addition to defining where the db2diag.log, traps, and dump files will be written, with DB2 UDB ESE V8.1 the DIAGPATH parameter controls where the new Administration Notification log is written. The name of this log is the combination of the instance name and *.nfy*; for example, db2inst1.nfy. This new type of diagnostic log is used to write notification messages intended for use by database and system administrators. For example, any alerts generated by the new health indicators are written to this log. Detailed diagnostic information is still written to the db2diag.log.

It is recommended to separate the diagnostic information and data to different disks to avoid disk contention, preferentially in local disks. This separation can also be used to ensure that information written into the diagnostics directory will not be able to fill up your data or log file systems, and can cause DB2 UDB to crash.

The DIAGPATH is a parameter that affects all databases in the instance. In a partitioned database environment that involves more than one physical machine, the path you specify must either reside on a shared file system accessible by all machines, or the same path must be defined on each machine. If the path is on a shared file system, diagnostic information from all partitions will be written to the same diagnostic logs. If the same path is defined on each machine on local file systems, the diagnostic logs on each machine will contain only the information from the partitions that are located on it.

3.3.11 Recommended parameters for performance

The information below gives hints, tips, and suggestions for your partitioned database and have been gathered from various IBM experiences with customers using DB2 UDB ESE V8.1.

The topics included are:

- DB2 UDB parameters
- Linux parameters
- Network parameters

DB2 UDB parameters

In this topic, we describe the main recommended DB2 UDB parameters, which include:

- DB2 variables
- DB2 database manager parameters

- DB2 database parameters

DB2 variables

Follow the main recommended DB2 variables.

> **Attention:** A **`db2stop`** and **`db2start`** must be done in order for the DB2 variables to take effect.

- DB2_STRIPED_CONTAINERS

  When creating a DMS table space container, a one-page tag is stored at the beginning of the container. The remaining pages are available for storage by DB2 UDB, and are grouped into extent-size blocks of data.

  When using RAID devices for table space containers, it is suggested that the table space be created with an extent size that is equal to, or a multiple of, the RAID stripe size. However, because of the one-page container tag, the extents will not line up with the RAID stripes, and it may be necessary during the I/O request to access more physical disks than would be optimal. This is particularly important when the RAID devices are not cached, and do not have special sequential detection and prefetch mechanisms (such as native SSA disks).

  The reason this variable does not affect DMS file containers is that the pages allocated for the DMS file container are still allocated by the file system, so we cannot guarantee that the pages in the container will be contiguous across the disk(s). With raw devices, DB2 UDB controls this and allocates the pages contiguously.

  When DB2_STRIPED_CONTAINERS is set to ON when the table space is created, the initial container tag will take up a full extent. This will avoid the problem described above.

  > **Note:** We recommend that DB2_STRIPED_CONTAINERS is set to on. With ESS, sequential detection and prefetch may alleviate this problem.

- DB2_PARALLEL_IO

  We recommend setting the DB2_PARALLEL_IO variable if using multiple physical disks for each container. For example, if the container is created on a RAID device that is composed of more than one physical disk, you may want to issue parallel read and write calls.

  When you specify the table space's ID to the DB2_PARALLEL_IO registry variable, the database is able to issue parallel I/O requests to that table space, even though it consists of a single container. DB2 UDB achieves this by enabling multiple prefetchers simultaneously for the table space, which results in improved utilization of the underlying physical drives. If DB2_PARALLEL_IO is on, then the number of prefetchers that DB2 UDB dispatches is equal to prefetchsize/extentsize.

You can enable parallel I/O for every table space using the DB2_PARALLEL_IO variable, as shown in the following example:

```
db2set DB2_PARALLEL_IO=*
```

You can also enable parallel I/O for a list of table spaces' IDs (comma-separated), as in the following example:

```
db2set DB2_PARALLEL_IO=1,2,4
```

A list of table space IDs can be obtained by running the LIST TABLESPACES command when connected to your database, or you could run this command:

```
SELECT TBSPACE,TBSPACEID FROM SYSCAT.TABLESPACES
```

After setting the DB2_PARALLEL_IO registry variable, you must run **db2stop** and then **db2start** for the change to take effect.

DB2_PARALLEL_IO should also be enabled for striped containers when PREFETCHSIZE > EXTENTSIZE.

- DB2_FORCE_FCM_BP

  Enable the DB2_FORCE_FCM_BP variable since it is recommended for partitioned database environments, which have multiple database partitions in a physical machine. It uses shared memory for inter-partition communication instead of the network, and provides a faster communication between the database partitions. When the DB2_FORCE_FCM_BP variable is set to Yes, the FCM buffers are always created in a separate memory segment, so that communication between FCM daemons of different logical partitions on the same physical partition occurs through shared memory. Otherwise, FCM daemons on the same partition communicate through Linux sockets, and communicating through shared memory is faster.

  The following command enables the DB2_FORCE_FCM_BP variable:

  ```
  db2set DB2_FORCE_FCM_BP=YES
  ```

DB2 database manager parameters

Follow the main recommended DB2 database manager configuration parameters:

- SHEAPTHRES and SHEAPTHRES_SHR

  In a DSS environment, the main memory allocation is for buffer pool and sorting (SHEAPTHRES, SHEAPTHRES_SHR and SORTHEAP). For information about buffer pool estimate size, see "Buffer pools considerations" on page 108.

SHEAPTHRES is used to control the total amount of memory allocated for sorting on the DB2 UDB server, and applies on a per-partition basis. This parameter should be set to the maximum amount of memory for all SORTHEAP that should be allowed to be allocated at any given time, specifically when using connection concentrator.

The rule of thumb to calculate SHEAPTHRES is:

```
SHEAPTHRES = (SORTHEAP * # concurrent sorts)
```

The SHEAPTHRES parameter is used differently for private and shared sorts:

– 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. When the total private-sort memory consumption for an instance reaches this limit, the memory allocated for additional incoming private-sort requests will be considerably reduced.

– 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. When this limit is reached, no further shared-sort memory requests will be allowed (until the total shared-sort memory consumption falls below the limit specified by SHEAPTHRES).

This parameter is completely related to the SORTHEAP parameter. For sort tune tips, see "DB2 database configuration recommendations" on page 114, SORTHEAP parameter.

SHEAPTHRES_SHR represents a hard limit on the total amount of database shared memory that can be used for sorting at any one time. When the total amount of shared memory for active shared sorts reaches this limit, subsequent sorts will fail (SQL0955C). If the value of SHEAPTHRES_SHR is 0, the threshold for shared sort memory will be equal to the value of the SHEAPTHRES database manager configuration parameter, which is also used to represent the sort memory threshold for private sorts. If the value of SHEAPTHRES_SHR is non-zero, then this non-zero value will be used for the shared sort memory threshold. SHEAPTHRES_SHR is only meaningful in two cases:

– If the intra_parallel database manager configuration parameter is set to yes, because when INTRA_PARALLEL is set to no, there will be no shared sorts.

– If the concentrator is on (that is, when max_connections is greater than max_coordagents), because sorts that use a cursor declared with the WITH HOLD option will be allocated from shared memory.

► INTRA_PARALLEL

This should only be set to on if you have more than two CPUs per partition and you have low query concurrency. This is particularly important for large

SMPs, where we could end up with hundreds of agents dispatched for a single query, and thousands of DB2 UDB processes running concurrently, in which case, task switching becomes a significant overhead. The large numbers of processes running is another reason to consider physical partitioning.

When running INTRA_PARALLEL or DFT_DEGREE set to YES with multiple data partitions per SMP, set MAX_QUERYDEGREE to the number of processors per data partition; for example, on a 32-way p690 with eight data partitions, set MAX_QUERYDEGREE to 4.

- CPUSPEED

  Set this database manager configuration parameter to –1. This will be the most recommended value for DB2 UDB. By doing this, DB2 UDB will calculate the value.

- FCM_NUM_BUFFERS

  If you have multiple database partitions on the same machine, you may find it necessary to increase the value of this parameter.

  It is recommended to start with a default value (4,096). If you run out of message buffers because of the number of users on the system, or the number of database partition servers on the system, or the complexity of the applications, you may find it necessary to increase it.

DB2 database configuration recommendations

Follow the main DB2 database configuration recommendations:

- SORTHEAP

  SORTHEAP is allocated to each agent, and agents are started per partition for a query accessing the table on a partition. The agent does not necessarily allocate a full SORTHEAP. If the optimizer calculates a smaller amount of SORTHEAP, and this amount is enough for the sort, then a smaller amount of sort memory will be requested by the agent at run time.

Important: It is important to allocate as much memory as possible to the SORTHEAP (and set the threshold accordingly) without over allocating memory and causing memory paging to occur. It is possible for a sort to be done entirely in sort memory. We will recommend setting SHEAPTHRES to 50 percent of the DB2 UDB memory, and then divide by the number of concurrent sorts expected to get SORTHEAP. Typically, this gives SORTHEAP in the region of 5000 to 10000 pages.

However, if this causes the operating system to perform page swapping, then you can lose the advantage of a large SORTHEAP. So, whenever you adjust the SORTHEAP and SHEAPTHRES configuration parameters, use an operating system monitor to track any changes in the system paging.

In the sort phase, the sort can be categorized as overflowed or non-overflowed. When considering the return of the results of the sort phase, the sort can be categorized as piped or non-piped:

– Overflowed and non-overflowed

  If the information being sorted cannot fit entirely into the SORTHEAP (a block of memory that is allocated each time a sort is performed), it overflows into temporary database tables. Sorts that do not overflow (non-overflowed) always perform better than those that do.

– Piped and non-piped

  If sorted information can return directly without requiring a temporary table to store a final, sorted list of data, it is referred to as a *piped sort*. If the sorted information requires a temporary table to be returned, it is referred to as a *non-piped sort*. A piped sort always performs better than a non-piped sort.

The DB2 optimizer will determine if a non-overflowed sort can be performed, and if a piped sort can be performed by comparing the expected result set with the value of the SORTHEAP database configuration parameter.

Tip: For good sort performance, always try to have non-overflowed and piped sorts.

To help you to tune SORTHEAP and SHEAPTHRES, you can use the system performance monitor (snapshot, for example):

– To check if you have overflowed sorts, you have to consider the total sorts and sort overflows elements, as shown in Example 3-4 on page 116.

  You can calculate the percentage of overflow sorts by sort overflows/total sorts, and use the value to determine if the optimizer is attempting to use a

SORTHEAP and fails. If the percentage is high, consider increasing the SORTHEAP and/or SHEAPTHRES values.

Example 3-4 Snapshot for database output

```
db2 get snapshot for database on pfp |grep -i sort
Total Private Sort heap allocated          = 2059
Total Shared Sort heap allocated           = 0
Shared Sort heap high water mark           = 0
Total sorts                                = 5
Total sort time (ms)                       = 55749
Sort overflows                             = 0
Active sorts                               = 0
```

– To check piped and non-piped sorts, you have to consider the *Piped sorts requested* and *Piped sorts accepted* elements, as shown in Example 3-5.

You can calculate the percentage of piped sorts accepted by piped sorts accepted/piped sorts requested, and determine if piped sorts are being chosen by the optimizer, but not accepted. If the percentage is low, consider increasing the SORTHEAP and/or SHEAPTHRES.

Example 3-5 Snapshot output for database manager

```
db2 get snapshot for database manager |grep -i sort
Private Sort heap allocated                    = 0
Private Sort heap high water mark              = 2059
Post threshold sorts                           = 0
Piped sorts requested                          = 1
Piped sorts accepted                           = 1
Sorting Information                     (SORT) = ON  12-06-2002 16:54:21.712838
```

Important: On a partitioned database, you can take a snapshot of the current partition, a specified partition, or all partitions using global snapshot.

- NUM_IOSERVERS

  The NUM_IOSERVERS database configuration parameter specifies the number of the I/O servers that perform prefetch I/O for the database. You should normally set the same number for the NUM_IOSERVERS parameter as the number of physical disks. If PREFETCHSIZE is increased above the suggestions outlined so far (for *additional* prefetching), then the NUM_IOSERVERS database configuration parameter should be set to at least PREFETCHSIZE /EXTENTSIZE. For a large system, beware not to make this too big. Remember that it is specified per partition. To calculate it, if you stripe every table space across each rank, you can easily end up with a total number of prefetchers (NUM_IOSERVERS*#partitions) that would be as many times as the total number of disks. A realistic number may be a maximum of 20–30 I/O servers per partition.

Note: To help you to tune NUM_IOSERVERS, use the snapshot or event monitor information.

Check the data element prefetch_wait_time. It shows the time an application spent waiting for an I/O server (prefetcher) to finish loading pages into the buffer pool. If the prefetch wait time is considerable, you might increase the NUM_IOSERVERS.

- NUM_IOCLEANERS

  This parameter allows you to specify the number of asynchronous page cleaners for a database. These page cleaners write changed pages from the buffer pool to disk before the space in the buffer pool is required by a database agent. This means that the agents will not wait for changed pages to be written out. As a result, your application's transaction can run faster.

  You may use the database system monitor to help you tune the num_iocleaners configuration parameter using information from the event monitor about write activity from a buffer pool:

  – The parameter can be reduced if both of the following conditions are true:
    - pool_data_writes is approximately equal to pool_async_data_writes.
    - pool_index_writes is approximately equal to pool_async_index_writes.
  – The parameter should be increased if either of the following conditions are true:
    - pool_data_writes is much greater than pool_async_data_writes.
    - pool_index_writes is much greater than pool_async_index_writes.

Tip: Since page cleaner I/O is now asynchronous, a smaller value can be set in num_iocleaners than in DB2 UDB V7. You can start with 2.

- Multi-page file allocation

  SMS table spaces are expanded on demand. This expansion is done a single page at a time by default. However, in certain workloads (for example, when doing a bulk insert) you can increase performance by using the db2empfa tool to tell DB2 UDB to expand the table space in groups of pages or extents. The db2empfa tool is located in the bin subdirectory of the sqllib directory. Running it causes the multipage_alloc database configuration parameter to be set to YES. In a multipartition database, db2empfa must be run on every partition.

Linux performance tuning

In our tests we did not spend much time tuning the Linux operating system. There are many tuning options available for Linux. For more information refer to:

- *Tuning SUSE LINUX Enterprise Server on IBM eServer xSeries Servers*, REDP-3862
- *Tuning Red Hat Enterprise Linux on IBM eServer xSeries Servers,* REDP-3861

3.3.12 Naming conventions

A major factor in keeping your building blocks symmetrical and easy to manage is the use of well-planned naming conventions. Many DB2 commands and SQL statements provide a method to do character string substitution on statements, allowing you to send the same statement to all of your partitions with one command. With a well-planned naming convention, such as using combinations of letters and numbers, you will be able to utilize these features more effectively or code script with input parameters.

Naming conventions are more important for some objects than others. In general, poor naming conventions for objects that touch the operating system (file systems, for example) will be more problematic than for database objects. Also, naming conventions are more important on some database objects than on others. For example, partition group names and buffer pools are more or less informational or descriptive. Strict naming conventions for these objects is not as critical.

Server names

We recommend using a meaningful combination of letters and numbers rather then using server name “themes” such as planets, Greek gods, or Disney characters. For example, we chose DB2RB*xx* (DB2 Redbook) for servers used in lab exercises, where *xx* is a number from “01” through “12”.

If you have multiple environments (production, development, test, QA) use a character to designate the environment.

Develop an abbreviation scheme for each project. For example, ACT (accounting), FIN (finance), etc.

A scheme like this is easier to remember and script than “Dopey, Sleepy, Grumpy and Doc.” You will not have to expend mental energy trying to remember server names and you will not have to invent a name for an eighth dwarf (server.)

Partition groups

We recommend using pg_*xxx*, where *xxx* is a descriptive name. For example:

- pg_all - A partition group on all partitions
- pg_zero - A partition group on partition zero (0) only
- pg_cat - A partition group for the catalog

Buffer pools

We recommend that you use bp*n*k, where *n* is the page size. In general you will only need one buffer pool for each page size. In the case of a special-purpose buffer pool (for example, for pinning code tables in memory), use a descriptive name to avoid use of that buffer pool by other tables. For example:

- bp32k - Buffer pool with 32-K page size
- bp4k - Buffer pool with 4-K page size
- bp4kcode - Special purpose 4-k buffer pool

File Systems and mount points

Tip: Keep your failover and scaling requirements in mind as you plan your file system names. Your failover process will likely involve mounting one partition's file systems on a new server, possibly one that already has another database partition. Likewise, if you move database partitions to new servers in order to scale, your naming convention must allow this to happen without file system name collision.

We recommend using unique naming conventions for you file systems. Below are some examples from the laboratory work for this publication.

- File systems and mount points for database logs

  ```
  /db2log<n>p<p>
  ```

  Where:

  – <n> = 1 for primary copy of logs
  – <n> = 2 for mirrored copy of logs

    Note on <n>: This refers to the "Path to log files" and MIRRORLOGPATH database configuration parameters, not LOGPRIMARY and LOGSECOND database configuration parameters.

  – <p> = Database partition number (see "Script Tip" on page 120)

- File systems and mount points for data table spaces

  Based on database size and storage configuration, you may have only a few file systems for table spaces or you may have hundreds. You may also be required to conform to a corporate naming convention. The key for user table space file system names is to be consistent and to avoid file system name collision if you are forced to move data partitions between servers for failover or scaling.

We used the following naming convention:

```
/db2fs<n>p<p>
```

Where:

– <n> = Sequential number
– <p> = Database partition number (see "Script Tip" on page 120)

For example, /db2fs1p1, /db2fs1p2, /db2fs3p1.

- File systems and mount points for system temp table spaces

  We recommend that you use the same naming convention that you use for the end-user data, but reserve one of the file system numbers (<n>) for system temp table spaces across all partitions. Again, the number or letter used is not as important as consistency, ease of use for scripting, and name collision avoidance.

  We chose the following:

```
/db2fs2p<p>
```

  Where <p> = database partition number (see "Script Tip" on page 120). Examples: /db2fs2p1, /db2fs2p2.

- File system and mount points for catalog table space

  We recommend that you use the same naming convention that you use for the end-user data and temp data, again reserving one of the file system numbers (<n>) for the catalog. This file system name will go unused on non-catalog partitions.

  We used the following naming convention:

```
/db2fs0p<p>
```

  Where:

  <p> = database catalog partition number (see "Script Tip" on page 120). Examples: /db2fs0p0

Script tip: Do not pad partition numbers with zeros in file system names. The **`db2_all`** command can perform character substitution on partition number. This allows you to send one **`db2_all`** command that will substitute the unpadded partition number for the string "##", as shown in Example 3-6.

Example 3-6 Using db2_all "##" substitution

```
## Send db2_all command to update logpath - unpadded partition number will
## be substituted for "##" in the actual command sent to each partition

db2_all "\"db2 update db cfg for tpch using newlogpath /db2log1p##/db2inst2/"

DB20000I  The UPDATE DATABASE CONFIGURATION command completed successfully.
db2rb09: db2 update db cfg for tpch using newlogpath /db2log1p##/db2inst2/
completed ok

DB20000I  The UPDATE DATABASE CONFIGURATION command completed successfully.
db2rb10: db2 update db cfg for tpch using newlogpath /db2log1p##/db2inst2/
completed ok

DB20000I  The UPDATE DATABASE CONFIGURATION command completed successfully.
db2rb09: db2 update db cfg for tpch using newlogpath /db2log1p##/db2inst2/
completed ok

DB20000I  The UPDATE DATABASE CONFIGURATION command completed successfully.
db2rb10: db2 update db cfg for tpch using newlogpath /db2log1p##/db2inst2/
completed ok

db2_all "db2 get db cfg for tpch |grep -i newlogpath"

Changed path to log files (NEWLOGPATH) = /db2log1p0/db2inst2/NODE0000/
db2rb09: db2 get db cfg for tpch |grep -i newlogpath completed ok

Changed path to log files (NEWLOGPATH) = /db2log1p1/db2inst2/NODE0001/
db2rb10: db2 get db cfg for tpch |grep -i newlogpath completed ok

Changed path to log files (NEWLOGPATH) = /db2log1p2/db2inst2/NODE0002/
db2rb09: db2 get db cfg for tpch |grep -i newlogpath completed ok

Changed path to log files (NEWLOGPATH) = /db2log1p3/db2inst2/NODE0003/
```

Note that the unbalanced double-quotes (“) in the above commands are correct, not typographical errors. There should be three double-quotes on a **db2_all** commands with this type of substitution.

Table spaces

Using meaningful names such as “fact” or “small” or “large” for table space is a good practice. However, simply numbering the user table spaces as “tsxx” should be considered when you have databases with large numbers of table spaces. This makes scripting easier. The goal is for easy maintenance and avoiding mistakes.

We chose the following naming convention:

- Catalog Tablespace - ts00

- User data - ts01, ts02, ts03, etc.
- System Temporary - tmp4k, tmp8k, etc.
- User Temporary - utmp4k, utmp8k, etc.

Table space containers

DB2 DDL has simplified syntax for defining containers that can be used if good naming conventions are followed. (See Example 3-6 on page 120) We recommend using a table space prefix followed by partition and container numbers. Some examples of table space prefixes might be "ts" (general table spaces), "dat" (data only), "idx" or "ndx" (index only).

DMS raw devices

In order to use the simplified DDL syntax and, more importantly, get the best performance, it is important to think at the partition level when creating the logical volumes. The file system names should be coordinated with the DB2 container partition numbers. We recommend the following convention:

```
/<device>/<tsprefix>c<c#>p<p#>
```

Where:

- <tsprefix> is a table space prefix (for example, dat for data, idx for index, etc.).
- <c#> is the container number within the data partition (for example, 1, 2, 3).
- <p#> is the data partition number without leading zeros.

For example:

```
/dev/ts01c1p1, /dev/rdatc2p9
```

Important: Change the ownership of both the character devices and block devices to <instanceID> and <instanceGroup>.

DMS file and SMS

For DMS file and SMS containers we recommend using the mount point name in the same way the file system name is used on DMS raw containers. When there will be multiple databases and/or instances on the same server you will have to adjust the names to avoid collision. We recommend the following convention:

```
/<mount point>/<tsprefix>c#p#
```

Or:

```
/<mount point>/<instance>/<database>/<tsprefix>c<c#>
```

Or:

```
/<mount point>/<instance>/<database>/<tsprefix>c<c#>p<p#>
```

Where:

- <tsprefix> is a table space prefix (for example, dat for data, idx for index, etc.).
- <c#> is the container number within the data partition (for example, 1, 2, 3).
- <p#> is the data partition number without leading zeros.

For example:

```
/db2fs1p1/db2inst1/tpch/ts01c00
/db2fs1p1/db2inst1/tpch/ts01c01
/db2fs1p1/ts01c1p1
/db2fs1p1/ts01c1p1/data/
/db2fs2p3/db2inst1/tpch/tmp4kc00
```

4

Implementation

This chapter describes our lab environment for DB2 Integrated Cluster Environment and guides you through how to implement DB2 Linux clusters. It focuses on multiple partition environments only. For more information about how to implement single partition DB2 environment, please refer to the IBM Redbook *Up and Running with DB2 for Linux,* SG24-6899.

In this chapter, we discuss the following topics:

- Lab environment.
- Prepare your Linux OS for DB2 Integrated Cluster Environment.
- DB2 partitioned database setup.
- Post installation tasks - Initial tuning.

4.1 Lab environment

When preparing this publication, we set up a lab environment to demonstrate DB2 Integrated Cluster Environment. This section covers the hardware environment, software environment, and topology of the lab environment.

4.1.1 Hardware environment

Our hardware environment includes IBM @servers, and storage and network devices. The detailed configuration is below.

Linux servers

The IBM @servers we use are:

- 4x IBM @servers X335, each has
  - 2x 3.06GHz Intel Xeon CPUs
  - 4-GB memory
  - 1x IBM 18.2 GB 10 K SCSI HDD
  - 2x Gigabit Ethernet Adapter
  - 1x QLogic QLA2312 Fibre Channel Adapter
- 4x IBM @servers X335; each has:
  - 2x 3.06GHz Intel Xeon CPUs
  - 4-GB memory
  - 1x IBM 18.2 GB 10 K SCSI HDD
  - 2x Gigabit Ethernet Adapter
  - 1x TopSpin 10 Gbps Infiniband Host Channel Adapter
- 4x IBM @servers e325; each has:2x AMD Opteron 248 (2.2 GHz) CPUs
  - 6-GB memory
  - 1x 18.2 G 10 K SCSI HDD
  - 2x Gigabit Ethernet Adapter
  - 1x QLogic QLA2312 Fibre Channel Adapter

Storage

The external storage used are:

- 1x IBM DS4400 (FAStT700)
- 3x IBM EXP700
- 28x IBM 36 GB 15 K 2 GB FC HDDs
- 14x IBM 73.4 GB 10 K 2 GB FC HDDs

Network

Network equipment is:

- 1x Cisco 1000Base-X switch
- 1x TOPSPIN Infiniband switch

4.1.2 Software environment

In this lab environment, we use two Linux distributions:

- SuSE Linux Enterprise Server (SLES) 9
- Red Hat Linux Enterprise Linux (RHEL) 3 update 2

Note: We strongly recommend using Linux enterprise distributions when implementing DB2 Integrated Cluster Environment. Also remember to only use official updates of Linux enterprise distributions. For the latest list of supported distributions, please refer to:

`http://www.ibm.com/db2/linux/validate`

In addition, we use IBM *Tivoli System Automation* for Linux to implement high availability of DB2 servers. DB2 UDB V8.2 has numerous enhancements that can deliver benefits in Linux environments. We used DB2 UDB V8.2 in our lab environment to leverage Linux kernel features. The following is detailed software version information:

- IBM DB2 UDB V8.2 Enterprise Sever Edition (ESE) with Database Partitioning Feature (DPF)
- Tivoli System Automation for Linux, V1.2

When you implement DB2 Integrated Cluster Environment, ensure that you use the latest available DB2 FixPak. For DB2 FixPaks information, please refer to the DB2 Universal Database for Linux, UNIX, and Windows support Web page:

`http://www.ibm.com/software/data/db2/udb/support.html`

4.1.3 Topology of the lab environments

We use three basic configuration units in our lab environment. The first configuration is four x335 servers connected to a DS4400 storage server via Fiber Channel (FC). In the second configuration, we use a TopSpin InfiniBand switch to connect four x335 servers with a DS4400 storage server. The third one is four e325 servers accessing EXP700 storage directly by fiber channel. This section discusses the topology of our lab environment. Note that the amount of storage used in these configurations is quite minimal, and for real deployments you will likely need more storage servers and expansion units.

Configuration 1

As illustrated in Figure 4-1, we have four servers, which are named db2rb01, db2rb02, db2rb03, and db2rb04. For the storage we have a DS4400 storage server and EXP700 storage expansion units; each has 14x IBM 36GB 15K 2GB FC HDDs. Each x335 server is connected to the DS4400 storage server using Fibre Channel. We run RHEL 3 on these four servers.

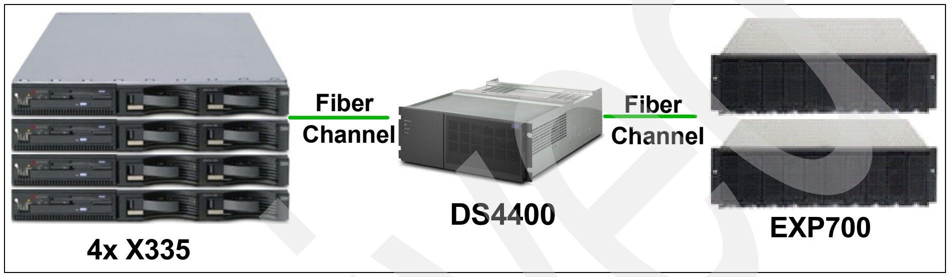


Figure 4-1 X335 servers connected directly to DS4000 Series storage

Configuration 2

In this configuration (Figure 4-2 on page 129), we utilize the same set of servers as in configuration 1, but use InfiniBand to connect the x335s servers to each other and to the storage infrastructure. Later, we expand this configuration to a 8-node configuration for the lab work for Chapter 6, “High availability” on page 309. The added servers are named db2rb05, db2rb06, db2rb07 and db2rb08. We run RHEL on these eight servers.

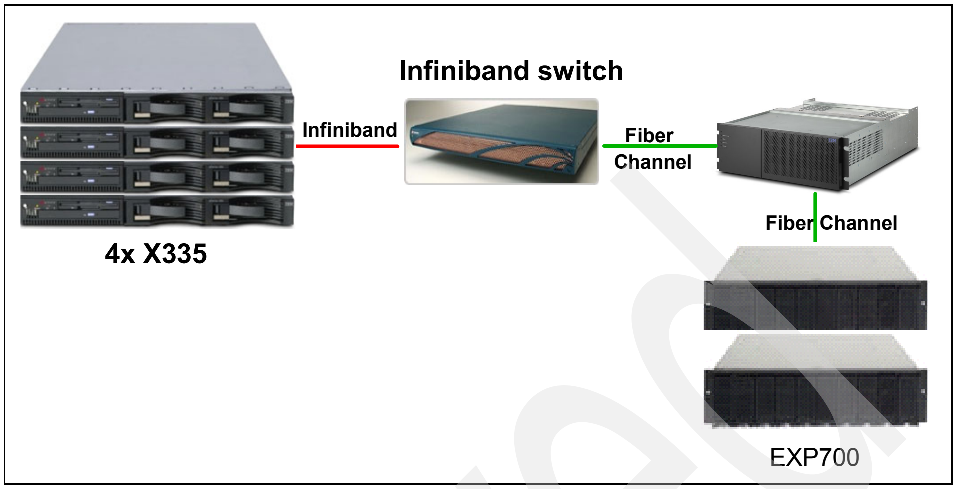


Figure 4-2   X335 servers connect to DS4000 Series storage via Infiniband switch

Configuration 3

In this configuration (Figure 4-3), e325 servers are directly connected to a EXP700 storage expansion unit with 14 IBM 73.4GB 10K 2GB FC HDDs. Four servers are connected to each other using Gigabit Ethernet. The server names are db2rb09, db2rb10, db2rb11 and db2rb12. We run SLES on these servers.

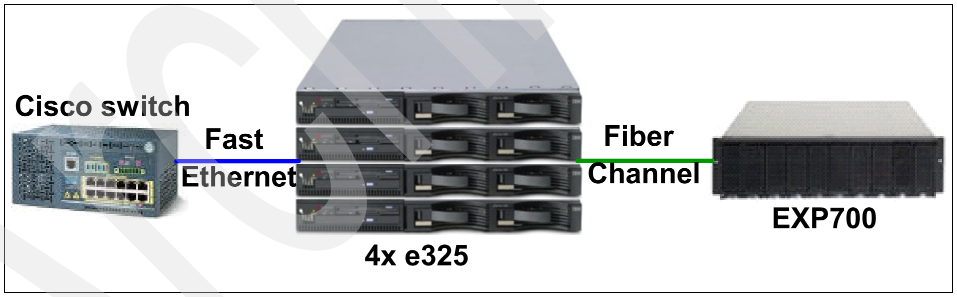


Figure 4-3   e325 servers directly connected to EXP700

In this chapter, we use configuration 1 as our example to explain how to prepare a Linux operating system and set up DB2.

4.2 Prepare your Linux OS for DB2 Integrated Cluster Environment

This section outlines requirements for installing DB2 Integrated Cluster Environment for Linux, and gives you some recommendations and procedures to prepare your Linux OS for DB2 Integrated Cluster Environment. It includes the following information:

- Basic requirements
- Installation planning and preparation

4.2.1 Basic requirements

Prior to install DB2 Integrated Cluster Environment, it is important to ensure that your computer meet both hardware and software requirements. Here are detailed requirements.

Required disk space

The disk space required for your product depends on the type of installation you choose and the type of file system you have. The DB2 Setup wizard provides dynamic size estimates based on the components selected during a typical, compact, or custom installation. Table 4-1 lists required disk space in the Linux operation system.

Table 4-1 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.	370 to 470 M
Compact	Installs only basic DB2 features and functions and does not include graphical tools.	310 to 380 M
Custom	This option allows you to select the features you wish to install.	310 to 890 M

When planning your disk space, remember to include disk space for required software, communication products, and documentation.

DB2 Information Center for DB2 UDB V8.2 is a fully searchable server that provides quick access to DB2 product information. You can install it through a separate document, CD-ROMs, or your can access it through the IBM Web site where the most recently release version is available:

`http://publib.boulder.ibm.com/infocenter/db2help/`

DB2 is usually installed to /opt/IBM/db2/V8.1 on the Linux operating system. Ensure that you have enough space on the proper file system, such as /opt. You can see disk space usage through the **`df -k`** command.

Memory requirements

At a minimum, DB2 UDB requires 256 MB of RAM. 512 MB of RAM memory is recommended if you use the GUI tools. When determining memory requirements, be aware of the following:

- For DB2 client support, these memory requirements are for a base of five concurrent client connections. You will need an additional 16 MB of RAM per five client connections.
- Additional memory is required for other software that is running on your system.
- Additional memory might be required to improve the performance of the DB2 GUI tools.
- Specific performance requirements can determine the amount of memory needed.
- Memory requirements are affected by the size and complexity of your database system.
- Memory requirements are affected by the extent of database activity and the number of clients accessing your system.
- On Linux, a SWAP space of at least twice as large as your RAM is recommended, but not required.

For DB2 Integrated Cluster Environment for Linux, we recommend that you use a 64-bit instance if your physical memory per partition is beyond 4 G. You can use the **`cat /proc/meminfo`** command or **`free`** command to see your memory usage information. Example 4-1 and Example 4-2 on page 132 are example output.

Example 4-1 cat /proc/meminfo output

```
# cat /proc/meminfo
        total:    used:    free:  shared: buffers:  cached:
Mem:  4222025728 1255759872 2966265856        0 49082368 976203776
Swap: 2089209856        0 2089209856
MemTotal:      4123072 kB
```

```
MemFree:       2896744 kB
MemShared:           0 kB
Buffers:         47932 kB
Cached:         953324 kB
SwapCached:          0 kB
Active:         348292 kB
ActiveAnon:     155084 kB
ActiveCache:    193208 kB
Inact_dirty:    596680 kB
Inact_laundry:  179180 kB
Inact_clean:         0 kB
Inact_target:   224828 kB
HighTotal:     3276720 kB
HighFree:      2181180 kB
LowTotal:       846352 kB
LowFree:        715564 kB
SwapTotal:     2040244 kB
SwapFree:      2040244 kB
HugePages_Total:     0
HugePages_Free:      0
Hugepagesize:     2048 kB
```

Example 4-2 free command output

```
# free
             total       used       free     shared    buffers     cached
Mem:       4123072    1226460    2896612          0      48012     953324
-/+ buffers/cache:     225124    3897948
Swap:      2040244          0    2040244
```

Communication requirements

TCP/IP is required to access remote databases. Your particular Linux distribution provides TCP/IP connectivity if it is selected during the installation. If your Linux computer is installed on an existing network and is required to use a static IP address, then information similar to that found in Table 4-2 should be collected from your network administrator.

Table 4-2 Example of TCP/IP settings

Name	Example number
Host IP Address	192.168.0.3
Subnet mask	255.255.255.0
Gateway	192.168.0.1

This information should be specified either during the installation of your Linux distribution, or after the installation is finished using your distribution's setup utility.

For DB2 Integrated Cluster Environment, we recommend you use faster network for communications between partitions, such as Gigabit Ethernet.

Additional software requirements

To set up DB2 partitioned servers, some additional software is required.

Linux packages

Table 4-3 and Table 4-4 list the package requirements for SuSE Linux and Red Hat distributions for DB2 partitioned servers (Linux) Version 8.2. The pdksh package is required for all DB2 systems. The rsh-server and nfs-utils packages are required for partitioned database systems. Both packages should be installed and running to continue with the setup of DB2 on partition database systems. To have the rsh-server running, inetd (or xinetd) must be installed and running as well.

For more information, see your Linux distribution documentation.

Table 4-3 Package requirements for SuSE Linux

Package name	Description
pdksh	Korn Shell. This package is required for partitioned database environments.
rsh-server	This package contains a set of server programs that allow users to run commands on remote computers, log in to other computers, and copy files between computers (rsh, rexec, rlogin, and rcp).
nfs-utils	Network File System support package. It allows access for local files to remote computers.

Table 4-4 Package requirements for Red Hat

Directory	Package name	Description
/System Environment/Shell	pdksh	Korn Shell. This package is required for partitioned database environments.

Directory	Package name	Description
/System Environment/Daemons	rsh-server	This package contains a set of programs that allow users to run commands on a remote computer. Required for partitioned database environments.
/System Environment/Daemons	nfs-utils	Network File System support package. It allows access for local files to remote computers.

IBM Software Development Kit (SDK) for Java

You require the appropriate level of IBM Software Development Kit (SDK) for Java to use Java-based tools like the DB2 Control Center, and to create and run Java applications, including stored procedures and user-defined functions.

If the IBM SDK for Java is required by a component being installed and the SDK for Java is not already installed, the SDK for Java will be installed if you use either the DB2 Setup wizard or a response file to install the product. The SDK for Java is not installed with the DB2 Run-Time client.

DB2 UDB V8.2 supports SDK 1.4.x. For the most up-to-date SDK for Java information, see the DB2 UDB system requirements Web page at:

http://www.ibm.com/software/data/db2/udb/sysreqs.html

For the most up-to-date Linux SDK information, see the IBM Developer Kit for Linux Web page at:

http://www.ibm.com/developerworks/java/jdk/index.html

Note: On Linux hybrid platforms, the 32-bit version of the SDK for Java is installed for you. You must install the 64-bit version of the SDK for Java yourself. See the "Installing the IBM Developer Kit for Java (UNIX)" section of the *Quick Beginnings for DB2 Servers,* GC09-4836-01.

When using Red Hat Enterprise Linux 3 on x86 hardware, Update 2 is required.

Web browser

The following browser is required to view online help: Mozilla Version 1.0 or later.

X Window System

X Window System software capable of rendering a graphical user interface is required if you want to use the DB2 Setup wizard to install DB2 Enterprise Server Edition or if you want to use any DB2 graphical tools on the Linux operating system.

Kerberos authentication software

If you plan to use Kerberos authentication, Red Hat Enterprise Linux Advanced Server 3 (x86 only) with IBM Network Authentication Service 1.4 is required.

DB2 Administration Server (DAS) requirements

A DAS must be created on each physical system for the Control Center and the Task Center to work properly.

Each DAS must be created under a user ID. If the same user ID is to be used on all physical systems, then that user ID's home directory cannot be shared (cross mounted) with the other systems. If a different user ID is used for each DAS, then the home directories of the user IDs that are used can be shared (cross mounted).

As long as a DAS is created on each system, it does not matter whether:

- A different user ID is used for each DAS, or
- The same user ID is used, and the user ID's home directory is not shared.

For easy management, we suggest that you use the same user ID on all physical systems. If you consider a high-availability configuration for your DB2 Integrated Cluster Environment, we suggest that you create a DAS on each physical machine, and do not fail over DAS between physical machines. One DAS can manage all the logical partitions on the physical server.

4.2.2 Installation planning and preparation

If your system meets all the prerequisites required by DB2 Integrated Cluster Environment, you can begin the enviroment set up. In this section, we guide you to prepare your Linux operating system for DB2 Integrated Cluster Environment installation, and give you procedures and instructions to do preparations for both Red Hat Linux and SuSE Linux. If not specified, procedures and instructions apply to both Red Hat and SuSE. We cover the following topics:

- Plan your DB2 Integrated Cluster Environment for Linux.
- Modify kernel parameters.
- Set up storage and logical volume.
- Table on page 146.
- Ensure NFS is running.
- Enable remote shell.

- Create groups and users.

Plan your DB2 Integrated Cluster Environment for Linux

Before you install DB2 Integrated Cluster Environment, a good practice is to plan it first. We recommend that you make a worksheet when you implement DB2 Integrated Cluster Environment. In this worksheet, you need to answer questions about the following:

- Hardware and network configuration
- Partition number and location
- Users and groups
- Storage layout
- Naming convention

Hardware and network configuration

At first, you must know your hardware and network configuration, which includes server configuration, software configuration, network configuration, storage and topology. Ensure that all these are meet DB2 Integrated Cluster Environment installation requirements. In case you do not know this information, please contact your system administrator to get it. We suggest that you record your network configuration on your worksheet. Table 4-5 describes part of our network configuration in our lab environment. For hardware configuration in our lab environment, please see 4.1, "Lab environment" on page 126.

Table 4-5 Network configuration

Host name	IP address	Net mask	Default gateway	Comment
db2rb01	9.26.162.46	255.255.254.0	9.26.162.1	
	172.21.0.1	255.255.255.0		Used for FCM
db2rb02	9.26.162.61	255.255.254.0	9.26.162.1	
	172.21.0.2	255.255.255.0		Used for FCM
db2rb03	9.26.162.60	255.255.254.0	9.26.162.1	
	172.21.0.3	255.255.255.0		Used for FCM
db2rb04	9.26.162.59	255.255.254.0	9.26.162.1	
	172.21.0.4	255.255.255.0		Used for FCM

Partition number and location

When you design your partition's number in your system, you should consider your hardware configuration. The most important things are CPU and memory. In general, one partition should have one CPU at least, and have enough physical memory. After you decide your partitions's number and locations, you can record your design in your worksheet. Table 4-6 is our design for configuration 1 in our lab environment.

Table 4-6 Partitions' numbers and locations

Partition number	Server	Comments
0	db2rb01	Catalog partition 1
1	db2rb02	Two logical partitions
2	db2rb02	
3	db2rb03	Two logical partitions
4	db2rb03	
5	db2rb04	Two logical partitions
6	db2rb04	

Users and groups

Three users and groups are needed to operate DB2. Table 4-7 lists users and groups we used in our lab environment. Make sure user IDs and group IDs are not used in your system. The user ID and group ID for DB2 instance owner must be the same in all physical machines.

Table 4-7 Users and groups

Usage	User name	User ID	Group name	Group ID
DB2 Instance owner	db2inst1	1001	db2iadm1	990
DB2 Instance fenced user	db2fenc1	1002	db2fadm1	991
DB2 Administration Server	dasusr1	1003	dasadm1	992

Storage layout

After you decide how many partitions you need and their locations, you need to prepare storage for these partitions. Generally speaking, you need an instance home directory to create an instance. If you use more than one machine in your cluster, it should be exported from one server and mounted on each physical server through NFS. Also, you need to prepare your storage to create databases.

When you are planning your storage layout, a good practice is obeying a naming convention for your mount points and directories. Table 4-8 shows the naming convention we used when we developed this publication. Here is explanation for this table:

<m>	A numeric number start from 1; it depicts the file system number for a specific partition.
<n>	The partition number.
<mm>	The tablespace ID.
<nn>	The container number.
<nnnn>	The partition number start from 0000.
<instance>	The name of instance.

This is only an example of a naming convention. You can develop your own naming convention for file systems and raw devices used in your system.

Table 4-8 Naming convention

Usage	Mount point	Path name
File system for Instance home	/db2home	/db2home/<instance>
File systems for DB2 LOG	/db2log<m>p<n>	/db2log<m>p<n>/<instance>/<database>
File systems for table space containers	/db2fs<m>p<n>	/db2fs<m>p<n>/<instance>/<database>/ts<mm>c<nn>
File system for DB directory	/db2db/<instance>/NODE<nnnn>	/db2db/<instance>/NODE<nnnn>

Table 4-9 depicts our storage layout for server db2rb01. It includes the Instance home directory, storages used for databases on partition 0.

Table 4-9 Storage layout for server 1

Usage	Partition	Storage type	Mount point	Comments
Instance home directory	Shared across all partitions	File system on RAID 1 array	/db2sharedhome	Local mount point; NFS mount point is /db2home/db2inst1
DB directory	0	File system on RAID 5 array	/db2db/db2inst1/NODE0000	

Usage	Partition	Storage type	Mount point	Comments
DB log	0	File system on RAID 1 array	/db2log1p0	Log files in partition 0
DB catalog table spaces	0	File system on RAID 1 array	/db2fs0p0	Only exists in catalog partition
DB regular table spaces	0	File system on RAID 5 array	/db2fs1p0	For data files in partition 0
DB temporary table spaces	0	File system on RAID 5 array	/db2fs2p0	For temporary tablespaces

Modify kernel parameters

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

Note: When you install DB2 UDB V8, it is not a prerequisite to modify Linux kernel parameters to increase IPC limits. DB2 UDB automatically raises the IPC limits when necessary. If you still want to modify these parameters, please follow the instructions below.

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) = 262144
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 explain how to update the *kernel.shmmax, kernel.sem,* and *kernel.msgmni* parameters on Red Hat and set them after each reboot.

1. Log on 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 the 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 would like 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 on 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. 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 run level.
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
```

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.

Set up storage and logical volume

How to setup storage and logical volume is a big topic, and it depends on which type of device you use. It exceeds the scope of this redbook. In our lab environment, we used a IBM DS4400 storage server and EXP700 expansion unit. For more information of working with IBM Disk Storage on Linux, please refer to:

- *Implementing Linux with IBM Disk Storage,* SG24-6261
- *Linux with xSeries and FAStT: Essentials*, SG24-7026
- *IBM TotalStorage: FAStT Best Practices Guide*, REDP-3690

DB2 can utilize three types of disk storage:

- Raw disk
- Logical Volume Manager (LVM) managed logical volume
- File system

Raw disk

Raw disk is either a whole physical disk or a physical disk partition without file system. A possible path is either /dev/sda or /dev/sda1. DB2 can utilize raw disk as DMS table space containers. When you implement a failover environment, raw disk can be identified by all physically attached servers with the same path. There is no need for start or stop methods to make the disk visible to the host; a monitor method is also unnecessary. So it is suitable to adopt raw disk in failover environment. Another advantage of using raw disk is that possible time-consuming **`fsck`** is never required. Normally if you use a raw disk as your DMS table space container, you can gain some performance improvement. A drawback for raw disk is that you cannot dynamically change the size.

Logical Volume Manager (LVM)

There are currently several logical volume managers available for Linux. We discuss the LVM currently available as part of most Linux 2.4-based distributions.

Here are several concepts that we need to clarify:

- Physical Volume (PV): The physical disk drive
- Physical Extent (PE): Partition that is defined as a basic storage unit of the Logical Volume Manager
- Volume Group (VG): A logical storage pool of one or more PVs
- Logical Volume (LV): Logical unit that can be used as raw device or creating fils system

LVM has many advantages:

- Performance: LVM can stripe data across physical disks in one VG; it increases I/O throughput.
- Redundancy: LVM can make a copy from one disk to another disk. You can also use mirroring function. It enables a server to continue running even after HDD failure.
- Flexibility: You can resize your VG and LV dynamically. It provides you with great flexibility to manage your storage.

For more information on LVM, please refer to the documents in your Linux distribution.

Note: When you implement a DB2 instance with failover capability for DB2 Integrated Cluster Environment, it is important to remember that, currently, LVM is not cluster aware. The implication is that, as root, it is very easy to cause an unrecoverable data loss in cluster environments. In shared disk environments (the most common being shared SCSI for small two-node clusters and shared fiber channel for larger clusters) more than one machine has physical access to a set of disks. It is critical that LVM be shut down on all nodes other than the node from which any LVM administration is performed. It is also critical that, at most, only one node have a file system mounted on a given logical volume (at any given time).

File system

There are many file systems supported by Linux currently, such as ext2, ext3, xfs, reiserfs, and so on. When implementing DB2 Integrated Cluster Environment, you can choose any file system to use. We recommend that you use the ext3 file system in Red Hat Linux, and use reiserfs on SuSE Linux.

For our lab environment, Table 4-10 shows the storage configuration for EXP700. Each EXP700 has a similar configuration. We use Linux partitions to manage the RAID 1 array, which is used for instance home directory, database log directory, and catalog table space containers. We use LVM to manage the other three RAID 5 arrays. One VG is created on each RAID 5 array; one VG contains three LVs. As we recommended, we use the ext3 file system on Red Hat Linux, and use reiserfs on SuSE Linux.

Table 4-10 Storage allocation for EXP700

EXP700	LVM	File system	Usage
RAID 1 (2x 36.4GB)	No	ext3 on Red Hat Linux, reiserfs on SuSE Linux	Instance home directory, log directories, catalog tablespace containers
RAID 5 (3x 36.4G)	1 VG, 3 LVs		Data tablespace containers, temporary tablespace containers, DB directory
RAID 5 (3x 36.4G)	1 VG, 3 LVs		
RAID 5 (3x 36.4G)	1 VG, 3 LVs		
RAID 5 (3x 36.4G)	1 VG, 3 LVs		

If you use raw disk as your storage, use Linux partition tools, such as **`fdisk`** or **`parted`**, to partition your hard disk. In SuSE Linux, you can use the GUI tool **`yast2`** to manage your partitions. For detailed command usage refer to your Linux distribution documents.

If you want to use file systems, you can create a file system over a disk partition with the **mkfs** command. For example, you have a partition called /dev/sda2, in Red Hat Linux, you can create an ext3 file system with the command **mkfs -t ext3 /dev/sda2**; in SuSE Linux, you can create a reiserfs with the command **mkfs -t reiserfs /dev/sda2**. You can also use mkfs.*fstype* command to create different file systems; for example, you can use **mkfs.ext3 /dev/sda2** and **mkfs.reiserfs /dev/sda2** to implement the above task separately.

If you try to use LVM to manage your storage, here is an example. We have a RAID 5 array, which is identified by the Linux operating system as /dev/sdc. We will create a VG on it, and then create three LV and file systems.

1. Create PV over /dev/sdc. Before you can use the physical disk or physical disk partition in your LVM, you need to initialize it as a PV. Log on as root, and enter the command **pvcreate /dev/sdc** to initialize /dev/sdc as a PV. You can use the **pvdisplay /dev/sdc** command to display PV information.
2. Create a VG datavg1 over /dev/sdc. Log in as root, then enter the following command in your command-line window (terminal):

   ```
   vgcreate datavg1 /dev/sdc
   ```

   After that, you can use **vgdisplay** to display this VG's information:

   ```
   #vgdisplay datavg1
   --- Volume group ---
   VG Name               datavg1
   VG Access             read/write
   VG Status             available/resizable
   VG #                  0
   MAX LV                256
   Cur LV                3
   Open LV               3
   MAX LV Size           255.99 GB
   Max PV                256
   Cur PV                1
   Act PV                1
   VG Size               67.72 GB
   PE Size               4 MB
   Total PE              17336
   Alloc PE / Size       0 / 0 GB
   Free  PE / Size 17336 / 69344 MB
   VG UUID               OAtlcX-YlUm-iMpw-tUe7-Z1nm-nSG5-4oWcWu
   ```

   You can only use a VG when it is active on your system. You can use **vgchange** to change VG status. To activate a VG, use the command **vgchange -a y <vgname>**; to deactivate a VG, use the **vgchange -a n <vgname>** command.

3. Create three LVs named usrdata, tempdata, and dbdir. The sizes are 60 GB, 5 GB and 2 GB separately. Here are the commands:

```
#lvcreate -n usrdata  datavg1 -L 60G
#lvcreate -n tempdata  datavg1 -L 5G
#lvcreate -n dbdir  datavg1 -L 2G
```

   You can use the command **lvdisplay** to display the information of a LV:

```
# lvdisplay /dev/datavg1/usrdata
--- Logical volume ---
LV Name                /dev/datavg1/usrdata
VG Name                datavg1
LV Write Access        read/write
LV Status              available
LV #                   1
# open                 1
LV Size                60 GB
Current LE             15360
Allocated LE           15360
Allocation             next free
Read ahead sectors     1024
Block device           58:0
```

4. Create three file systems. If Linux operating system is Red Hat, we use the **mkfs.ext3** command to create a *ext3* file system; if it is SuSE Linux, then we use the **mkfs.reiserfs** command to create a *reiserfs* file system.

   In Red Hat Linux, enter the following commands to create the ext3 file system:

```
#mkfs.ext3 /dev/datavg1/usrdata
#mkfs.ext3 /dev/datavg1/tempdata
#mkfs.ext3 /dev/datavg1/dbdir
```

   In SuSE Linux, use the following commands to create the ext3 file system:

```
#mkfs.reiserfs /dev/datavg1/usrdata
#mkfs.reiserfs /dev/datavg1/tempdata
#mkfs.reiserfs /dev/datavg1/dbdir
```

Now the file system is ready for use. The only thing you need to do is mount these file systems. To mount these file systems, you need to do the following:

1. Create mount points for these file systems. In our example, according to our naming convention, we use the mount points listed in Table 4-11.

Table 4-11 Mount points

Device name (file system)	Mount point
/dev/datavg1/usrdata	/db2fs1p1

Device name (file system)	Mount point
/dev/datavg1/tempdata	/db2fs2p1
/dev/datavg1/dbdir	/db2db/db2inst1/NODE0001

In your command line, enter the following command:

```
#mkdir -p /db2fs1p1
#mkdir -p /db2fs2p1
#mkdir -p /db2db/db2inst1/NODE0001
```

2. Add entries to the /etc/fstab file. In Red Hat Linux, we add the following entries:

```
/dev/datavg1/usrdata /db2fs1p1 ext3 noauto 1 0
/dev/datavg1/tempdata /db2fs2p1 ext3 noauto 1 0
/dev/datavg1/dbdir /db2db/db2inst1/NODE0001 ext3 noauto 1 0
```

   In SuSE Linux, you need to replace *ext3* with *reiserfs*. The option *noauto* in the forth column means that the Linux operating system will not mount this file system when it issues a **mount -a** command in the startup scripts. You must mount it yourself. It is very important to specify this option when you implement a DB2 failover environment with IBM Tivoli System Automation for Linux or other high-availability software. You should mount and unmount these file systems through HA software. If you do not implement a DB2 instance with failover capability, then you can edit your /etc/fstab file and replace the *noauto* option with the *auto* option.

3. Now you are ready to mount these file systems. You can use the **mount** command to mount these file systems:

```
#mount /db2fs1p1
#mount /db2fs2p1
#mount /db2db/db2inst1/NODE0001
```

   If you want to unmount these file systems, use the **umount** command.

To manage a large DB2 cluster, we suggest that you keep all the planning documents and use shell scripts to create your storage and file system, so you can have a records of your operation; it is useful for your maintenance job.

Set up TCP/IP

You can set up TCP/IP settings when you install Linux operating system, or you can change it later. For Red Hat Linux, you can use the **redhat-config-network** command to start the Network Configuration GUI tool to configure your network settings. For SuSE Linux, use the **yast2** command to start the YaST Control Center to set up your TCP/IP address and host name.

To check your current host name, use the **hostname** command:

```
# hostname
DB2RB01.torolab.ibm.com
```

To check your IP configuration, use the **ifconfig -a** command.

When you prepare your DB2 Integrated Cluster Environment, add all physical machines in the Linux cluster to your /etc/hosts/ file, so you can use the host name or alias to connect them between these machines. In our example, the /etc/hosts file is like this:

```
127.0.0.1       localhost
9.26.162.46     db2rb01 db2rb01.torolab.ibm.com
9.26.162.61     db2rb02 db2rb02.torolab.ibm.com
9.26.162.60     db2rb03 db2rb03.torolab.ibm.com
9.26.162.59     db2rb04 db2rb04.torolab.ibm.com

172.21.0.1      db2rb01_ge
172.21.0.2      db2rb02_ge
172.21.0.3      db2rb03_ge
172.21.0.4      db2rb04_ge
```

Ensure NFS is running

When you install a partitioned DB2 on the Linux operating system, Network File System (NFS) must be running on each computer.

To verify whether NFS is running, use the following command:

```
showmount -e hostname
```

The *hostname* is the machine you want to check. If you run it without *hostname,* it will check your local machine. If NFS is not running, you will receive a message similar to this:

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

To start your NFS server, use the command **/etc/init.d/nfs start**.

Once you have verified that NFS is running on each system, check for the specific processes required by DB2. The required process is *rpc.statd*. You can use the **ps -ef |grep rpc.statd** command to check for this process:

```
# ps -ef |grep rpc.statd
rpcuser   1759     1  0 Jul09 ?        00:00:01 rpc.statd
```

You must have a file system that is available to all machines that will participate in your partitioned database system. This file system will be used as the instance home directory.

For configurations that use more than one machine for a single database instance, NFS is used to share this file system. Typically, one machine in a cluster is used to export the file system, and the remaining machines in the cluster mount the NFS file system from this machine. The machine that exports the file system has the file system mounted locally. If you want to implement a DB2 instance with failover capability, then this file system must reside on shared storage so that other machines can access this file system when this machine fails.

Here is the procedure for creating the instance owner directory and exporting it through the NFS server:

1. Create a local file system. Either use disk partition or LVM. Please refer to "Set up storage and logical volume" on page 141. In our example, we use the mount point /db2sharedhome for mounting the local file system; use /db2home/db2inst1 as our NFS mount point across all the machines.
2. To automatically export an NFS file system on Linux at boot time, add an entry to the /etc/exports file. Be sure to include all of the host names participating in the cluster. Also, ensure that each machine in the cluster has root authority on the exported file system by using the *no_root_squash* option. In our example, we use db2rb01 as our NFS server.

   The /etc/exports is an ASCII file that contains the following type of information:

   ```
   /db2sharedhome db2rb*.torolab.ibm.com(rw,sync,no_root_squash)
   ```

   To export the NFS directory, run the command:

   ```
   /usr/sbin/exportfs -r
   ```

Now you are ready to mount your DB2 instance home directory through NFS. You can mount the DB2 instance home directory by adding entries to /etc/fstab, or you can use *autofs* to mount your NFS home directory. When you implement a DB2 instance with failover capability, it is better to use *autofs* to mount your NFS home directory, because you do not need to umount and mount your home directory through start and stop scripts used in HA software. *autofs* will mount your home directory dynamically when you access your DB2 instance home directory. In our lab environment, we use *autofs* to mount the DB2 instance home directory automatically.

Here is the procedure to mount DB2 instance home directory through /etc/fstab:

1. On each of the remaining machines in the cluster, add an entry to the /etc/fstab file to NFS mount the file system automatically at boot time. When you specify the mount point options, ensure that the file system is mounted at boot time, is read-write, is mounted hard, includes the bg (background)

option, and that setuid programs can be run properly. Here is example entry in our /etc/fstab:

```
db2rb01:/db2sharedhome /db2home/db2inst1 nfs
rw,timeo=600,retrans=2,hard,nointr,noac,bg,nolock,suid 0 0
```

2. Create mount point /db2home/db2inst1 on each machine.
3. NFS mount the exported file system on each machine in the cluster by entering the following command:

   ```
   mount /db2home/db2inst1
   ```

   If the mount command fails, use the **showmount** command to check the status of the NFS server. For example:

   ```
   showmount -e db2rb01
   ```

   This **showmount** command should list the file systems that are exported from the machine named db2rb01. If this command fails, the NFS server may not have been started. Run the following command as root on the NFS server to start the server manually:

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

If you want to mount your DB2 instance home directory through autofs, please follow up with the following procedures:

1. Make sure your Linux operating system supports *autofs*. You need the *autofs* package installed in your Linux operation system. To check whether you installed the *autofs* package, enter the command:

   ```
   rpm -qa |grep autofs
   ```

   If you have not install it, please install the autofs package provided in your Linux installation CD-ROMs, or you can download the RPM package from the Internet. Now all latest Linux distributions and kernels except Mandrake Linux support *autofs*. Mandrake Linux distributions ship a default kernel with *supermount*, a similar software as *autofs*.
2. Make directory /db2home to be used by *autofs* on each machine.

   Note: /db2home is used by autofs exclusively. You should not make other directories under it.

3. Edit configuration files on each machine. Open the /etc/auto.master file and add the following entries:

   ```
   /db2home /etc/auto.db2home  --timeout=0
   ```

   Each entry in auto.master has three fields. The first field is the mount point. The second field is the location of the map file, and the third field is optional. The third field can contain information such as a time-out value.

Edit the map file /etc/auto.db2home, and add the following entries:

```
db2inst1        -rw,timeo=600,retrans=2,hard,nointr,noac,bg,nolock,suid
db2rb01:/db2sharedhome
```

The first field in /etc/auto.db2home is the name of the *db2inst1* subdirectory. This directory is created dynamically by automount. It should not actually exist on the client machine. The second field contains mount options such as rw for read and write access. The third field is the location of the NFS export including the hostname and directory.

Note: Use the proper options as used in our example; otherwise, you may encounter problems such as file permission and so on.

4. Start the *autofs* service on each machine, and enter the command:

   ```
   /etc/init.d/autofs start
   ```

   Now you can test your configuration by entering the command **`cd /db2home/db2inst1`**. You should be able to enter your DB2 instance home directory now.

   To view the active mount points, you can use command:

   ```
   /etc/init.d/autofs status
   ```

   If you modified your configuration file when autofs is running, please use the following command to tell the automount daemon to reload the configuration files.

   ```
   /etc/init.d/autofs reload
   ```

You can configure the NFS service to start automatically at system boot time, assuming that we need to start that NFS service automatically when run level is 3 and 5. First, use **`chkconfig --list nfs`** to list your NFS service status:

```
# chkconfig --list nfs
nfs             0:off   1:off   2:off   3:off   4:off   5:off   6:off
```

We can see that the NFS service will not start automatically in all run levels. We use the following command to configure the NFS service to start automatically when the run level is 3 and 5:

```
# chkconfig --level 35 nfs on
# chkconfig --list nfs
nfs             0:off   1:off   2:off   3:on    4:off   5:on    6:off
```

Now the NFS service will start automatically at run levels 3 and 5.

Before you create your instance, ensure that the following steps were successful:

- On a single machine in the cluster, you have created a file system to be used as the instance and home directory.
- If you have a configuration that uses more than one machine for a single database instance, you have exported this file system using NFS.
- You have mounted the exported file system on each machine in the cluster.

Enable remote shell

DB2 Integrated Cluster Environment needs a remote shell to communicate among partitions. Before you create your partitioned DB2 instance, make sure the remote shell service is enabled on all machines. Check if *xinetd* is enabled first:

```
# /etc/init.d/xinetd status
xinetd (pid 2175) is running...
```

If xinetd is not running, start it:

```
# /etc/init.d/xinetd start
```

Same as for the NFS service—you can make a remote shell service to start automatically at system boot time:

```
#chkconfig --level 35 xinetd on
```

Now we need to check whether remote shell service is enabled on the *xinetd* service. Open the /etc/xinetd.d/rsh file and check whether it includes the following line:

```
disable = yes
```

If included, comment out the line or change *yes* to *no*.

Create groups and users

Before you create your instance, you need to create users and groups for your instance. For example, we use three users and groups in our lab environment (Table 4-7 on page 137). You can create these users and groups by following this procedure:

1. Log on to the primary computer.
2. Create a group for the instance owner (for example, db2iadm1), the fenced user who will execute the user-defined function (UDF) or stored procedures (for example, db2fadm1), and the Administration Server (for example, dasadm1) by entering the following commands:

```
groupadd -g 990 db2iadm1
groupadd -g 991 db2fadm1
groupadd -g 992 dasadm1
```

3. Create a user that belongs to each group that you created in the previous step using the following commands. The home directory for db2inst1 will be the instance home directory that you previously created and shared (db2sharedhome). Make sure all servers mounted the instance home directory to /db2home/db2inst1 before you create users.

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

4. Set an initial password for each user that you created by entering the following commands:

   ```
   passwd db2inst1
   passwd db2fenc1
   passwd dasusr1
   ```

5. Log out.
6. Log on to the primary computer as each user (db2inst1, db2fenc1, and dasusr1) to test the ID and password created.
7. Log out.
8. Create the exact same user and group accounts on each computer that will participate in your partition database system. For our example, perform this task on db2rb02, db2rb03 and db2rb04.
9. Add trusted remote users to the execute command through **rsh** without a password. In a partitioned DB2 environment, each database partition must have the authority to execute remote commands on all the participating machines without a password. We must add the DB2 instance owner and hostname as trusted remote users across all participating machines. To do this, update the .rhosts file of the instance owner. The format is:

   ```
   hostname instance-owner-name
   ```

   Because the DB2 instance home directory is a NFS shared directory, we only need to edit it once. In our example, we add the following entries to the .rhosts file:

   ```
   db2rb01.torolab.ibm.com  db2inst1
   db2rb02.torolab.ibm.com  db2inst1
   db2rb03.torolab.ibm.com  db2inst1
   db2rb04.torolab.ibm.com  db2inst1
   ```

   Then we need to change the .rhosts file permission to 600:

   ```
   chmod 600 ~/db2inst1/.rhosts
   ```

Now you can test whether we can execute the command through **`rsh`** without a password among participating machines:

```
$ rsh db2rb02 date
Tue Jul 20 11:32:17 EDT 2004
```

Important: Make sure the .rhosts file is updated correctly; otherwise you may encounter a communication error when you try to start your instance.

4.3 DB2 partitioned database setup

This section describes the general implementation steps for a DB2 partitioned database. The implementation steps are summarized below.

- Preparations for DB2 product installation
- DB2 product installation
- Preparations for instance creation
- Instance creation
- Partition configuration
- Preparations for database creation
- Database creation
- Data loading

4.3.1 Considerations on DB2 installation

DB2 provides two installation methods. Before you begin installation, you must choose one from the below:

- **`db2setup`** program
- **`db2_install`** script

`db2setup` is a GUI-based installer that guides you through installation steps. With **`db2setup`** you can:

- Choose which DB2 components to install.
- Create an administration server with its owner user and group.
- Create an instance with its owner user and group.
- Configure miscellaneous monitoring settings.

Though **`db2setup`** is an interactive installer, you can run it silently with a pre-created response file. This feature is useful when you have to install DB2 on multiple computers. To prepare a response file, you just run **`db2setup`**, directing it to create a response file, and go through installation steps.

On the other hand, **`db2_install`** is a simple shell script that just installs RPM packages. The script simply installs all the RPM packages contained in the DB2 installation medium. It is useful when you just need to install the product, but not create an administration server or instances.

If you plan to manage your computers with IBM Cluster Systems Management (CSM), the following CSM commands help you install DB2 on multiple computers.

- **`dcp`** distributes files to managed computers.
- **`dsh`** runs commands on managed computers.

You can use **`dcp`** to distribute a response file to computers and **`dsh`** to run db2setup remotely with the response file on all computers. If you plan to install DB2 with the **`db2_install`** script, you just have to run it with **`dsh`** on all computers.

> **Note:** Because installation steps using **`db2_install`** are fairly simple and easy, the following sections focus on installation steps using **`db2setup`** and response files. Also, tasks described there must be done as root user unless otherwise noted.

4.3.2 Preparations for DB2 UDB product installation

Before installation, a few preparations are needed.

- Ensure remote shell service availability.

  Make sure that remote shell service is enabled on all computers on which you plan to install DB2 UDB. See “Enable remote shell” on page 151 for details.

- Ensure NFS is running.

  Make sure that NFS is running on each computer. See “Ensure NFS is running” on page 147.

- Prepare DB2 UDB installation media.

  Prepare a DB2 UDB installation medium for each computer. One of the easiest ways to do this is to mount an installation medium on one computer and share it with other computers via NFS. Or you can simply copy the whole content of a medium onto each computer, if you have enough temporary space on each computer.

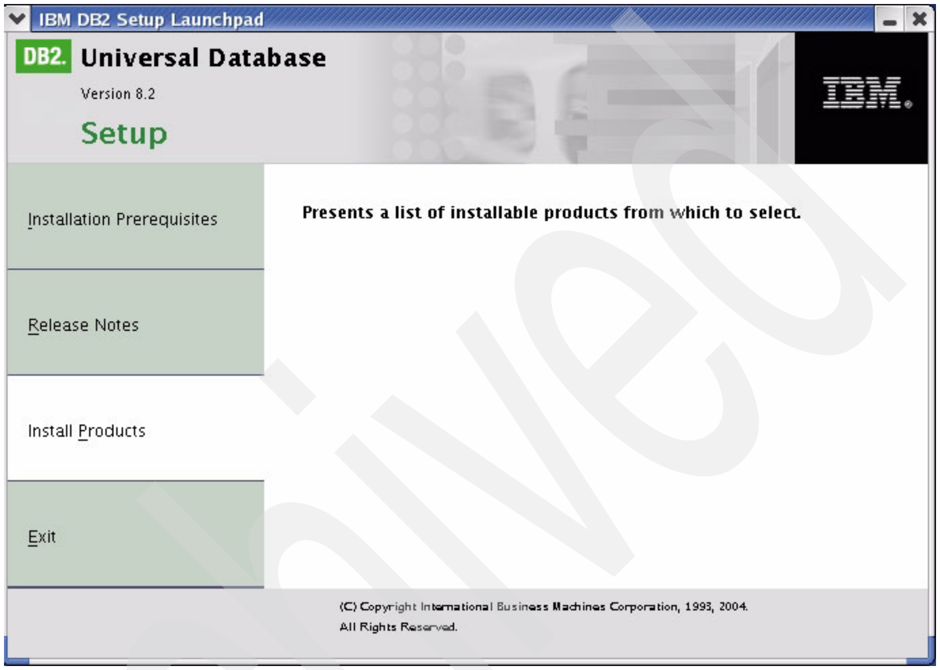
4.3.3 DB2 UDB product installation

To install DB2, follow the steps below.

1. Launch the **`db2setup`** program.

Choose one computer as the primary computer where you will create an instance. Log in to the primary computer as root and launch the **`db2setup`** program. db2setup is located in the root directory of the installation medium.

Choose **Install Products**, as shown in Figure 4-4.

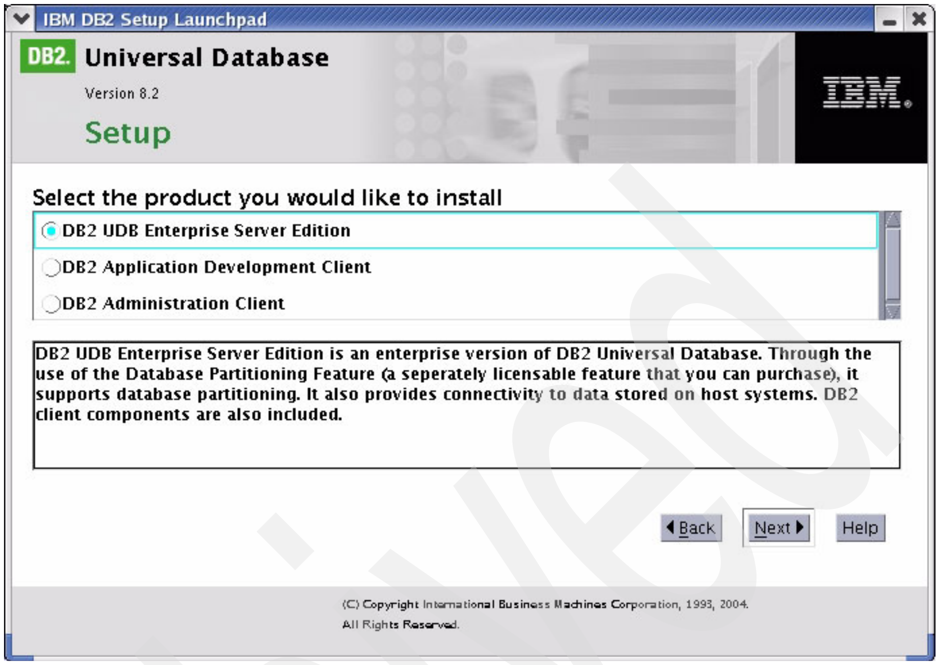


Presents a list of installable products from which to select.

(C) Copyright International Business Machines Corporation, 1993, 2004.
All Rights Reserved.

Figure 4-4 IBM DB2 Setup Launchpad

In the next dialog you are asked to choose a product to install. Choose **DB2 UDB Enterprise Server Edition**, as shown in Figure 4-5 on page 156.


IBM DB2 Setup Launchpad

DB2. Universal Database

Version 8.2

Setup

Select the product you would like to install

(●) DB2 UDB Enterprise Server Edition
( ) DB2 Application Development Client
( ) DB2 Administration Client

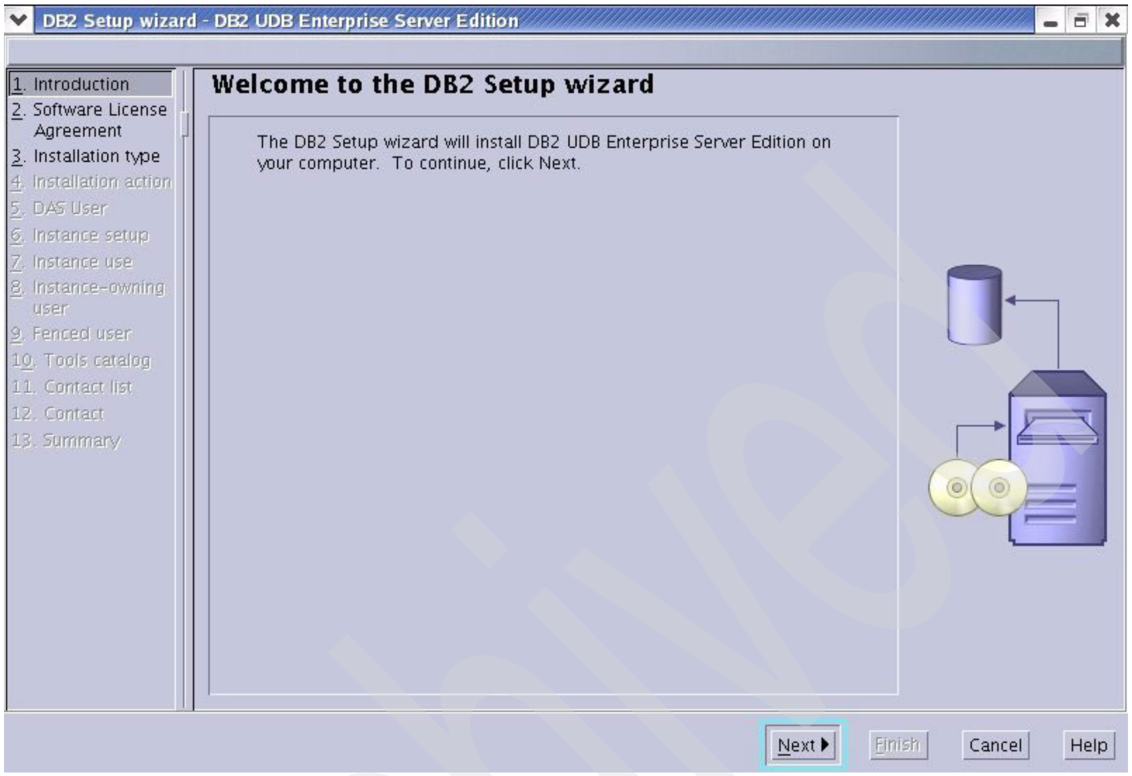
DB2 UDB Enterprise Server Edition is an enterprise version of DB2 Universal Database. Through the use of the Database Partitioning Feature (a seperately licensable feature that you can purchase), it supports database partitioning. It also provides connectivity to data stored on host systems. DB2 client components are also included.

(C) Copyright International Business Machines Corporation, 1993, 2004.
All Rights Reserved.

Figure 4-5 Product selection

2. Install DB2 and create a response file.

   Then DB2 Setup wizard begins installation, as shown in Figure 4-6 on page 157.


DB2 Setup wizard - DB2 UDB Enterprise Server Edition

1. Introduction
2. Software License Agreement
3. Installation type
4. Installation action
5. DAS User
6. Instance setup
7. Instance use
8. Instance-owning user
9. Fenced user
10. Tools catalog
11. Contact list
12. Contact
13. Summary

Welcome to the DB2 Setup wizard

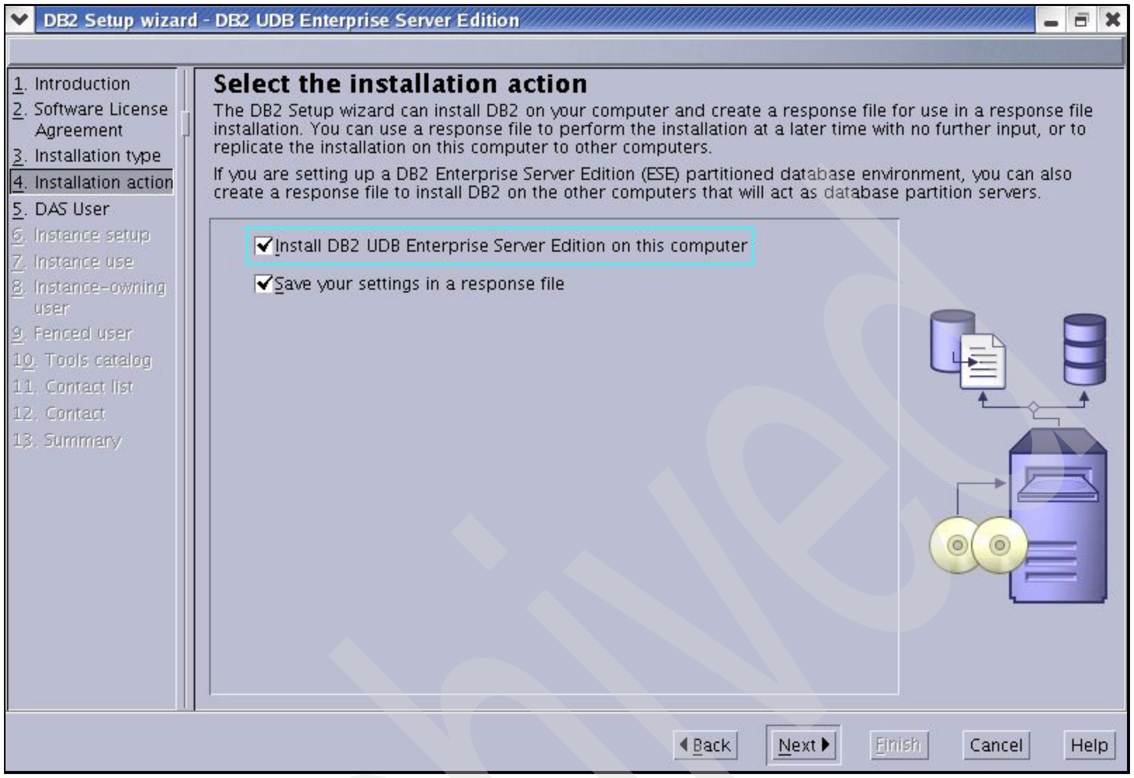
The DB2 Setup wizard will install DB2 UDB Enterprise Server Edition on your computer. To continue, click Next.

Figure 4-6 DB2 Setup wizard

From here on, you can follow the dialog and choose the options you like, except a few steps below.

1. The install action.

   After you choose the install type, you must specify the install action. Be sure to check the “Save your settings in a response file” checkbox, as well as the “Install DB2 UDB Enterprise Server on this computer” checkbox, as shown in Figure 4-7 on page 158.


DB2 Setup wizard - DB2 UDB Enterprise Server Edition

Select the installation action

The DB2 Setup wizard can install DB2 on your computer and create a response file for use in a response file installation. You can use a response file to perform the installation at a later time with no further input, or to replicate the installation on this computer to other computers.

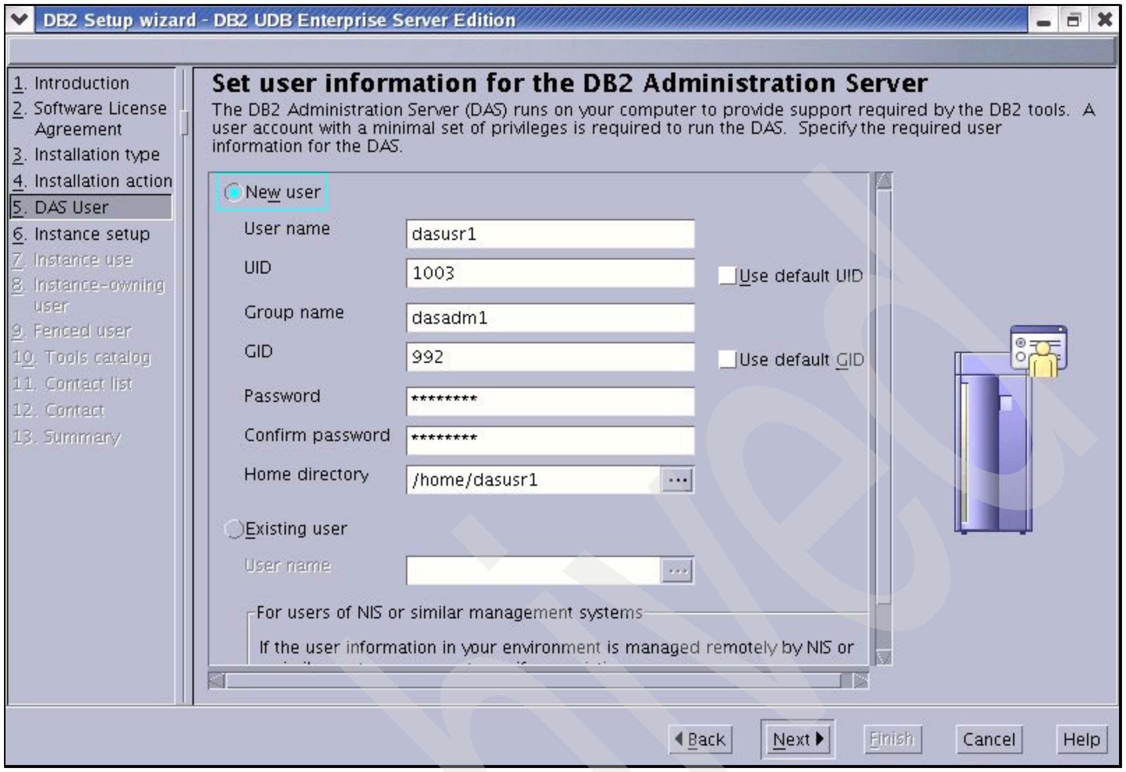
If you are setting up a DB2 Enterprise Server Edition (ESE) partitioned database environment, you can also create a response file to install DB2 on the other computers that will act as database partition servers.

- [✓] Install DB2 UDB Enterprise Server Edition on this computer
- [✓] Save your settings in a response file

Figure 4-7 Selecting the install action

2. User information for the DB2 Administration server.

   During installation, the DB2 Administration server is created. You can assign an existing user and group ID, or create a new one for its owner. When you create a new user and group, it is safer to assign IDs to them explicitly, as shown in Figure 4-8 on page 159, so that the owner, user, and group IDs are consistent across all the computers. This is not required but recommended for the sake of simplicity.


DB2 Setup wizard - DB2 UDB Enterprise Server Edition

Set user information for the DB2 Administration Server

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

[✓ New user]

User name	dasusr1	
UID	1003	Use default UID
Group name	dasadm1	
GID	992	Use default GID
Password	*********	
Confirm password	*********	
Home directory	/home/dasusr1	

Existing user

User name

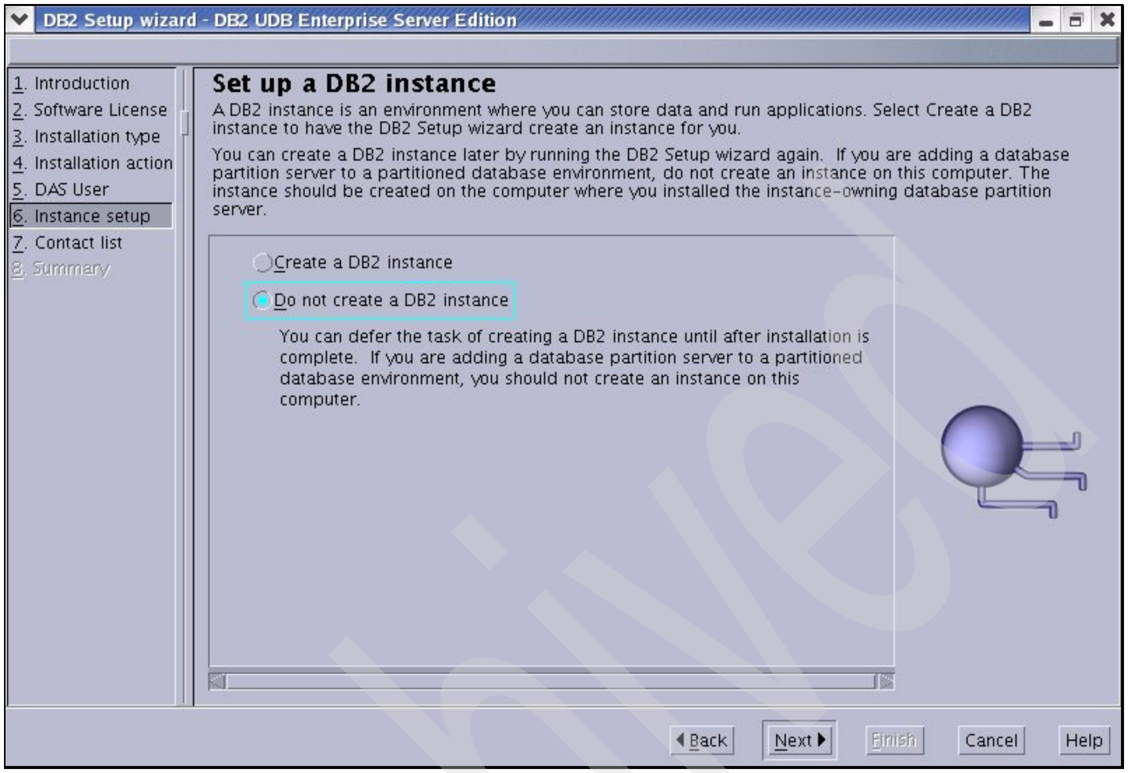
For users of NIS or similar management systems

If the user information in your environment is managed remotely by NIS or

Figure 4-8 Assigning a user ID for the administration server owner explicitly

3. Set up DB2 instances.

   You should not create an instance during installation. Tell the wizard not to create an instance, as shown in Figure 4-9 on page 160.


DB2 Setup wizard - DB2 UDB Enterprise Server Edition

Set up a DB2 instance

A DB2 instance is an environment where you can store data and run applications. Select Create a DB2 instance to have the DB2 Setup wizard create an instance for you.

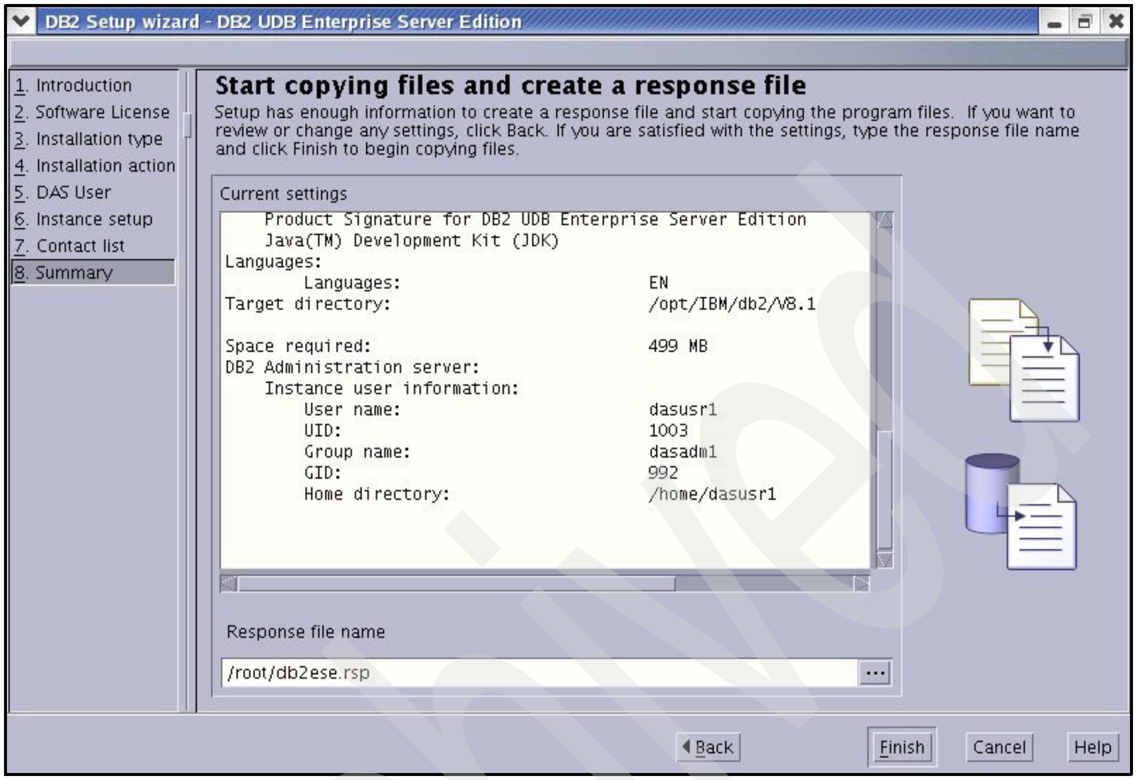
You can create a DB2 instance later by running the DB2 Setup wizard again. If you are adding a database partition server to a partitioned database environment, do not create an instance on this computer. The instance should be created on the computer where you installed the instance-owning database partition server.

Create a DB2 instance [✓ Do not create a DB2 instance]

You can defer the task of creating a DB2 instance until after installation is complete. If you are adding a database partition server to a partitioned database environment, you should not create an instance on this computer.

Figure 4-9 No DB2 instance creation

After you feed **db2setup** enough information, **db2setup** tells you it is ready to begin copying files, as shown in Figure 4-10 on page 161.



Start copying files and create a response file

Setup has enough information to create a response file and start copying the program files. If you want to review or change any settings, click Back. If you are satisfied with the settings, type the response file name and click Finish to begin copying files.

Current settings

```
    Product Signature for DB2 UDB Enterprise Server Edition
    Java(TM) Development Kit (JDK)
Languages:
        Languages:                                    EN
Target directory:                                     /opt/IBM/db2/V8.1

Space required:                                       499 MB
DB2 Administration server:
    Instance user information:
        User name:                                    dasusr1
        UID:                                          1003
        Group name:                                   dasadm1
        GID:                                          992
        Home directory:                               /home/dasusr1
```

Response file name

/root/db2ese.rsp

Figure 4-10 Ready to begin installation

Do not forget to specify the location to save the response file. Click **Finish** to begin copying files.

Install DB2 servers on other participating computers

Use the response file you have created to install DB2 on other participating computers. Distribute the response file to the other computers and invoke db2setup with the -r option.

```
# ./db2setup -r /root/db2ese.rsp
DBI1191I db2setup is installing and configuring DB2 according to
          the response file provided. Please wait.
```

Repeat this procedure for all computers where DB2 is to be installed.

If you have enabled remote shell for root user, you can accomplish these tasks from the primary computer, without logging into each computer. Here is an example from our lab environment:

```
# rcp /root/db2ese.rsp db2rb02:/root/db2ese.rsp
# rsh db2rb02 /mnt/cdrom/db2setup -r /root/db2ese.rsp
```

In this example, we assume that:

- db2rb02 is the hostname of the computer to install DB2.
- The installation medium is accessible under /mnt/cdrom.
- The response file is saved as /root/db2ese.rsp.

If you are managing computers with CSM, installation can be done more easily with **dcp** and **dsh** commands. In this case you just create a response file on one of the computers, possibly on the management server, without actually installing, and distribute it to all managed computers with **dcp**. Then you can invoke **db2setup** on all managed computers at once with **dsh**, as below.

```
# dcp -a db2ese.rsp /root
# dsh -a /mnt/cdrom/db2setup -r /root/db2ese.rsp
```

Apply the latest FixPak if available

At the time of writing, no fixpak is available for DB2 UDB V8.2. But you should always check fixpak availability prior to installation. Check the URL below for FixPak availability.

```
http://www.ibm.com/software/data/db2/udb/suppolrt.html
```

If any fixpak is available, we strongly recommend that you apply the fixpak before you create instances.

4.3.4 Preparation for instance creation

Before creating an instance, a few preparations are needed.

1. Prepare user IDs and group IDs.

   Before you create an instance, you must have the instance owner user and fenced user prepared, along with their groups. They must have identical ID across all computers. See "Create groups and users" on page 151.

2. Prepare instance owner's home directory.

   All participating computers must share the instance owner's home directory. Typically the home directory is created in a dedicated file system. This file system is mounted on one of the participants and exported to the others via NFS. See "Ensure NFS is running" on page 147.

Note: If you plan to configure your instance with failover capability, two cautions must be taken to make NFS service highly available. First, you must place the home directory in a shared storage, not in a local disk, so that either computer can take over the home directory for NFS service. Second, every computer should include the same entries for the home directory in /etc/exports and /etc/fstab for the home directory.

3. Verify that all related host names can be resolved.

   Verify that all related host names and switch names can be resolved on all participating computers using DNS or the /etc/hosts file.

4. Prepare the .rhosts file.

   In a partitioned database environment, each database partition must have the authority to execute remote commands on all the other participating computers. Update the .rhosts file of the instance owner to accept remote command invocations (see "Enable remote shell" on page 151).

4.3.5 Instance creation

To create an instance, follow the steps below.

1. Create an instance.

   Only the root user can create an instance. The instance creation command syntax is as below.

   ```
   /opt/IBM/db2/V8.1/instance/db2icrt -s ESE -u fenceuser owneruser
   ```

   fenceuser is a fence user name and *owneruser* is an instance owner user name.

   Switch to the owner user and test if you can start the instance.

   ```
   # su - db2inst1
   # db2start
   2004-07-17 17.51.30     0   0   SQL1063N  DB2START processing was
   successful.
   SQL1063N  DB2START processing was successful.
   ```

   In this example, we assume the owner is db2inst1. Do not forget to stop the instance for now.

2. Enable client connections.

   To make a DB2 instance accept connections from database clients over TCIP/IP, you must update a registry variable and a database manager configuration parameter.

   ```
   # db2set DB2COMM=TCPIP
   ```

```
# db2 update dbm cfg using SVCENAME servicename
```

servicename is either a service name in /etc/services or a port number. We recommend that you register a service name in /etc/services and use it. Make sure that all participating computers have an entry for this service name in /etc/services.

3. Enable communication between database partitions.

   When an instance is created, service names for Fast Communication Manager (FCM) are automatically defined in /etc/services. For our lab environment, the service names are defined as below.

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

   Each entry has a name like *DB2_instancename_number* and corresponds to each participating computer. In this example, ports are assigned for up to four computers. If you have prepared more computers than that, modify these entries so that each computer is assigned one dedicated port. Note that the service name for the first computer does not include a number, and the name for the last includes "END" instead of a number.

4.3.6 Node configuration

The node configuration file *db2nodes.cfg* contains configuration information that tells DB2 which computer hosts a database partition in an instance. db2nodes.cfg is located in the *sqllib* directory under the instance owner's home directory.

Each line in db2nodes.cfg corresponds to one partition in an instance. By default, the node configuration file has only one entry, as below:

```
0 hostname 0
```

The first *0* is a database partition number, and *hostname* is the hostname of the computer that hosts this partition. The second *0* specifies a logical port number.

You must update the db2nodes.cfg file according to your configuration. For example, in our lab environment, we deploy four partitions on four computers named db2rb01, db2rb02, db2rb03, and db2rb04, respectively. So db2nodes.cfg should look like below:

```
0 db2rb01 0
1 db2rb02 0
2 db2rb03 0
3 db2rb04 0
```

If you want to deploy more than one partition on one computer, assign a unique logical port to each partition. For example, if we deploy four partitions on two computers, db2rb01 and db2rb02, where each computer hosts two partitions, db2nodes.cfg should look like below:

```
0 db2rb01 0
1 db2rb01 1
2 db2rb02 0
3 db2rb02 1
```

If each participating computer has a dedicated network interface for FCM, you can specify the interface information in db2nodes.cfg. For example, if each computer in the above example has an alternate network interface named *hostname*_ge, db2nodes.cfg should include the names shown below:

```
0 db2rb01 0 db2rb01_ge
1 db2rb01 1 db2rb01_ge
2 db2rb02 0 db2rb02_ge
3 db2rb02 1 db2rb02_ge
```

After you update db2nodeds.cfg, verify that you can start the instance.

```
# su - db2inst1
# db2start
2004-07-17 20.19.36     0   0   SQL1063N  DB2START processing was successful.
2004-07-17 20.19.37     2   0   SQL1063N  DB2START processing was successful.
2004-07-17 20.19.37     1   0   SQL1063N  DB2START processing was successful.
2004-07-17 20.20.01     3   0   SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.
```

Do not forget to stop the instance for now.

4.3.7 Preparation for database directory (optional)

This step is optional unless you plan to configure your instance with failover capability. If you would like to create your database elsewhere other than a default place, you must place it in shared storage. The tricky part is that you must dedicate one file system per partition and carefully choose its mount point.

When you create a database on /db2db in the instance db2inst1, which has four partitions, the whole database directory structure is built as shown in Figure 4-11 on page 166.

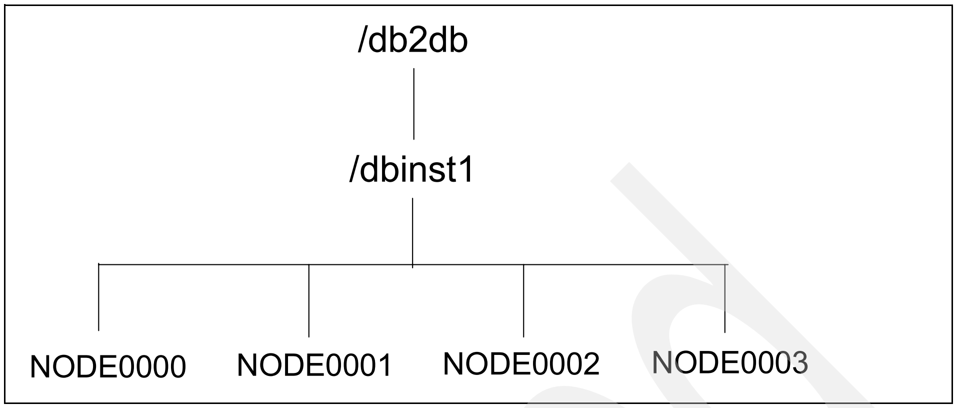


Figure 4-11 Database directory structure

NODE*xxxx* is the actual directory where database files for the partition *xxxx* are stored.

To configure your instance with failover capability, create the directory hierarchy similar to the one in Figure 4-11 on each computer. And prepare one dedicated file system for one database partition. On each server, mount the corresponding file system on *path-to-database-directory/instancename/*NODE*xxxx*, where *xxxx* is a partition number. For every participating computer to take over any failed partition, add entries in /etc/fstab so that each computer can mount any partition's database directory. For our lab environment, the entries for our database partitions in /etc/fstab should look as below.

```
/dev/datavg0/dbdir /db2db/db2inst1/NODE0000 ext3 noauto 1 0
/dev/datavg1/dbdir /db2db/db2inst1/NODE0001 ext3 noauto 1 0
/dev/datavg2/dbdir /db2db/db2inst1/NODE0002 ext3 noauto 1 0
/dev/datavg3/dbdir /db2db/db2inst1/NODE0003 ext3 noauto 1 0
```

4.3.8 Database creation

To create a database, follow the steps below.

1. Create a database.

   Create a database with the CREATE DATABASE command. The command syntax is:

   CREATE DATABASE *databasename* ON *path-to-databasedirectory*

   Add any database creation options if necessary. For example:

   ```
   CREATE DATABASE TPCH
   USING CODESET IBM-1252
   TERRITORY US
   ```

```
COLLATE USING SYSTEM
CATALOG TABLESPACE
  MANAGED BY SYSTEM
  USING ('/db2fs0p0/db2inst2/tpch/ts00')
  EXTENTSIZE 32 PREFETCHSIZE 96
USER TABLESPACE
  MANAGED BY SYSTEM
  USING ('/db2fs1p $N/db2inst2/tpch/ts01c00')
  EXTENTSIZE 32 PREFETCHSIZE 96
TEMPORARY TABLESPACE
  MANAGED BY SYSTEM
  USING ('/db2fs2p $N/db2inst2/tpch/4ktmpc00')
  EXTENTSIZE 32 PREFETCHSIZE 96
WITH "Redbook TPCH";
```

2. Create node groups.

   By default, there are three node groups in a database:

   – IBMDEFAULTGROUP contains all partitions and is used for user tablespaces.

   – IBMTEMPGROUP contains all partitions and is used for temporary tablespaces.

   – IBMCATGROUP contains only a catalog partition and is used for a catalog tablespace only.

   If default node groups are not suitable for your database, create node groups with the CREATE NODEGROUP command.

   ```
   CREATE NODEGROUP nodegroupname ON NODES(partitionlist)
   ```

   nodegroupname is the name for a nodegroup and *partitionlist* is a comma-separated list of the partition numbers this node group contains.

3. Create tablespaces.

   Create a tablespace with the CREATE TABLESPACE command. Do not forget to specify the IN clause if you want to use node groups other than default node groups.

   ```
   CREATE TABLESPACE tablespace IN nodegroupname ....
   ```

   The IN clause comes just after the table space name, followed by other options. You can also create buffer pools and associate them with table spaces if necessary.

   Note: If you plan to configure your database with a failover requirement, place table space containers in shared storage, so that any participation computer can access them when it takes over a failed partition.

For example:

```
CONNECT TO TPCH;

CREATE REGULAR TABLESPACE ts01 IN DATABASE PARTITION GROUP pg_all
   PAGESIZE 32K
   MANAGED BY SYSTEM
      USING ('/db2fs1p0/db2inst1/tpch/ts01c00') ON DBPARTITIONNUM(0)
      USING ('/db2fs1p1/db2inst1/tpch/ts01c00') ON DBPARTITIONNUM(1)
   BUFFERPOOL BP32K
   EXTENTSIZE 32
   PREFETCHSIZE 64;

CREATE REGULAR TABLESPACE ts02 IN DATABASE PARTITION GROUP pg_zero
   PAGESIZE 4K
   MANAGED BY SYSTEM
      USING ('/db2fs1p0/db2inst1/tpch/ts02c00') ON DBPARTITIONNUM(0)
      USING ('/db2fs1p1/db2inst1/tpch/ts02c00') ON DBPARTITIONNUM(1)
   BUFFERPOOL IBMDEFAULTBP
   EXTENTSIZE 32
   PREFETCHSIZE 64;

CONNECT RESET;
```

4. Create tables.

   To create tables, use the CREATE TABLE command with their partitioning keys specified.

   `CREATE TABLE` *`tablename`* `(`*`column-definition`*`) IN` *`tablespacename`* `PARTITIONING KEY (`*`partitioningkeys`*`)`

   tablename and *tablespacename* are the names of the table and the table space, respectively. *partitioningkeys* is a comma-separated list of the column names used as the portioning keys for this table. For example:

```
CREATE TABLE TPCH.PART
   (P_PARTKEY     INTEGER NOT NULL ,
    P_NAME        VARCHAR(55) NOT NULL ,
    P_MFGR        CHAR(25) NOT NULL ,
    P_BRAND       CHAR(10) NOT NULL ,s
    P_TYPE        VARCHAR(25) NOT NULL ,
    P_SIZE        INTEGER NOT NULL ,
    P_CONTAINER   CHAR(10) NOT NULL ,
    P_RETAILPRICE DECIMAL(15,2) NOT NULL ,
    P_COMMENT     VARCHAR(23) NOT NULL )
   PARTITIONING KEY ("P_PARTKEY")
   USING HASHING  IN "TS01" ;
```

4.3.9 Data loading

To populate created tables with data, use *import* or *load*. While import simply repeats an insert transaction, load writes page images directly into the table space. To feed a large amount of data into tables, load is preferable.

To load data into a partitioned database, use the LOAD command.

```
LOAD FROM filename OF format INSERT INTO tablename
```

filename is the path to a file that contains data to be loaded. Three options available for *format* are ASC (non-delimited ASCII format), DEL (delimited ASCII format), and IXF (interchange exchange format). See the *Command Reference,* SC09-4828-01, for these format details. With INSERT options, data in a file is added to the specified table. Use the REPLACE option if you would like to replace whole data in a table.

```
LOAD FROM filename OF format REPLACE INTO tablename
```

Data files should reside on the database server. To load data in a file that resides on a client computer where the LOAD command is typed, use the CLIENT option.

```
LOAD CLIENT FROM filename OF format INSERT INTO tablename
```

4.4 Post installation tasks - Initial tuning

In this section, we guide you to do initial tuning for your database system. Always remember that DB2 performance is limited by your hardware configuration. You cannot improve your performance if your system is bound to hardware. This section only describes how to do initial tuning with DB2 *Configuration Advisor*. For more information about DB2 performance tuning, please refer to DB2 manual *Administration Guide: Performance,* SC09-4821-01, which can be downloaded from web page at:

http://www.ibm.com/cgi-bin/db2www/data/db2/udb/winos2unix/support/v8pubs.d2w/en_main

There is also a very good article “Best practices for tuning DB2 UDB v8.1 and its databases” on the IBM developerworks Web site:

http://www.ibm.com/developerworks/db2/library/techarticle/dm-0404mcarthur/index.html

After you create the database, you can use Configuration Advisor to help you tune your DB2 system. DB2 provides the Configuration Advisor GUI tool that allows you to configure a database and instance based on the answers to a series of questions. You can also invoke Configuration Advisor through the command-line processor, too. We describe these two methods separately. In addition, we introduce some commonly used DB2 registry and environment variables related to performance.

4.4.1 Tuning your DB2 through Configuration Advisor GUI tool

Once your database is created, we recommend that you use Configuration Advisor to check the database and instance parameter settings. The default settings frequently are not enough. In our lab environment, we created a database TPCH. We do initial configuration for the TPCH database and instance using Configuration Advisor.

1. Start your control center. From the menu, click **Tools** → **Wizards**. The Wizards window appears (Figure 4-13); then select **Configuration Advisor**, and click **OK.**

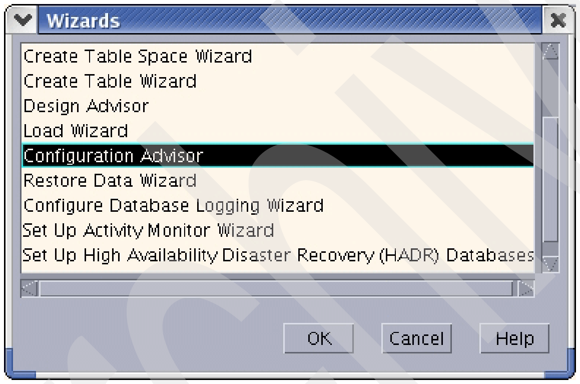
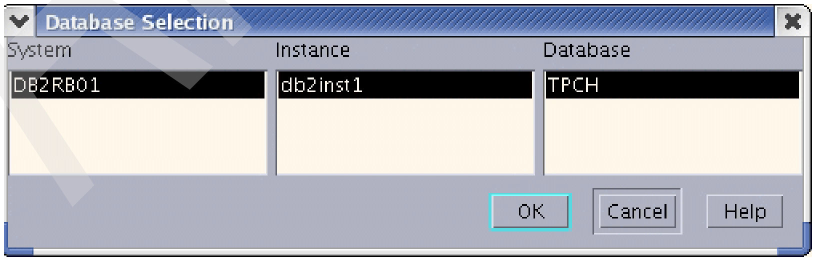

Wizards
Create Table Space Wizard
Create Table Wizard
Design Advisor
Load Wizard
Configuration Advisor
Restore Data Wizard
Configure Database Logging Wizard
Set Up Activity Monitor Wizard
Set Up High Availability Disaster Recovery (HADR) Databases
OK
Cancel
Help

Figure 4-12 Wizards window

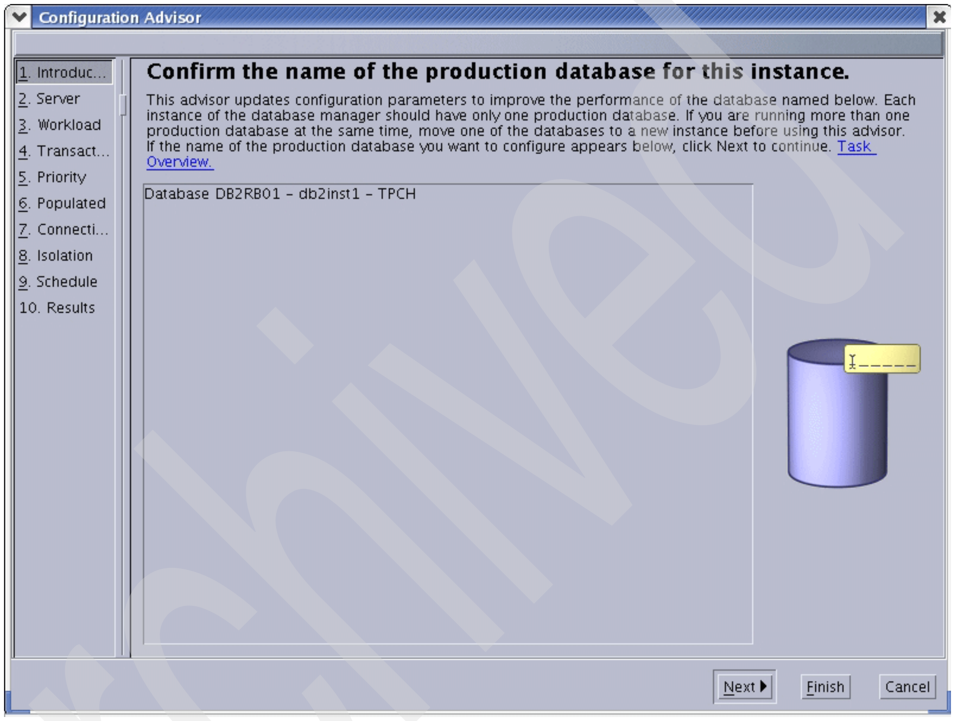
2. The Database Selection window appears (Figure 4-14 on page 171). Select the system, instance and database in the list box, then click **OK**.



System	Instance	Database
DB2RB01	db2inst1	TPCH

Figure 4-13 Select database

3. The Configuration Advisor window appears (Figure 4-15 on page 172). The Configuration Advisor needs some information from you. In the next few steps, the tool will ask for your input. If you use the GUI equivalent command AUTOCONFIGURE, these input are specified as the command options. We list these options in Table 4-12 on page 181 when we discuss the AUTOCONFIGURE command.



Confirm the name of the production database for this instance.

This advisor updates configuration parameters to improve the performance of the database named below. Each instance of the database manager should have only one production database. If you are running more than one production database at the same time, move one of the databases to a new instance before using this advisor. If the name of the production database you want to configure appears below, click Next to continue. Task Overview.

Database DB2RB01 - db2inst1 - TPCH

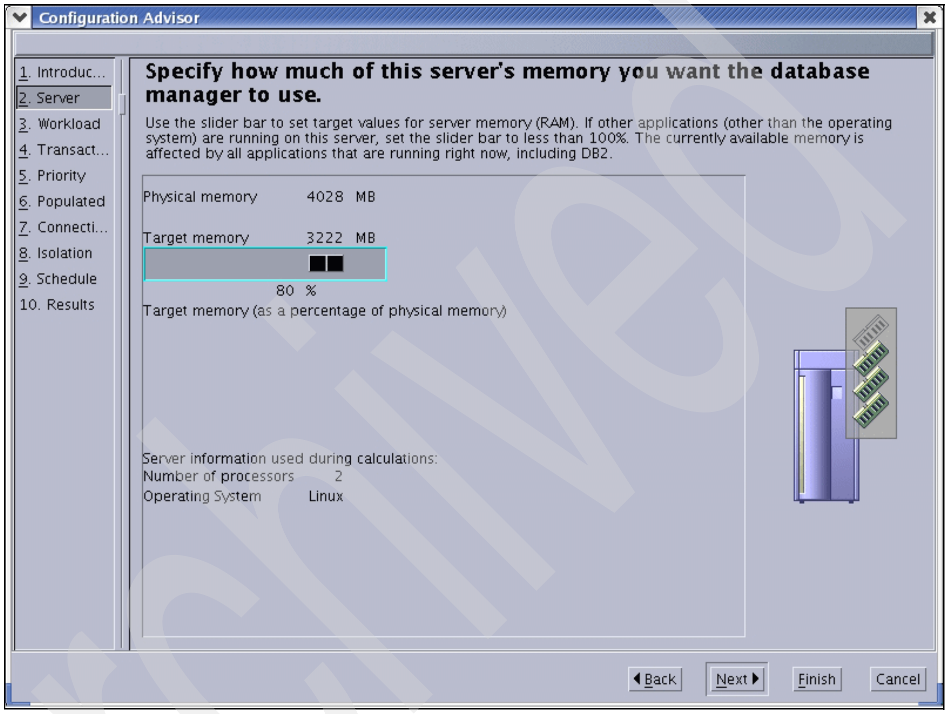
Figure 4-14 Configuration Advisor window

4. Specify the percent of memory used for this database. This is equal to the num_percent parameter in the AUTOCONFIGURE command. When you specify percent of memory, consider the following:

   a. On systems with multiple logical partitions, the mem_percent parameter refers to the percentage of memory that is to be used by all logical partitions. For example, if DB2 uses 80 percent of the memory on the system, specify 80 percent regardless of the number of logical partitions.

   b. The database configuration recommendations made will be adjusted for one logical partition. This command makes configuration recommendations for the currently connected database, assuming that the database is the only active database on the system. If more than one database is active on the system, adjust the mem_percent parameter to reflect the current database's share of memory. For example, if DB2 uses 80 percent of the system's memory and there are two active databases on the system that should share the resources equally, specify 40 percent (80 percent divided by two databases) for the parameter mem_percent.

In our lab environment, we have a dedicated database server with only one TPCH database, so we use 80 percent memory for this database.



Specify how much of this server's memory you want the database manager to use.

Use the slider bar to set target values for server memory (RAM). If other applications (other than the operating system) are running on this server, set the slider bar to less than 100%. The currently available memory is affected by all applications that are running right now, including DB2.

Physical memory 4028 MB

Target memory 3222 MB

80 %

Target memory (as a percentage of physical memory)

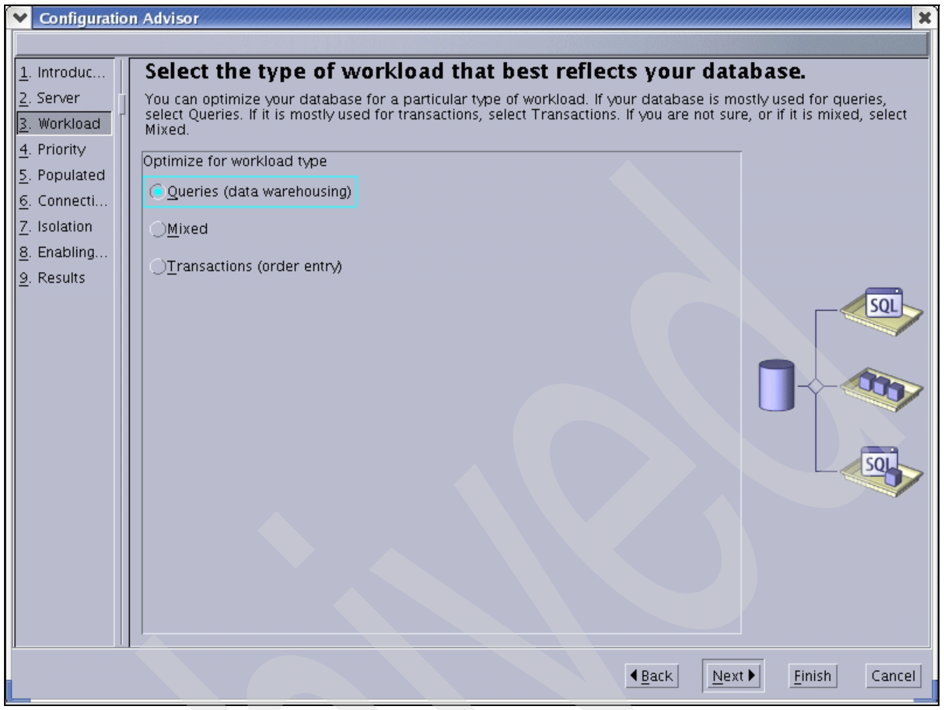
Server information used during calculations:
Number of processors 2
Operating System Linux

Figure 4-15 Specify percent of memory used for TPCH database

5. Specify the workload type for your database (Figure 4-16 on page 173). There are three options:

   – Queries (data warehousing)
   – Mixed
   – Transactions (order entry)

   If your system is a typical Online Analytical Processing (OLAP) system, most of the operations are complex queries; you select **Queries (data warehousing)**. If your system is a typical Online Transactional Processing (OLTP) system, you select **Transactions (order entry)**; If your system is neither a pure OLTP nor a pure OLAP system, it has some functions of both OLAP and OLTP, then you choose **Mixed**. In our example, it is a OLAP system, so we choose **Queries (data warehousing)**.


Configuration Advisor

1. Introduc...
2. Server
3. Workload
4. Priority
5. Populated
6. Connecti...
7. Isolation
8. Enabling...
9. Results

Select the type of workload that best reflects your database.

You can optimize your database for a particular type of workload. If your database is mostly used for queries, select Queries. If it is mostly used for transactions, select Transactions. If you are not sure, or if it is mixed, select Mixed.

Optimize for workload type

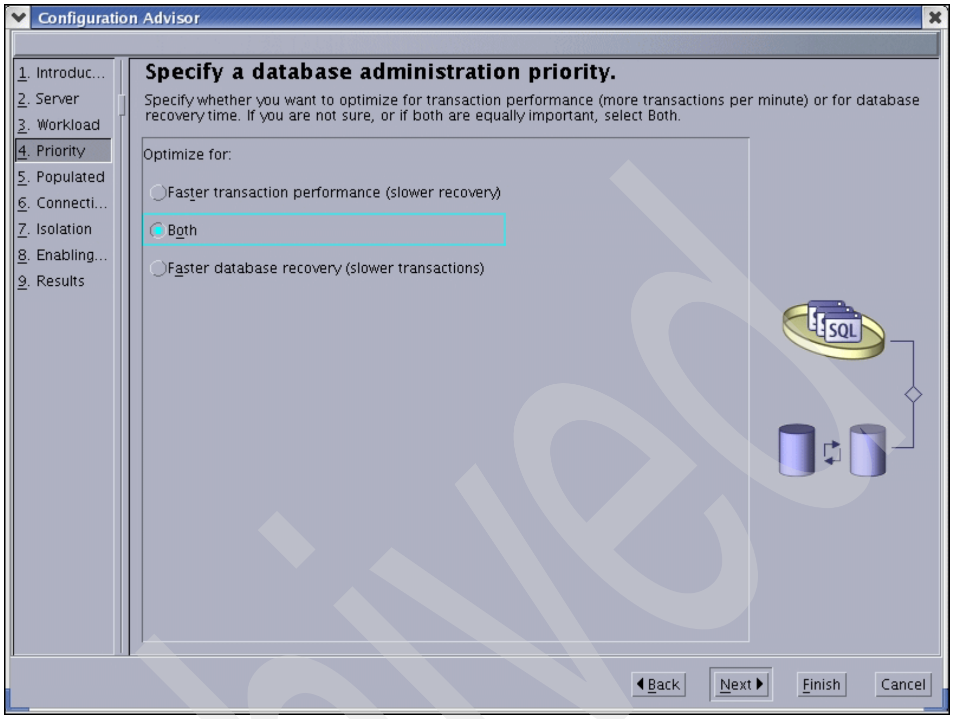
[✓ Queries (data warehousing)]
Mixed
Transactions (order entry)

Figure 4-16 Specify workload type for your database

6. Specify a database administration priority (Figure 4-17 on page 174). There are three options:

   – Faster transaction performance (slower recovery)
   – Both
   – Faster database recovery (slower transaction)

   Select the proper administration priority according to your requirements. This question mainly affects the database configuration parameters such as LOGBUFSZ and MINCOMMIT. How to select it depends on your system's requirements. We select **Both** in our example.



Specify a database administration priority.

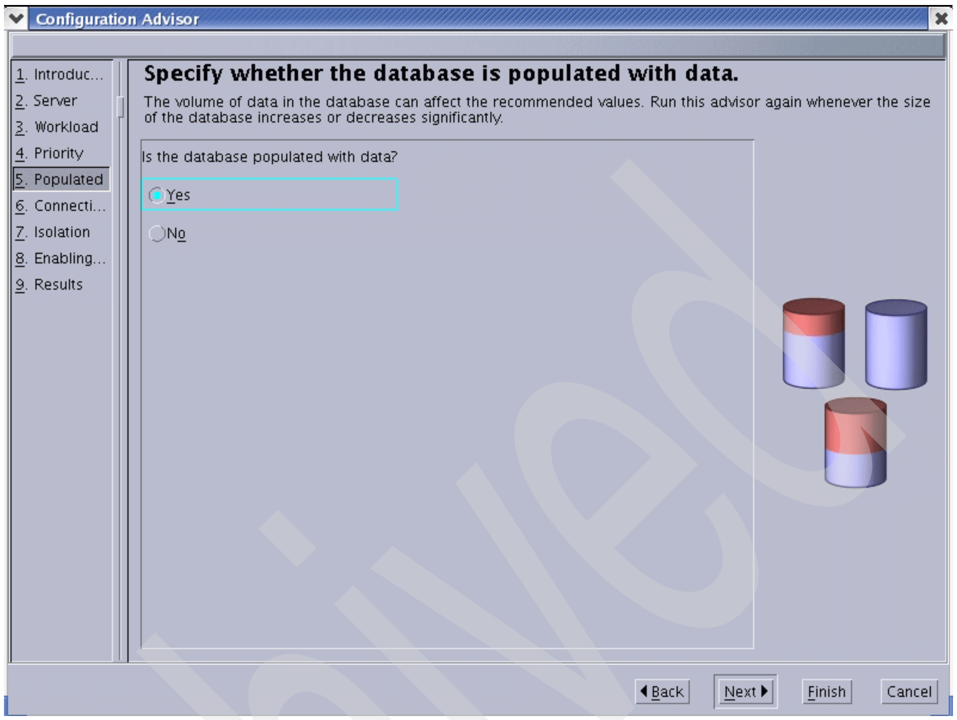
Specify whether you want to optimize for transaction performance (more transactions per minute) or for database recovery time. If you are not sure, or if both are equally important, select Both.

Optimize for:

Faster transaction performance (slower recovery) [✓ Both] Faster database recovery (slower transactions)

Figure 4-17 Specify a database administration priority

7. Specify whether your database is populated with data (Figure 4-18 on page 175). You should select Yes or No according to whether your database is populated with data or not. In our example, we have loaded data into database TPCH, so we select **Yes**.



Specify whether the database is populated with data.

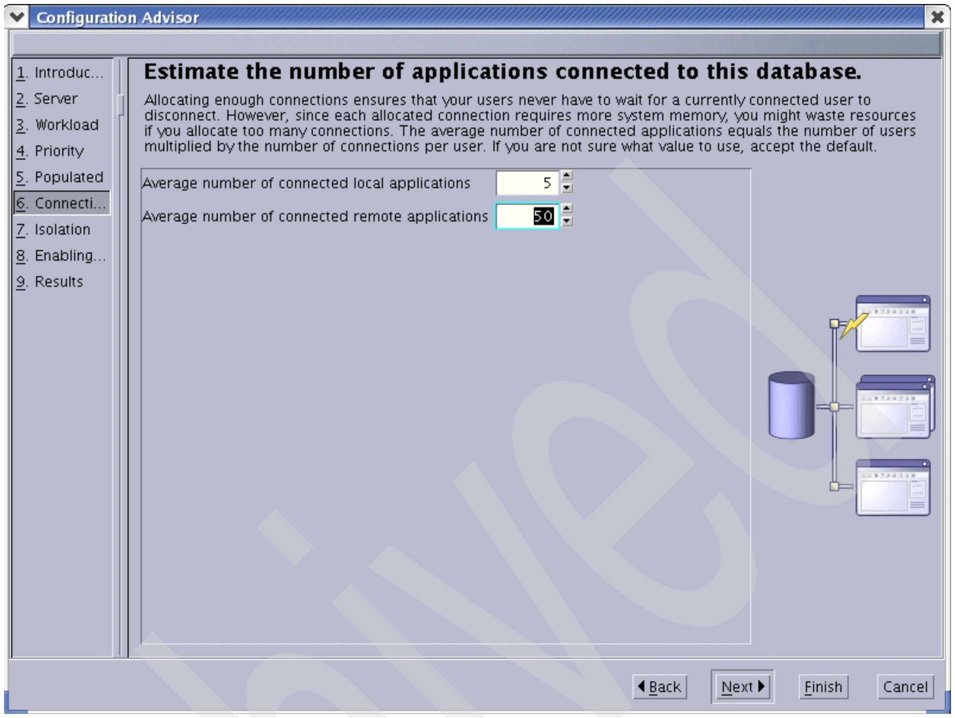
The volume of data in the database can affect the recommended values. Run this advisor again whenever the size of the database increases or decreases significantly.

Is the database populated with data?

[✓ Yes] No

Figure 4-18 Specify whether your database is populated with data

8. Specify the number of applications connected to this database (Figure 4-19 on page 176). There are two types of applications; the first type is local applications, which connect to the database through IPC; the other is remote applications, which connect to the database through TCP/IP. The rule to distinguish whether your application is a remote application and local application is by database catalog type it connects to. If the database catalog type is REMOTE, then it is a remote application. If the database catalog type is INDIRECT, then it is a local application. In our example, we select five local applications and 50 remote applications.



Estimate the number of applications connected to this database.

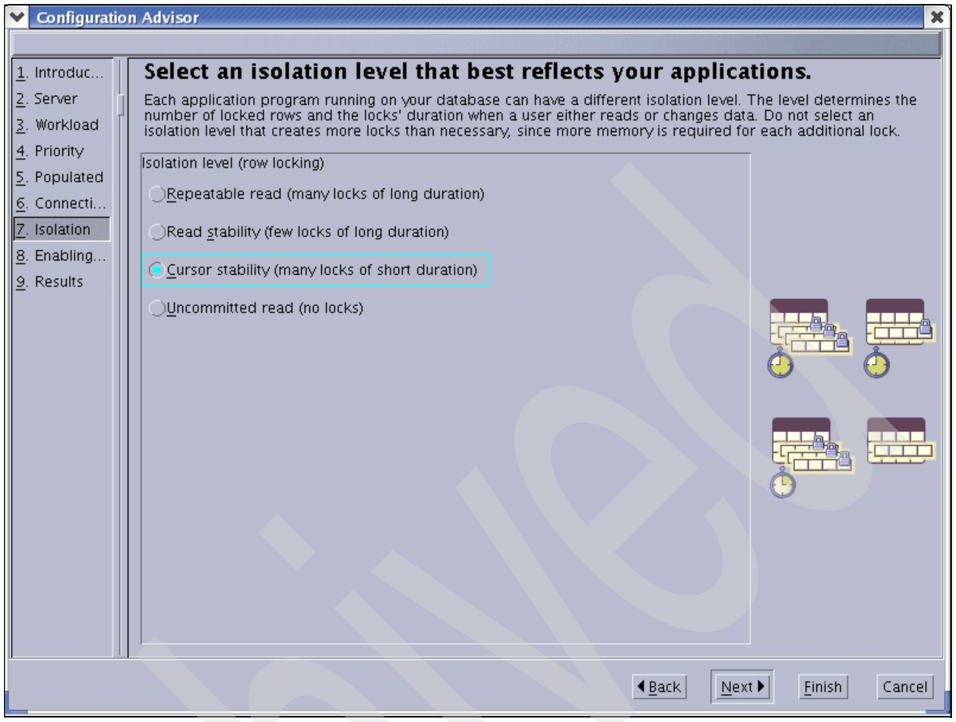
Allocating enough connections ensures that your users never have to wait for a currently connected user to disconnect. However, since each allocated connection requires more system memory, you might waste resources if you allocate too many connections. The average number of connected applications equals the number of users multiplied by the number of connections per user. If you are not sure what value to use, accept the default.

Average number of connected local applications: 5

Average number of connected remote applications: 50

Figure 4-19 Specify connections to your database

9. Specify the isolation level (Figure 4-20 on page 177). There are four isolation levels in DB2. They are Repeatable Read (RR), Read Stability (RS), Cursor Stability (CS), and Uncommitted Read (UR). Isolation level affects DB2's lock mechanism. Use the CS isolation level unless you have special requirements. We use CS isolation level in our example.



Select an isolation level that best reflects your applications.

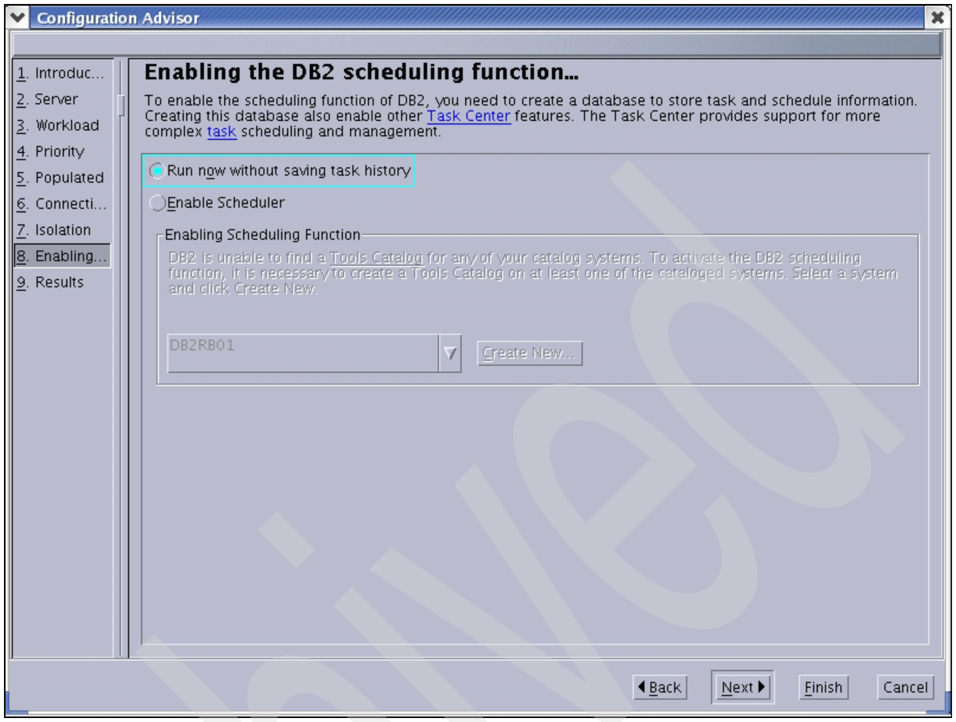
Each application program running on your database can have a different isolation level. The level determines the number of locked rows and the locks' duration when a user either reads or changes data. Do not select an isolation level that creates more locks than necessary, since more memory is required for each additional lock.

Isolation level (row locking)

- Repeatable read (many locks of long duration)
- Read stability (few locks of long duration)
- [✓ Cursor stability (many locks of short duration)]
- Uncommitted read (no locks)

Figure 4-20 Specify isolation level

10. Specify whether you want to run immediately or save and schedule this task. If you have not created a tools catalog, you may see the following window (Figure 4-21 on page 178). You can enable it in this window by selecting the Enable scheduler radio button. We suggest that you enable the scheduler if you have not done it before. After you enable the scheduler, then you will see a window similar to Figure 4-22 on page 179. Select the “Create this as a task in the Task Center” radio button, then select **Run system** and **Scheduler system**; select **Save task only**, then click **Next**.



Enabling the DB2 scheduling function...

To enable the scheduling function of DB2, you need to create a database to store task and schedule information. Creating this database also enable other Task Center features. The Task Center provides support for more complex task scheduling and management.

[✓ Run now without saving task history]

Enable Scheduler

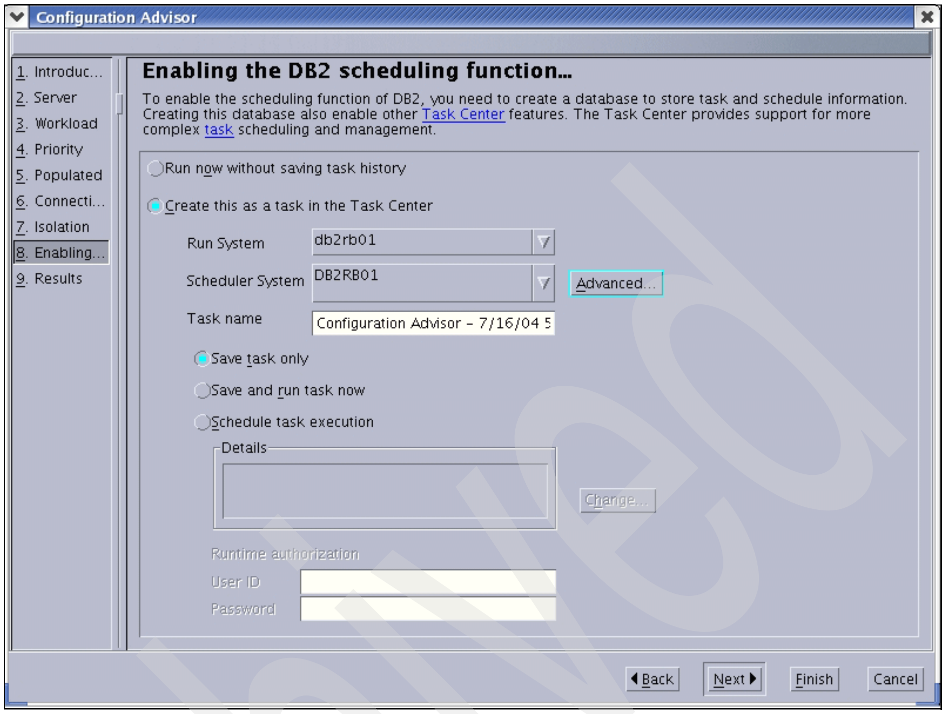
Enabling Scheduling Function

DB2 is unable to find a Tools Catalog for any of your catalog systems. To activate the DB2 scheduling function, it is necessary to create a Tools Catalog on at least one of the cataloged systems. Select a system and click Create New.

DB2RB01

Create New...

Figure 4-21 Enabling the DB2 scheduling function window


Configuration Advisor

1. Introduc…
2. Server
3. Workload
4. Priority
5. Populated
6. Connecti…
7. Isolation
8. Enabling…
9. Results

Enabling the DB2 scheduling function…

To enable the scheduling function of DB2, you need to create a database to store task and schedule information. Creating this database also enable other Task Center features. The Task Center provides support for more complex task scheduling and management.

( ) Run now without saving task history
(●) Create this as a task in the Task Center

Run System: db2rb01
Scheduler System: DB2RB01
Task name: Configuration Advisor - 7/16/04 5

(●) Save task only
( ) Save and run task now
( ) Schedule task execution

Details

Runtime authorization
User ID:
Password:

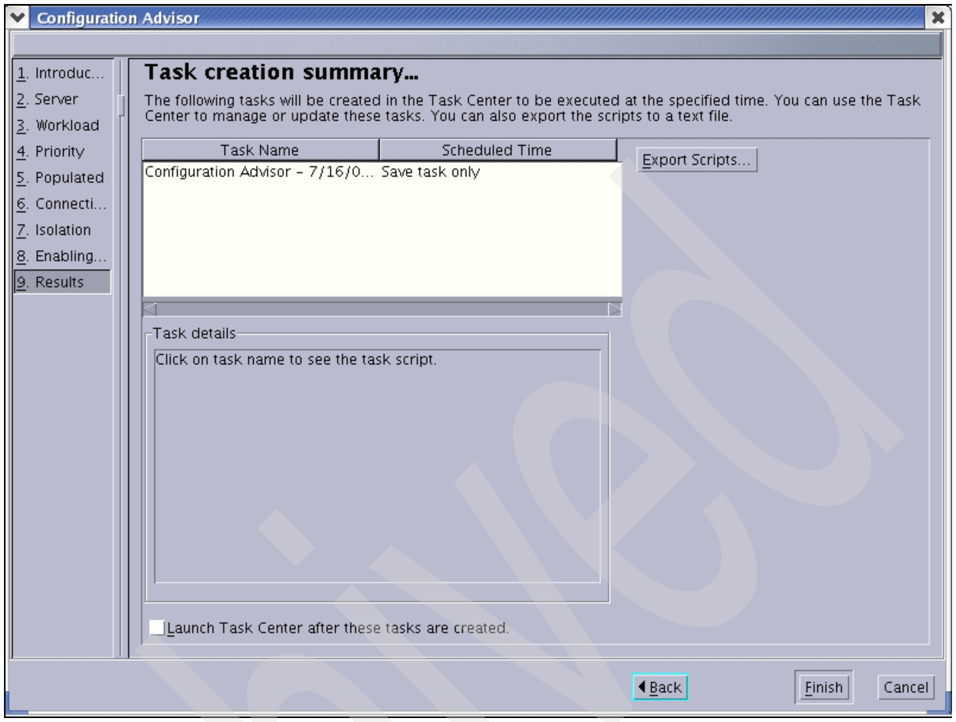
Figure 4-22 Define your task

11. Task creation summary (Figure 4-23 on page 180) is next. Click **Export Scripts**, and save it to the DB2 instance home directory. We save it to the ~db2inst1/scripts/autoconfigure.db2 directory in our example. Click **Finish**. After you created your autoconfigure script, you can run it to update DB and DBM configuration across all the partitions by using the following command:

```
db2_all “\”db2 -tvf ~db2inst1/scripts/autoconfigure.db2
>>~db2inst1/scripts/autoconfigure##.log“
```

You can check the command result from the autoconfigure*.log file. Remember that you may need to restart your DB2 instance to make some changes take effect.

After restarting your instance, you can run a benchmark against your database to see the performance improvement.



Task creation summary...

The following tasks will be created in the Task Center to be executed at the specified time. You can use the Task Center to manage or update these tasks. You can also export the scripts to a text file.

Task Name	Scheduled Time
Configuration Advisor - 7/16/0...	Save task only

Task details

Click on task name to see the task script.

Launch Task Center after these tasks are created.

Figure 4-23 Configuration results

4.4.2 Tuning your DB2 by using the autoconfigure command

You can also use the **autoconfigure** command to tune your DB2 system. The command syntax is:

```
>>-AUTOCONFIGURE--+---------------------------------------+----->
                  |         .---------------------------.  |
                  |         V                           |  |
                  '-USING----input-keyword--param-value-+-'

>--APPLY--+-DB ONLY----+---------------------------------------><
          +-DB AND DBM-+
          '-NONE-------'
```

Table 4-12 on page 181 shows autoconfigure options. We use the **autoconfigure** command to configure our DB2 system with the same option we use in Configuration Advisor GUI tool, Example 4-3 shows the command.

Example 4-3 autoconfigure example

```
CONNECT TO TPCH;
```

```
AUTOCONFIGURE USING mem_percent 80 workload_type complex num_stmts 10 tpm 20
admin_priority both num_local_apps 5 num_remote_apps 50 isolation CS APPLY DB
AND DBM;
CONNECT RESET;
```

Table 4-12 Autoconfigure options

Keyword	Valid value	Default value	Description
mem_percent	1–100	25	Percentage of memory to dedicate. If other applications (other than the operating system) are running on this server, set this to less than 100.
workload_type	Simple, mixed, complex	Mixed	Simple workloads tend to be I/O intensive and mostly transactions, whereas complex workloads tend to be CPU intensive and mostly queries.
num_stmts	1–1,000,000	10	Number of statements per unit of work.
tpm	1–200,000	60	Transactions per minute.
admin_priority	Performance, recovery, both	Both	Optimize for better performance (more transactions per minute) or better recovery time.
is_populated	Yes, no	Yes	Is the database populated with data?
num_local_apps	0–5,000	0	Number of connected local applications.
num_remote_apps	0–5,000	10	Number of connected remote applications.
isolation	RR, RS, CS, UR	RR	Isolation level of applications connecting to this database (Repeatable Read, Read Stability, Cursor Stability, Uncommitted Read).

Keyword	Valid value	Default value	Description
bp_resizeable	Yes, no	Yes	Are buffer pools resizable?

Because this command requires a database connection and only configures the partition you connect to, you need to connect to each partition and issue this command. In our lab environment, we use the **db2_all** command to tune our partitioned database. We save Example 4-3 on page 180 to *~db2inst1/autoconfigure.db2*, so we can issue the following command to tune our database across all of the partitions:

```
db2_all “db2 -tvf ~db2inst1/autoconfigure.db2 >>autoconfigure.log”
```

4.4.3 Important DB2 registry and environment variables

Typically, DB2 registry profile variables affect the DB2 engine and the optimizer's behavior. They are used for a specific purpose. We introduce some commonly used DB2 registry variables relating to performance. For a detailed description of DB2 registry and environment variables, please refer to Appendix A. “DB2 Registry and Environment Variables”, in *Administration Guide: Performance*, SC09-4821, or you can find it on DB2 Information Center.

DB2_PARALLEL_IO

When you use disk array as your storage, you can use the DB2_PARALLEL_IO registry variable to help promote parallel access to your table spaces.

The degree of parallelism is determined by the prefetch size and extent size for the containers in the table space. For example, if the prefetch size is four times the extent size, then there are four extent-sized prefetch requests. The number of containers in the table space does not affect the number of prefetchers. To enable parallel I/O for all table spaces, use the wildcard character “*”. To enable parallel I/O for a subset of all table spaces, enter the list of table spaces. If there is more than one container, extent-size pieces of any full prefetch request are broken down into smaller requests executed in parallel based on the number of prefetchers.

When this variable is not enabled, the number of prefetcher requests created is based on the number of containers in the table space.

In our lab environment, we use disk array for all of our table spaces. We set the value to “*” to enable parallel I/O for all table spaces. Use **db2set DB2_PARALLEL_IO=*** to enable it.

DB2LINUXAIO

Use this registry variable to enable asynchronous I/O (AIO) support. Because asynchronous I/O needs Linux kernel support, you should make sure that your Linux distributions meet the minimum requirements. At the time of writing, Linux distributions and kernel requirements are as listed in Table 4-13.

Table 4-13 Supported Linux distributions for AIO

Distributions	Kernel	Library
RHEL V3 Update 1 or later	2.4.21 or later	libaio 0.3.96 or later for x86, x86-64 and PPC platform; libaio 0.3.9802.1 for IA64 and zSeries platform
SLES V8 or later	2.4.9 or later	
All DB2 supported distributions	2.6.x	

AIO is not enabled by default. You can use the following command and restart DB2 instance to enable it:

```
db2set DB2LINUXAIO=true
```

Once you enable AIO, DB2 UDB page cleaners can work more efficiently because it can continue scanning other buffer pool pages to be write out and do not need to wait for the I/O operation finish. We recommend that you enable direct I/O (DIO) for table spaces that do not contain LOB or LONG VARCHAR data.

DB2 UDB V8.2 supports DIO on Linux with 2.6 kernel. With DIO, DB2 avoids using the file system cache to retrieve data. It eliminates the copy operation from the file system cache to the DB2 buffer pool, then reduces CPU utilization.

To enable direct I/O for your tablespace, you can specify the NO FILE SYSTEM CACHING parameter in your CREATE/ALTER TABLESPACE or CREATE DATABASE commands. Example 4-4 shows how to create a SMS table space with DIO enabled.

Example 4-4 Create tablespace with DIO enabled

```
CONNECT TO TPCH;

CREATE REGULAR TABLESPACE DATA1
MANAGED BY SYSTEM
USING (‘/db2fs1p $N /db2inst1/tpch/ts01c00’)
NO FILE SYSTEM CACHING;

CONNECT RESET;
```

```
TERMINATE;
```

DB2_SCATTERED_IO

This registry variable turns on scattered I/O, which uses readv() to read from disk. If you are running on a system that contains the vectored raw I/O performance improvement Linux kernel patch, then you should turn this variable on to increase performance. Use the command **`db2set DB2_SCATTERED_IO=ON`** to enable scattered I/O. Table 4-14 shows Linux distributions that can support Vector I/O.

Table 4-14 Supported Linux distributions for scattered I/O

Distributions	Base kernel
Red Hat AS 2.1 or later	2.4.9
RHEL 3 or later	2.4.21
SLES 8 or later	2.4.19
All distributions supported by DB2	2.6.x

DB2_LGPAGE_BP

This registry variable enables large page memory support on DB2. Before you use this parameter, ensure that your Linux system supports large page memory. Table 4-15 lists the Linux distributions that support large page buffer pool.

Table 4-15 Supported Linux distributions for large page buffer pool

Distributions	Base kernel	Library	Comment
Red Hat AS 2.1 or later	2.4.9	libcap.so	You can install libcap.so from the libcap RPM package.
RHEL 3 or later	2.4.21	libcap.so	
SLES 8 Service Pack 3 or later	2.4.21	libcap.so	
All distributions supported by DB2	2.6.x	libcap.so	

You need to configure your Linux operating system to enable large memory support. We describe how to enable large memory support on different Linux distributions separately.

SuSE Linux Enterprise Server 8 with Service Pack 3 (all platforms except IA64) and Red Hat Advanced Server 2.1

You can enable large memory support by the following two steps:

1. Use a text editor to open your boot manager configuration file, for example, /boot/grub/menu.lst file for grub boot manager, add *bigpages=xxxM* to your boot parameters, where *xxx* is the desired amount of large pages in Megabytes. Reboot your server.
2. After you reboot your server as root, run the **`echo 1 > /proc/sys/kernel/shm-use-bigpages`** command to dynamically enable large page support in the kernel.

The amount of *bigpages* to use will depend on the size of your buffer pools and an additional small amount for overhead. We recommend that you monitor the /proc/meminfo output. Pay special attention to the *BigFree* entry to monitor the amount of free large pages remaining.

SuSE Linux Enterprise Server 8 with Service Pack 3 (IA64 only)

Configure large memory support in this environment is as same as in the distributions based on mainline 2.6.x kernel. Please refer to “Distributions based on mainline 2.6.x Kernel” on page 185.

Red Hat Enterprise Linux 3

Large memory is supported by default in the RHEL V3 kernel. You only need to specify the amount of large memory; specify the */proc/sys/vm/hugetlb_pool* kernel parameter dynamically. The value for this parameter is the amount of large page memory MB. For example, if you issue the command **`echo 500 > /proc/sys/vm/hugetlb_pool`** as root, it means that we specify 500-MB large page memory. You can see similar output from the **`cat /proc/meminfo`** command:

```
HugePages_Total: 250
HugePages_Free: 250
Hugepagesize: 2048 KB
```

Note: The amount of large page memory must be available in a contiguous block of memory. If it is not, the value is automatically reduced to the largest contiguous block that is available. Because system memory gets fragmented throughout regular system operation, it might be necessary to reboot the computer and set hugetlb_pool as soon as possible.

Distributions based on mainline 2.6.x Kernel

Large page support in the mainline 2.6.x kernel is very similar to that of Red Hat Enterprise Linux 3. The only difference is that the kernel parameter is called *nr_hugepages* instead of *hugetlb_pool*, and it specifies hugepage units to use instead of Megabytes.

To specify 500 large pages memory, use the following command:

```
cat 500 > /proc/sys/vm/hr_hugepages
```

If you issue **`cat /proc/meminfo`**, you should see information similar to the following:

```
HugePages_Total:   500
HugePages_Free:    500
Hugepagesize:     2048 kB
```

Configure DB2 to use large page memory for buffer pools

You can enable DB2 to take advantage of large pages by issuing the command **`db2set DB2_LGPAGE_BP=YES`** before starting the instance. After you enable this registry variable, DB2 will allocate all memory used for buffer pools and other control segments from the pool of large pages.

> **Attention:** Keep monitoring large memory pool usage through the /proc/meminfo file. Ensure that you have enough memory, but not too much in large page pool for DB2 buffer pools.

DB2_EVALUNCOMMITTED

This registry affects DB2's row lock mechanism for CS and RS isolation levels. It is OFF by default. Use the command **`db2set DB2_EVALUNCOMMITTED=ON`** to enable it.

These registry variable settings apply at compile time for dynamic SQL and at bind time for static SQL. With this feature enabled, DB2 can apply predicate evaluation on uncommitted data. Also, deleted rows are skipped unconditionally on table scan access while deleted keys are not skipped for type-2 index scans unless the registry variable DB2_SKIPDELETED is also set. Enabling this registry variable can improve concurrency greatly.

DB2_SKIPDELETED

When enabled, DB2 allows statements using either CS or RS isolation levels to unconditionally skip deleted keys during index access and deleted rows during table access. With DB2_EVALUNCOMMITTED enabled, deleted rows are automatically skipped, but uncommitted pseudo-deleted keys in type-2 indexes are not unless DB2_SKIPDELETED is also enabled.

Note: DB2_EVALUNCOMMITTED and DB2_SKIPDELETED do not impact the behavior of cursors on the DB2 catalog tables.

4.4.4 Validating logical database design with the DB2 Design advisor

The DB2 Design advisor is a tool that can help you significantly improve your workload performance. The task of selecting which indexes, MQTs, clustering dimensions, or partitions to create for a complex workload can be quite daunting. The Design advisor identifies all of the objects needed to improve the performance of your workload. Given a set of SQL statements in a workload, the Design advisor will generate recommendations for:

- New indexes
- New materialized query tables (MQTs)
- Conversion to multidimensional clustering (MDC) tables
- Repartitioning of tables
- Deletion of indexes and MQTs unused by the specified workload (through the GUI tool)

The Design advisor considers the costs and advantages of each feature when formulating design recommendations. It takes into account the characteristics of your workload, the characteristics of the database, and the hardware resources. Then it weighs the performance improvements against the costs associated with implementing the recommendations.

Before implementing any of the Design advisor recommendations, consider whether you provided the Design advisor with a representative query workload. If the workload that you submit to the Design advisor is not representative of the overall database activity, the recommendations might not provide performance enhancements for queries not included in the workload. Also, it is critical that the statistics on any object involved in the workload be up to date. Inaccurate statistics might negatively impact the quality of the Design advisor recommendations.

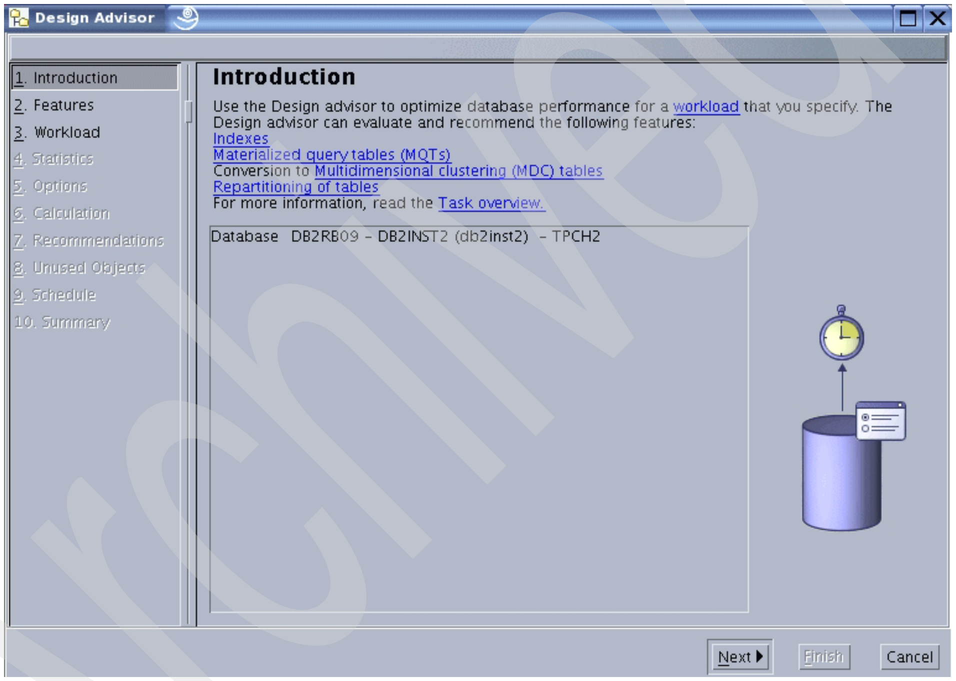
Because the Design advisor performs the cost-benefit analysis for you, all features are selected by default for consideration. It is recommended that you leave this default setting and allow the Design advisor to determine which features provide the greatest overall performance advantages for the workload that you specify.

If you do not want to implement specific performance enhancement features, you can choose not to select those features for evaluation. Eliminating those features will reduce the amount of time that the Design advisor spends evaluating different performance enhancement possibilities for your database.

We now show how do run the GUI of Design advisor and then show the command-line version.

Design advisor wizard

To start the design advisor go to the DB2 Control Center (db2cc), right-click the database for which you want to get recommendations, and select **Design Advisor**. When the Design advisor is started, it shows you the Introduction page, as shown in Figure 4-24. To get to the next page click **Next**.

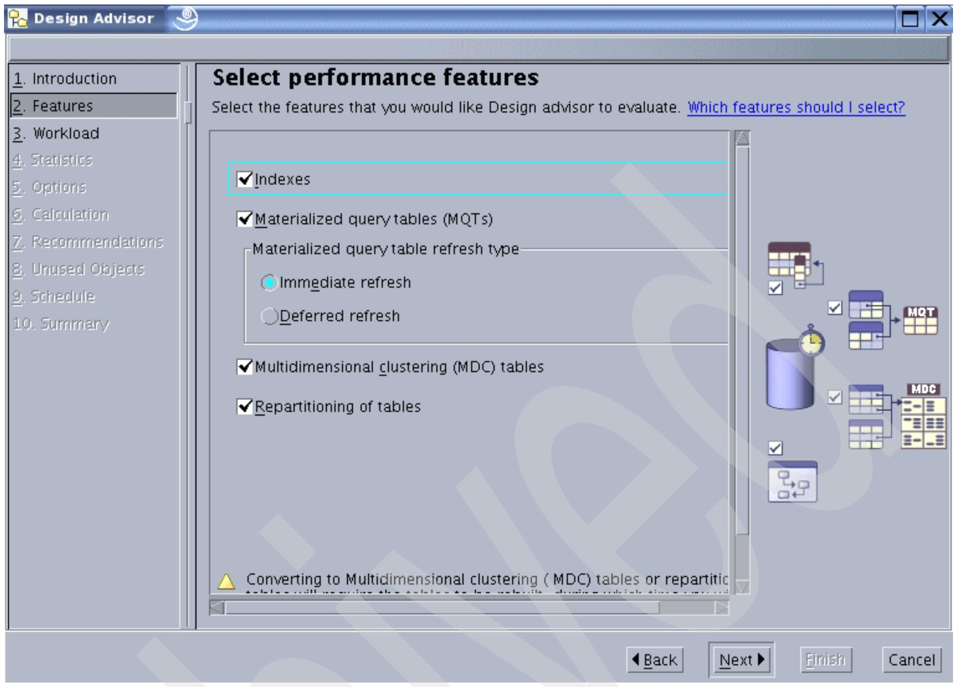


Introduction

Use the Design advisor to optimize database performance for a workload that you specify. The Design advisor can evaluate and recommend the following features:
Indexes
Materialized query tables (MQTs)
Conversion to Multidimensional clustering (MDC) tables
Repartitioning of tables
For more information, read the Task overview.

Database DB2RB09 - DB2INST2 (db2inst2) - TPCH2

Figure 4-24 Design advisor - Introduction



Design Advisor

1. Introduction
2. Features
3. Workload
4. Statistics
5. Options
6. Calculation
7. Recommendations
8. Unused Objects
9. Schedule
10. Summary

Select performance features

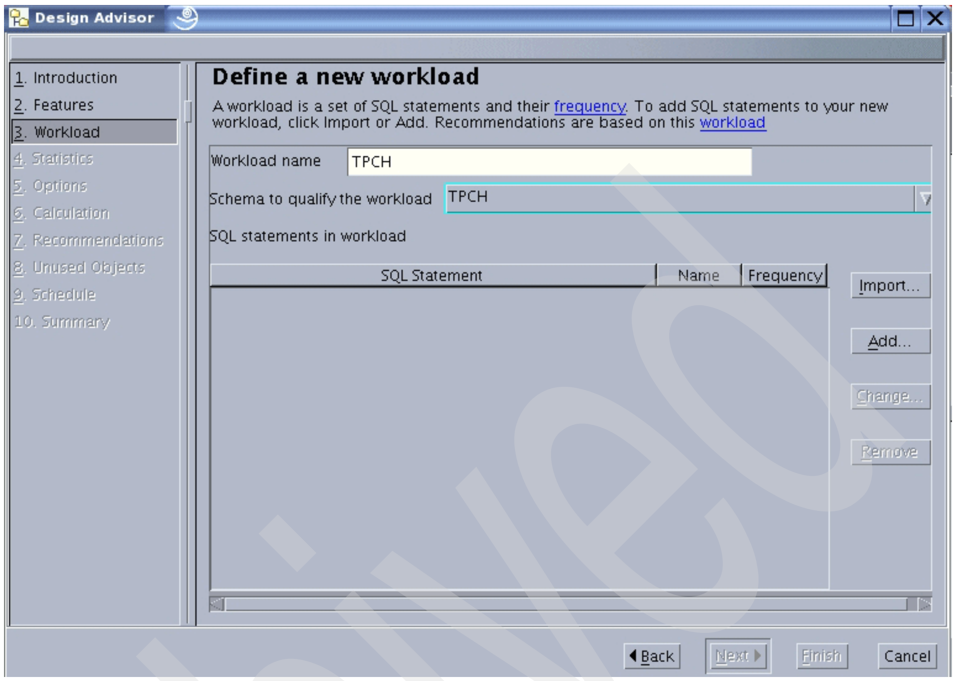
Select the features that you would like Design advisor to evaluate. Which features should I select?

- [✓] Indexes
- [✓] Materialized query tables (MQTs)
  - Materialized query table refresh type
    - [● Immediate refresh]
    - Deferred refresh
- [✓] Multidimensional clustering (MDC) tables
- [✓] Repartitioning of tables

Converting to Multidimensional clustering (MDC) tables or repartitic

Figure 4-25 Design advisor - Features

The Features page gives you the option to select the features you want the Design advisor to take into account. You can select Indexes, Materialized query tables (MQTs), Multidimensional clustering (MDC) tables and Repartitioning of tables. We recommended that you accept all of the options selected. Use the Next button to get to the Workload page.



Define a new workload

A workload is a set of SQL statements and their frequency. To add SQL statements to your new workload, click Import or Add. Recommendations are based on this workload

Workload name: TPCH

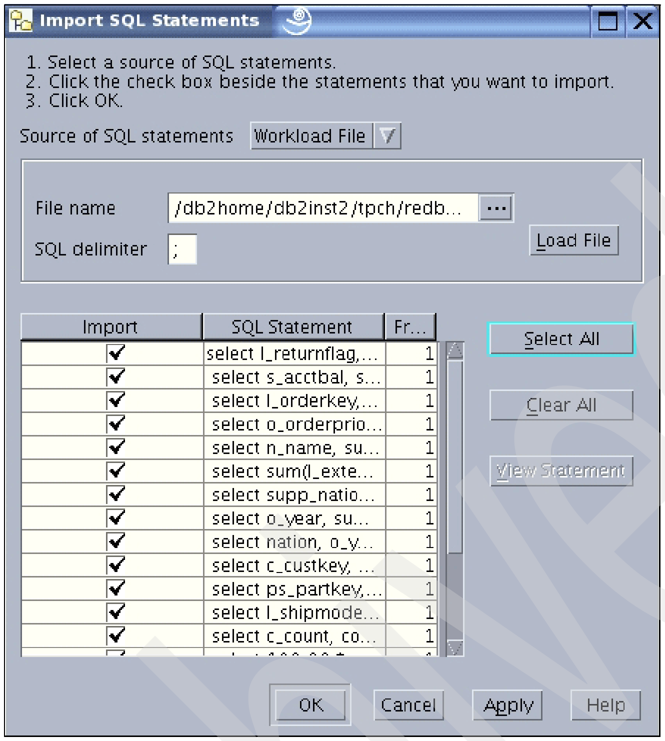
Schema to qualify the workload: TPCH

SQL statements in workload

SQL Statement	Name	Frequency

Figure 4-26 Design advisor - Workload

Give your workload a name in the Workload name field and select the schema for the workload, as shown in Figure 4-26. To enter the workload you have two choices, you can either use the Add option to enter the SQL command in the opening window or use the Import option to import a workload file (delimited text file, containing the SQL commands). We show how to import a workload file. Use the **Import** button to do so.



Import SQL Statements

1. Select a source of SQL statements.
2. Click the check box beside the statements that you want to import.
3. Click OK.

Source of SQL statements: Workload File

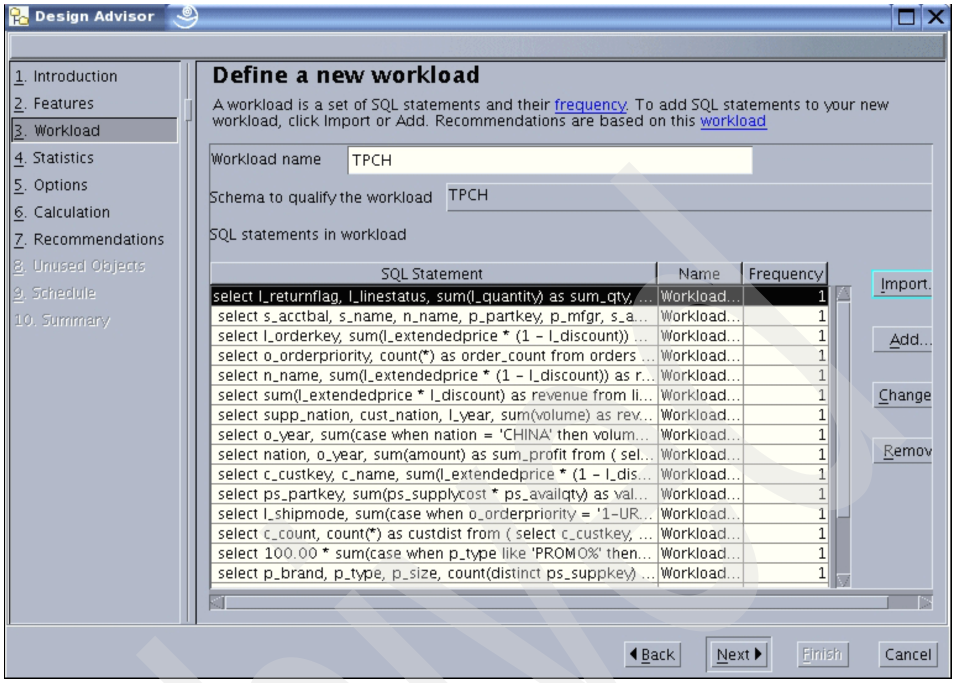
File name: /db2home/db2inst2/tpch/redb...

SQL delimiter: ;

Import	SQL Statement	Fr...
✓	select l_returnflag,...	1
✓	select s_acctbal, s...	1
✓	select l_orderkey,...	1
✓	select o_orderprio...	1
✓	select n_name, su...	1
✓	select sum(l_exte...	1
✓	select supp_natio...	1
✓	select o_year, su...	1
✓	select nation, o_y...	1
✓	select c_custkey, ...	1
✓	select ps_partkey,...	1
✓	select l_shipmode...	1
✓	select c_count, co...	1

Figure 4-27 Design advisor - Import workload file

You have to enter (or browse with the **...** button) the File name and select the SQL delimiter used in your text file. You can then click **Load File** to load the file. The content will then be shown in a window similar to Figure 4-27. The checkboxes in the Import column are not yet selected; do so with the **Select All** button or just select the SQL statements you want to use. When finished, click **OK** to get back to the Workload page. Please note that the import option also allows for import from package and recent SQL statements.



Define a new workload

A workload is a set of SQL statements and their frequency. To add SQL statements to your new workload, click Import or Add. Recommendations are based on this workload

Workload name: TPCH

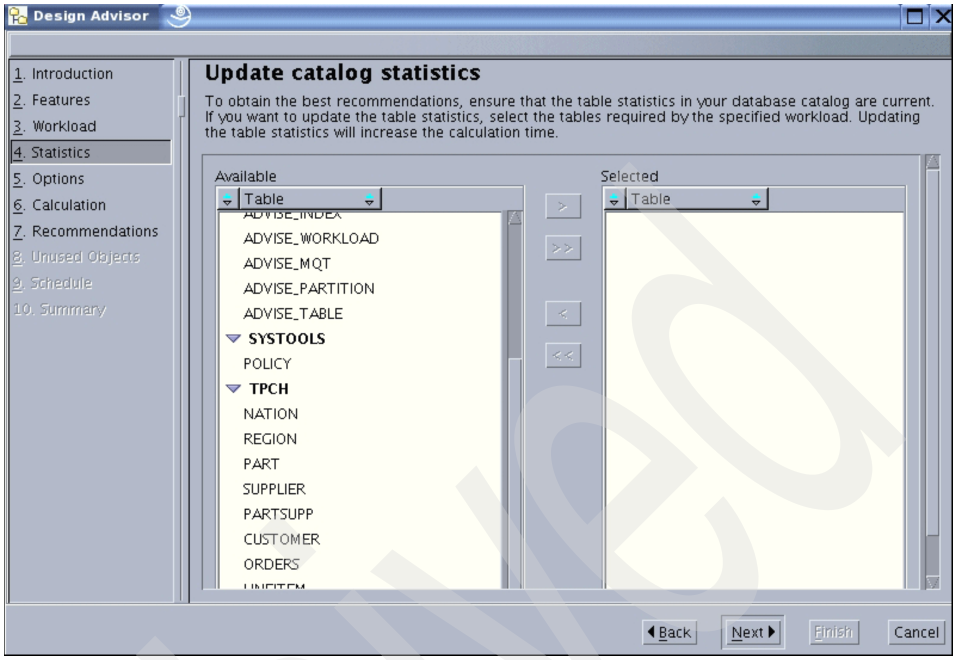
Schema to qualify the workload: TPCH

SQL statements in workload

SQL Statement	Name	Frequency
select l_returnflag, l_linestatus, sum(l_quantity) as sum_qty, ...	Workload...	1
select s_acctbal, s_name, n_name, p_partkey, p_mfgr, s_a...	Workload...	1
select l_orderkey, sum(l_extendedprice * (1 - l_discount)) ...	Workload...	1
select o_orderpriority, count(*) as order_count from orders ...	Workload...	1
select n_name, sum(l_extendedprice * (1 - l_discount)) as r...	Workload...	1
select sum(l_extendedprice * l_discount) as revenue from li...	Workload...	1
select supp_nation, cust_nation, l_year, sum(volume) as rev...	Workload...	1
select o_year, sum(case when nation = 'CHINA' then volum...	Workload...	1
select nation, o_year, sum(amount) as sum_profit from (sel...	Workload...	1
select c_custkey, c_name, sum(l_extendedprice * (1 - l_dis...	Workload...	1
select ps_partkey, sum(ps_supplycost * ps_availqty) as val...	Workload...	1
select l_shipmode, sum(case when o_orderpriority = '1-UR...	Workload...	1
select c_count, count(*) as custdist from (select c_custkey, ...	Workload...	1
select 100.00 * sum(case when p_type like 'PROMO%' then...	Workload...	1
select p_brand, p_type, p_size, count(distinct ps_suppkey) ...	Workload...	1

Figure 4-28 Design advisor - Imported workload

The SQL statements we imported are now shown in the Workload page, as shown in Figure 4-28. You can change a statement or its frequency with the Change button, add a new statement with the Add button, or remove a statement with the Remove button. If customization of the workload has finished, continue by clicking the **Next** button.



Update catalog statistics

To obtain the best recommendations, ensure that the table statistics in your database catalog are current. If you want to update the table statistics, select the tables required by the specified workload. Updating the table statistics will increase the calculation time.

Available: Table

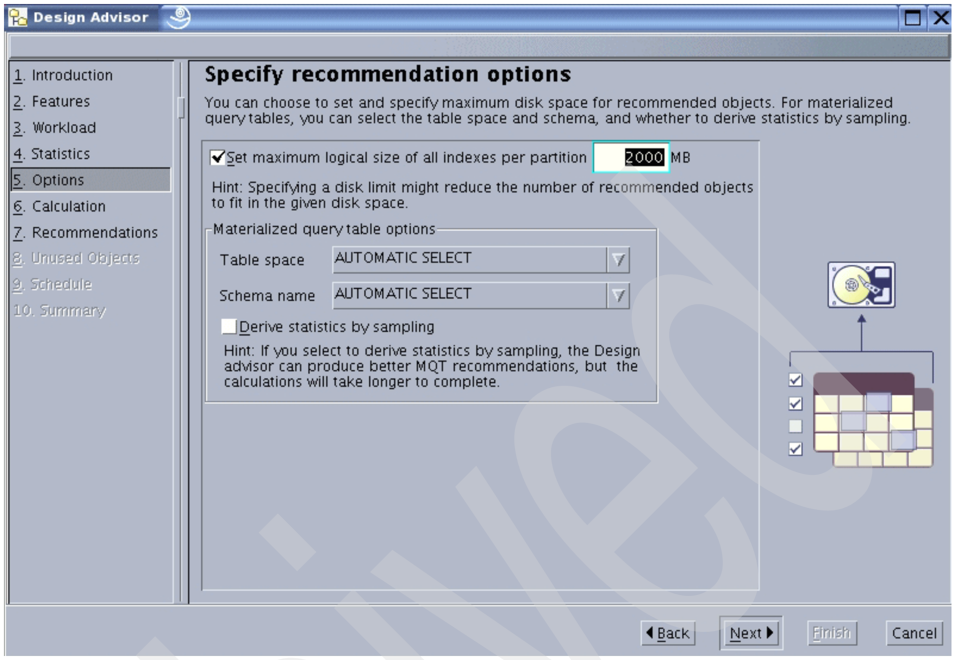
ADVISE_INDEX
ADVISE_WORKLOAD
ADVISE_MQT
ADVISE_PARTITION
ADVISE_TABLE
SYSTOOLS
POLICY
TPCH
NATION
REGION
PART
SUPPLIER
PARTSUPP
CUSTOMER
ORDERS
LINEITEM

Selected: Table

Figure 4-29 Design advisor - Statistics

On the Statistics page (Figure 4-29) you can select tables on which you want to run statistics before the Design advisor calculates recommendations. Current statistics are essential to get correct recommendations. When you have selected the necessary tables click **Next**.

Tip: Do runstats on the tables used by your workload before you use the Design advisor. Although the Design advisor can collect the statistic data for you, you have no influence on how it collects them. When using the runstats utility you can specify the options desired.


Design Advisor

1. Introduction
2. Features
3. Workload
4. Statistics
5. Options
6. Calculation
7. Recommendations
8. Unused Objects
9. Schedule
10. Summary

Specify recommendation options

You can choose to set and specify maximum disk space for recommended objects. For materialized query tables, you can select the table space and schema, and whether to derive statistics by sampling.

[✓] Set maximum logical size of all indexes per partition 2000 MB

Hint: Specifying a disk limit might reduce the number of recommended objects to fit in the given disk space.

Materialized query table options

Table space: AUTOMATIC SELECT

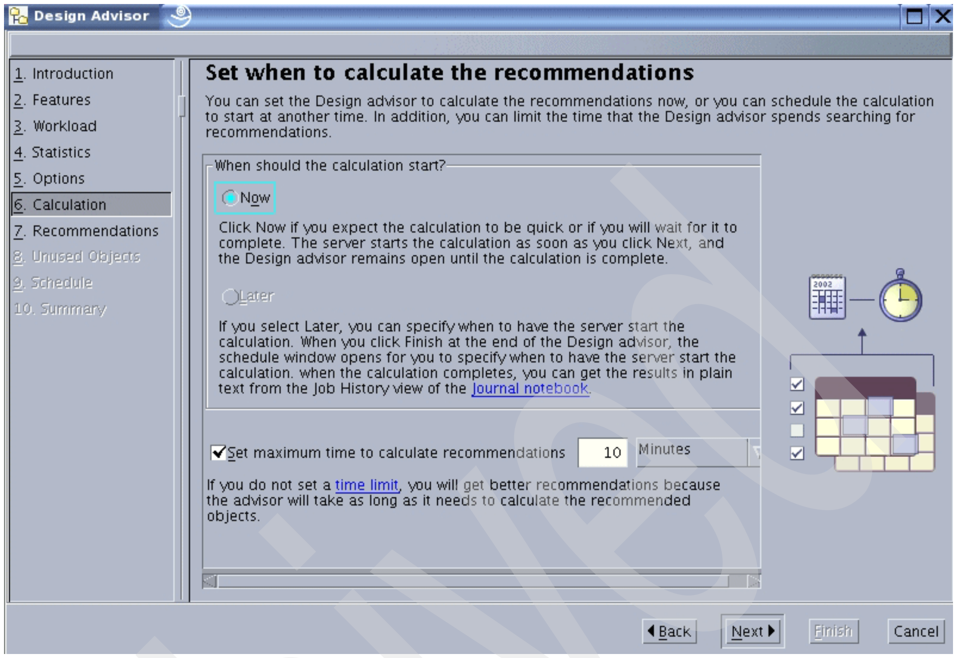
Schema name: AUTOMATIC SELECT

[ ] Derive statistics by sampling

Hint: If you select to derive statistics by sampling, the Design advisor can produce better MQT recommendations, but the calculations will take longer to complete.

Figure 4-30 Design advisor - Options

On the Options page, shown in Figure 4-30, you can limit the space used for the recommended indexes (if you do not select the option the Design advisor will take 20 percent of the database size) and select some options for MQTs. Once the options are selected, proceed by clicking **Next**.



Set when to calculate the recommendations

You can set the Design advisor to calculate the recommendations now, or you can schedule the calculation to start at another time. In addition, you can limit the time that the Design advisor spends searching for recommendations.

When should the calculation start?

[✓ Now]

Click Now if you expect the calculation to be quick or if you will wait for it to complete. The server starts the calculation as soon as you click Next, and the Design advisor remains open until the calculation is complete.

Later

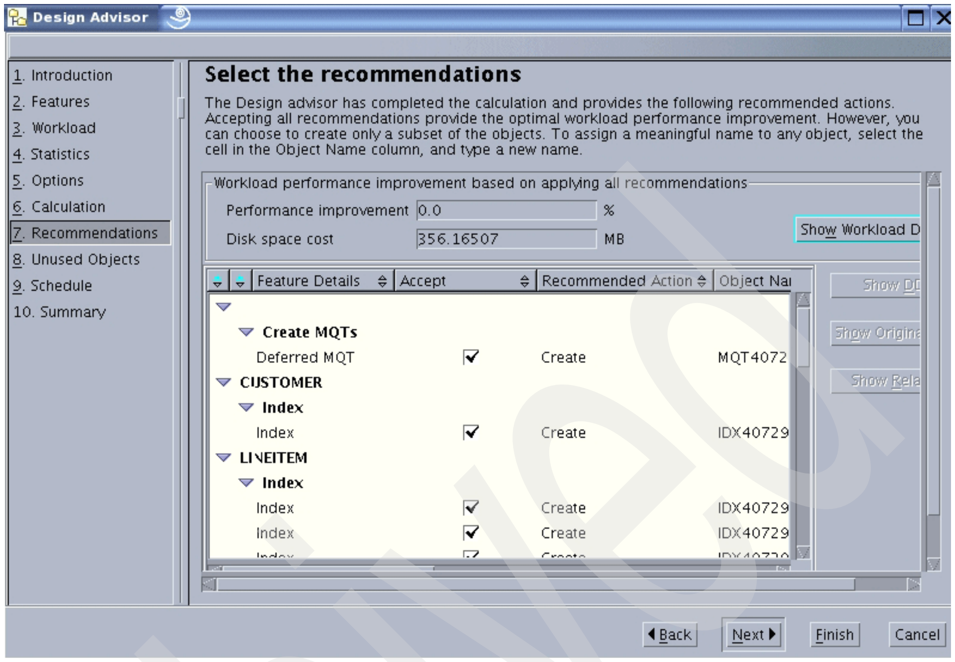
If you select Later, you can specify when to have the server start the calculation. When you click Finish at the end of the Design advisor, the schedule window opens for you to specify when to have the server start the calculation. when the calculation completes, you can get the results in plain text from the Job History view of the Journal notebook.

[✓] Set maximum time to calculate recommendations 10 Minutes

If you do not set a time limit, you will get better recommendations because the advisor will take as long as it needs to calculate the recommended objects.

Figure 4-31 Design advisor - Calculation

The Calculation page (Figure 4-31) gives you the opportunity to limit the run time of the Design advisor and to schedule the calculation for a later time. Click the **Next** button to start the calculation (when you select Now, or else you get to the Schedule page).


Design Advisor

1. Introduction
2. Features
3. Workload
4. Statistics
5. Options
6. Calculation
7. Recommendations
8. Unused Objects
9. Schedule
10. Summary

Select the recommendations

The Design advisor has completed the calculation and provides the following recommended actions. Accepting all recommendations provide the optimal workload performance improvement. However, you can choose to create only a subset of the objects. To assign a meaningful name to any object, select the cell in the Object Name column, and type a new name.

Workload performance improvement based on applying all recommendations

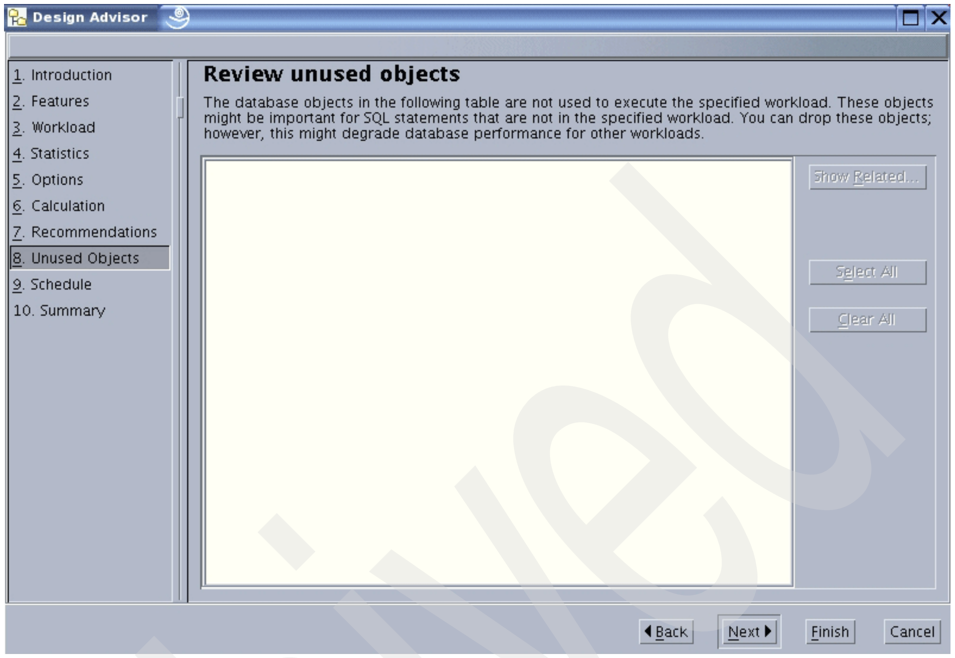
Performance improvement 0.0 %

Disk space cost 356.16507 MB

Feature Details	Accept	Recommended Action	Object Nai
Create MQTs			
Deferred MQT	✓	Create	MQT4072
CUSTOMER			
Index			
Index	✓	Create	IDX40729
LINEITEM			
Index			
Index	✓	Create	IDX40729
Index	✓	Create	IDX40729

Figure 4-32 Design advisor - Recommendations

When the calculation finishes, the Recommendation page is presented to you, as shown in Figure 4-32. By default all recommendations are accepted; that means they will be applied when you finish the wizard. You can certainly deselect options you do not want to be implemented. You also can change the index name. Click **Next** to get to the Unused Objects page.

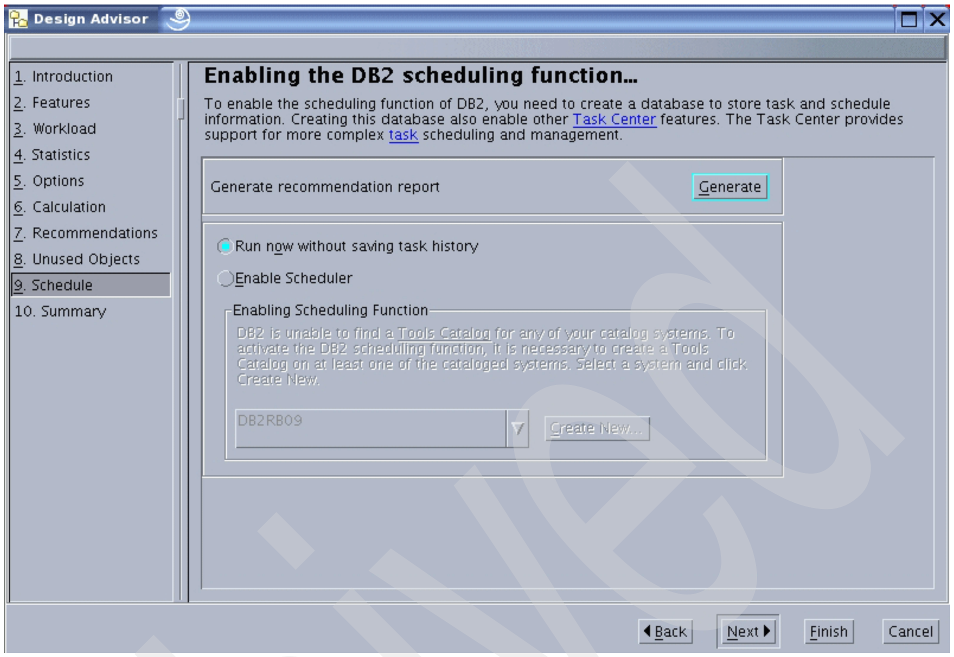


Review unused objects

The database objects in the following table are not used to execute the specified workload. These objects might be important for SQL statements that are not in the specified workload. You can drop these objects; however, this might degrade database performance for other workloads.

Figure 4-33 Design advisor - Unused objects

On the page shown in Figure 4-33 objects that are not used by your workload are shown. The ones you select will be deleted when you finish the wizard.



Enabling the DB2 scheduling function...

To enable the scheduling function of DB2, you need to create a database to store task and schedule information. Creating this database also enable other Task Center features. The Task Center provides support for more complex task scheduling and management.

Generate recommendation report [Generate]

(●) Run now without saving task history
( ) Enable Scheduler

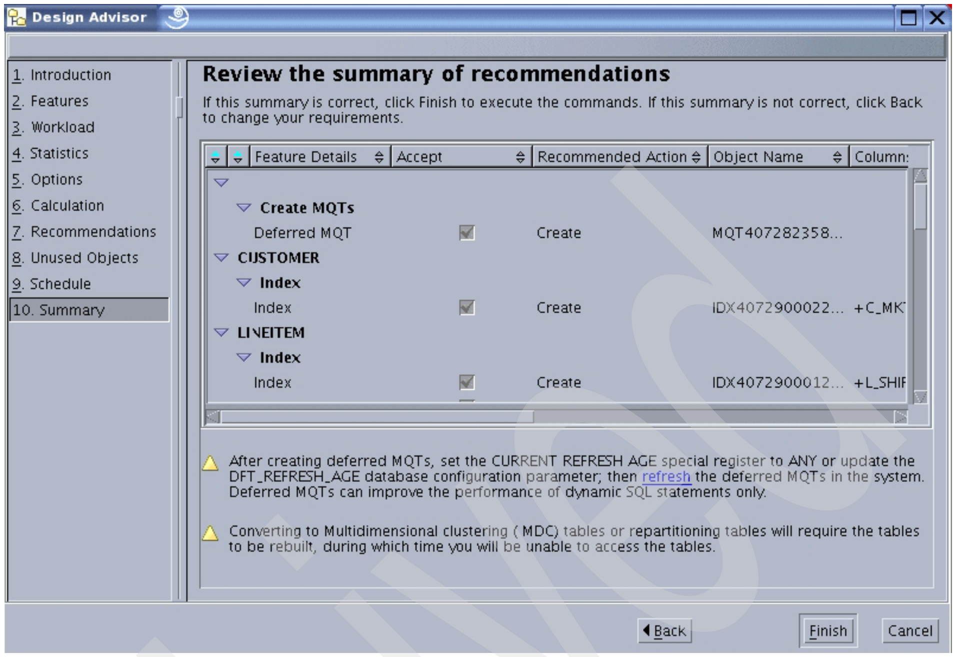
Enabling Scheduling Function

DB2 is unable to find a Tools Catalog for any of your catalog systems. To activate the DB2 scheduling function, it is necessary to create a Tools Catalog on at least one of the cataloged systems. Select a system and click Create New.

DB2RB09 [Create New...]

Figure 4-34 Design advisor - Schedule

The Schedule page shown in Figure 4-34 allows you to run the task now or at a later time. Please note that if the task is run now, the indexes are created and runstats commands are also issued for all affected tables. If you have big tables, the process may have a performance impact. Scheduling the task at the off-peak time should then be considered. Proceed with the **Next** button.



Review the summary of recommendations

If this summary is correct, click Finish to execute the commands. If this summary is not correct, click Back to change your requirements.

Feature Details	Accept	Recommended Action	Object Name	Column:
Create MQTs				
Deferred MQT	✓	Create	MQT407282358...	
CUSTOMER				
Index				
Index	✓	Create	IDX4072900022...	+C_MK
LINEITEM				
Index				
Index	✓	Create	IDX4072900012...	+L_SHIF

After creating deferred MQTs, set the CURRENT REFRESH AGE special register to ANY or update the DFT_REFRESH_AGE database configuration parameter; then refresh the deferred MQTs in the system. Deferred MQTs can improve the performance of dynamic SQL statements only.

Converting to Multidimensional clustering (MDC) tables or repartitioning tables will require the tables to be rebuilt, during which time you will be unable to access the tables.

Figure 4-35 Design advisor - Summary

The Summary page (shown in Figure 4-35) lists every object that will be created or deleted according to your selection. You can control your selection again, and when you are sure you want to apply the settings click the **Finish** button to do so.

Design advisor command

Now we also show the command-line version, which has the following syntax:

```
>>-db2advis--+--d--+--database-name--+-------------------+------>
             '--db-'                 +--w--workload-name-+
                                     +--s--"statement"---+
                                     +--i--filename------+
                                     +--g----------------+
                                     '--qp---------------'
>--+------------------------+--+-----------------+--+------+--->
   '--a--userid--+---------+-'  '--m--advise-type-'  '--x---'
                 '-/passwd-'
>--+------+--+----------------+--+----+------------------------>
   '--u---'  '--l--disk-limit-'  '--n-'
>--+--------------------+--+-------------+--+------+---------->
   '--t--max-advise-time-'  '--k--+-HIGH-+-'  '--f---'
                                  +-MED--+
                                  +-LOW--+
                                  '-OFF--'
```

```
>--+------+--+-----------------+--+-----------------+----------->
   '--r---'  '--n--schema-name-'  '--q--schema-name-'
>--+---------------------+--+---------------------+--+----+----->
   '--b--tablespace-name-'  '--c--tablespace-name-'  '--h-'
>--+----+--+-------------+-------------------------------------><
   '--p-'  '--o--outfile-'
```

The following are options:

- -d: Database TPCH
- -w: Workload TPCH
- -m: The advise needed on Indexes (I), MQTs (M), MDCs (C) and partitioning (P)
- -q: Using the SCHEMA TPCH for unqualified queries in the workload
- -l: Calculating for a maximum of 15 minutes
- -o: Writing the recommendations to the *recommendation.out* file
- -f: Forces db2advis to drop existing catalog table copies

The command is as follows:

```
db2advis -d TPCH -w TPCH -m IMCP -q TPCH -t 15 -f -o recommendation.out
```

With small modifications the output file can be run as a DB2 command file:

```
db2 -tvf recommendation.out
```

More information about the **db2advis** command can be found in the *IBM DB2 UDB Command Reference V8*, SC09-4828.

5

Operations and administration

Autonomic computing is an approach to self-managed computing systems with a minimum of human interference. The term derives from the body's autonomic nervous system, which controls key functions without conscious awareness or involvement.

This does not mean that Database Administrators (DBAs) are needless. This technique is intended to improve the productivity and effectiveness of the DBAs.

This chapter describes the new autonomic features of DB2 UDB V8.2 and some of the tools on the system that can be used for monitoring.

It contains the following topics:

- Autonomic features
- Automatic database maintenance
- Manageability enhancements
- DB2 monitoring tools
- Monitoring your Linux

The automatic database maintenance features allow you to automatically run backup, runstats, and reorg, depending on a defined policy.

To set up the automatic database maintenance you start the Automatic Maintenance wizard from the Health Center or go in the Control Center, right-click the database you want to configure, and select **Configure Automatic Maintenance**, as shown in Figure 5-1.

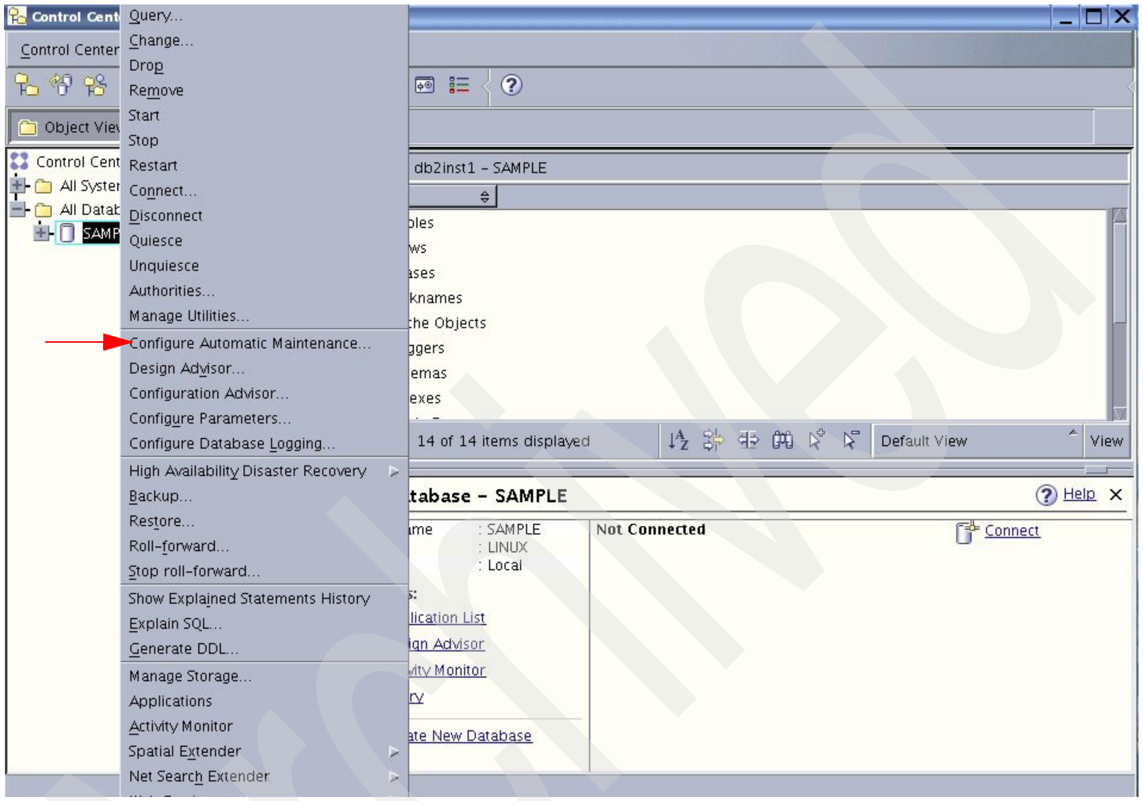
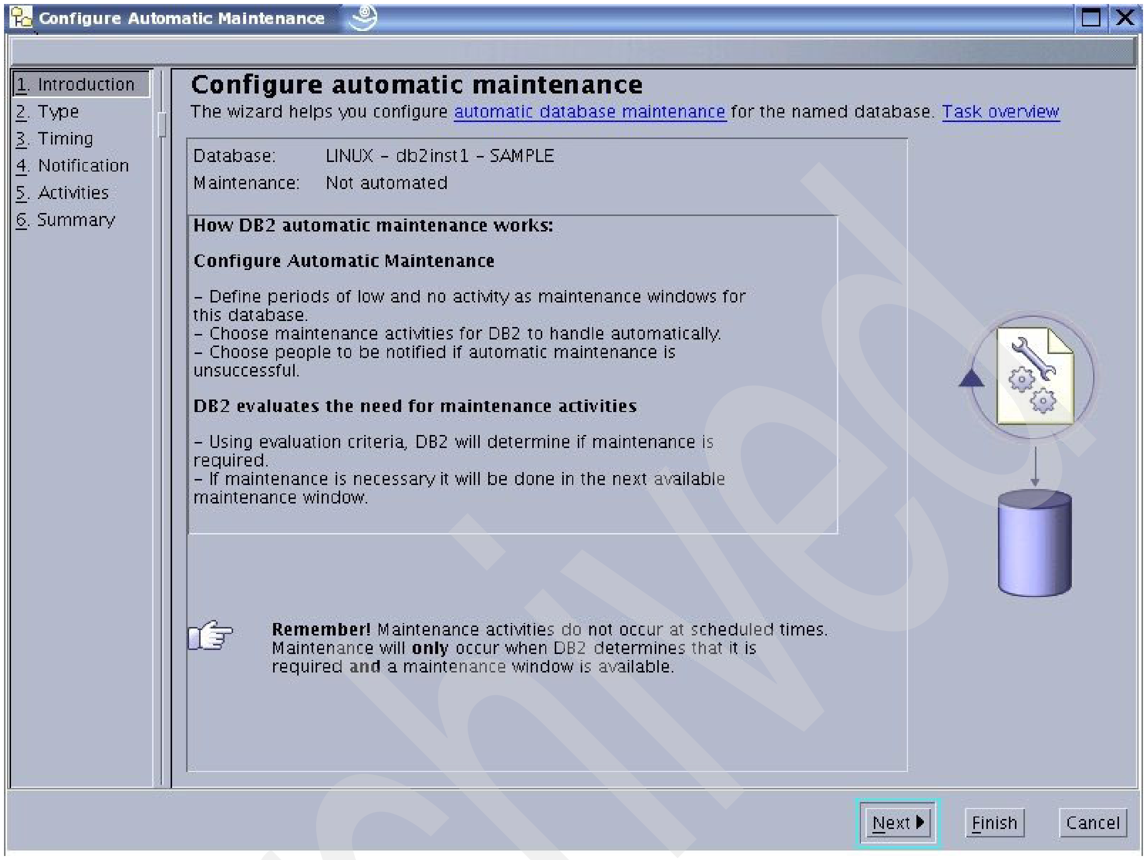


Figure 5-1 Configure automatic maintenance start

After having started the automatic maintenance you will be shown the Introduction page, as in Figure 5-2 on page 203.



Configure automatic maintenance

The wizard helps you configure automatic database maintenance for the named database. Task overview

Database: LINUX - db2inst1 - SAMPLE

Maintenance: Not automated

How DB2 automatic maintenance works:

Configure Automatic Maintenance

- Define periods of low and no activity as maintenance windows for this database.
- Choose maintenance activities for DB2 to handle automatically.
- Choose people to be notified if automatic maintenance is unsuccessful.

DB2 evaluates the need for maintenance activities

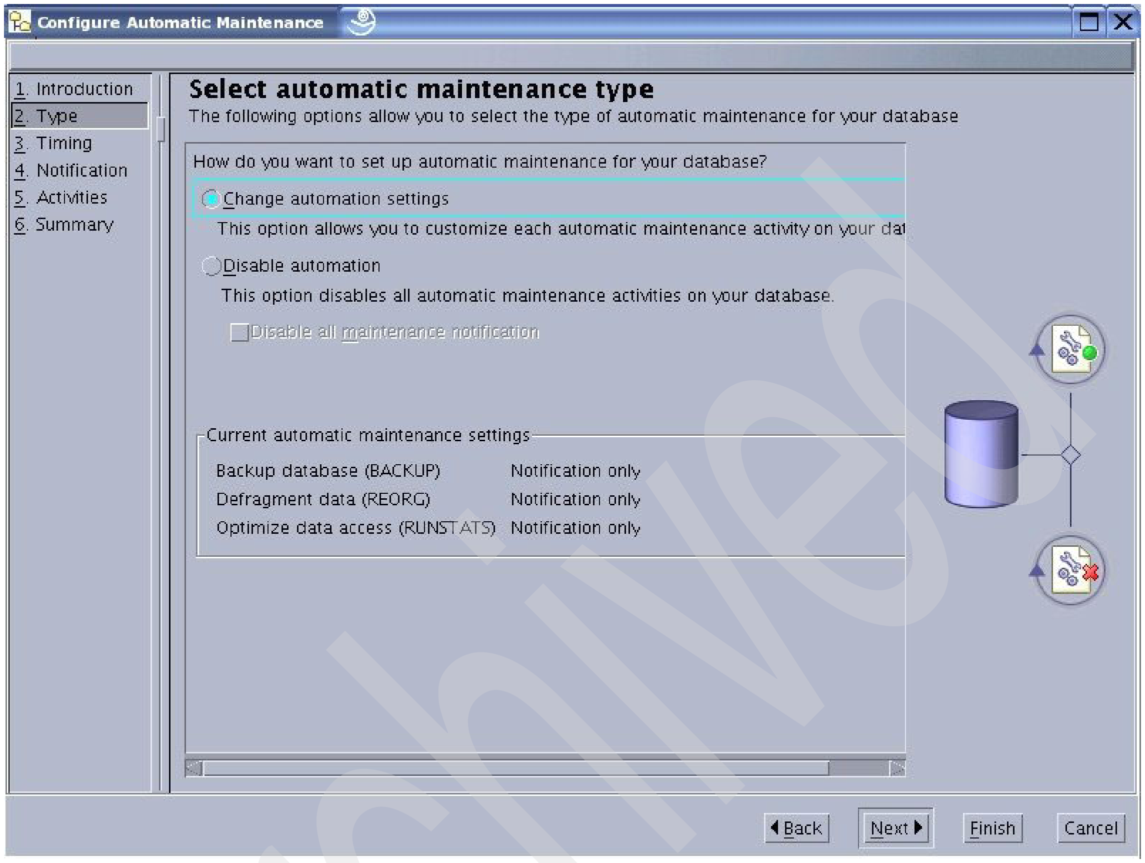
- Using evaluation criteria, DB2 will determine if maintenance is required.
- If maintenance is necessary it will be done in the next available maintenance window.

Remember! Maintenance activities do not occur at scheduled times. Maintenance will **only** occur when DB2 determines that it is required **and** a maintenance window is available.

Figure 5-2 Introduction to automatic maintenance

When you have read the introduction, proceed by clicking the **Next** button. You will get to Type screen, where you should choose the “Change automation settings” radio button and then click the **Next** button (see Figure 5-3 on page 204).

You can also switch the automation off here by selecting the Disable automation radio button.



Select automatic maintenance type

The following options allow you to select the type of automatic maintenance for your database

How do you want to set up automatic maintenance for your database?

(●) Change automation settings

This option allows you to customize each automatic maintenance activity on your dat

( ) Disable automation

This option disables all automatic maintenance activities on your database.

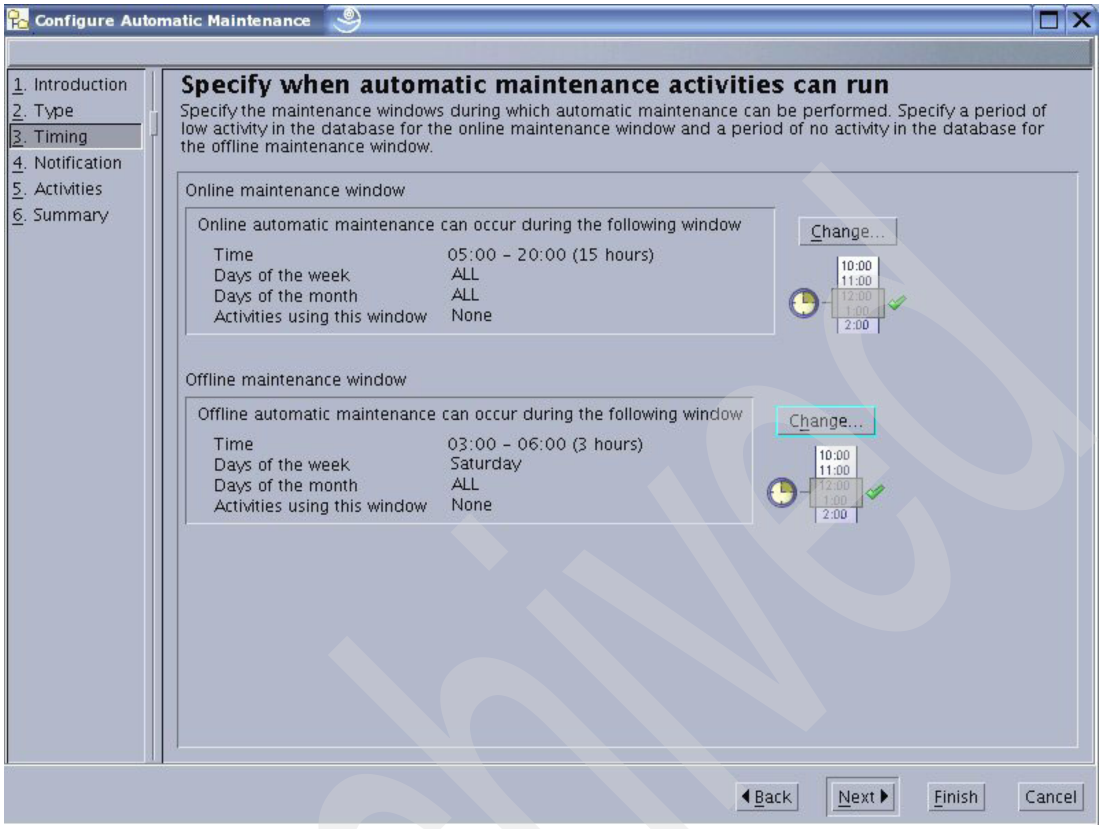
[ ] Disable all maintenance notification

Current automatic maintenance settings

Backup database (BACKUP)	Notification only
Defragment data (REORG)	Notification only
Optimize data access (RUNSTATS)	Notification only

Figure 5-3 Select automation maintenance type

The next screen, Figure 5-4 on page 205, will give you the option to define your online and offline maintenance windows.


Configure Automatic Maintenance

1. Introduction
2. Type
3. Timing
4. Notification
5. Activities
6. Summary

Specify when automatic maintenance activities can run

Specify the maintenance windows during which automatic maintenance can be performed. Specify a period of low activity in the database for the online maintenance window and a period of no activity in the database for the offline maintenance window.

Online maintenance window

Online automatic maintenance can occur during the following window

Time	05:00 - 20:00 (15 hours)
Days of the week	ALL
Days of the month	ALL
Activities using this window	None

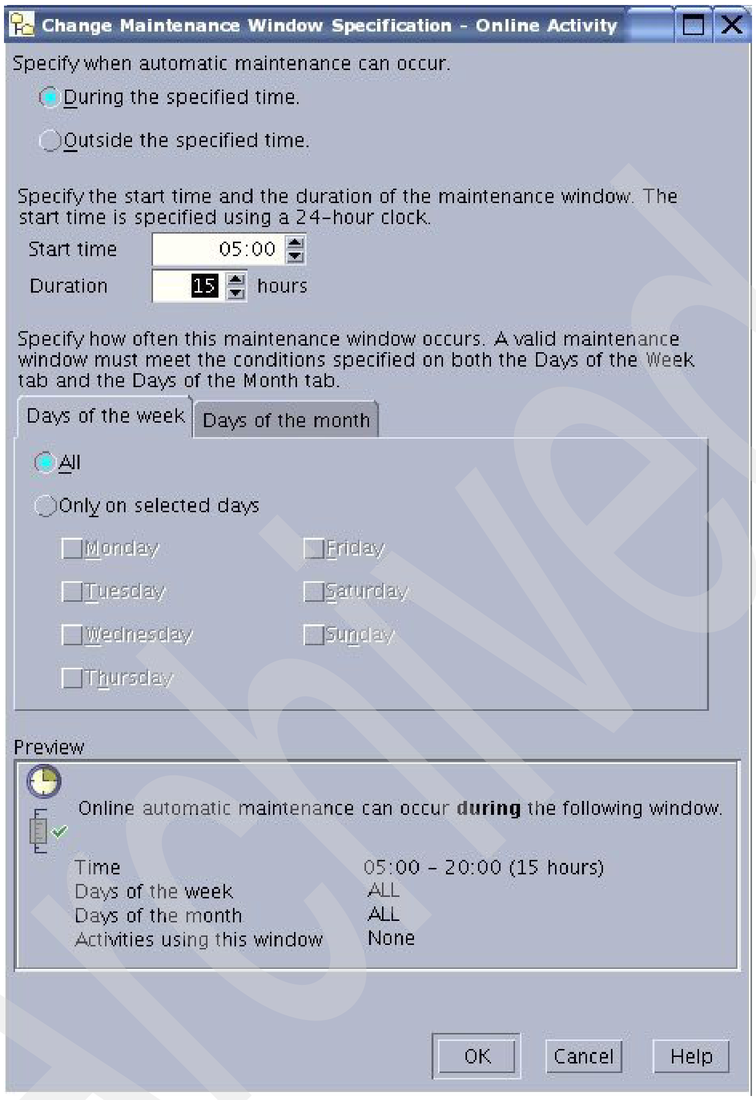
Offline maintenance window

Offline automatic maintenance can occur during the following window

Time	03:00 - 06:00 (3 hours)
Days of the week	Saturday
Days of the month	ALL
Activities using this window	None

Figure 5-4 Timing of automatic maintenance activities

By choosing the **Change** button beside the online maintenance window, you will see the screen shown in Figure 5-5 on page 206, or the offline maintenance window. What you will see then is shown in Figure 5-6 on page 207,. You can define the time range for your particular environment. The online maintenance will consume resources. You may want to consider the time frame in which the maintenance activities are performed. You also can use the UTIL_IMPACT_LIM database manager configuration parameter to influence the amount of load that the online tools will generate on your system. However, the online runstats is running with maximum 7 percent impact irrespective of UTIL_IMPACT_LIM.


Change Maintenance Window Specification - Online Activity

Specify when automatic maintenance can occur.

[✓ During the specified time.]

Outside the specified time.

Specify the start time and the duration of the maintenance window. The start time is specified using a 24-hour clock.

Start time	05:00
Duration	15 hours

Specify how often this maintenance window occurs. A valid maintenance window must meet the conditions specified on both the Days of the Week tab and the Days of the Month tab.

[✓ All]

Only on selected days

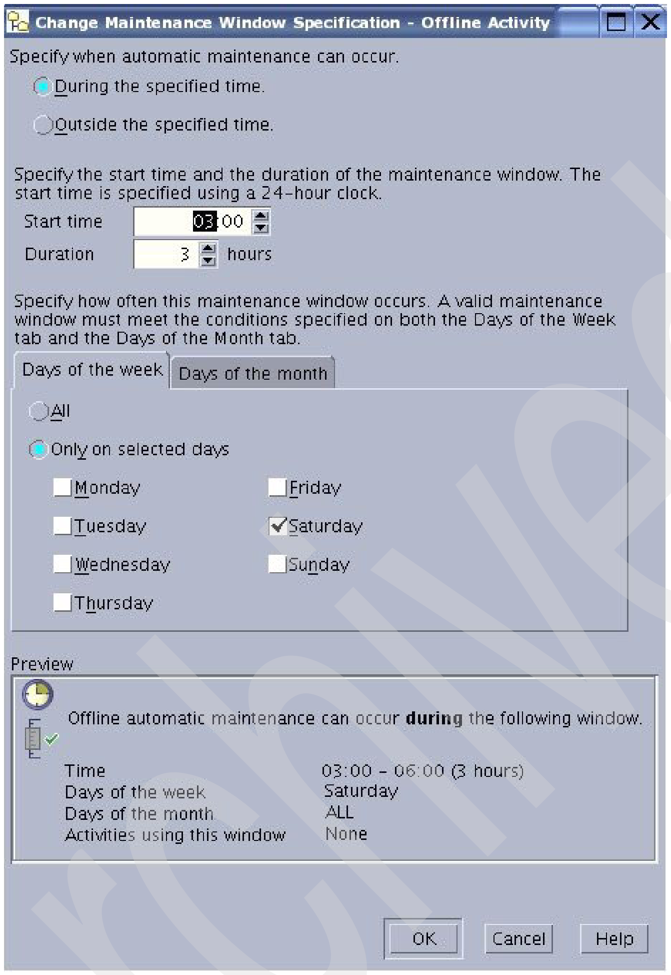
Preview

Online automatic maintenance can occur **during** the following window.

Time	05:00 - 20:00 (15 hours)
Days of the week	ALL
Days of the month	ALL
Activities using this window	None

Figure 5-5 Changing online maintenance time

Automatic statistics collection as well as online database backups are run in the online maintenance time period. To minimize the impact on the system, they are throttled by the adaptive utility throttling mechanism. The internal scheduling mechanism uses the online maintenance time period to start the online jobs. These features run to completion even if they go beyond the time period specified.


Change Maintenance Window Specification - Offline Activity

Specify when automatic maintenance can occur.

[✓ During the specified time.]

Outside the specified time.

Specify the start time and the duration of the maintenance window. The start time is specified using a 24-hour clock.

Start time	03:00
Duration	3 hours

Specify how often this maintenance window occurs. A valid maintenance window must meet the conditions specified on both the Days of the Week tab and the Days of the Month tab.

All

[✓ Only on selected days]

☐ Monday	☐ Friday
☐ Tuesday	☑ Saturday
☐ Wednesday	☐ Sunday
☐ Thursday	

Preview

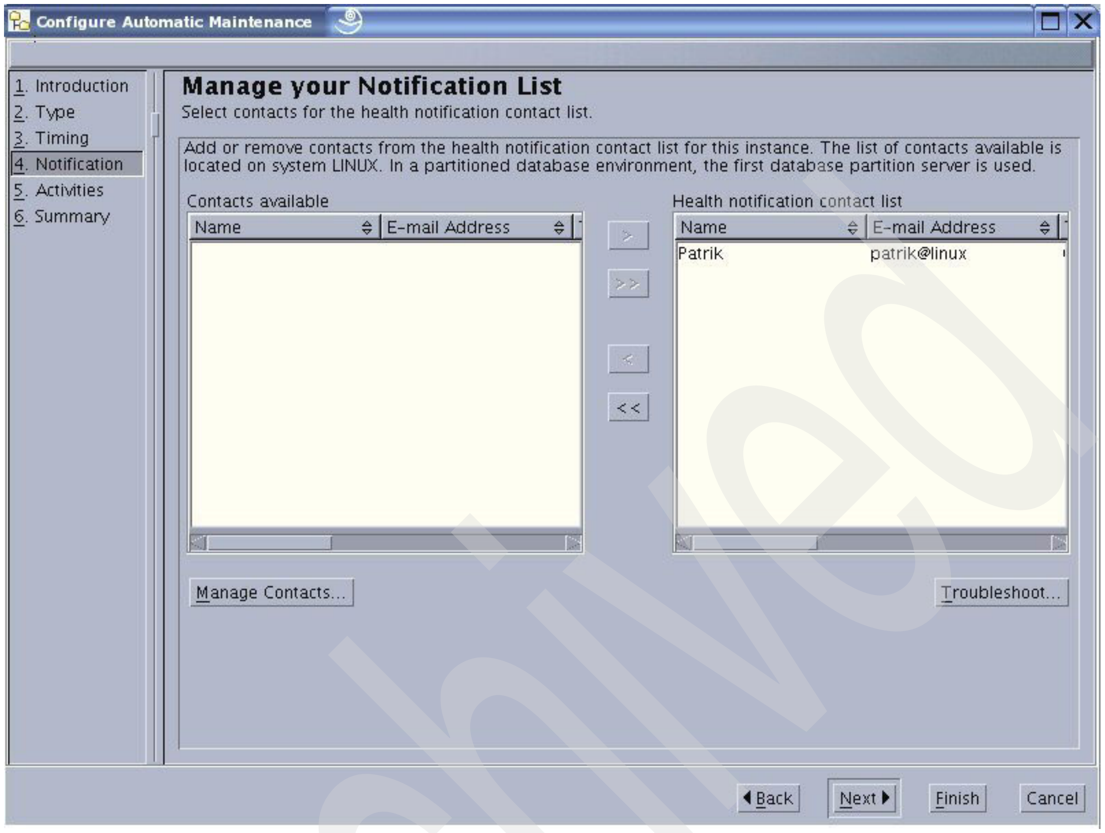
Offline automatic maintenance can occur **during** the following window.

Time	03:00 - 06:00 (3 hours)
Days of the week	Saturday
Days of the month	ALL
Activities using this window	None

Figure 5-6 Changing offline maintenance time

Offline database backups and table and index reorganization are run in the offline maintenance time period. These features run to completion even if they go beyond the time period specified. The internal scheduling mechanism learns over time and estimates job completion times. If the offline time period is too small for a particular database backup or reorganization activity, the scheduler will not start the job the next time around and relies on the health monitor to provide notification of the need to increase the offline maintenance time period.

When you adjusted the settings for both online and offline maintenance windows, confirmed them by clicking **OK** in each case, we proceed to the *Notification* step by clicking the **Next** button (Figure 5-4 on page 205).



Configure Automatic Maintenance

Manage your Notification List

Select contacts for the health notification contact list.

Add or remove contacts from the health notification contact list for this instance. The list of contacts available is located on system LINUX. In a partitioned database environment, the first database partition server is used.

Contacts available

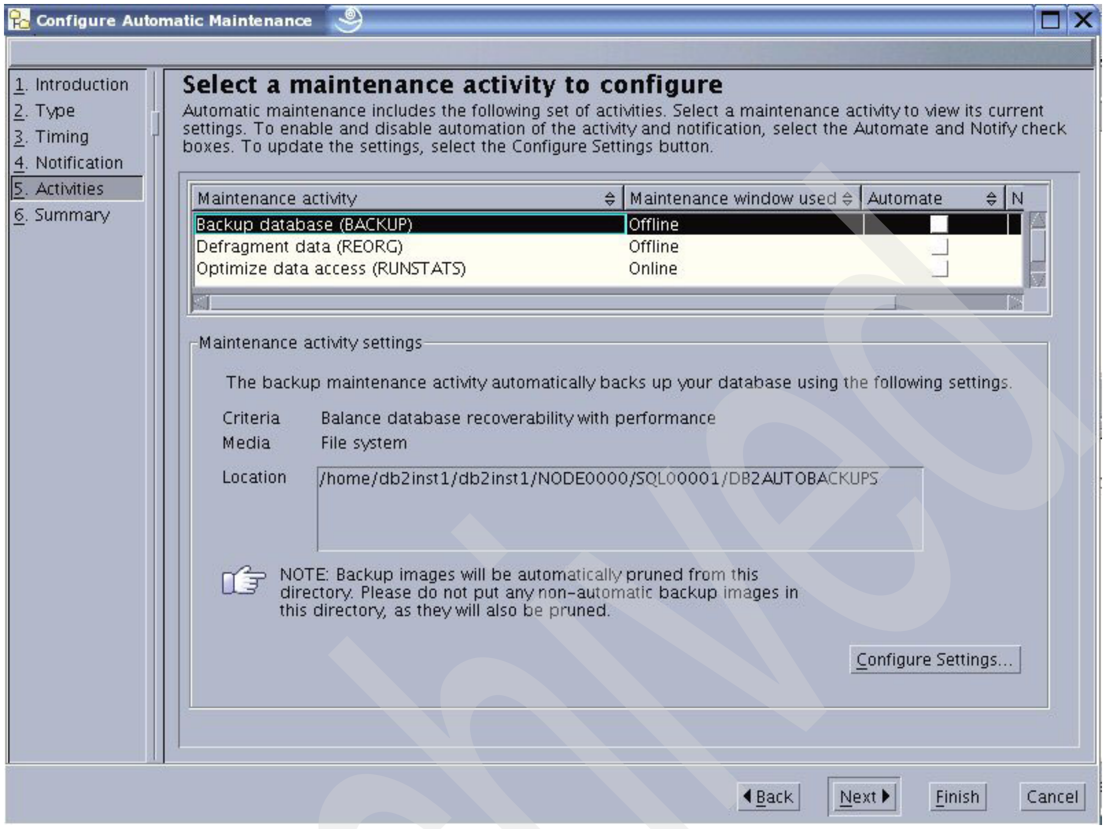
Name	E-mail Address

Health notification contact list

Name	E-mail Address
Patrik	patrik@linux

Figure 5-7 Manage your notification list

On the *Manage your Notification List* page (shown in Figure 5-7), you can manage the people who should be notified if messages are generated. You can add additional contacts by using the **Manage Contacts** button. When you finish with this click the **Next** button, which will bring you to the Activities page.


Configure Automatic Maintenance

1. Introduction
2. Type
3. Timing
4. Notification
5. Activities
6. Summary

Select a maintenance activity to configure

Automatic maintenance includes the following set of activities. Select a maintenance activity to view its current settings. To enable and disable automation of the activity and notification, select the Automate and Notify check boxes. To update the settings, select the Configure Settings button.

Maintenance activity	Maintenance window used	Automate	N
Backup database (BACKUP)	Offline	☐	
Defragment data (REORG)	Offline	☐	
Optimize data access (RUNSTATS)	Online	☐	

Maintenance activity settings

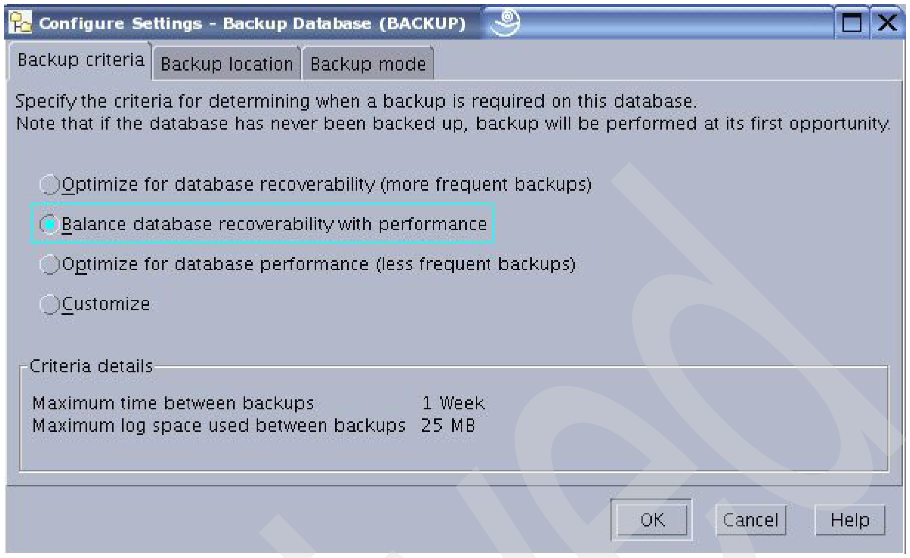
The backup maintenance activity automatically backs up your database using the following settings.

Criteria	Balance database recoverability with performance
Media	File system
Location	/home/db2inst1/db2inst1/NODE0000/SQL00001/DB2AUTOBACKUPS

NOTE: Backup images will be automatically pruned from this directory. Please do not put any non-automatic backup images in this directory, as they will also be pruned.

Figure 5-8 Configure maintenance activity

First select in the window the activity you want to configure, then click the **Configure Settings** button to get into the corresponding configuration dialog. In this redbook we first do the configuration for the backup.


Configure Settings - Backup Database (BACKUP)

Backup criteria | Backup location | Backup mode

Specify the criteria for determining when a backup is required on this database.
Note that if the database has never been backed up, backup will be performed at its first opportunity.

- ○ Optimize for database recoverability (more frequent backups)
- ● Balance database recoverability with performance
- ○ Optimize for database performance (less frequent backups)
- ○ Customize

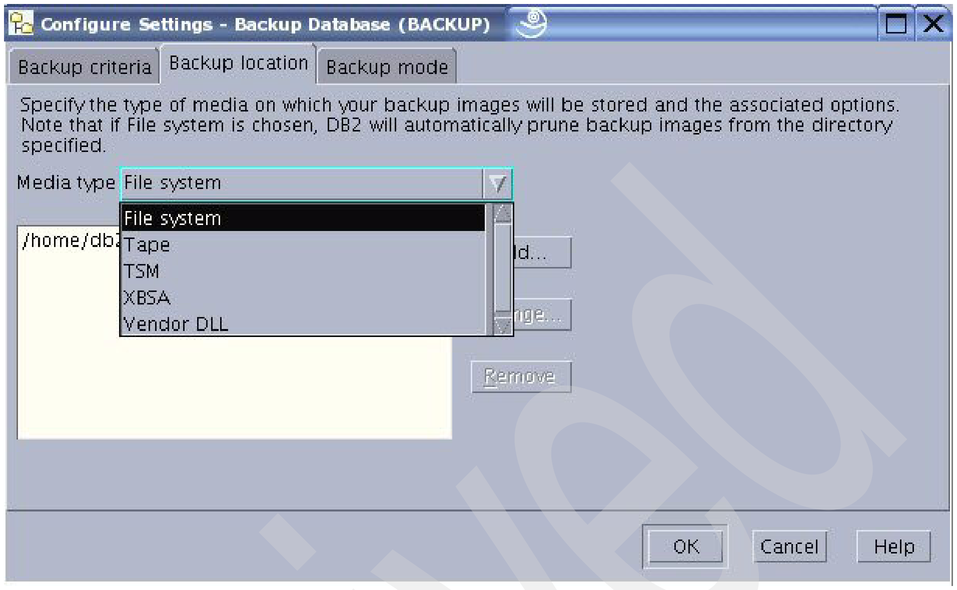
Criteria details

Maximum time between backups	1 Week
Maximum log space used between backups	25 MB

Figure 5-9 Backup priority

The first screen for the backup configuration, shown in Figure 5-9, will give you four options for the frequency of the backup, of which three are predefined and one is to customize the settings to your needs based on maximum time between backups and maximum log space used between backups. The log space used option is only available if archival logging is enabled, which is described in “Archival logging” on page 257. Choose the option that meets your needs or customize it the way that meets your requirements. If done, click the **Backup location** tab, which is shown in Figure 5-10 on page 211.

.


Configure Settings - Backup Database (BACKUP)

Backup criteria | Backup location | Backup mode

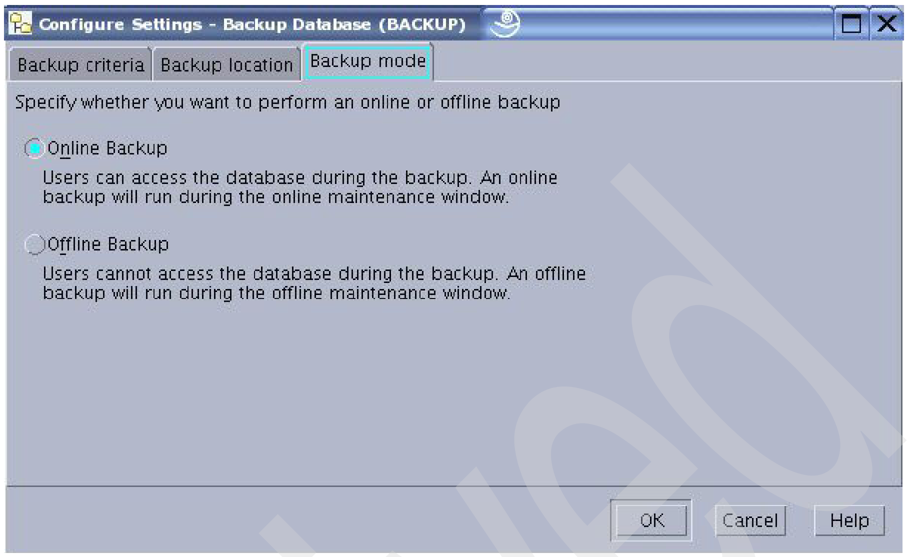
Specify the type of media on which your backup images will be stored and the associated options. Note that if File system is chosen, DB2 will automatically prune backup images from the directory specified.

Media type: File system

Figure 5-10   Backup location

You have the choice to write your backup to the file system, to tape, to TSM, to use the XBSA, or to use a vendor dll. If backup to disk is selected, the automatic backup feature will regularly delete backup images from the directory specified in the Configure Automatic Maintenance wizard. Only the most recent backup image is guaranteed to be available at any given time. It is recommended that this directory be kept exclusively for the automatic backup feature and not be used to store other backup images.

The last of the three tabs, the Backup mode (shown in Figure 5-11 on page 212), lets you choose between an online or an offline backup. Please remember that you can only do online backups when you have enabled archival logging, which is described in “Archival logging” on page 257. A new feature in DB2 UDB V8.2 is that you can include the log files required in the online backup image, which then can be used during database restore time.


Configure Settings - Backup Database (BACKUP)

Backup criteria | Backup location | Backup mode

Specify whether you want to perform an online or offline backup

(●) Online Backup

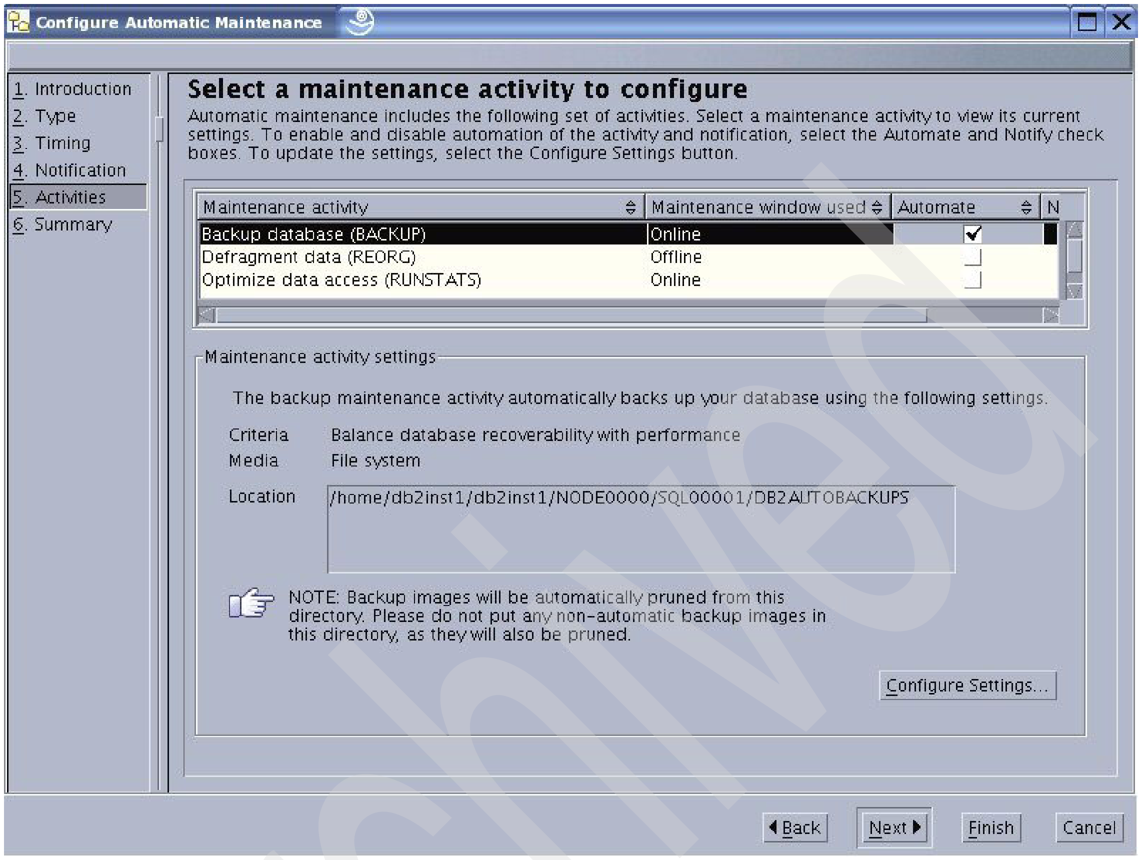
Users can access the database during the backup. An online backup will run during the online maintenance window.

( ) Offline Backup

Users cannot access the database during the backup. An offline backup will run during the offline maintenance window.

Figure 5-11 Backup mode

After having done the configuration, you apply the settings by clicking **OK**. You get back to the Activities page with an updated line for the backup, as shown in Figure 5-12 on page 213. Please make sure that the box for backup database in the column Automate is checked.



Select a maintenance activity to configure

Automatic maintenance includes the following set of activities. Select a maintenance activity to view its current settings. To enable and disable automation of the activity and notification, select the Automate and Notify check boxes. To update the settings, select the Configure Settings button.

Maintenance activity	Maintenance window used	Automate	N
Backup database (BACKUP)	Online	✓	
Defragment data (REORG)	Offline		
Optimize data access (RUNSTATS)	Online		

Maintenance activity settings

The backup maintenance activity automatically backs up your database using the following settings.

Criteria	Balance database recoverability with performance
Media	File system
Location	/home/db2inst1/db2inst1/NODE0000/SQL00001/DB2AUTOBACKUPS

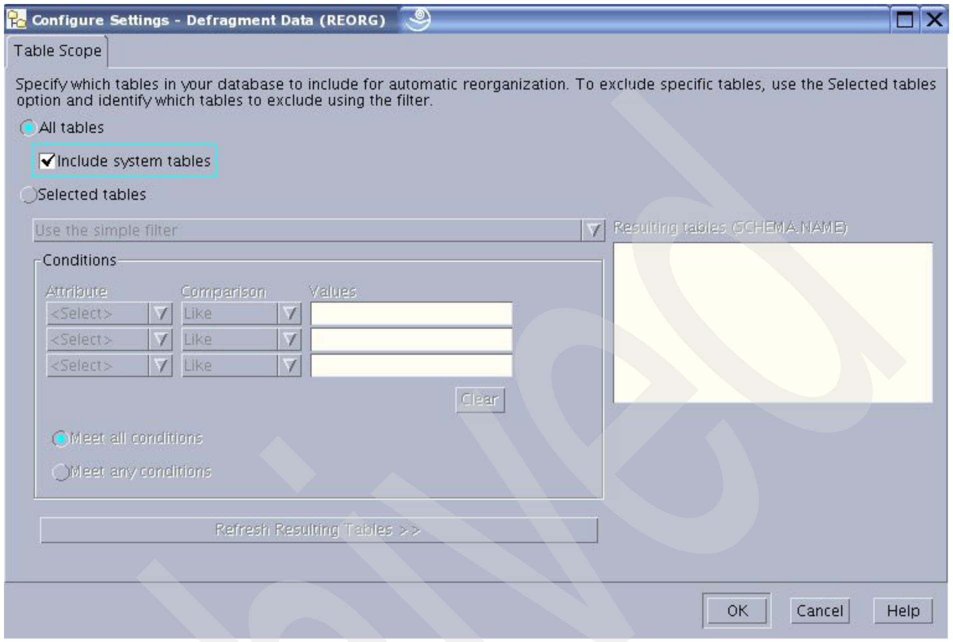
NOTE: Backup images will be automatically pruned from this directory. Please do not put any non-automatic backup images in this directory, as they will also be pruned.

Figure 5-12 Configured backup maintenance

To configure the automatic defragmentation of data (reorg), select **Defragment data (REORG)** in the window and click the **Configure Settings** button. You will get to the reorg configuration page, as shown in Figure 5-13 on page 214. There you can select all tables, with or without system tables, or specify conditions that have to be met for the table before a reorg is run. Click the **OK** button to confirm the settings and get back to the activities page.

To configure the automatic runstats select **Optimize data access RUNSTATS)** in the maintenance activity window and use the **Configure settings** button to get to the runstats configuration page, as shown in Figure 5-14 on page 215. The configuration is similar to the one for reorg. You can select all tables with or without the system tables, or define conditions that have to be fulfilled before a runstats is run. By clicking **OK**, you get back to the activity overview, which should then look similar to the one shown in Figure 5-15 on page 216. Now click **Finish** to apply the configuration or go first to the Summary page by clicking the

Next button. There you can review you settings and, if all is well, apply them with the **Finish** button.



Specify which tables in your database to include for automatic reorganization. To exclude specific tables, use the Selected tables option and identify which tables to exclude using the filter.

[✓ All tables]

[✓ Include system tables]

Selected tables

Figure 5-13 Configure reorg

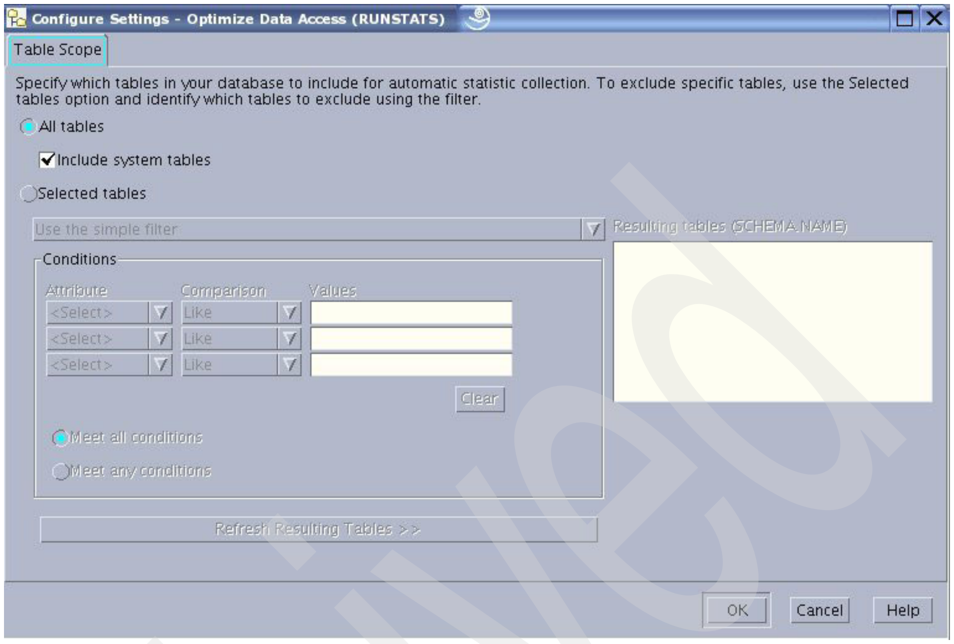

Configure Settings - Optimize Data Access (RUNSTATS)

Table Scope

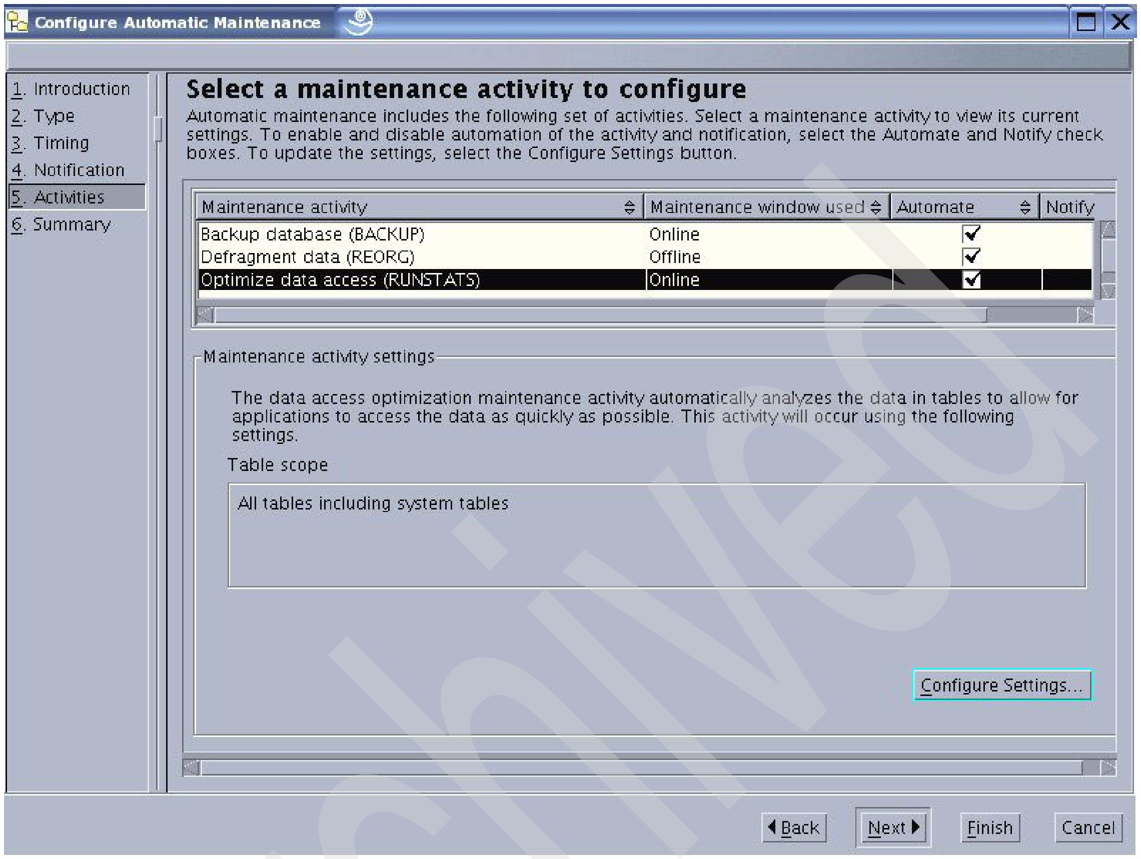
Specify which tables in your database to include for automatic statistic collection. To exclude specific tables, use the Selected tables option and identify which tables to exclude using the filter.

[✓ All tables]

[✓ Include system tables]

Selected tables

Figure 5-14 Configure runstats



Select a maintenance activity to configure

Automatic maintenance includes the following set of activities. Select a maintenance activity to view its current settings. To enable and disable automation of the activity and notification, select the Automate and Notify check boxes. To update the settings, select the Configure Settings button.

Maintenance activity	Maintenance window used	Automate	Notify
Backup database (BACKUP)	Online	✓	
Defragment data (REORG)	Offline	✓	
Optimize data access (RUNSTATS)	Online	✓	

Maintenance activity settings

The data access optimization maintenance activity automatically analyzes the data in tables to allow for applications to access the data as quickly as possible. This activity will occur using the following settings.

Table scope: All tables including system tables

Figure 5-15 All activities configured

The automatic statistics collection and reorganization features store working data in tables in your database. These tables are created in the SYSTOOLSPACE table space. The SYSTOOLSPACE table space is created automatically with default options when the database is activated. Storage requirements for these tables are proportional to the number of tables in the database and should be calculated as approximately 1 KB per table. If this is a significant size for your database, you may want to drop and re-create the table space and allocate storage appropriately. The automatic maintenance and health monitor tables in the table space are automatically re-created. Any history captured in those tables is lost when the table space is dropped.

5.1 Manageability enhancements

In this section, we introduce the new or enhanced DB2 UDB V8 tools, which can help you manage your database system more efficiently by automation.

5.1.1 Health Center enhancements

The Health Center for DB2 UDB V8.2 has a number of new features:

- Health indicator configuration launchpad

  The health indicator configuration launchpad provides easy access to various levels of health indicator settings. This launchpad also helps you to understand how these configurations apply to health monitoring, allowing you to make changes appropriate to your database system environment.

- Recommendation advisor

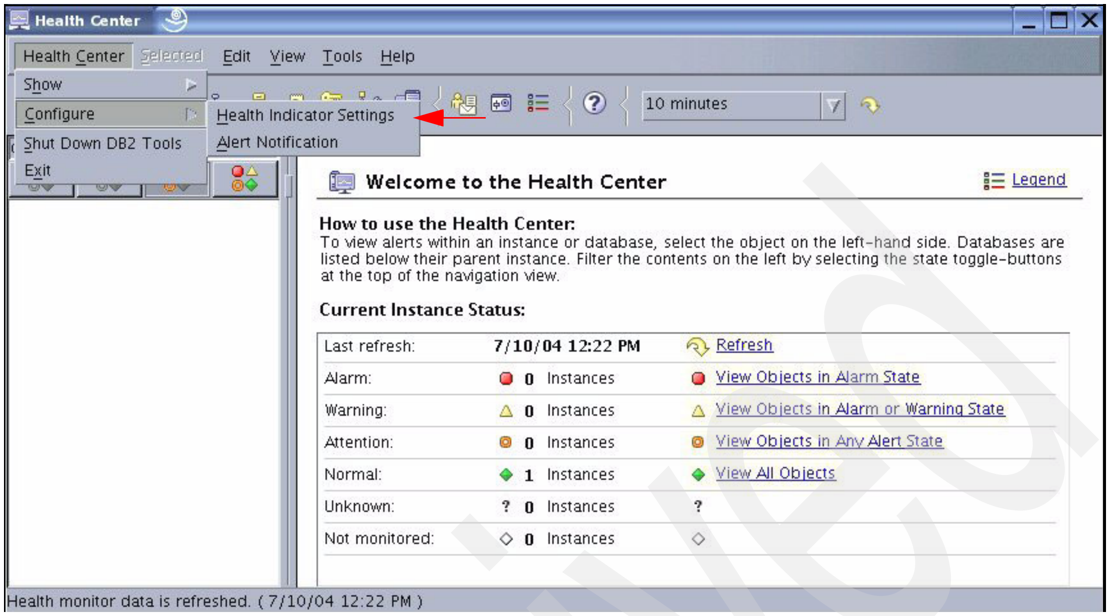
  The Recommendation advisor helps you to resolve health alerts on DB2 objects. This advisor provides recommendations that can correct the issue causing a health alert. In addition, the Recommendation advisor helps you to implement the recommendation that you select, whether this requires launching a tool, running a script, or adjusting the configuration parameter settings on an instance or a database.

- Troubleshoot Health Alert Notification wizard

  The Troubleshoot Health Alert Notification wizard helps you to troubleshoot various problems related to health alert notification, which ensures that notifications are received properly by the correct contacts.

Health indicator configuration launchpad

In this section we show how to use the health indicator configuration launchpad. First of all we have to start the Health Center. This can be done in two ways. One is to start it from the control center via **Tools** → **Health Center**. The second way is to type `db2hc` in a command window to start the Health Center directly. When the Health Center has started, click **Health Center** → **Configure** → **Health Indicator Settings**, as shown in Figure 5-16 on page 218, to start the launchpad.



Welcome to the Health Center

How to use the Health Center:
To view alerts within an instance or database, select the object on the left-hand side. Databases are listed below their parent instance. Filter the contents on the left by selecting the state toggle-buttons at the top of the navigation view.

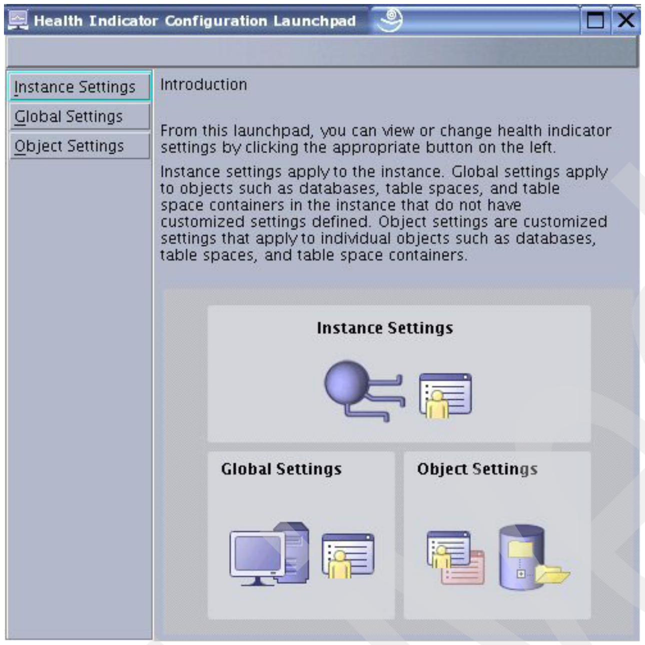
Current Instance Status:

Last refresh:	7/10/04 12:22 PM		Refresh
Alarm:	●	0 Instances	View Objects in Alarm State
Warning:	△	0 Instances	View Objects in Alarm or Warning State
Attention:	◎	0 Instances	View Objects in Any Alert State
Normal:	◆	1 Instances	View All Objects
Unknown:	?	0 Instances	?
Not monitored:	◇	0 Instances	◇

Health monitor data is refreshed. (7/10/04 12:22 PM)

Figure 5-16 Health Center

Figure 5-17 on page 219 shows the started launchpad. We go through the three settings in the following order: Instance settings, global settings, and at the end the object settings.



Introduction

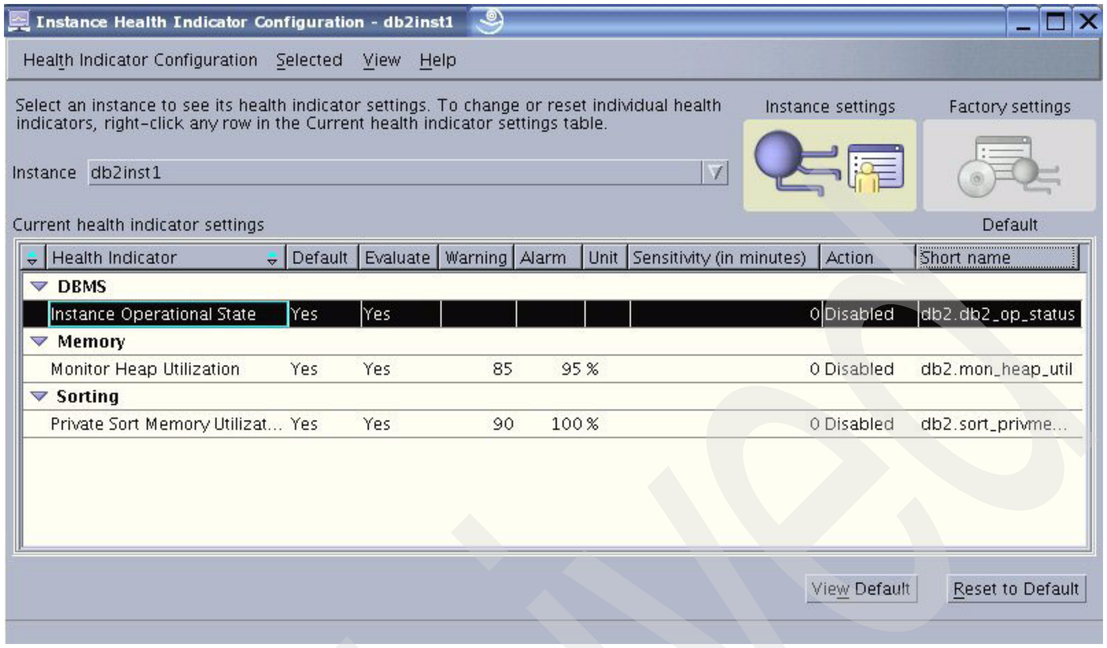
From this launchpad, you can view or change health indicator settings by clicking the appropriate button on the left.

Instance settings apply to the instance. Global settings apply to objects such as databases, table spaces, and table space containers in the instance that do not have customized settings defined. Object settings are customized settings that apply to individual objects such as databases, table spaces, and table space containers.

Figure 5-17 Health indicator configuration launchpad

Instance setting

To arrive at the instance settings click **Instance Settings** at the launchpad. At the instance settings screen, shown in Figure 5-18 on page 220, you have to select the instance you want to configure with the Instance pull-down menu. After having done so, the current health indicator settings will be shown, as in Figure 5-18 on page 220. If you want to change one of the settings right-click the indicator you want to change and you get the Edit command, as in Figure 5-19 on page 220. Now, click **Edit** and you get to the configuration page (Figure 5-20 on page 221).


Instance Health Indicator Configuration - db2inst1
Health Indicator Configuration
Selected
View
Help

Select an instance to see its health indicator settings. To change or reset individual health indicators, right-click any row in the Current health indicator settings table.

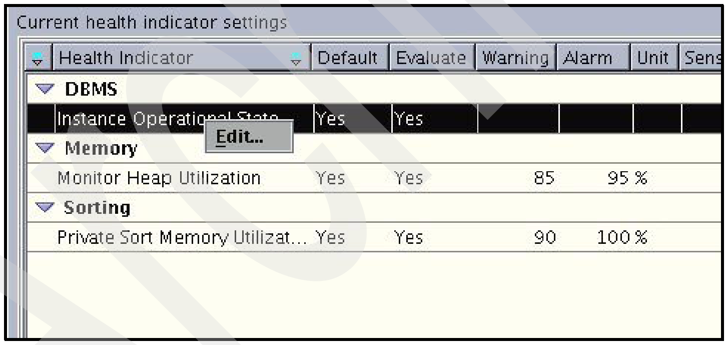
Instance settings
Factory settings
Default

Instance db2inst1

Current health indicator settings

Health Indicator	Default	Evaluate	Warning	Alarm	Unit	Sensitivity (in minutes)	Action	Short name
DBMS								
Instance Operational State	Yes	Yes				0	Disabled	db2.db2_op_status
Memory								
Monitor Heap Utilization	Yes	Yes	85	95	%	0	Disabled	db2.mon_heap_util
Sorting								
Private Sort Memory Utilizat...	Yes	Yes	90	100	%	0	Disabled	db2.sort_privme...

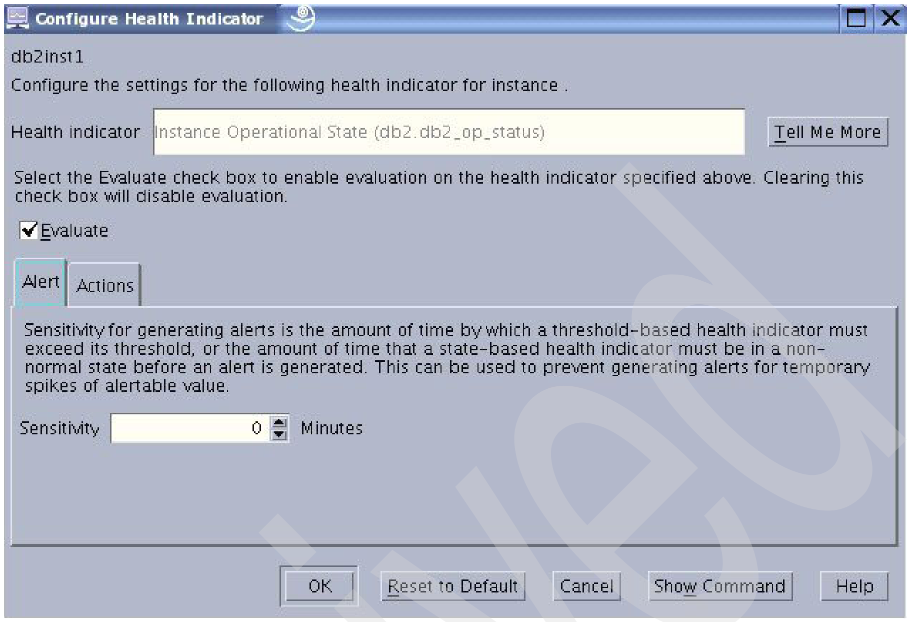
Figure 5-18 Instance configuration - Current settings



Current health indicator settings

Health Indicator	Default	Evaluate	Warning	Alarm	Unit	Sens
DBMS						
Instance Operational State	Yes	Yes				
Memory						
Monitor Heap Utilization	Yes	Yes	85	95	%	
Sorting						
Private Sort Memory Utilizat...	Yes	Yes	90	100	%	

Figure 5-19 Instance configuration - Edit


Configure Health Indicator

db2inst1

Configure the settings for the following health indicator for instance .

Health indicator: Instance Operational State (db2.db2_op_status)

Select the Evaluate check box to enable evaluation on the health indicator specified above. Clearing this check box will disable evaluation.

☑ Evaluate

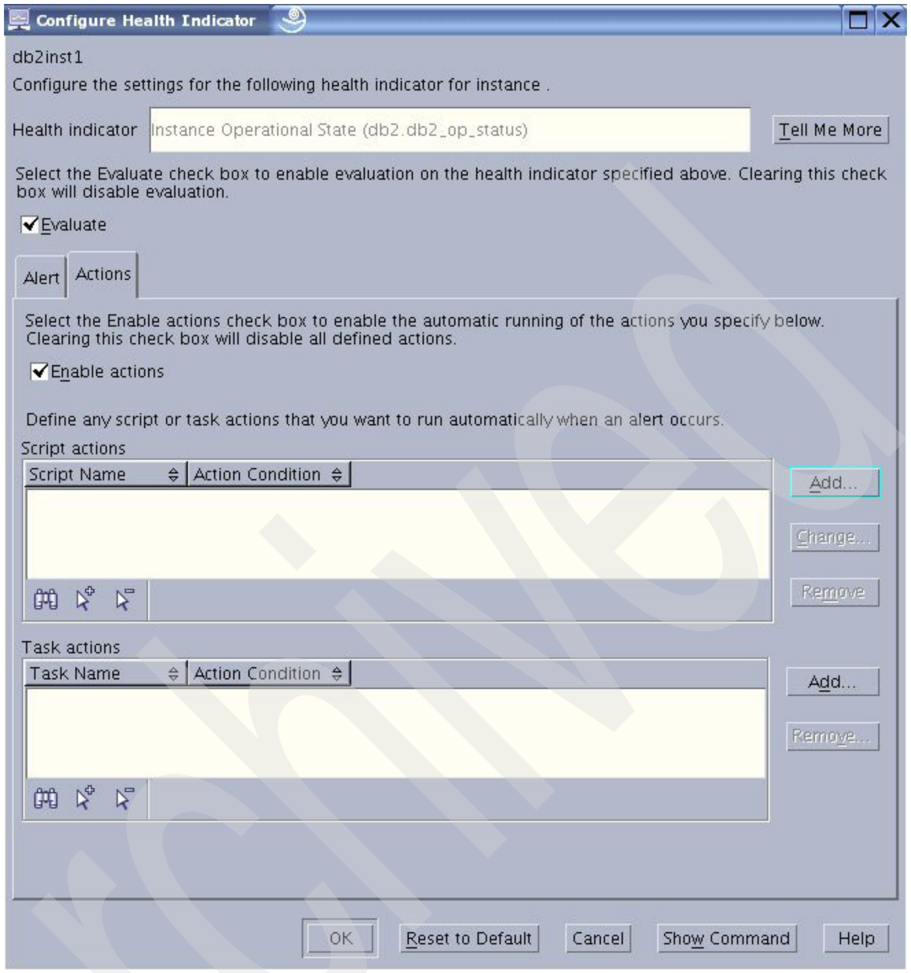
Sensitivity for generating alerts is the amount of time by which a threshold-based health indicator must exceed its threshold, or the amount of time that a state-based health indicator must be in a non-normal state before an alert is generated. This can be used to prevent generating alerts for temporary spikes of alertable value.

Sensitivity: 0 Minutes

Figure 5-20 Configure health indicator - Alert

On the configuration page we can enable or disable the health indicator by checking or unchecking the Evaluate box. On the Alert tab, we can set the sensitivity. To set the actions to take when an alert is triggered, click the **Actions** tab, which is shown in Figure 5-21 on page 222. To enable an action, the Enable actions checkbox has to be selected. Then we must define an action. We have two choices. We can add a script that will run or add a task (tasks can be defined in the Task Center). We show the add script option.

To add a script action, click **Add** beside the Script action box. The following page, shown in Figure 5-22 on page 223, allows you to define the condition on which the script will run via the Condition pull-down menu, the System name (also via pull-down menu), the Name of the script (including path), and the Type (DB2 command or OS command script).



db2inst1

Configure the settings for the following health indicator for instance .

Health indicator: Instance Operational State (db2.db2_op_status)

Select the Evaluate check box to enable evaluation on the health indicator specified above. Clearing this check box will disable evaluation.

[✓] Evaluate

Select the Enable actions check box to enable the automatic running of the actions you specify below. Clearing this check box will disable all defined actions.

[✓] Enable actions

Define any script or task actions that you want to run automatically when an alert occurs.

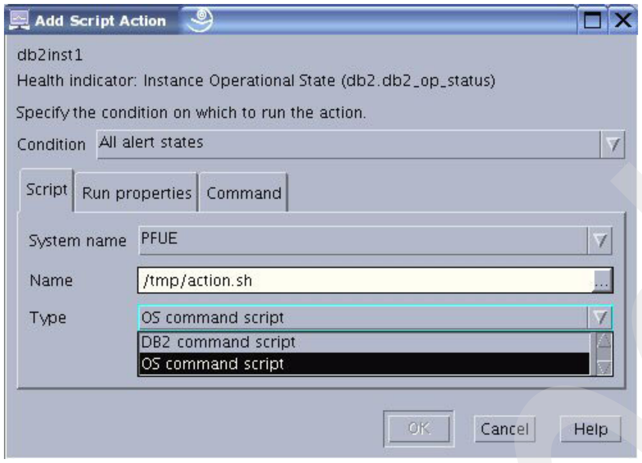
Script actions

Script Name	Action Condition

Task actions

Task Name	Action Condition

Figure 5-21 Configure health indicator - Actions


Add Script Action

db2inst1

Health indicator: Instance Operational State (db2.db2_op_status)

Specify the condition on which to run the action.

Condition: All alert states

Script | Run properties | Command

System name: PFUE

Name: /tmp/action.sh

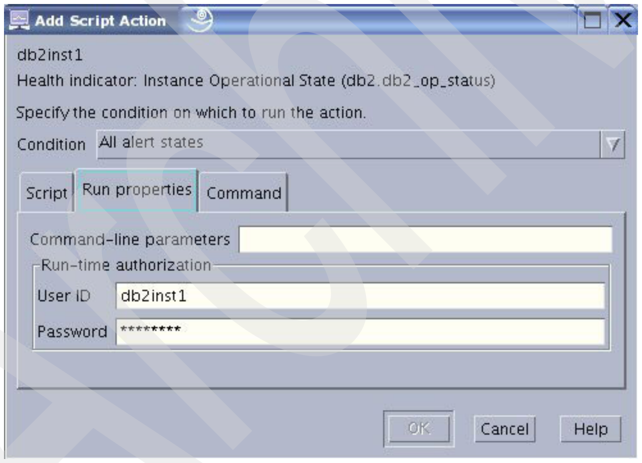
Type: OS command script

DB2 command script

OS command script

Figure 5-22 Add script - Script

To change to the Run properties tab shown in Figure 5-23 just click **Run properties**. On this tab you have the option to assign command-line parameters and enter the run-time authorization in the corresponding text boxes.


Add Script Action

db2inst1

Health indicator: Instance Operational State (db2.db2_op_status)

Specify the condition on which to run the action.

Condition: All alert states

Script | Run properties | Command

Command-line parameters:

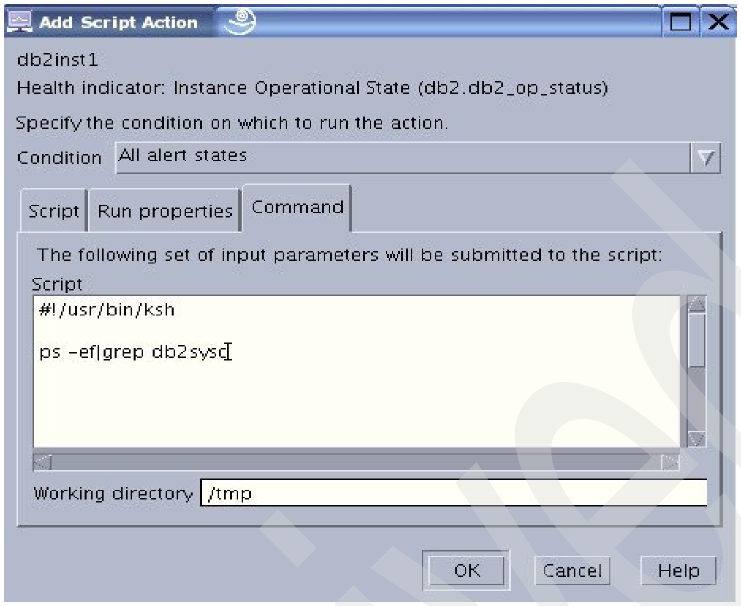
Run-time authorization

User ID: db2inst1

Password: ********

Figure 5-23 Add script - Run properties

The last tab is the Command tab, shown in Figure 5-24 on page 224. In this tab you can optionally, if changes are required, edit the script content in the Script text box. The second thing in this tab is the working directory, which you have to define. After configuring everything, click **OK**.



db2inst1

Health indicator: Instance Operational State (db2.db2_op_status)

Specify the condition on which to run the action.

Condition: All alert states

The following set of input parameters will be submitted to the script:

Script

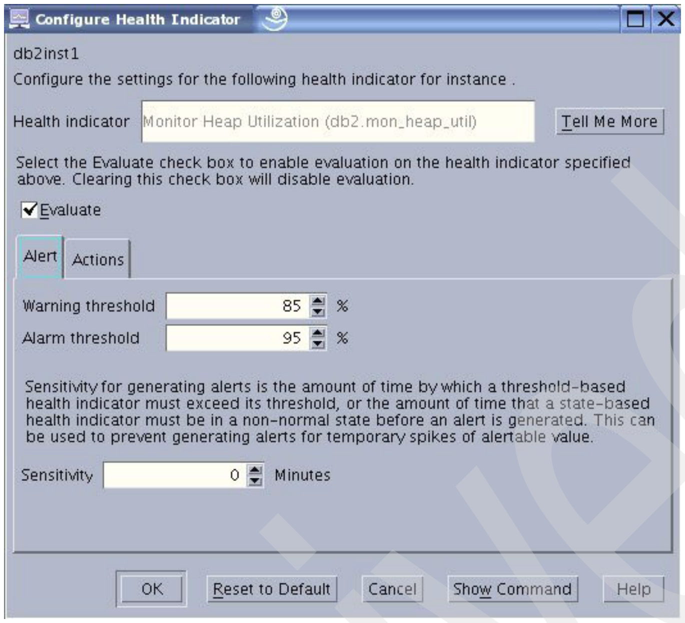
```
#!/usr/bin/ksh

ps -ef|grep db2sysc
```

Working directory: /tmp

Figure 5-24 Add script - Command

We also want to show you how the screen would look when you edit the setting for a health indicator that has a warning and alarm value. To show you this we took the monitor heap utilization health indicator, shown in Figure 5-25 on page 225. You see that the Alert tab is a bit different. Besides the option for the Sensitivity, you can also alter the warning threshold and the alarm threshold. The Actions tab and its configuration is identical to the one shown in Figure 5-21 on page 222 through Figure 5-24.


Configure Health Indicator

db2inst1

Configure the settings for the following health indicator for instance .

Health indicator: Monitor Heap Utilization (db2.mon_heap_util)

Select the Evaluate check box to enable evaluation on the health indicator specified above. Clearing this check box will disable evaluation.

✓ Evaluate

Warning threshold: 85 %

Alarm threshold: 95 %

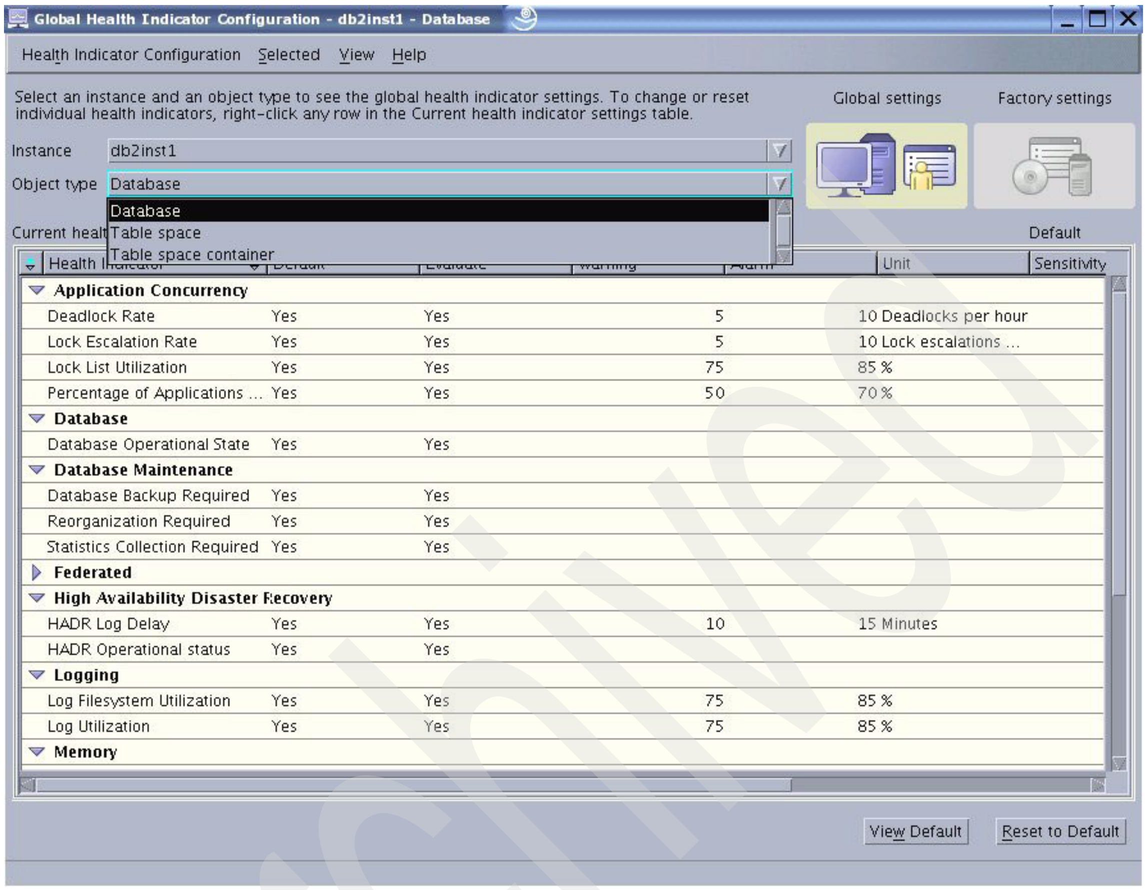
Sensitivity for generating alerts is the amount of time by which a threshold-based health indicator must exceed its threshold, or the amount of time that a state-based health indicator must be in a non-normal state before an alert is generated. This can be used to prevent generating alerts for temporary spikes of alertable value.

Sensitivity: 0 Minutes

Figure 5-25 Configure health indicator - Alert with thresholds

Global setting

We have now finished the instance settings section and proceed to the global settings. You get there by clicking **Global Settings** on the launchpad (Figure 5-17 on page 219). You get to the screen shown in Figure 5-26 on page 226, where you have to again select the instance and the object type. You can choose between database shown in Figure 5-26 on page 226, the table space shown in Figure 5-27 on page 227, and the table space container shown in Figure 5-28 on page 227. The design is equal to the one for the instance settings. The same is true for the editing of the health indicators; the procedures are the same as the ones for the instance setting. We will not show them here again.


Global Health Indicator Configuration - db2inst1 - Database
Health Indicator Configuration
Selected
View
Help
Global settings
Factory settings
View Default
Reset to Default

Select an instance and an object type to see the global health indicator settings. To change or reset individual health indicators, right-click any row in the Current health indicator settings table.

Instance: db2inst1

Object type: Database

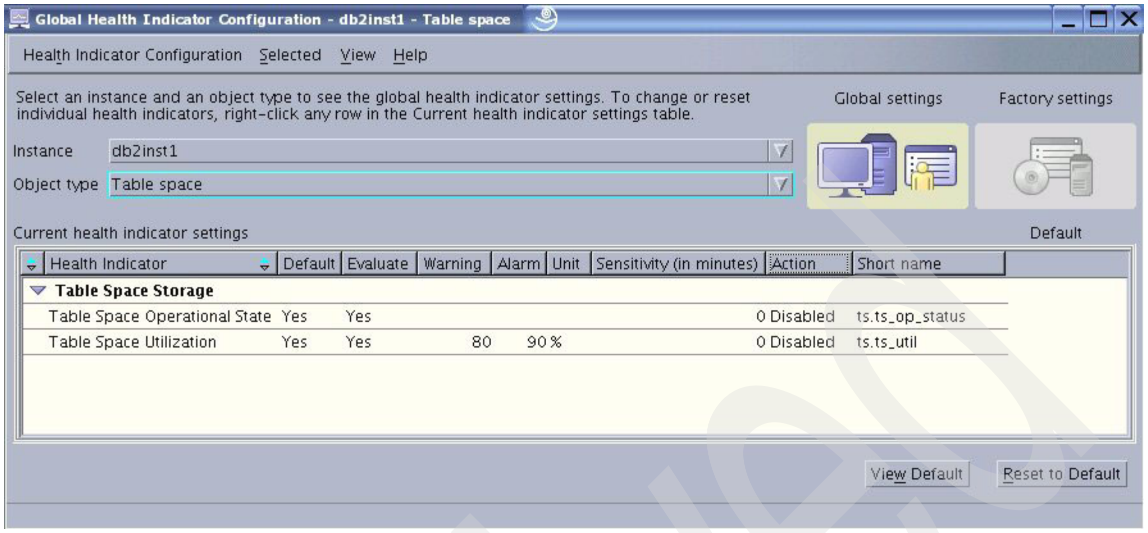
Database
Table space
Table space container

Current healt

Default

Health Indicator	Default	Evaluate	Warning	Alarm	Unit	Sensitivity
Application Concurrency						
Deadlock Rate	Yes	Yes	5	10	Deadlocks per hour	
Lock Escalation Rate	Yes	Yes	5	10	Lock escalations ...	
Lock List Utilization	Yes	Yes	75	85	%	
Percentage of Applications ...	Yes	Yes	50	70	%	
Database						
Database Operational State	Yes	Yes				
Database Maintenance						
Database Backup Required	Yes	Yes				
Reorganization Required	Yes	Yes				
Statistics Collection Required	Yes	Yes				
Federated						
High Availability Disaster Recovery						
HADR Log Delay	Yes	Yes	10	15	Minutes	
HADR Operational status	Yes	Yes				
Logging						
Log Filesystem Utilization	Yes	Yes	75	85	%	
Log Utilization	Yes	Yes	75	85	%	
Memory						

Figure 5-26 Global settings - Database


Global Health Indicator Configuration - db2inst1 - Table space
Health Indicator Configuration
Selected
View
Help
Global settings
Factory settings
Default
View Default
Reset to Default

Select an instance and an object type to see the global health indicator settings. To change or reset individual health indicators, right-click any row in the Current health indicator settings table.

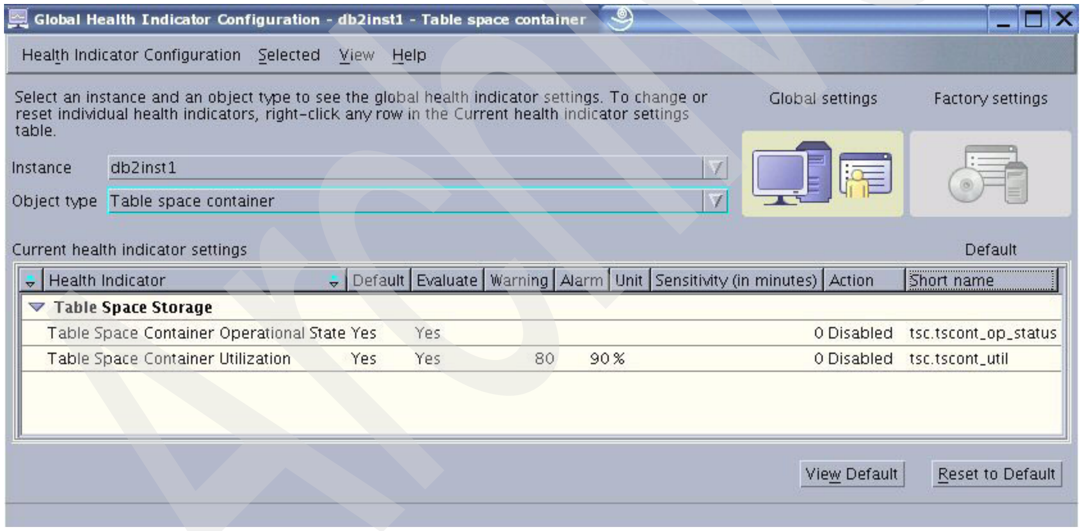
Instance: db2inst1

Object type: Table space

Current health indicator settings

Health Indicator	Default	Evaluate	Warning	Alarm	Unit	Sensitivity (in minutes)	Action	Short name
Table Space Storage								
Table Space Operational State	Yes	Yes				0	Disabled	ts.ts_op_status
Table Space Utilization	Yes	Yes	80	90	%	0	Disabled	ts.ts_util

Figure 5-27 Global settings - Table space


Global Health Indicator Configuration - db2inst1 - Table space container
Health Indicator Configuration
Selected
View
Help
Global settings
Factory settings
Default
View Default
Reset to Default

Select an instance and an object type to see the global health indicator settings. To change or reset individual health indicators, right-click any row in the Current health indicator settings table.

Instance: db2inst1

Object type: Table space container

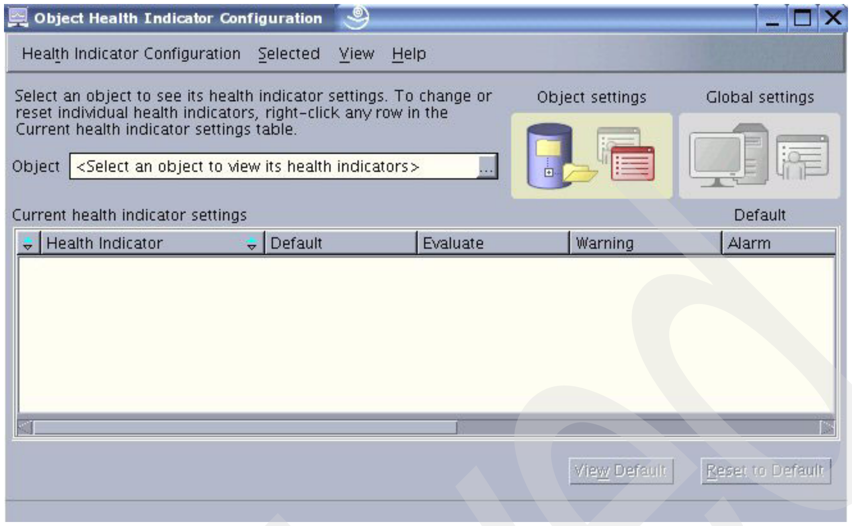
Current health indicator settings

Health Indicator	Default	Evaluate	Warning	Alarm	Unit	Sensitivity (in minutes)	Action	Short name
Table Space Storage								
Table Space Container Operational State	Yes	Yes				0	Disabled	tsc.tscont_op_status
Table Space Container Utilization	Yes	Yes	80	90	%	0	Disabled	tsc.tscont_util

Figure 5-28 Global settings - Table space container

Object setting

The last thing that now remains is the object setting. You get to object settings configuration page by clicking **Object setting** at the launchpad (Figure 5-17 on page 219). After having done so, the screen shown in Figure 5-29 on page 228 will pop up.



Select an object to see its health indicator settings. To change or reset individual health indicators, right-click any row in the Current health indicator settings table.

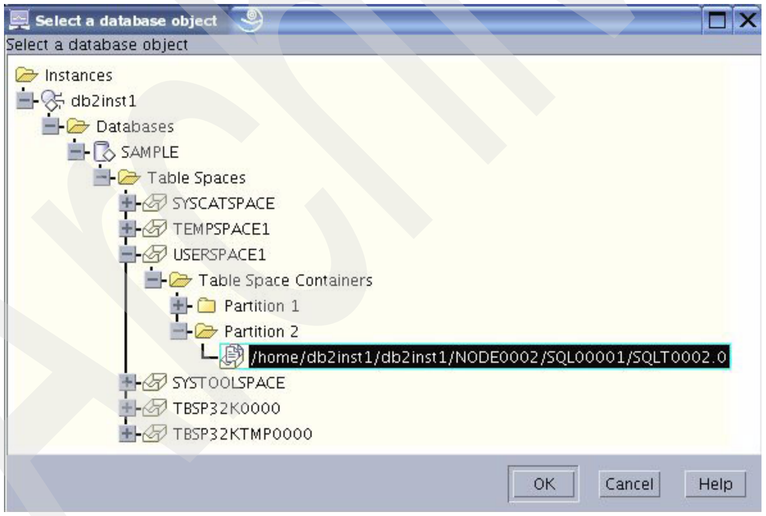
Object <Select an object to view its health indicators>

Current health indicator settings

Health Indicator	Default	Evaluate	Warning	Alarm

Figure 5-29   Object settings

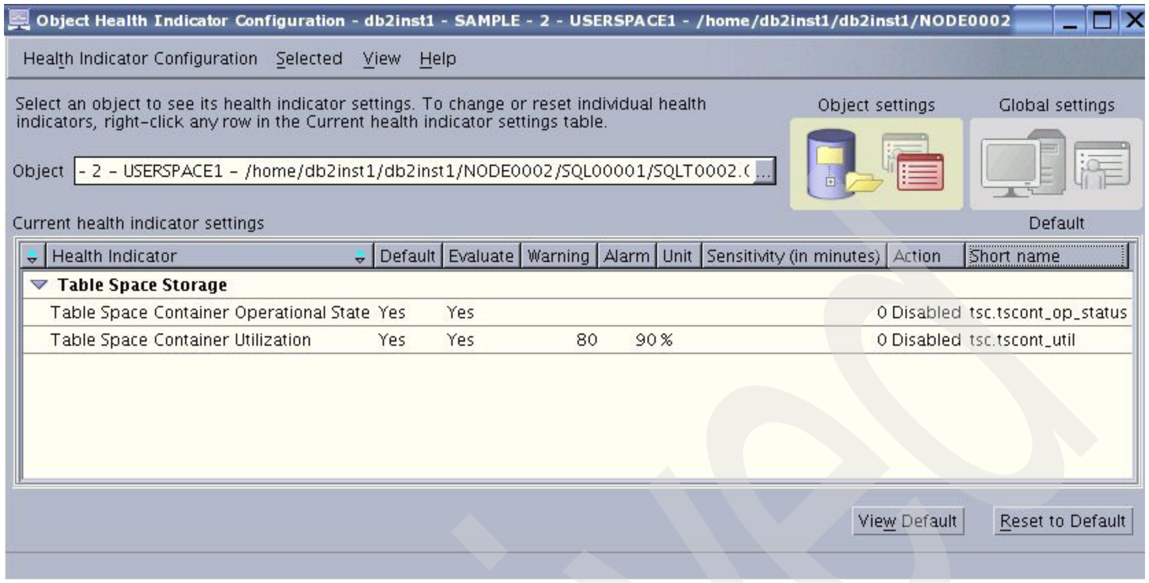
We now must select an object. Do so by clicking the **...** button beside the Object text box. The Select a database object page appears as shown in Figure 5-30.



```
Instances
  db2inst1
    Databases
      SAMPLE
        Table Spaces
          SYSCATSPACE
          TEMPSPACE1
          USERSPACE1
            Table Space Containers
              Partition 1
              Partition 2
                /home/db2inst1/db2inst1/NODE0002/SQL00001/SQLT0002.0
          SYSTOOLSPACE
          TBSP32K0000
          TBSP32KTMP0000
```

Figure 5-30   Object settings - Select a database object

Select an object by clicking it and then click the **OK** button. The object settings screen has been updated with the object we just selected. This can be seen in Figure 5-31 on page 229. Altering of the settings is done the same way as described for the instance settings, so we will not mention that again here.


Object Health Indicator Configuration - db2inst1 - SAMPLE - 2 - USERSPACE1 - /home/db2inst1/db2inst1/NODE0002

Health Indicator Configuration Selected View Help

Select an object to see its health indicator settings. To change or reset individual health indicators, right-click any row in the Current health indicator settings table.

Object: - 2 - USERSPACE1 - /home/db2inst1/db2inst1/NODE0002/SQL00001/SQLT0002.(

Object settings | Global settings

Current health indicator settings | Default

Health Indicator	Default	Evaluate	Warning	Alarm	Unit	Sensitivity (in minutes)	Action	Short name
Table Space Storage								
Table Space Container Operational State	Yes	Yes				0	Disabled	tsc.tscont_op_status
Table Space Container Utilization	Yes	Yes	80	90	%	0	Disabled	tsc.tscont_util

View Default Reset to Default

Figure 5-31   Object settings - Updated for one object

Note: The difference between the global and the object health indicator settings is that the global settings provide the default for objects of that type, and the object settings are specific to a single object

The configuration (for instance, global and object settings) can also be done in a command window with the syntax shown in Example 5-1.

Example 5-1   Update alert configuration command

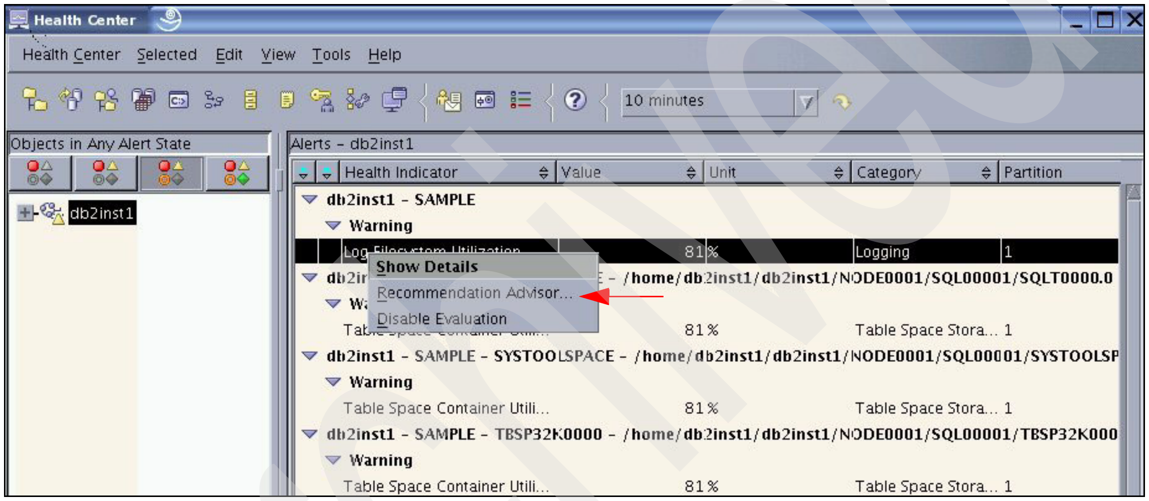
```
UPDATE ALERT CONFIGURATION FOR {{CONTAINERS | DATABASE MANAGER | DATABASES
|TABLESPACES} | {CONTAINER container-name FOR tblspace-name | DATABASE
|TABLESPACE tblspace-name} ON database-alias} USING health-indicator-name {SET
parameter-name value [ {,parameter-name value} ... ] | UPDATE ACTION {SCRIPT
pathname | TASK task-name} on {ALARM | ALLALERT | ATTENTION state | WARNING}
SET parameter-name value [ {,parameter-name value} ... ] [ {,{SCRIPT pathname |
TASK task-name} ON {ALARM | ALLALERT | ATTENTION state | WARNING} SET
parameter-name value [ {,parameter-name value} ... ]} ... ] | DELETE ACTION
{SCRIPT pathname | TASK task-name} ON {ALARM | ALLALERT | ATTENTION state |
WARNING} [ {,{SCRIPT pathname | TASK task-name} ON {ALARM | ALLALERT |
ATTENTION state | WARNING}} ... ] ADD ACTION {SCRIPT pathname
Add-Script-Details | TASK task-name} ON {ALARM | ALLALERT | ATTENTION state |
WARNING} [ {,{SCRIPT pathname add-script-details | TASK task-name} ON {ALARM |
ALLALERT | ATTENTION state | WARNING}} ... ] [ON hostname] USER username USING
password}
```

```
Add-Script-Details: TYPE {DB2 [STATEMENT TERMINATION CHARACTER character] |
OPERATING SYSTEM [COMMAND LINE PARAMETERS parms]} WORKING DIRECTORY pathname
```

Recommendation advisor

The Recommendation advisor supports you in finding the right solution for your problem. We show you here how this is done with the graphical user interface (GUI). We also present the command-line syntax at the end.

The Recommendation advisor is started out of the Health Center. Right-click a warning or alert (we took, as an example, a log file system utilization warning) and select the **Recommendation Advisor**, as shown in Figure 5-32.

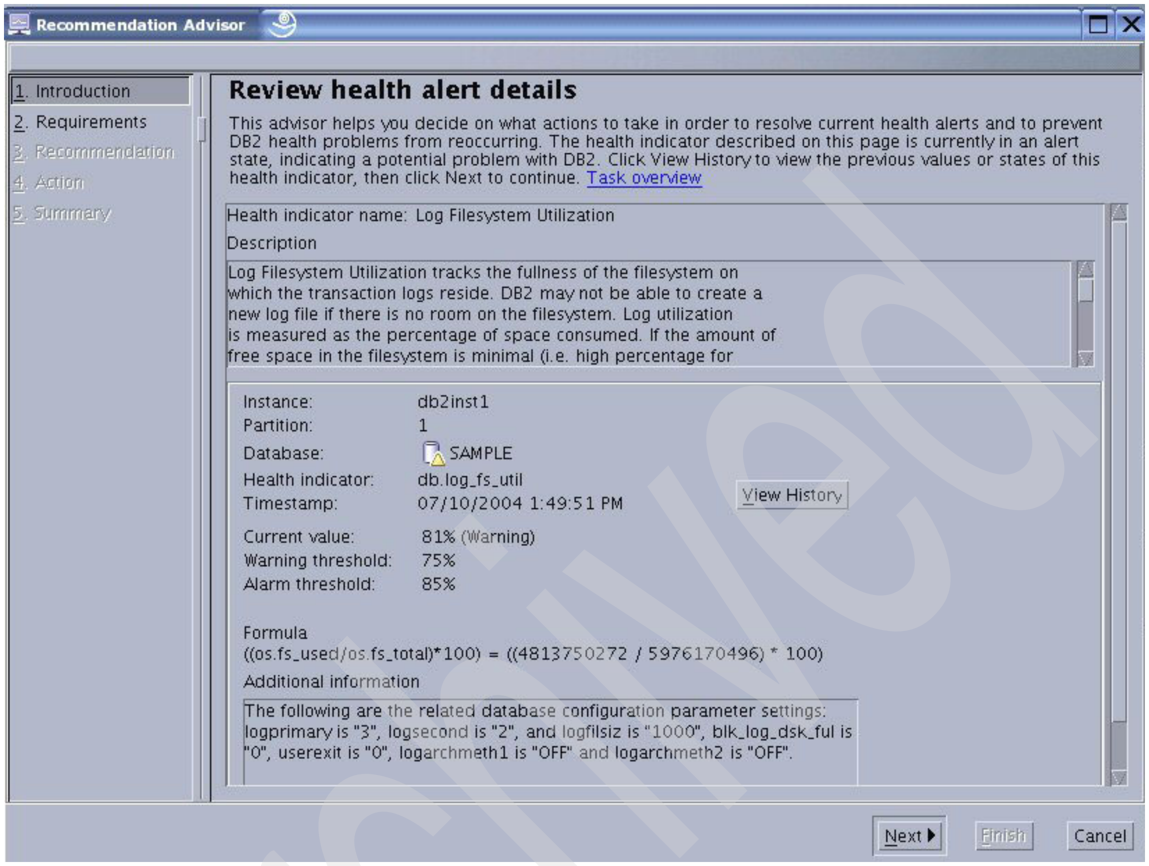


Health Center
Health Center Selected Edit View Tools Help
10 minutes
Objects in Any Alert State
db2inst1
Alerts - db2inst1
Show Details
Recommendation Advisor...
Disable Evaluation

Health Indicator	Value	Unit	Category	Partition
db2inst1 - SAMPLE				
Warning				
Log Filesystem Utilization	81 %		Logging	1
db2in... E - /home/db2inst1/db2inst1/NODE0001/SQL00001/SQLT0000.0				
Wa...				
Tab...	81 %		Table Space Stora...	1
db2inst1 - SAMPLE - SYSTOOLSPACE - /home/db2inst1/db2inst1/NODE0001/SQL00001/SYSTOOLSP				
Warning				
Table Space Container Utili...	81 %		Table Space Stora...	1
db2inst1 - SAMPLE - TBSP32K0000 - /home/db2inst1/db2inst1/NODE0001/SQL00001/TBSP32K000				
Warning				
Table Space Container Utili...	81 %		Table Space Stora...	1

Figure 5-32   Start the Recommendation Advisor

We are now on the first page of the Recommendation advisor, which gives a introduction to the problem, as shown in Figure 5-33 on page 231. After having read the introduction we go on by clicking the **Next** button.


Recommendation Advisor

1. Introduction
2. Requirements
3. Recommendation
4. Action
5. Summary

Review health alert details

This advisor helps you decide on what actions to take in order to resolve current health alerts and to prevent DB2 health problems from reoccurring. The health indicator described on this page is currently in an alert state, indicating a potential problem with DB2. Click View History to view the previous values or states of this health indicator, then click Next to continue. Task overview

Health indicator name: Log Filesystem Utilization

Description

Log Filesystem Utilization tracks the fullness of the filesystem on which the transaction logs reside. DB2 may not be able to create a new log file if there is no room on the filesystem. Log utilization is measured as the percentage of space consumed. If the amount of free space in the filesystem is minimal (i.e. high percentage for

Instance:	db2inst1
Partition:	1
Database:	SAMPLE
Health indicator:	db.log_fs_util
Timestamp:	07/10/2004 1:49:51 PM
Current value:	81% (Warning)
Warning threshold:	75%
Alarm threshold:	85%

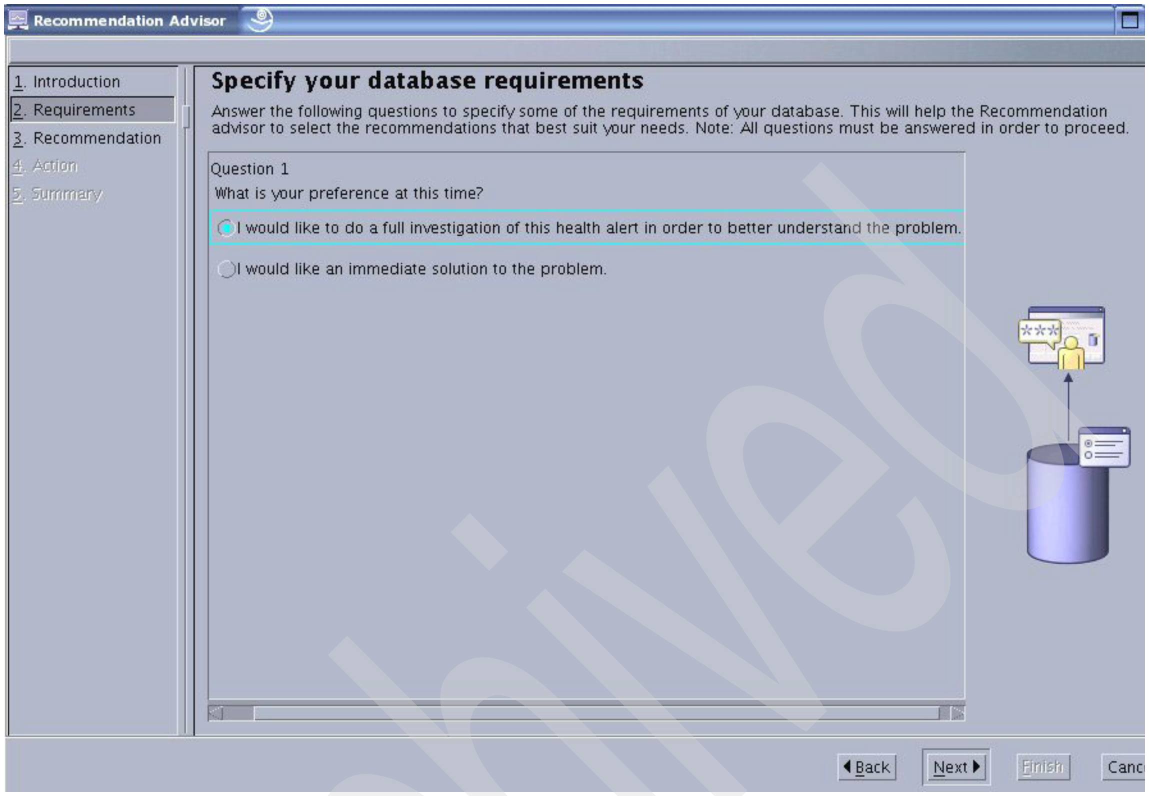
Formula
((os.fs_used/os.fs_total)*100) = ((4813750272 / 5976170496) * 100)

Additional information

The following are the related database configuration parameter settings: logprimary is "3", logsecond is "2", and logfilsiz is "1000", blk_log_dsk_ful is "0", userexit is "0", logarchmeth1 is "OFF" and logarchmeth2 is "OFF".

Figure 5-33 Recommendation advisor - Introduction

We are now on the Requirements page, shown in Figure 5-34 on page 232, where we have to decide if we want to have an immediate action or if we want to do a full investigation of the warning or alert. We show both in this publication, starting with the full investigation (which is shown in Figure 5-34 on page 232 through Figure 5-37 on page 235) and then discuss the immediate solution (which is shown in Figure 5-38 on page 236 through Figure 5-41 on page 239).



Specify your database requirements

Answer the following questions to specify some of the requirements of your database. This will help the Recommendation advisor to select the recommendations that best suit your needs. Note: All questions must be answered in order to proceed.

Question 1

What is your preference at this time?

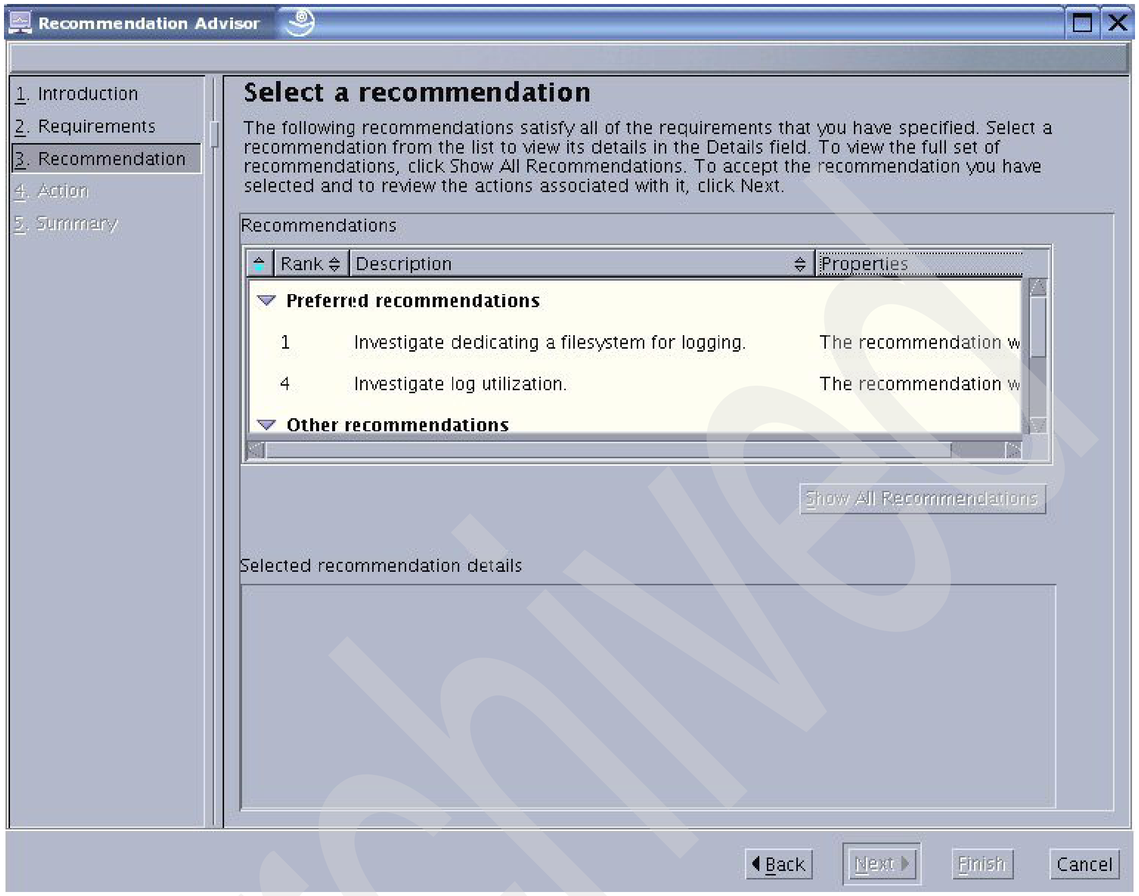
[✓ I would like to do a full investigation of this health alert in order to better understand the problem.]

I would like an immediate solution to the problem.

Figure 5-34 Recommendation advisor - Requirements - Full investigation

Full investigation

For the full investigation we select the “I would like to do a full investigation of this health alert in order to better understand the problem” radio button (also see Figure 5-34). To get to the Recommendation page, shown in Figure 5-35 on page 233, click the **Next** button.


Recommendation Advisor

1. Introduction
2. Requirements
3. Recommendation
4. Action
5. Summary

Select a recommendation

The following recommendations satisfy all of the requirements that you have specified. Select a recommendation from the list to view its details in the Details field. To view the full set of recommendations, click Show All Recommendations. To accept the recommendation you have selected and to review the actions associated with it, click Next.

Recommendations

Rank	Description	Properties
Preferred recommendations		
1	Investigate dedicating a filesystem for logging.	The recommendation w
4	Investigate log utilization.	The recommendation w
Other recommendations		

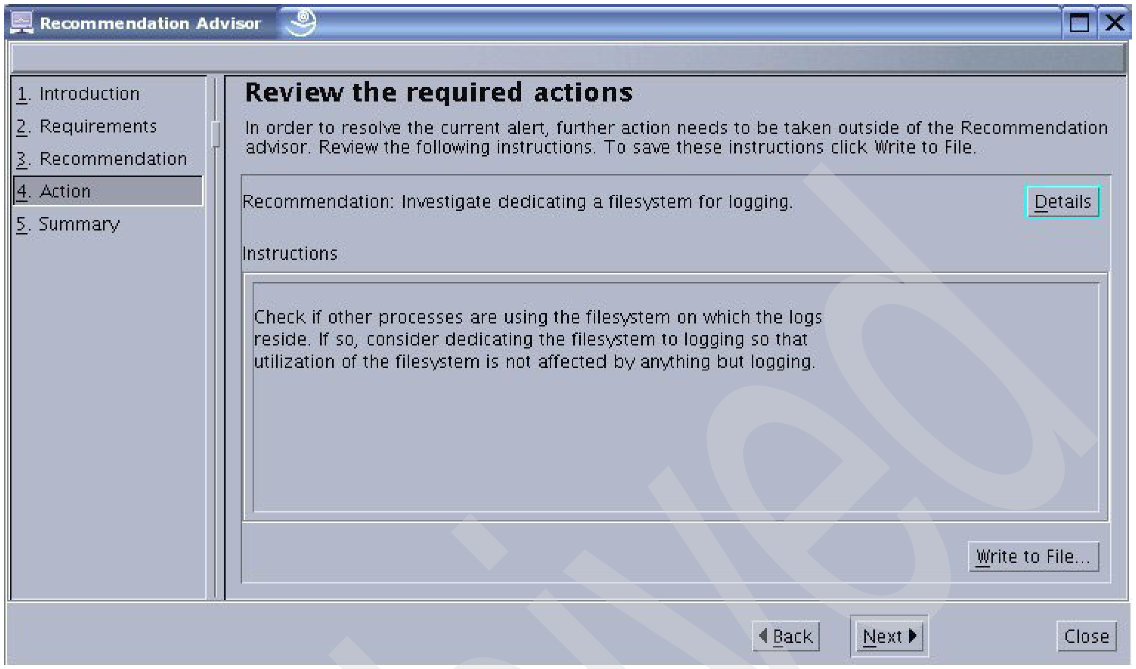
Show All Recommendations

Selected recommendation details

Back | Next | Finish | Cancel

Figure 5-35 Recommendation advisor - Recommendation - Full investigation

On the Recommendation page you see the recommendations that the Recommendation advisor provides. The recommendations are ranked with numbers (while 1 is the highest rank). When you click the **Show All Recommendations** button you also see the immediate actions under Other recommendations. We choose the action that is ranked with 1, the “Investigate dedicating a file system for logging”. Do so by clicking it, and then click the **Next** button to get to the Action page shown in Figure 5-36 on page 234.



Recommendation Advisor

Review the required actions

In order to resolve the current alert, further action needs to be taken outside of the Recommendation advisor. Review the following instructions. To save these instructions click Write to File.

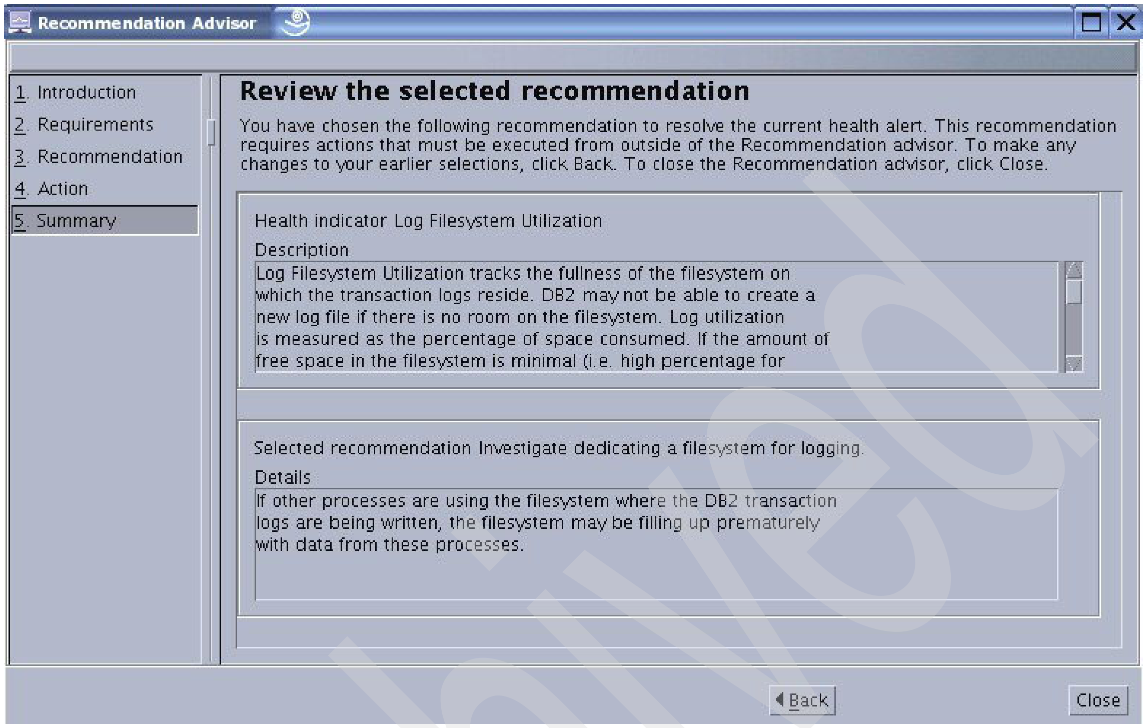
Recommendation: Investigate dedicating a filesystem for logging.

Instructions

Check if other processes are using the filesystem on which the logs reside. If so, consider dedicating the filesystem to logging so that utilization of the filesystem is not affected by anything but logging.

Figure 5-36 Recommendation advisor - Action - Full investigation

On the Action page we see the instructions to resolve the problem. You can also write the information to a file with the Write to File button. To get to the Summary page click the **Next** button. On the Summary page, shown in Figure 5-37 on page 235, you have a description of the health indicator that gave the warning or alert, and also details about the selected recommendation. To close the wizard click **Close**.



Review the selected recommendation

You have chosen the following recommendation to resolve the current health alert. This recommendation requires actions that must be executed from outside of the Recommendation advisor. To make any changes to your earlier selections, click Back. To close the Recommendation advisor, click Close.

Health indicator Log Filesystem Utilization

Description

Log Filesystem Utilization tracks the fullness of the filesystem on which the transaction logs reside. DB2 may not be able to create a new log file if there is no room on the filesystem. Log utilization is measured as the percentage of space consumed. If the amount of free space in the filesystem is minimal (i.e. high percentage for

Selected recommendation Investigate dedicating a filesystem for logging.

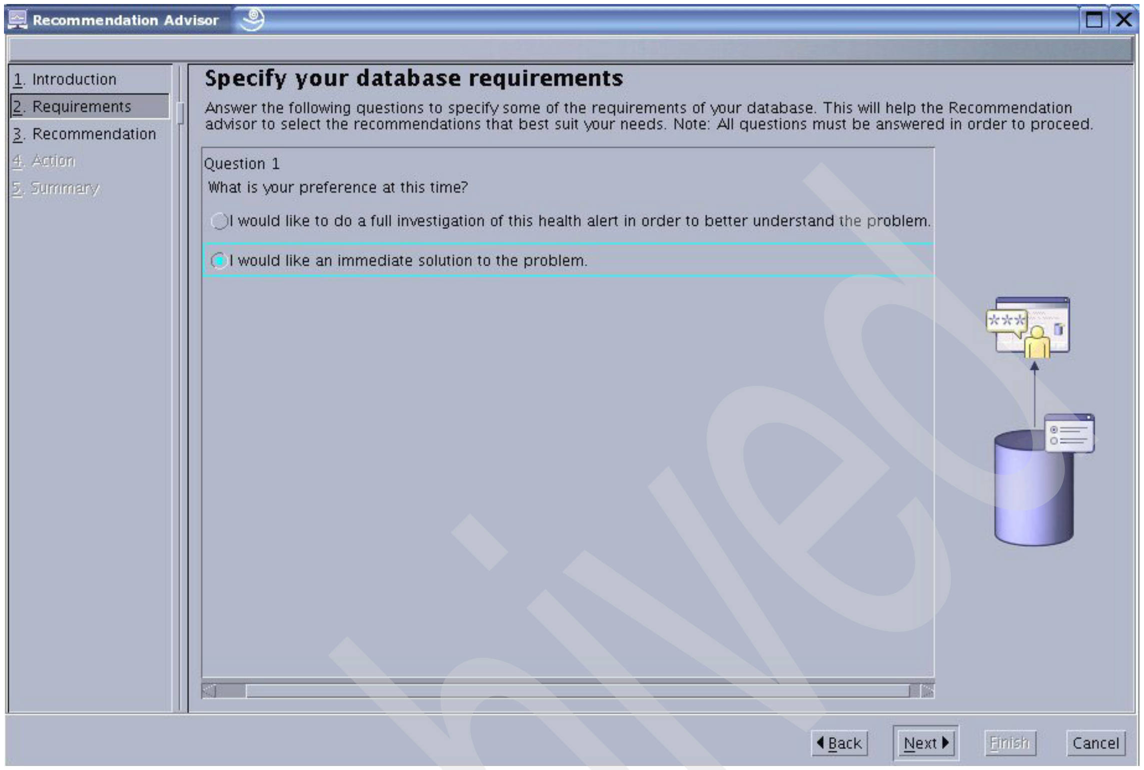
Details

If other processes are using the filesystem where the DB2 transaction logs are being written, the filesystem may be filling up prematurely with data from these processes.

Figure 5-37 Recommendation advisor - summary - full investigation

Immediate solution

We now have a look what we get when we use the “I would like an immediate solution to the problem” radio button on the Requirements page, as shown in Figure 5-38 on page 236. After selecting the radio button, click **Next** to get to the Recommendation page.



Specify your database requirements

Answer the following questions to specify some of the requirements of your database. This will help the Recommendation advisor to select the recommendations that best suit your needs. Note: All questions must be answered in order to proceed.

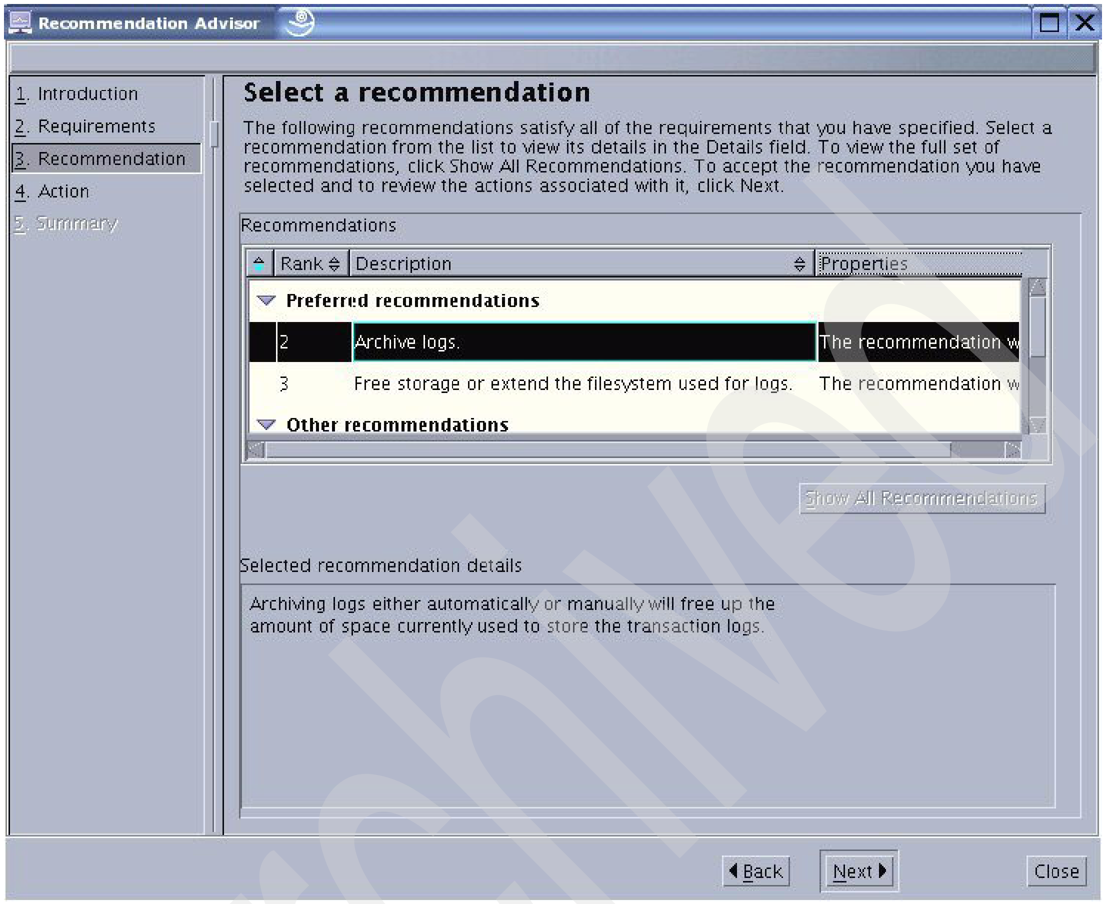
Question 1

What is your preference at this time?

- I would like to do a full investigation of this health alert in order to better understand the problem.
- [✓ I would like an immediate solution to the problem.]

Figure 5-38 Recommendation advisor - Requirements - Immediate solution

On the Recommendation page, shown in Figure 5-39 on page 237, we can again choose the action we want. We select the one with rank 2, the archive logs. So click on **Archive logs** and then click the **Next** button to get to the Action page, shown in Figure 5-40 on page 238.



Select a recommendation

The following recommendations satisfy all of the requirements that you have specified. Select a recommendation from the list to view its details in the Details field. To view the full set of recommendations, click Show All Recommendations. To accept the recommendation you have selected and to review the actions associated with it, click Next.

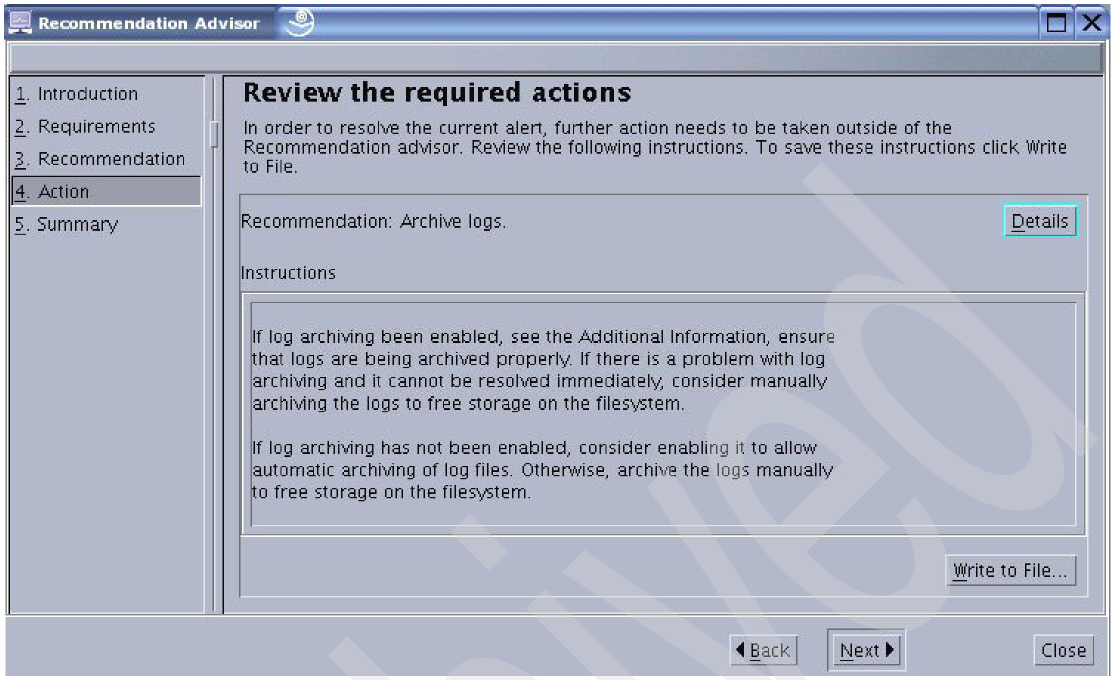
Recommendations

Rank	Description	Properties
Preferred recommendations		
2	Archive logs.	The recommendation w
3	Free storage or extend the filesystem used for logs.	The recommendation w
Other recommendations		

Selected recommendation details

Archiving logs either automatically or manually will free up the amount of space currently used to store the transaction logs.

Figure 5-39 Recommendation advisor - Recommendation - Immediate solution



Review the required actions

In order to resolve the current alert, further action needs to be taken outside of the Recommendation advisor. Review the following instructions. To save these instructions click Write to File.

Recommendation: Archive logs.

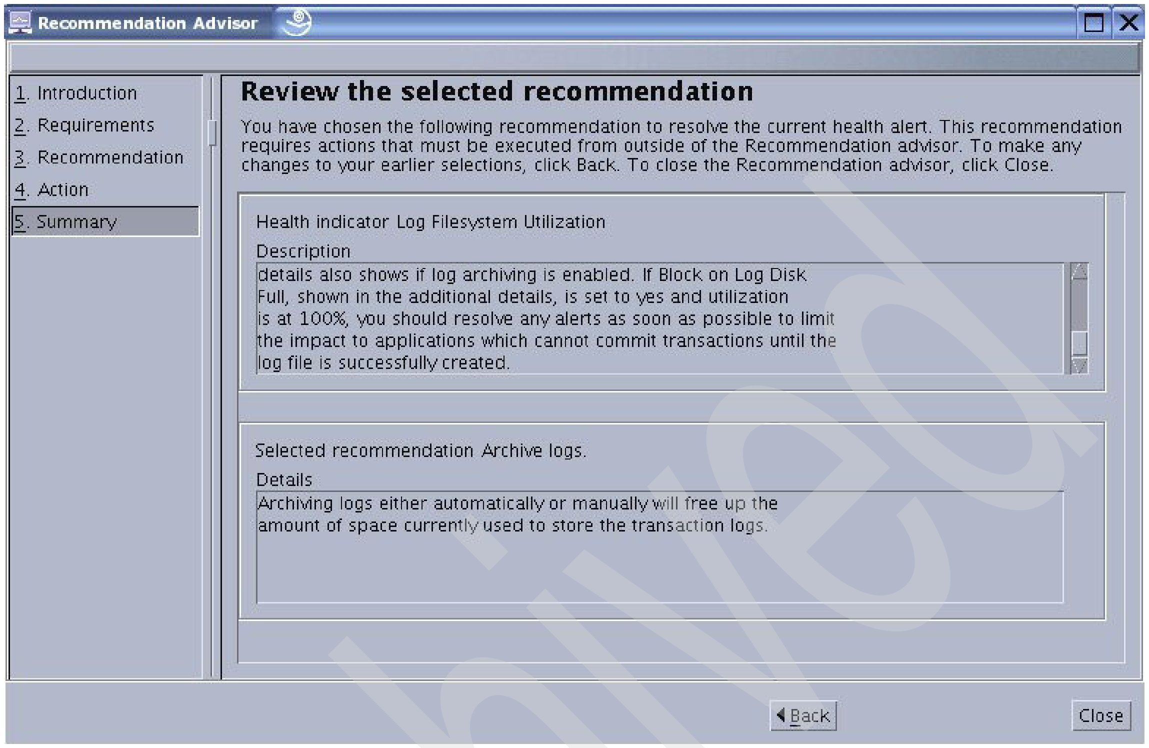
Instructions

If log archiving been enabled, see the Additional Information, ensure that logs are being archived properly. If there is a problem with log archiving and it cannot be resolved immediately, consider manually archiving the logs to free storage on the filesystem.

If log archiving has not been enabled, consider enabling it to allow automatic archiving of log files. Otherwise, archive the logs manually to free storage on the filesystem.

Figure 5-40 Recommendation advisor - Action - Immediate solution

On the Action page we get instructions on how to solve the problem, and also have the possibility to save the instructions to a file with the Write to File button. Click **Next** to get to the Summary page (Figure 5-41 on page 239), where we get a description of the health indicator which is in the warning or alert state, and also details of the actions that should be taken. Close the wizard by clicking **Close**.



Review the selected recommendation

You have chosen the following recommendation to resolve the current health alert. This recommendation requires actions that must be executed from outside of the Recommendation advisor. To make any changes to your earlier selections, click Back. To close the Recommendation advisor, click Close.

Health indicator Log Filesystem Utilization

Description

details also shows if log archiving is enabled. If Block on Log Disk Full, shown in the additional details, is set to yes and utilization is at 100%, you should resolve any alerts as soon as possible to limit the impact to applications which cannot commit transactions until the log file is successfully created.

Selected recommendation Archive logs.

Details

Archiving logs either automatically or manually will free up the amount of space currently used to store the transaction logs.

Figure 5-41 Recommendation advisor - Summary - Immediate solution

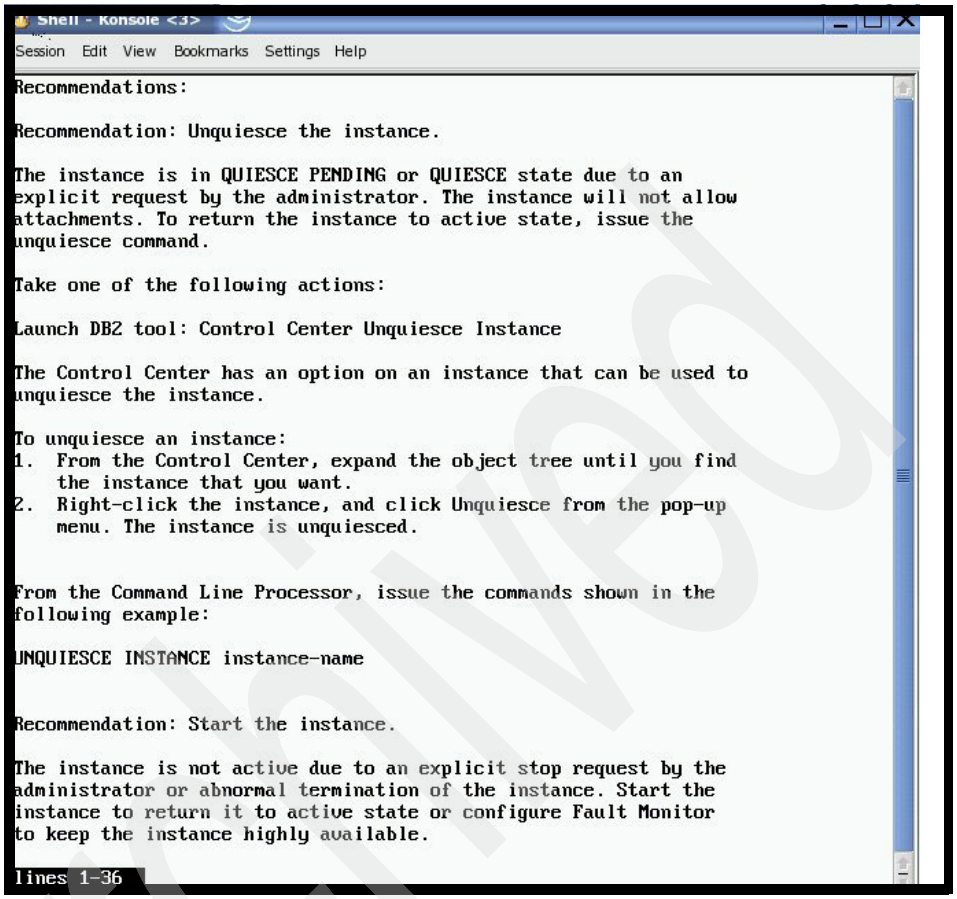
We can also get the recommendations through the command-line interface. The syntax looks like:

```
db2 GET RECOMMENDATIONS FOR HEALTH INDICATOR <health indicator short name> ON
database name
```

An example would be the following:

```
db2 GET RECOMMENDATIONS FOR HEALTH INDICATOR db2.db2_op_status ON sample
```

The output of the command is shown in Figure 5-42 on page 240.



```
Recommendations:

Recommendation: Unquiesce the instance.

The instance is in QUIESCE PENDING or QUIESCE state due to an
explicit request by the administrator. The instance will not allow
attachments. To return the instance to active state, issue the
unquiesce command.

Take one of the following actions:

Launch DB2 tool: Control Center Unquiesce Instance

The Control Center has an option on an instance that can be used to
unquiesce the instance.

To unquiesce an instance:
1.  From the Control Center, expand the object tree until you find
    the instance that you want.
2.  Right-click the instance, and click Unquiesce from the pop-up
    menu. The instance is unquiesced.


From the Command Line Processor, issue the commands shown in the
following example:

UNQUIESCE INSTANCE instance-name


Recommendation: Start the instance.

The instance is not active due to an explicit stop request by the
administrator or abnormal termination of the instance. Start the
instance to return it to active state or configure Fault Monitor
to keep the instance highly available.

lines 1-36
```

Figure 5-42 Recommendation advisor - Command-line processor

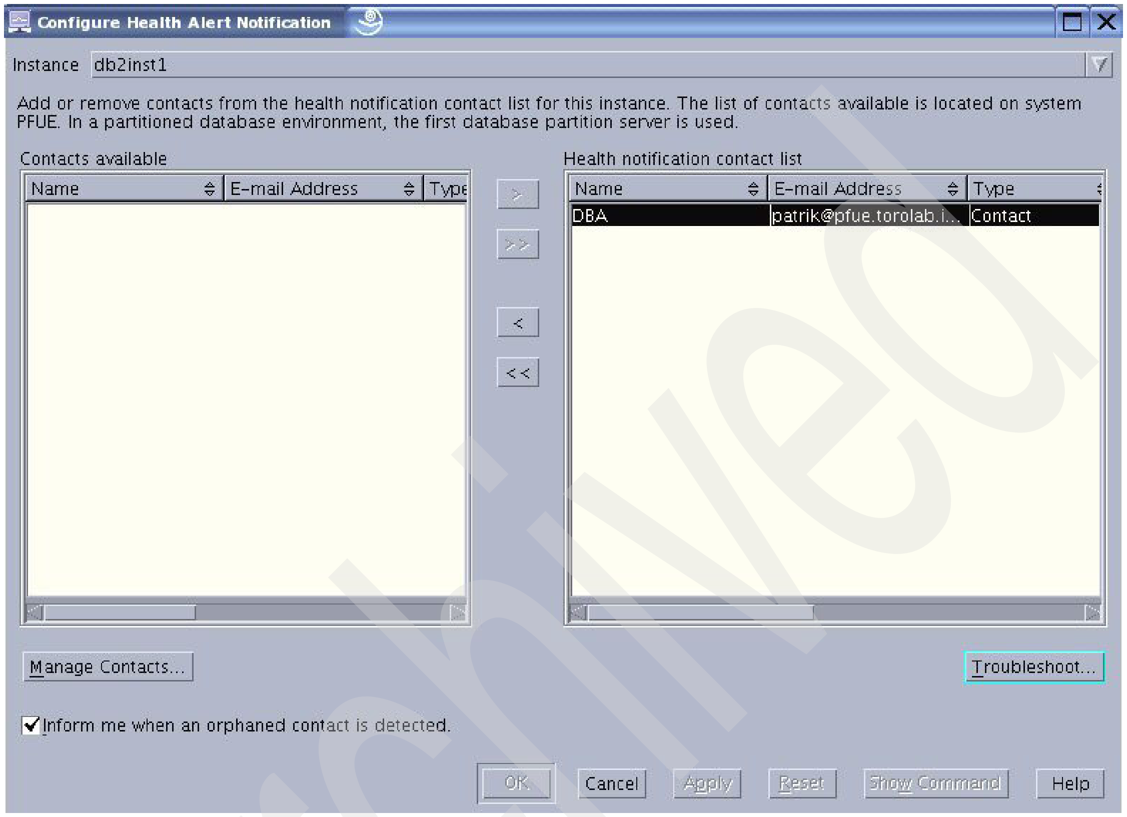
Troubleshoot Health Alert Notification wizard

Use the health alert notification troubleshooting wizard to correct problems that are related to health alert notification. Some problems that this wizard can help resolve include:

- E-mail or pager notifications are not being received.
- Fewer than expected contacts appear in the health notification contact list.
- The health notification contact list contains orphaned contacts.

To get to the troubleshoot wizard, open the Health Center. From the Health Center navigational view, right-click the instance for which you want to run the troubleshoot wizard and select **Configure** → **Alert Notification** from the pop-up menu. The configure health alert notification window opens as shown in

Figure 5-43. In the window click the **Troubelshoot** button to start the wizard, shown in Figure 5-44 on page 242.



Instance db2inst1

Add or remove contacts from the health notification contact list for this instance. The list of contacts available is located on system PFUE. In a partitioned database environment, the first database partition server is used.

Contacts available

Name	E-mail Address	Type

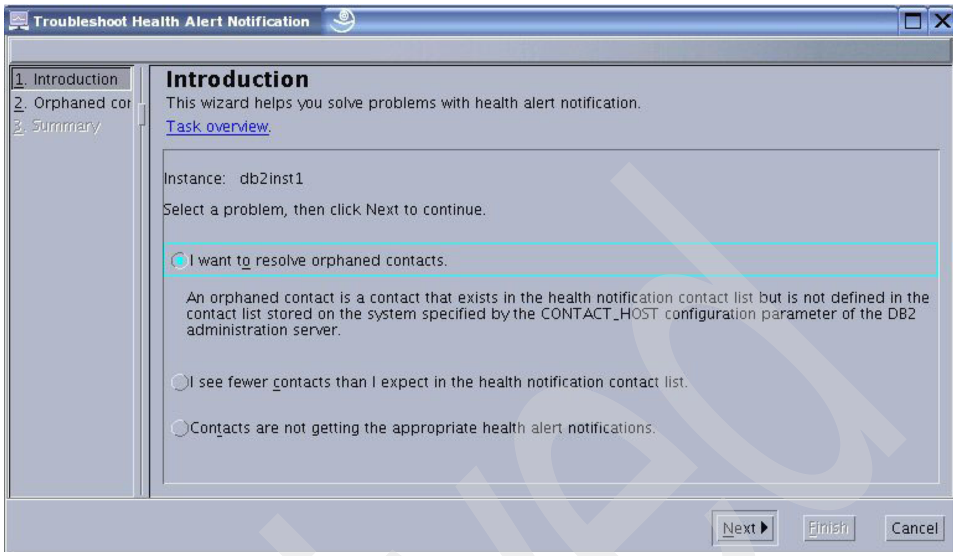
Health notification contact list

Name	E-mail Address	Type
DBA	patrik@pfue.torolab.i...	Contact

✓ Inform me when an orphaned contact is detected.

Figure 5-43 Configure health alert notification

In the troubleshoot wizard we have different choices, depending on what the problem is. For the first option click the “I want resolve orphaned contacts” radio button, as shown in Figure 5-44 on page 242, and then click **Next**.



Introduction

This wizard helps you solve problems with health alert notification.

Task overview.

Instance: db2inst1

Select a problem, then click Next to continue.

[✓ I want to resolve orphaned contacts.]

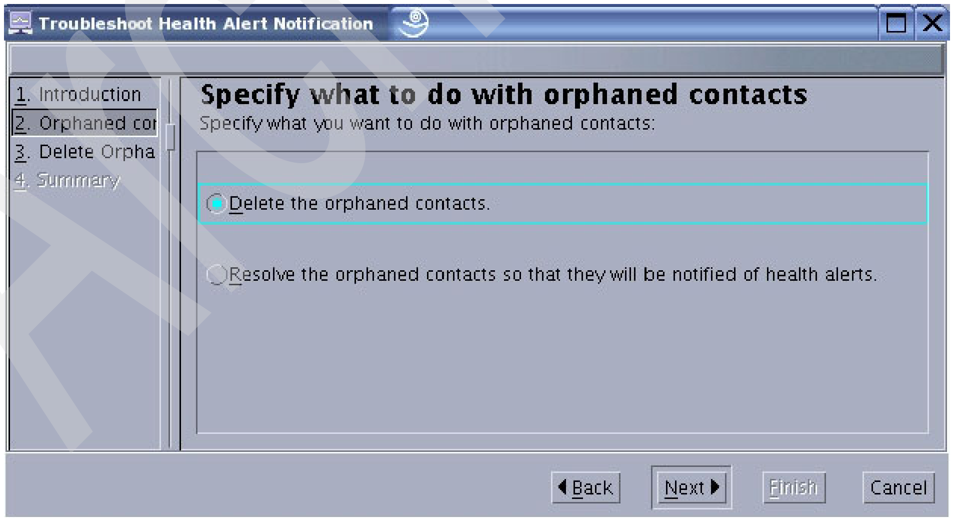
An orphaned contact is a contact that exists in the health notification contact list but is not defined in the contact list stored on the system specified by the CONTACT_HOST configuration parameter of the DB2 administration server.

I see fewer contacts than I expect in the health notification contact list.

Contacts are not getting the appropriate health alert notifications.

Figure 5-44 Troubleshoot health alert notification - Introduction

We then get to the Orphaned contact page, shown in Figure 5-45. We have to decide if we want to delete the orphaned contact or resolve the orphaned contact. We go through both options—first the delete option. Select **Delete the orphaned contact** and click **Next** to get to the Delete Orphaned page, which is shown in Figure 5-46 on page 243.



Specify what to do with orphaned contacts

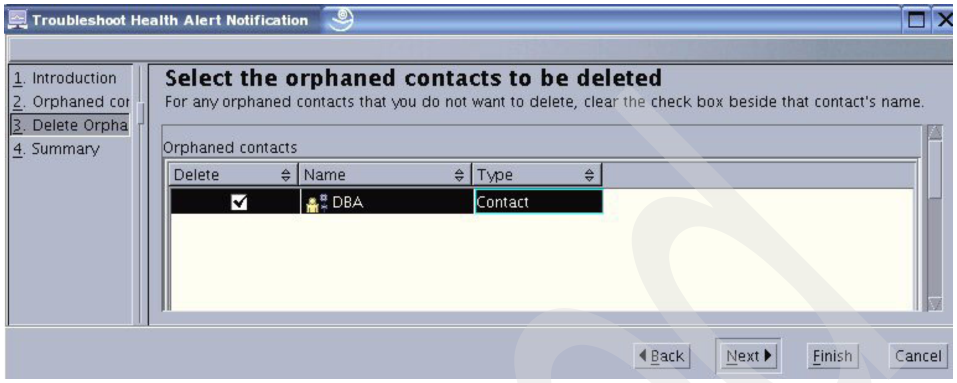
Specify what you want to do with orphaned contacts:

[✓ Delete the orphaned contacts.]

Resolve the orphaned contacts so that they will be notified of health alerts.

Figure 5-45 Troubleshoot health alert notification - Delete orphaned contacts

Check the Delete box in the Orphaned contact window for the contacts you want to delete. Then click **Next** to get to the Summary page, shown in Figure 5-47.



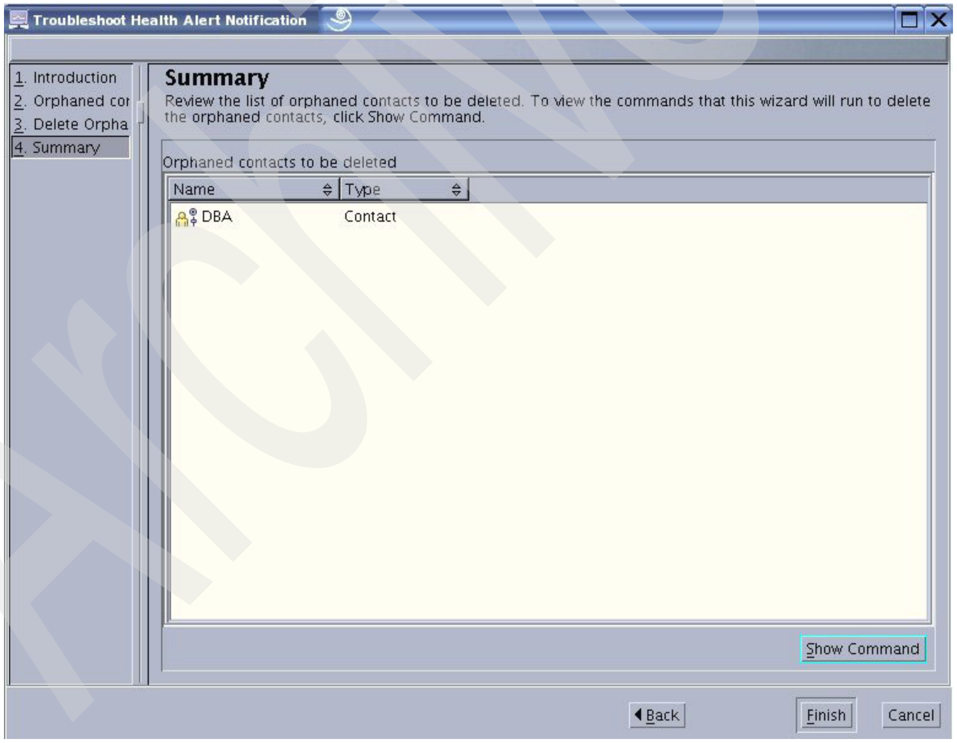
Select the orphaned contacts to be deleted

For any orphaned contacts that you do not want to delete, clear the check box beside that contact's name.

Orphaned contacts

Delete	Name	Type
✓	DBA	Contact

Figure 5-46 Troubleshoot health alert notification - Delete orphaned contacts



Summary

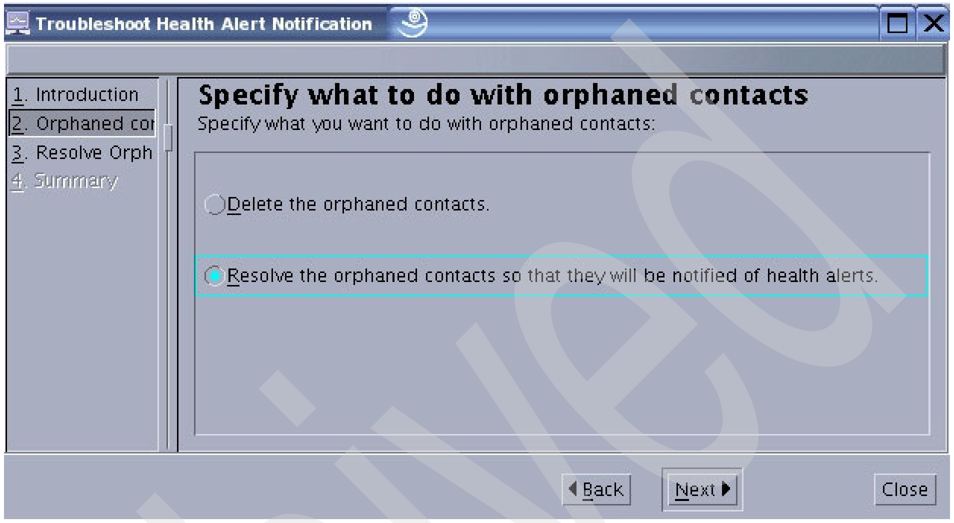
Review the list of orphaned contacts to be deleted. To view the commands that this wizard will run to delete the orphaned contacts, click Show Command.

Orphaned contacts to be deleted

Name	Type
DBA	Contact

Figure 5-47 Troubleshoot health alert notification - Summary

The Summary page shows you again all the contacts that have been selected to be deleted. Review it and, if all is well, click **Finish**; or if you first want to see the command that is executed click **Show Command**, and then when you close that window again click **Finish**.



Specify what to do with orphaned contacts

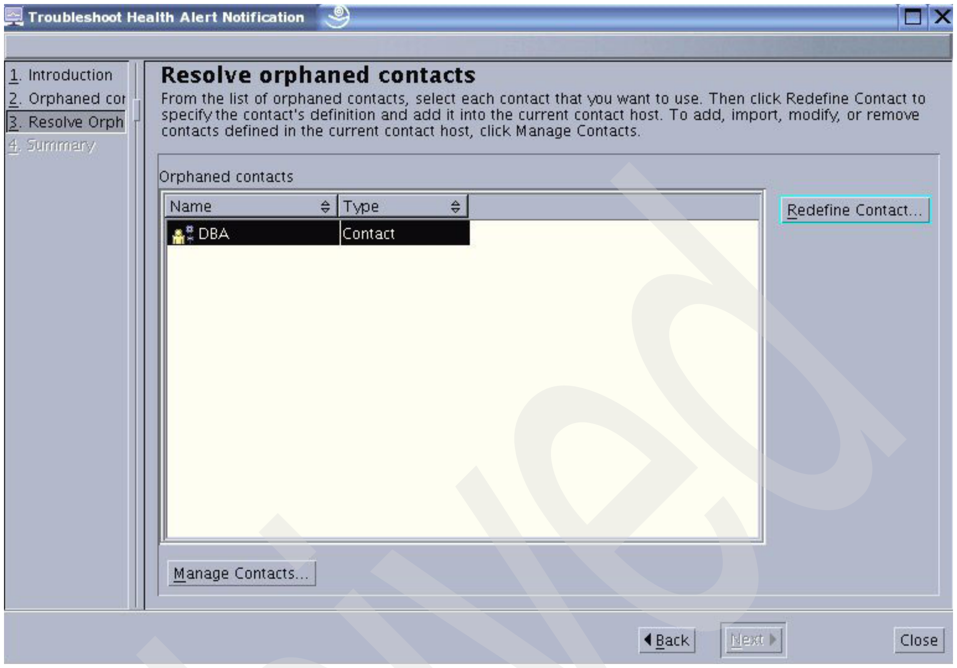
Specify what you want to do with orphaned contacts:

Delete the orphaned contacts. [✓ Resolve the orphaned contacts so that they will be notified of health alerts.]

Figure 5-48 Troubleshoot health alert notification - Resolve orphaned contacts

The second option is to resolve the contact instead of deleting it. So instead of selecting “Delete the orphan contacts”, as in Figure 5-45 on page 242, this time we select the “Resolve the orphaned contacts so that they will be notified of health alerts”, shown in Figure 5-48. Click **Next** to get to the next page, shown in Figure 5-49 on page 245, where we resolve the orphaned contacts. We again have two options. Use the Redefine Contact button or the Manage Contacts button. By clicking the **Redefine Contact** button we get the screen shown in Figure 5-50 on page 245, where we can check the settings of the contact and make corrections. When we click the **Manage Contacts** button we get to the screen shown in Figure 5-51 on page 246. When we there click the **Add Contact** button we again get to the screen shown in Figure 5-50 on page 245 to correct the settings for the contact.

When all orphaned contacts are resolved, our Resolve Orphaned Contacts page will look like Figure 5-52 on page 246.


Troubleshoot Health Alert Notification

1. Introduction
2. Orphaned cor
3. Resolve Orph
4. Summary

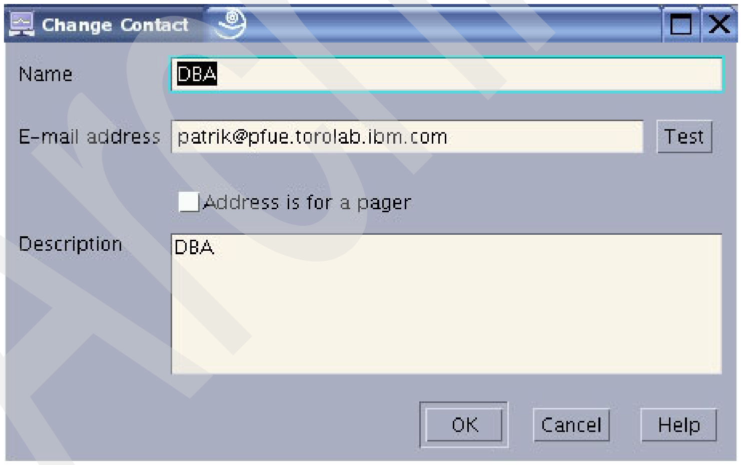
Resolve orphaned contacts

From the list of orphaned contacts, select each contact that you want to use. Then click Redefine Contact to specify the contact's definition and add it into the current contact host. To add, import, modify, or remove contacts defined in the current contact host, click Manage Contacts.

Orphaned contacts

Name	Type
DBA	Contact

Figure 5-49 Troubleshoot health alert notification - Resolve orphaned contacts


Change Contact

Name: DBA

E-mail address: patrik@pfue.torolab.ibm.com

☐ Address is for a pager

Description: DBA

Figure 5-50 Change Contact

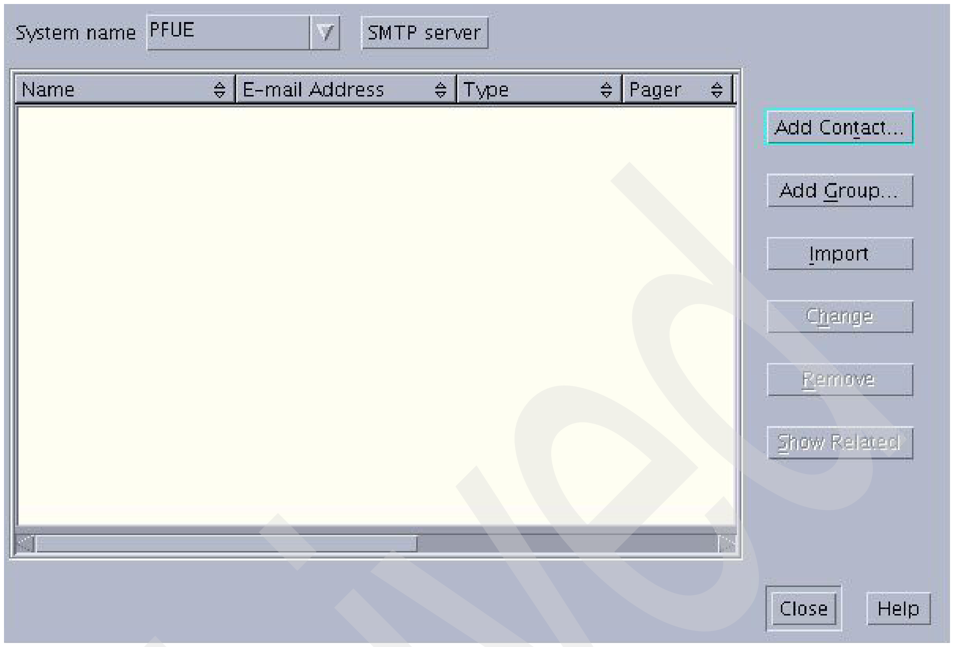
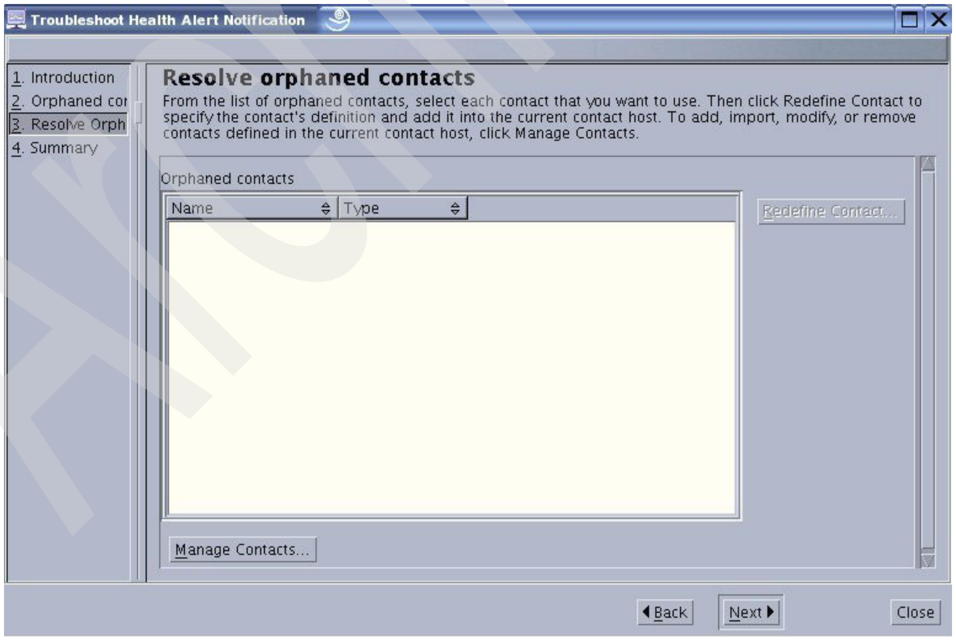


Figure 5-51   Manage contacts

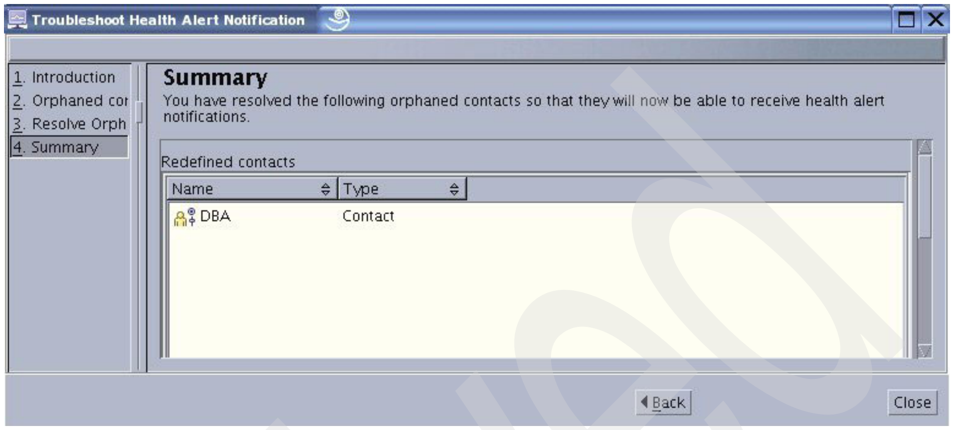


Resolve orphaned contacts

From the list of orphaned contacts, select each contact that you want to use. Then click Redefine Contact to specify the contact's definition and add it into the current contact host. To add, import, modify, or remove contacts defined in the current contact host, click Manage Contacts.

Figure 5-52   Troubleshoot health alert notification - Resolved contacts

When you finish resolving all orphaned contacts proceed by clicking the **Next** button. On the Summary page, shown in Figure 5-53, you again see all of the redefined contacts. The wizard can then be closed by clicking **Close**.



Summary

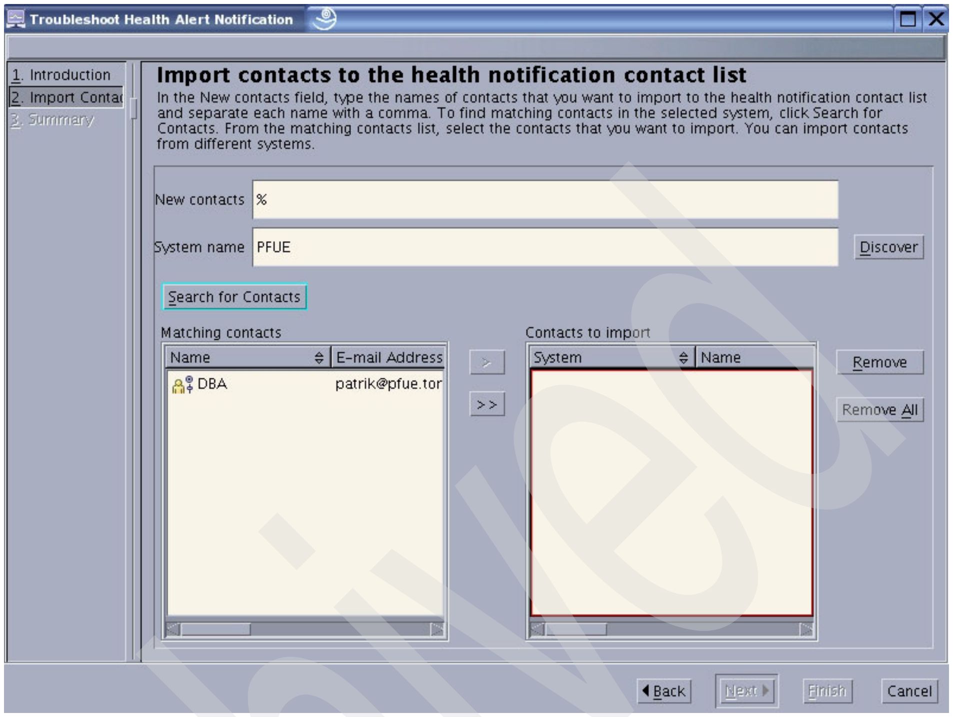
You have resolved the following orphaned contacts so that they will now be able to receive health alert notifications.

Redefined contacts

Name	Type
DBA	Contact

Figure 5-53 Troubleshooting health alert notification - Summary resolved contacts

The second option we have for the troubleshoot health alert notification wizard is the “I see fewer contacts that I expect in the health notification contact list”, Figure 5-44 on page 242. Select it and then click **Next**. The screen we see is shown in Figure 5-54 on page 248. On this page we can search for contacts on a system. In the New contacts field enter what you are searching for. You can use the % (percent) and _ (underscore) wild cards when specifying the new contacts to import. After clicking the **Search for Contacts** button, contacts matching your search criteria will be listed in the Matching contacts box. You can add them to your system with the arrow button (> means only the selected contact, while >> means all contacts found). When you finish adding the found contacts to the list of contacts to be imported click the **Next** button to get to the Summary page shown in Figure 5-55 on page 249.



Import contacts to the health notification contact list

In the New contacts field, type the names of contacts that you want to import to the health notification contact list and separate each name with a comma. To find matching contacts in the selected system, click Search for Contacts. From the matching contacts list, select the contacts that you want to import. You can import contacts from different systems.

New contacts: %

System name: PFUE

Matching contacts

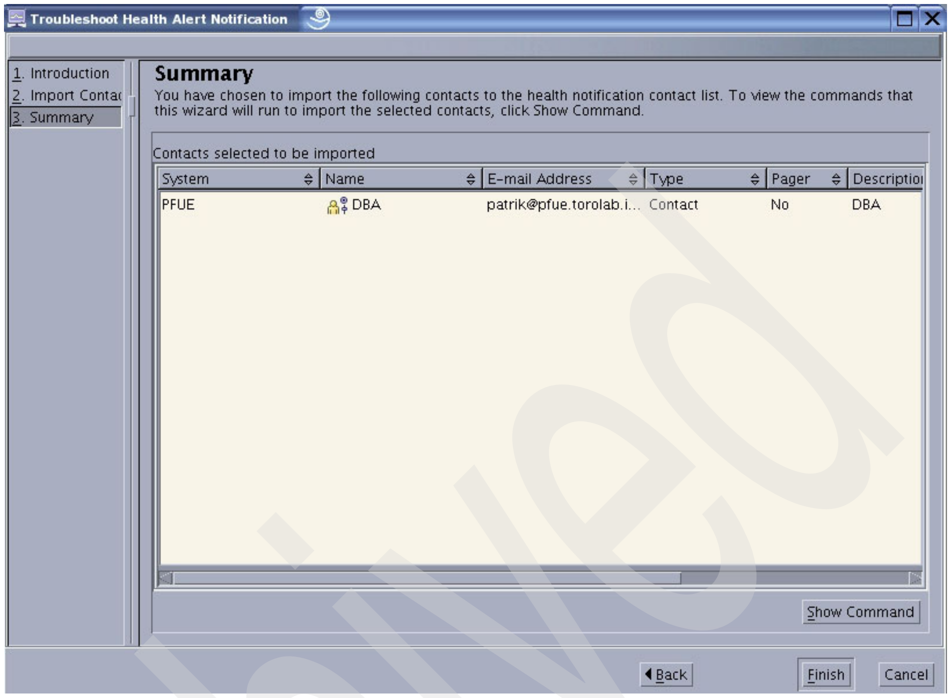
Name	E-mail Address
DBA	patrik@pfue.tor

Contacts to import

System	Name

Figure 5-54 Troubleshoot health alert notification - Import contacts

On the Summary page you see all of the contacts you have selected to import. You can look at the command that the wizard will use to import the contacts by clicking the **Show command** button. The wizard will execute the command and exit when you click the **Finish** button.



Summary

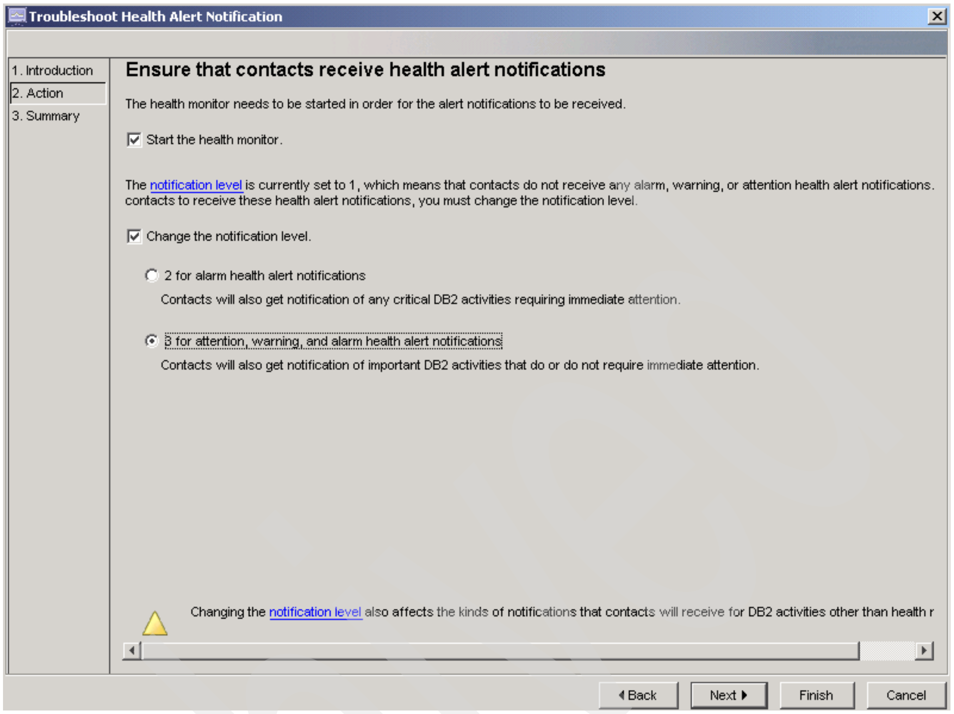
You have chosen to import the following contacts to the health notification contact list. To view the commands that this wizard will run to import the selected contacts, click Show Command.

Contacts selected to be imported

System	Name	E-mail Address	Type	Pager	Descriptio
PFUE	DBA	patrik@pfue.torolab.i...	Contact	No	DBA

Figure 5-55 Troubleshoot health alert notification - Summary import contacts

The last option that is given in Figure 5-44 on page 242, “Contact are not getting the appropriate health alert notification”, gives you the possibility, when selected, to start the health monitor if not yet started, or change the notification level. First of all you have to select that option and then click the **Next** button. You get the screen shown in Figure 5-56 on page 250. Update the page to your needs and click **Next** to get to the Summary page shown in Figure 5-57 on page 250. Review your settings and click **Finish** if all is well.


Troubleshoot Health Alert Notification

Ensure that contacts receive health alert notifications

The health monitor needs to be started in order for the alert notifications to be received.

[✓] Start the health monitor.

The notification level is currently set to 1, which means that contacts do not receive any alarm, warning, or attention health alert notifications. contacts to receive these health alert notifications, you must change the notification level.

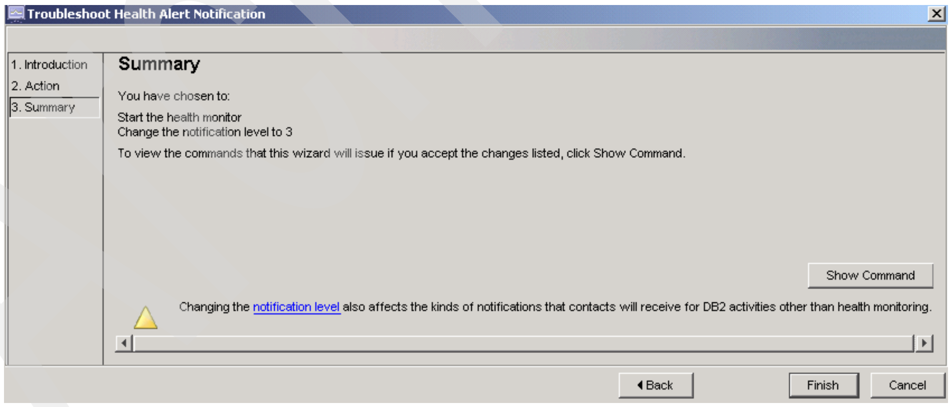
[✓] Change the notification level.

○ 2 for alarm health alert notifications
Contacts will also get notification of any critical DB2 activities requiring immediate attention.

● 3 for attention, warning, and alarm health alert notifications
Contacts will also get notification of important DB2 activities that do or do not require immediate attention.

Changing the notification level also affects the kinds of notifications that contacts will receive for DB2 activities other than health r

Figure 5-56 Contacts did not get the appropriate health alert notification - Action


Troubleshoot Health Alert Notification

Summary

You have chosen to:

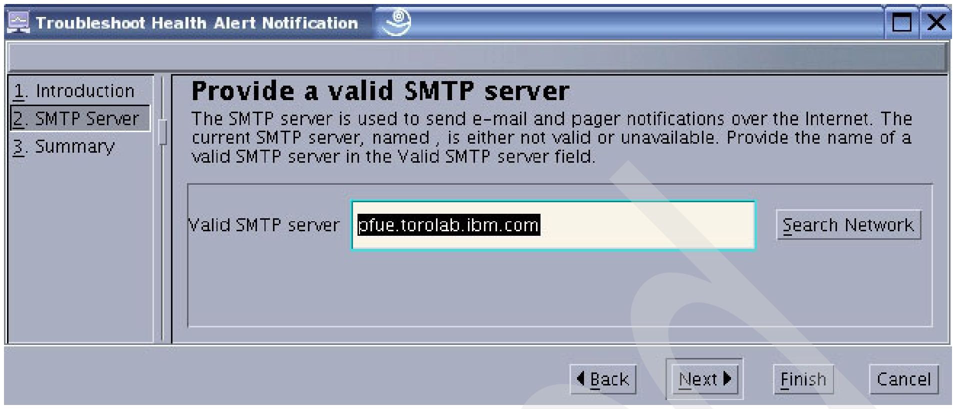
Start the health monitor
Change the notification level to 3

To view the commands that this wizard will issue if you accept the changes listed, click Show Command.

Changing the notification level also affects the kinds of notifications that contacts will receive for DB2 activities other than health monitoring.

Figure 5-57 Contacts did not get appropriate health alert notification - Summary

Depending on your configuration, the dialog can look completely different. If no SMTP (simple mail transfer protocol) server can be found, you will not get the dialog shown in Figure 5-56, but the one shown in Figure 5-58 on page 251.



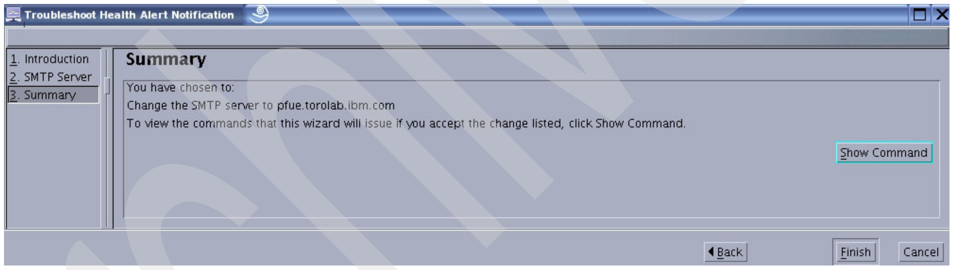
Provide a valid SMTP server

The SMTP server is used to send e-mail and pager notifications over the Internet. The current SMTP server, named , is either not valid or unavailable. Provide the name of a valid SMTP server in the Valid SMTP server field.

Valid SMTP server: pfue.torolab.ibm.com

Figure 5-58 SMTP server

Enter a valid SMTP server or let the wizard search through your network. After having entered a SMTP server click the **Next** button to get to the Summary page shown in Figure 5-59.



Summary

You have chosen to:
Change the SMTP server to pfue.torolab.ibm.com
To view the commands that this wizard will issue if you accept the change listed, click Show Command.

Figure 5-59 SMTP Server - Summary

Click the **Finish** button to change the SMTP server and close the wizard.

5.1.2 Automated log file management

Before configuring the logging of your database you must choose between two types of logging:

- *Circular logging,* which uses a circle or ring of log files to record changes to a database. When the last log is filled, the first file is reused in a circular fashion. Since transaction information can be overwritten, rollforward recovery is not possible with this type of logging. Circular logging is the default when a database is created.

- *Archival logging,* which does not overwrite log files. Instead, it creates additional logs to record all transactions since the last backup. This type of logging supports rollforward recovery.

When you change from circular to archival logging it is mandatory to make a backup of your database. The database logging wizard will do the backup for you.

On the following pages we show you how you how to configure both variants of logging with the in DB2 UDB V8.2 newly introduced database logging wizard.

To start the database logging wizard, go to the DB2 Control Center, right-click the database you want to configure, and select the **Configure Database Logging** option, as shown in Figure 5-60.

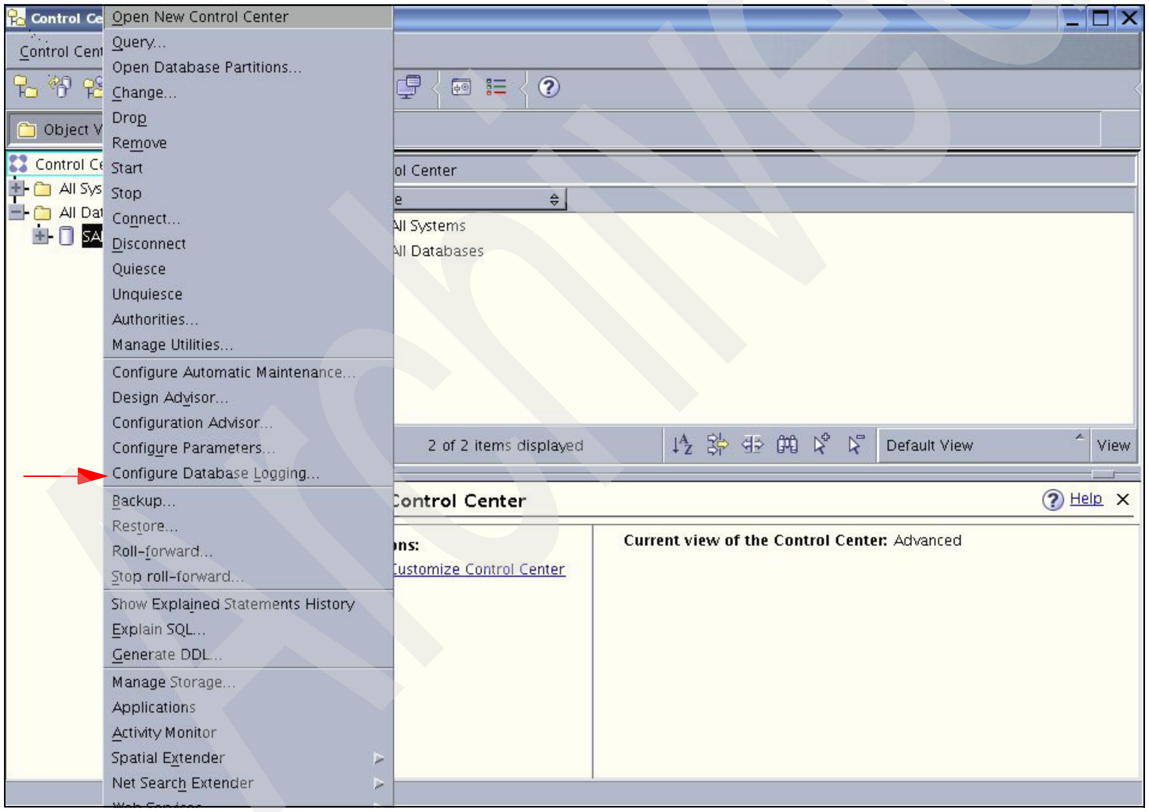
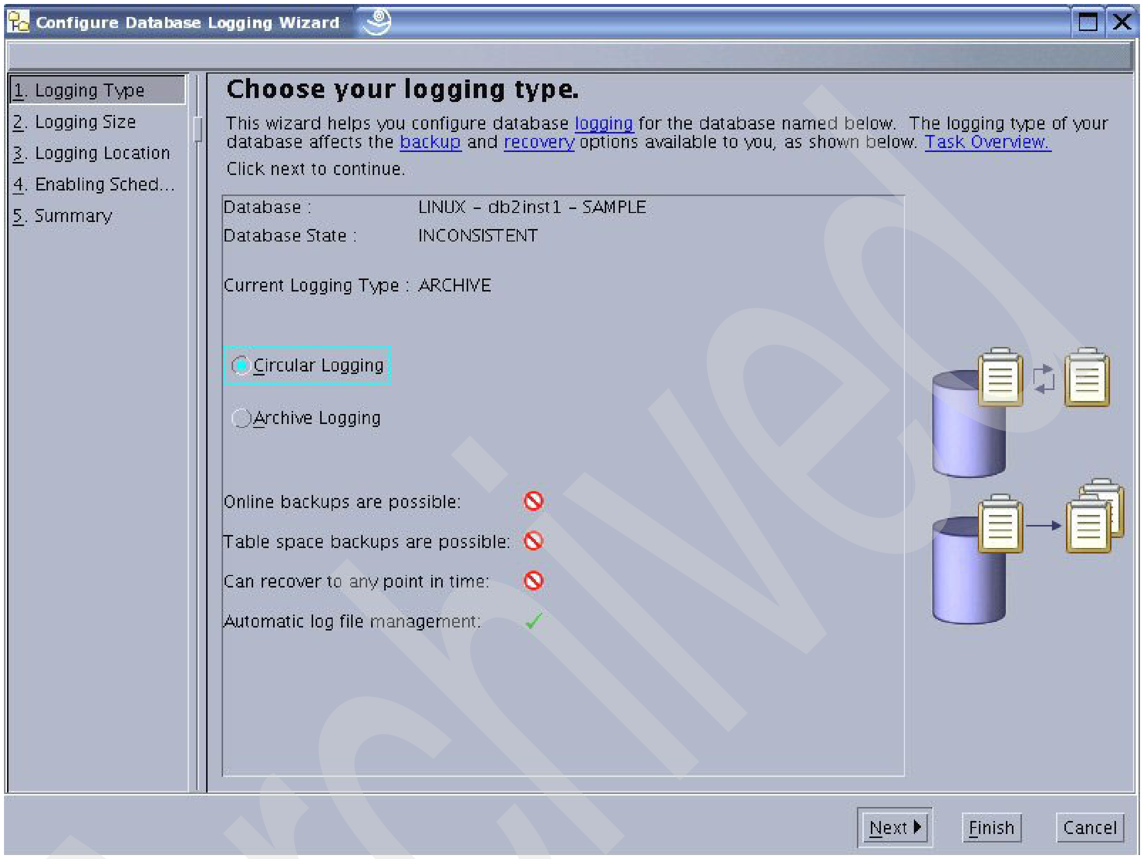


Figure 5-60 Configure database logging

When the wizard starts you will see a screen similar to the one in Figure 5-61 on page 253. At this point you have to tell the wizard which kind of logging you want to configure, circular or archival logging. The wizard shows you with signs which options will be available if you choose one or the other logging type.



Choose your logging type.

This wizard helps you configure database logging for the database named below. The logging type of your database affects the backup and recovery options available to you, as shown below. Task Overview.

Click next to continue.

Database : LINUX - db2inst1 - SAMPLE

Database State : INCONSISTENT

Current Logging Type : ARCHIVE

[✓ Circular Logging]

Archive Logging

Online backups are possible:	🚫
Table space backups are possible:	🚫
Can recover to any point in time:	🚫
Automatic log file management:	✓

Figure 5-61 Circular logging - Choosing type

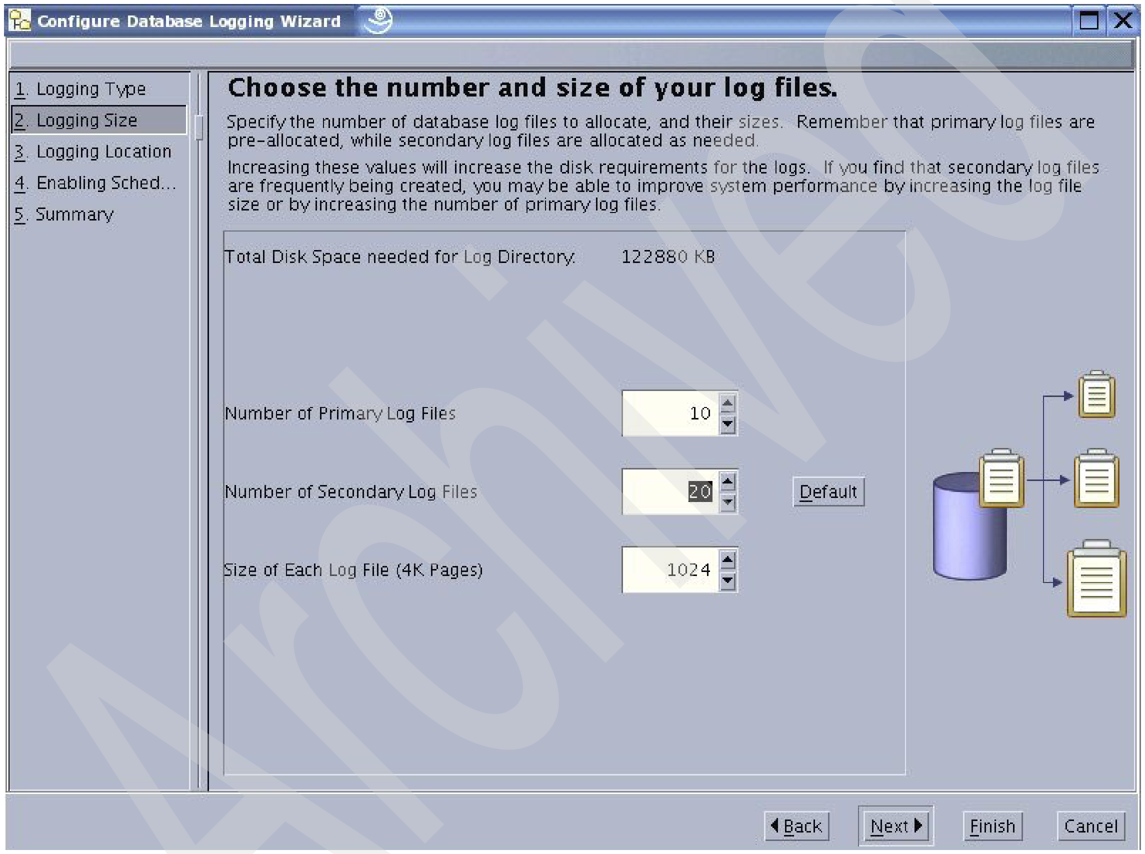
Circular logging

First we show you how to configure the circular logging. On the Logging Type page select the Circular Logging radio button and then click **Next**, and you will get to the next screen, Figure 5-62 on page 254.

You can now configure three parameters:

- Number of primary log files (logprimary)
- Number of secondary log files (logsecond)
- Size of the log files (logfilsiz)

You have to keep in mind that logprimary + logsecond cannot be bigger than 256, and that your file system holding the log files must be big enough to hold all your log files ((logprimary + logsecond) * (logfilsiz + 2) * 4 KB). Another point that you have to consider is that the primary logs are allocated when the database is activated, and the secondary logs are only allocated one-by-one if the logprimary are not enough to hold the uncommitted transaction data. We suggest that you choose logprimary (together with logfilsiz) in a way that they can hold the daily workload and set the logsecond in a way that you can handle your workload peaks. You can find additional information about these parameters in the *IBM DB2 UDB Administration Guide: Performance V8*, SC09-4821.



Choose the number and size of your log files.

Specify the number of database log files to allocate, and their sizes. Remember that primary log files are pre-allocated, while secondary log files are allocated as needed.

Increasing these values will increase the disk requirements for the logs. If you find that secondary log files are frequently being created, you may be able to improve system performance by increasing the log file size or by increasing the number of primary log files.

Total Disk Space needed for Log Directory: 122880 KB

Number of Primary Log Files: 10

Number of Secondary Log Files: 20

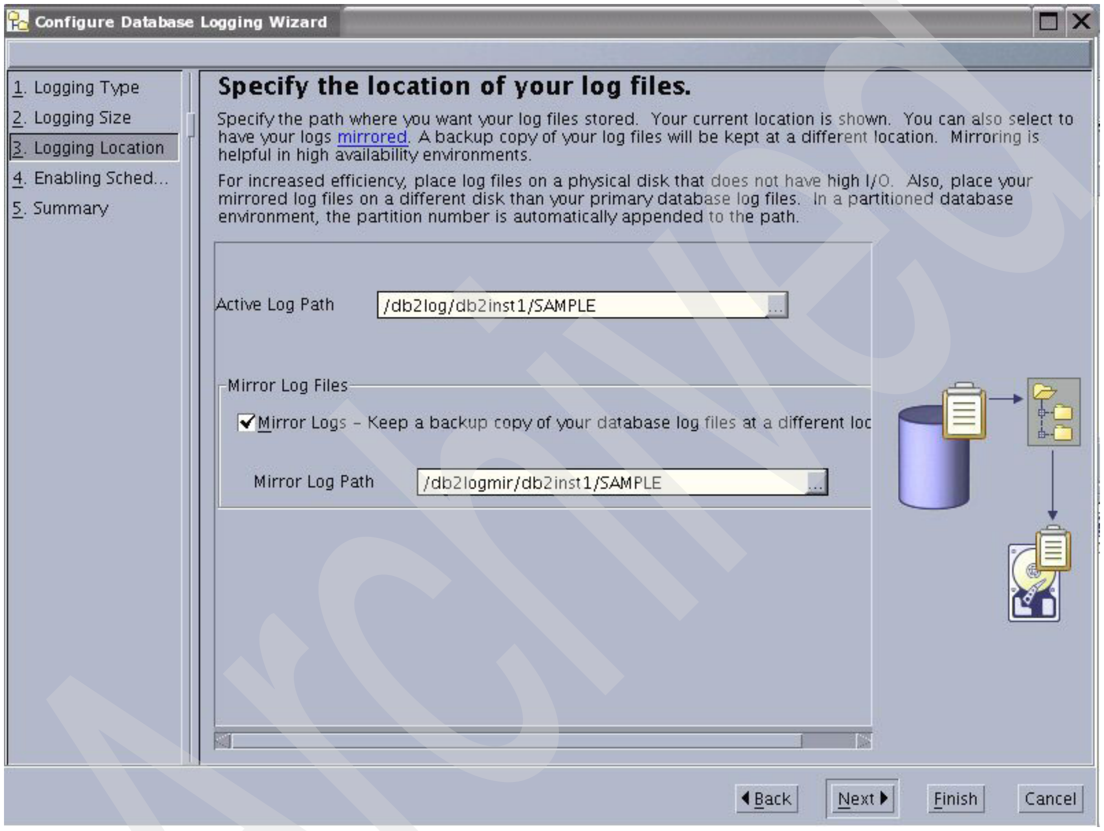
Size of Each Log File (4K Pages): 1024

Figure 5-62 Circular logging - Number and size of log files

When finished configuring logprimary, logsecond, and logfilsiz, we proceed to the Logging Location (Figure 5-63 on page 255), using the **Next** button.

You can now enter the path to where you want your active log files in the Active Log Path box, or you can just browse through your directories to select one as the active log path by clicking the button right next to the text box.

DB2 supports log mirroring at the database level, which you can activate on this page by selecting the Mirror Logs check box. The Mirror Log Path text box must be completed with a valid path. You should consider using the mirror log path when you are concerned that your active logs may get damaged (as a result of a disk crash). The mirror log path should be placed on a different disk from the active log path and, if possible, try to access the disk on which the mirror log path is located over a different disk controller.



Specify the location of your log files.

Specify the path where you want your log files stored. Your current location is shown. You can also select to have your logs mirrored. A backup copy of your log files will be kept at a different location. Mirroring is helpful in high availability environments.

For increased efficiency, place log files on a physical disk that does not have high I/O. Also, place your mirrored log files on a different disk than your primary database log files. In a partitioned database environment, the partition number is automatically appended to the path.

Active Log Path: /db2log/db2inst1/SAMPLE

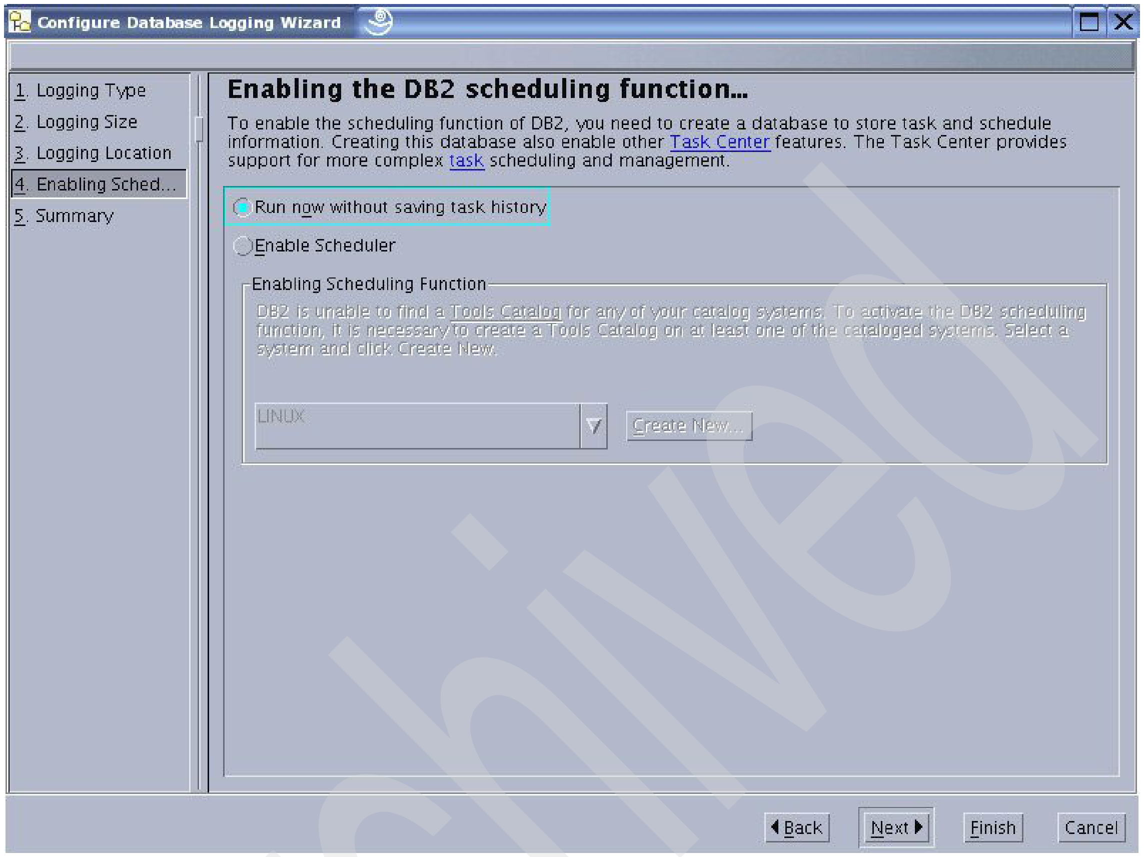
Mirror Log Files

[✓] Mirror Logs - Keep a backup copy of your database log files at a different loc

Mirror Log Path: /db2logmir/db2inst1/SAMPLE

Figure 5-63 Circular logging - Location

Having finished setting up the active log path and, if applicable, the mirror log path, then proceed to the Enabling Scheduling page (shown in Figure 5-64 on page 256) by clicking the **Next** button. If you want, you can select to run the task scheduled by choosing Enable Scheduler, or just run the task instantly by choosing the “Run now without saving task history” option. When you have made your selection, click the **Next** button to get to the Summary page shown in Figure 5-65 on page 257.



Enabling the DB2 scheduling function...

To enable the scheduling function of DB2, you need to create a database to store task and schedule information. Creating this database also enable other Task Center features. The Task Center provides support for more complex task scheduling and management.

[✓ Run now without saving task history]

Enabling Scheduler

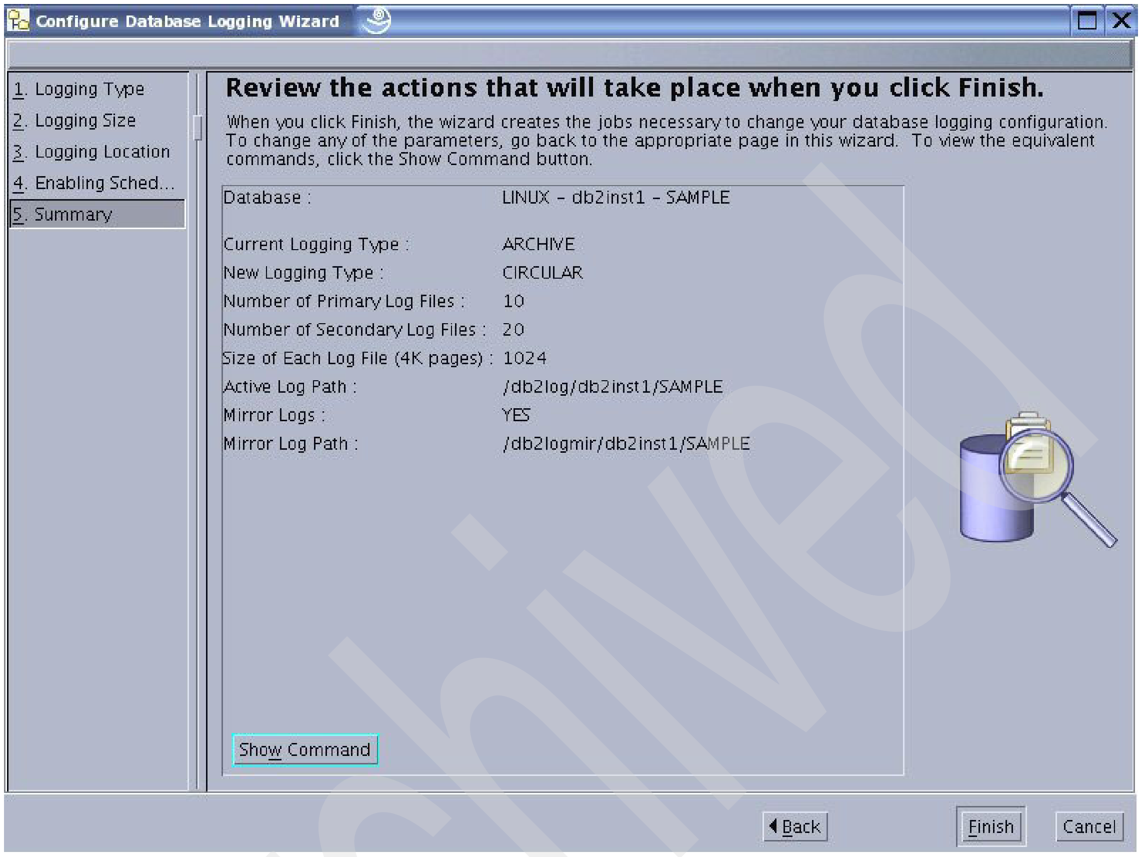
Enabling Scheduling Function

DB2 is unable to find a Tools Catalog for any of your catalog systems. To activate the DB2 scheduling function, it is necessary to create a Tools Catalog on at least one of the cataloged systems. Select a system and click Create New.

LINUX

Figure 5-64 Circular logging - Scheduling function

Check your settings on the Summary page and, if all is OK, click the **Finish** button to let the wizard made the changes in your database.



Review the actions that will take place when you click Finish.

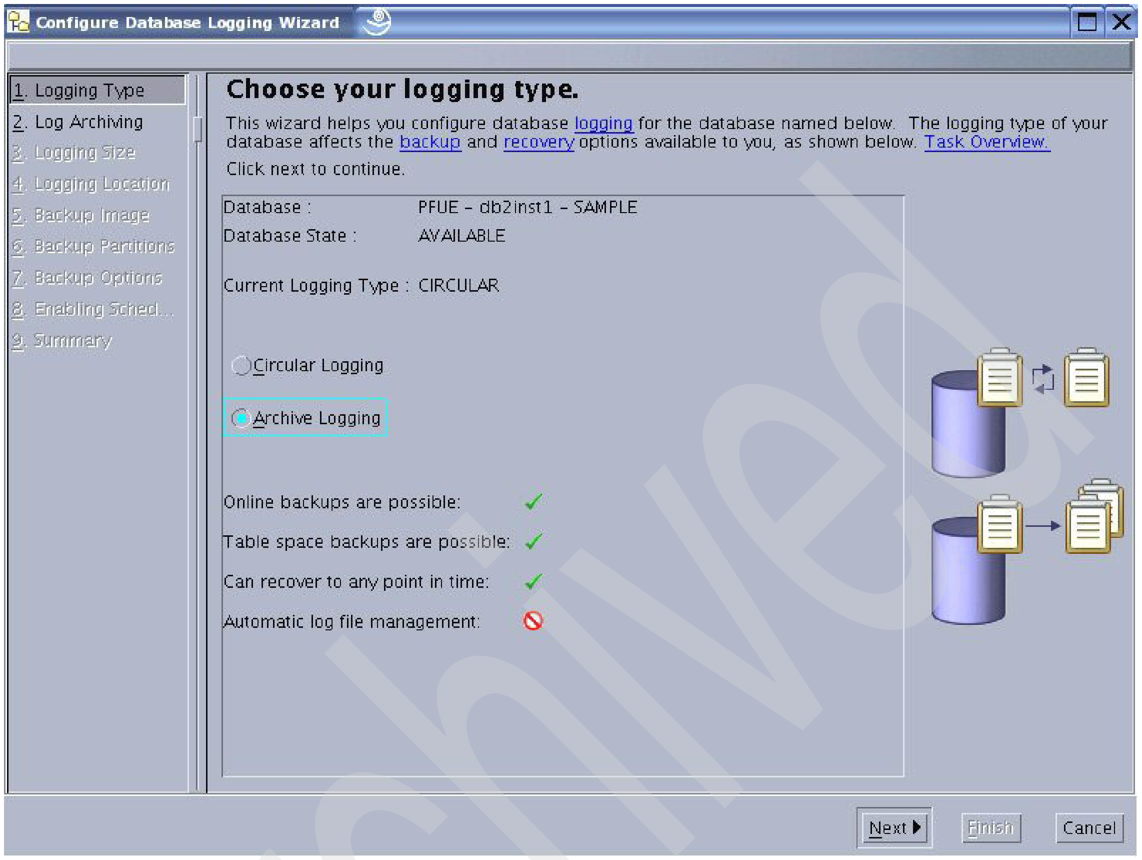
When you click Finish, the wizard creates the jobs necessary to change your database logging configuration. To change any of the parameters, go back to the appropriate page in this wizard. To view the equivalent commands, click the Show Command button.

Database :	LINUX - db2inst1 - SAMPLE
Current Logging Type :	ARCHIVE
New Logging Type :	CIRCULAR
Number of Primary Log Files :	10
Number of Secondary Log Files :	20
Size of Each Log File (4K pages) :	1024
Active Log Path :	/db2log/db2inst1/SAMPLE
Mirror Logs :	YES
Mirror Log Path :	/db2logmir/db2inst1/SAMPLE

Figure 5-65 Circular logging - Summary

Archival logging

We now show how to configure the archival logging. When the Archive Logging radio button is selected you see a screen like in Figure 5-66 on page 258.



Choose your logging type.

This wizard helps you configure database logging for the database named below. The logging type of your database affects the backup and recovery options available to you, as shown below. Task Overview.

Click next to continue.

Database : PFUE - db2inst1 - SAMPLE

Database State : AVAILABLE

Current Logging Type : CIRCULAR

[ ] Circular Logging [✓ Archive Logging]

Online backups are possible: ✓

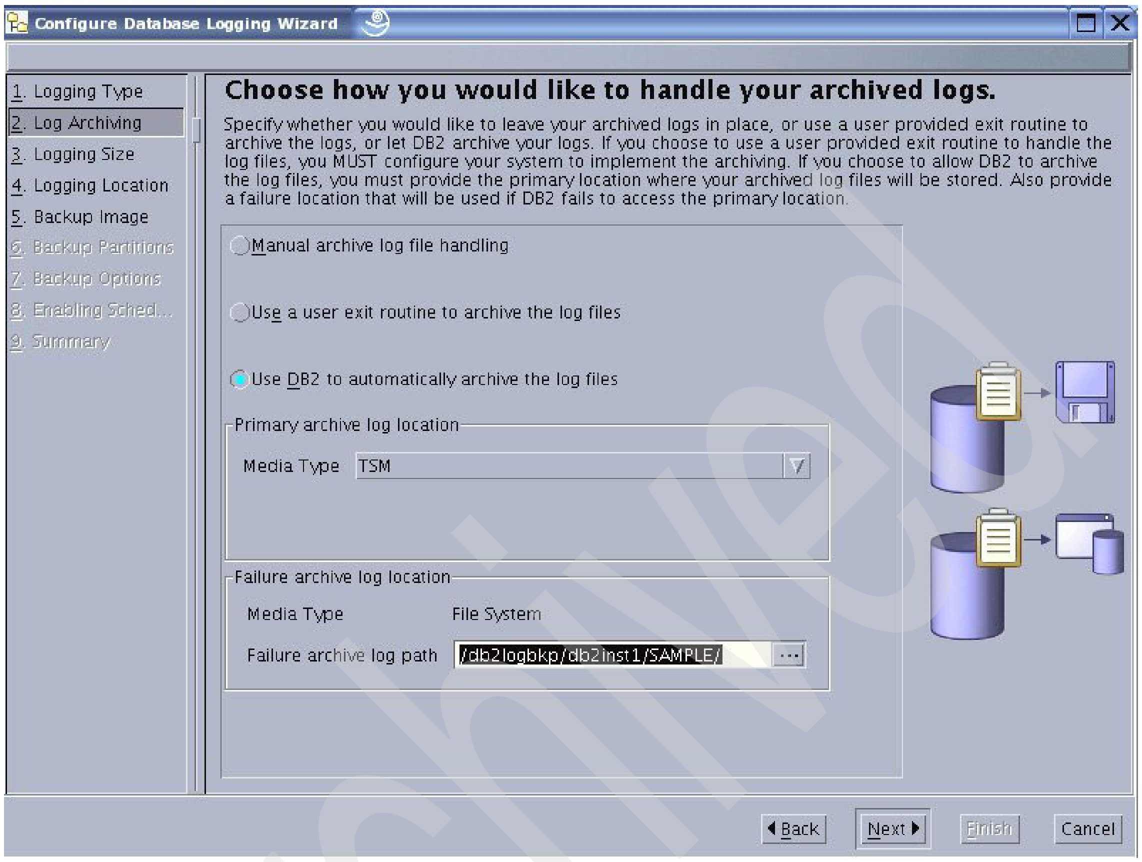
Table space backups are possible: ✓

Can recover to any point in time: ✓

Automatic log file management: 🚫

Figure 5-66 Archival logging - Choosing the type

To proceed with the configuration click **Next**. On the Log Archiving page, shown in Figure 5-67 on page 259, you decide how your logs should be archived. Fill out the required fields according to your environment. Click the **Next** button to get to the Logging Size page (Figure 5-68 on page 260).



Choose how you would like to handle your archived logs.

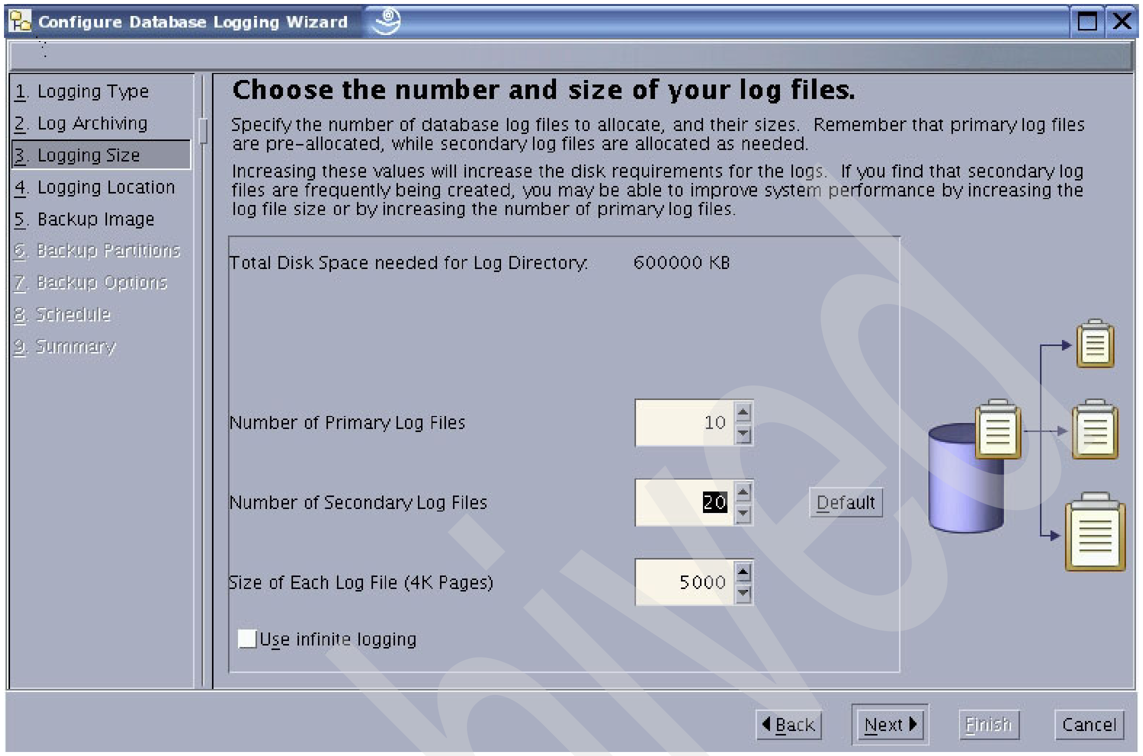
Specify whether you would like to leave your archived logs in place, or use a user provided exit routine to archive the logs, or let DB2 archive your logs. If you choose to use a user provided exit routine to handle the log files, you MUST configure your system to implement the archiving. If you choose to allow DB2 to archive the log files, you must provide the primary location where your archived log files will be stored. Also provide a failure location that will be used if DB2 fails to access the primary location.

- Manual archive log file handling
- Use a user exit routine to archive the log files
- [✓ Use DB2 to automatically archive the log files]

Primary archive log location
Media Type: TSM

Failure archive log location
Media Type: File System
Failure archive log path: /db2logbkp/db2inst1/SAMPLE/

Figure 5-67 Archival logging - Archiving method


Configure Database Logging Wizard

1. Logging Type
2. Log Archiving
3. Logging Size
4. Logging Location
5. Backup Image
6. Backup Partitions
7. Backup Options
8. Schedule
9. Summary

Choose the number and size of your log files.

Specify the number of database log files to allocate, and their sizes. Remember that primary log files are pre-allocated, while secondary log files are allocated as needed.

Increasing these values will increase the disk requirements for the logs. If you find that secondary log files are frequently being created, you may be able to improve system performance by increasing the log file size or by increasing the number of primary log files.

Total Disk Space needed for Log Directory: 600000 KB

Number of Primary Log Files: 10

Number of Secondary Log Files: 20

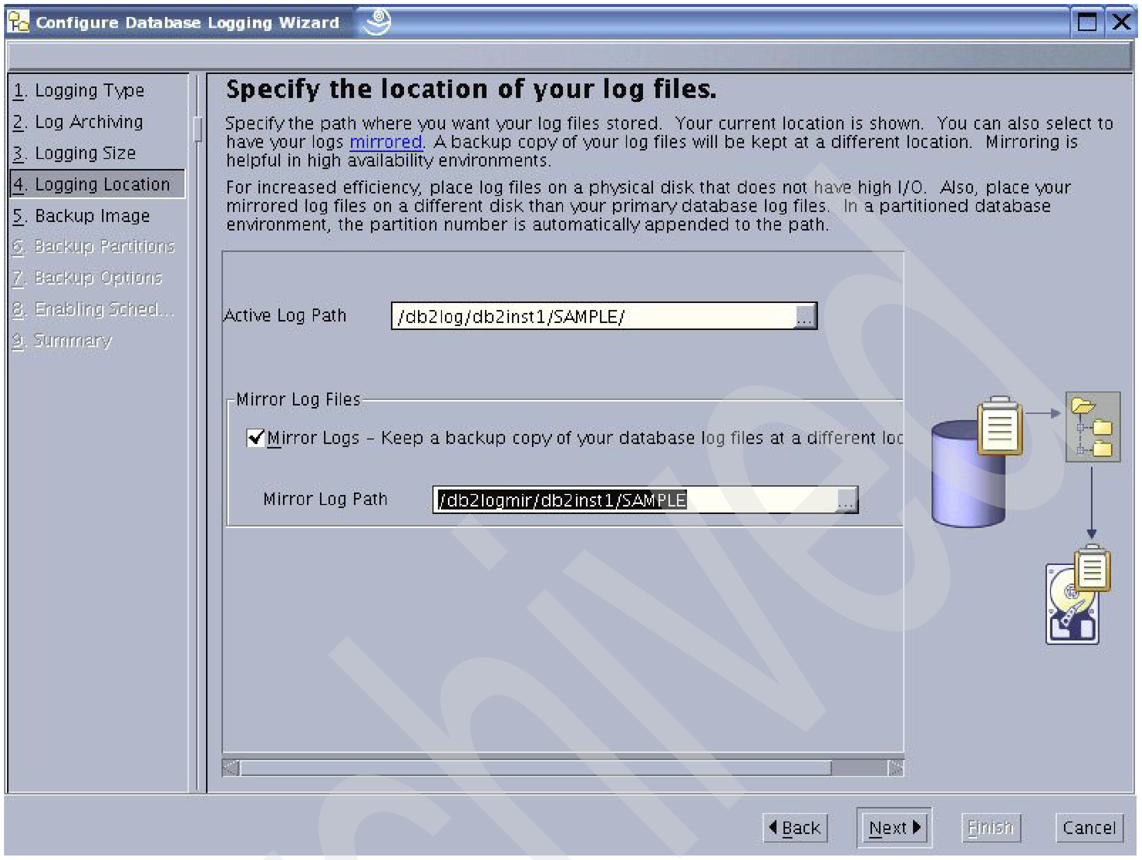
Size of Each Log File (4K Pages): 5000

Use infinite logging

Figure 5-68 Archival logging - Number and size of log files

For the configuration of the number of logs, the same rules apply as those discussed in "Circular logging" on page 253. One addition here is that you can activate infinite logging when using archival logging with the internal or your own userexit. The activation of infinite logging is done by enabling the Use infinite logging check box. A `-1` (minus one) will be entered in the Number of Secondary Log Files box. With infinite logging and the userexit configuration parameter enabled, you have the same disk space considerations as for circular logging. DB2 will keep at least the number of active log files specified by logprimary in the log path. Ensure that you provide extra disk space to allow the delay caused by archiving log files.

After the configuration of the logging size click **Next**. We proceed to the Logging Location page (Figure 5-69 on page 261), where you can change the active log path and also define a mirror log path. When done, click **Next**.


Configure Database Logging Wizard

1. Logging Type
2. Log Archiving
3. Logging Size
4. Logging Location
5. Backup Image
6. Backup Partitions
7. Backup Options
8. Enabling Sched...
9. Summary

Specify the location of your log files.

Specify the path where you want your log files stored. Your current location is shown. You can also select to have your logs mirrored. A backup copy of your log files will be kept at a different location. Mirroring is helpful in high availability environments.

For increased efficiency, place log files on a physical disk that does not have high I/O. Also, place your mirrored log files on a different disk than your primary database log files. In a partitioned database environment, the partition number is automatically appended to the path.

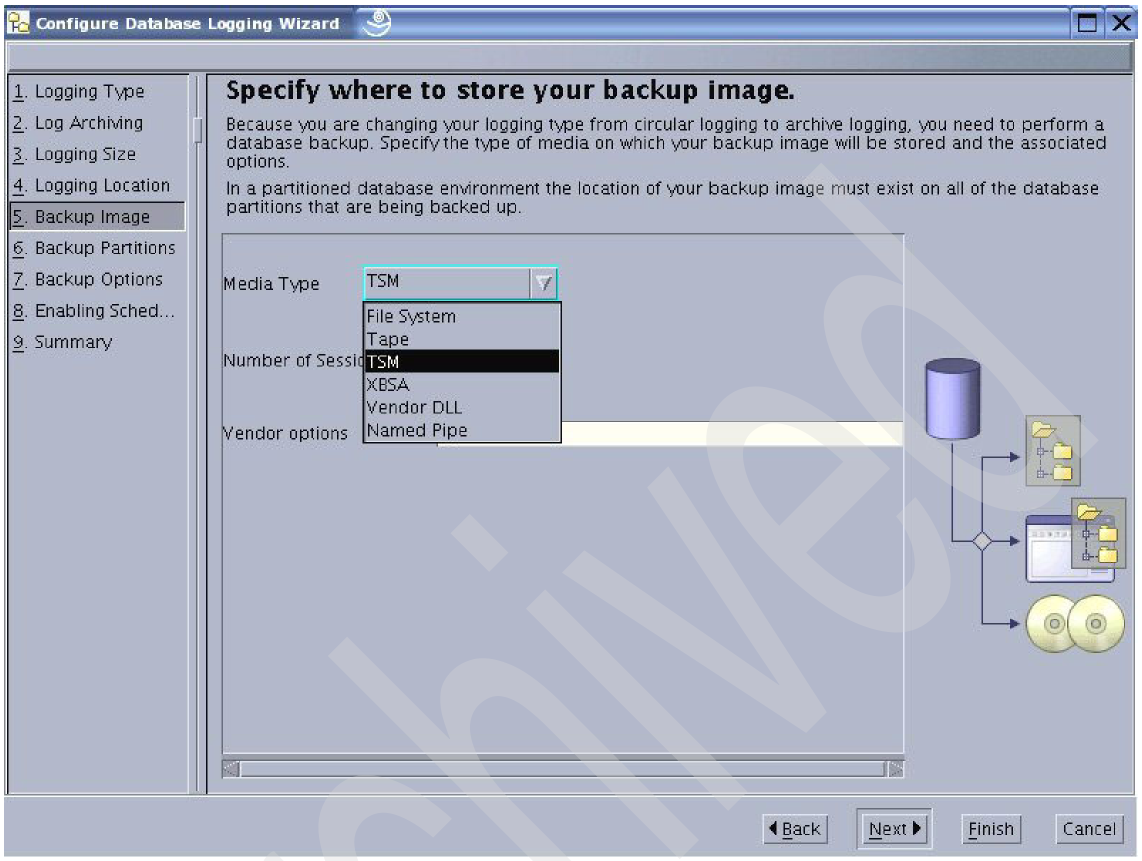
Active Log Path: /db2log/db2inst1/SAMPLE/

Mirror Log Files

[✓] Mirror Logs - Keep a backup copy of your database log files at a different loc

Mirror Log Path: /db2logmir/db2inst1/SAMPLE

Figure 5-69 Archival logging - Location of logs


Configure Database Logging Wizard

1. Logging Type
2. Log Archiving
3. Logging Size
4. Logging Location
5. Backup Image
6. Backup Partitions
7. Backup Options
8. Enabling Sched...
9. Summary

Specify where to store your backup image.

Because you are changing your logging type from circular logging to archive logging, you need to perform a database backup. Specify the type of media on which your backup image will be stored and the associated options.

In a partitioned database environment the location of your backup image must exist on all of the database partitions that are being backed up.

Media Type: TSM

Number of Sessio

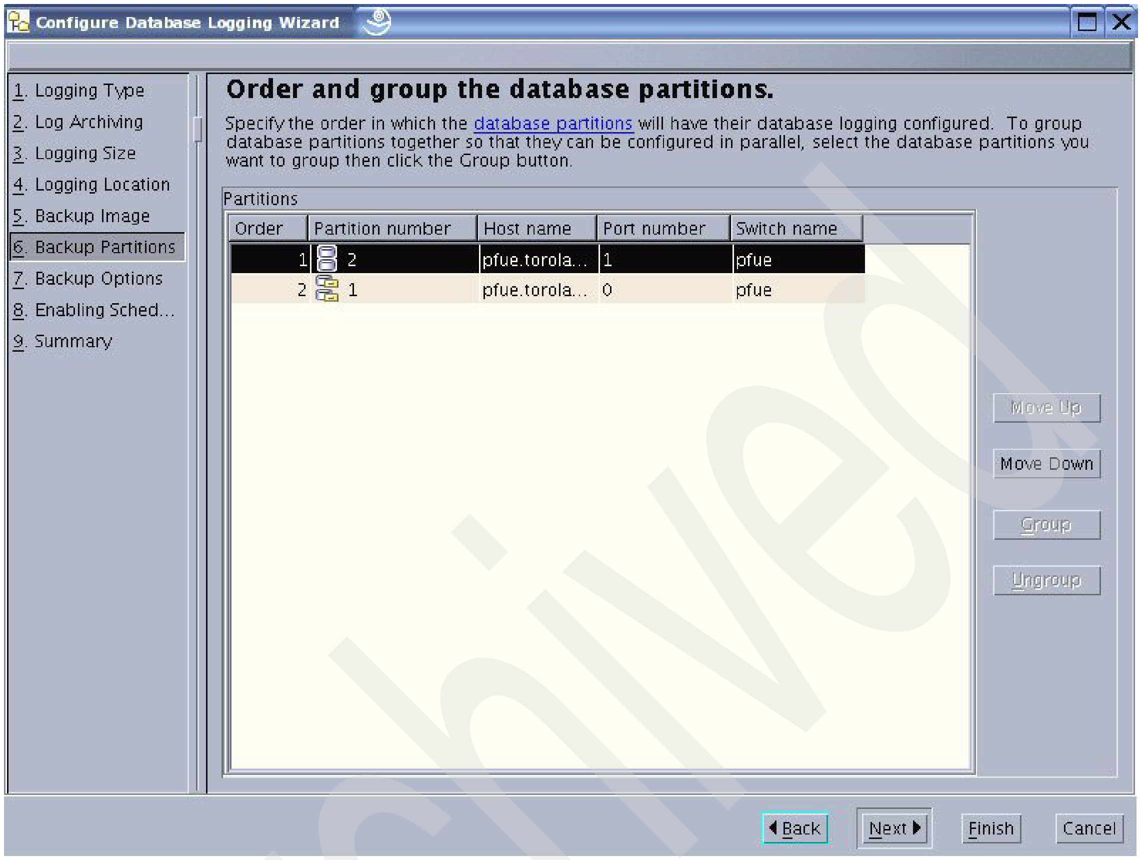
Vendor options

Figure 5-70 Archival logging - Backup image

Figure 5-70 shows the Backup Image page, which gives you the option to define where you want to put the mandatory backup. You see in Figure 5-70 that you have any choices. Customize it to your preferences and open the Backup Partitions page, shown in Figure 5-71 on page 263, by clicking **Next**. On the Backup Partitions page you can determine in which order the backups should run, and you can group partitions together that should run at the same time.

In Figure 5-71 on page 263 you see that we changed the order so that partition two is backed up first and partition one is backed up second. We did this only so that you can see the difference.

Attention: Always run your backup first on the catalog partition and then do the rest of your partitions grouped to your preference.


Configure Database Logging Wizard

1. Logging Type
2. Log Archiving
3. Logging Size
4. Logging Location
5. Backup Image
6. Backup Partitions
7. Backup Options
8. Enabling Sched...
9. Summary

Order and group the database partitions.

Specify the order in which the database partitions will have their database logging configured. To group database partitions together so that they can be configured in parallel, select the database partitions you want to group then click the Group button.

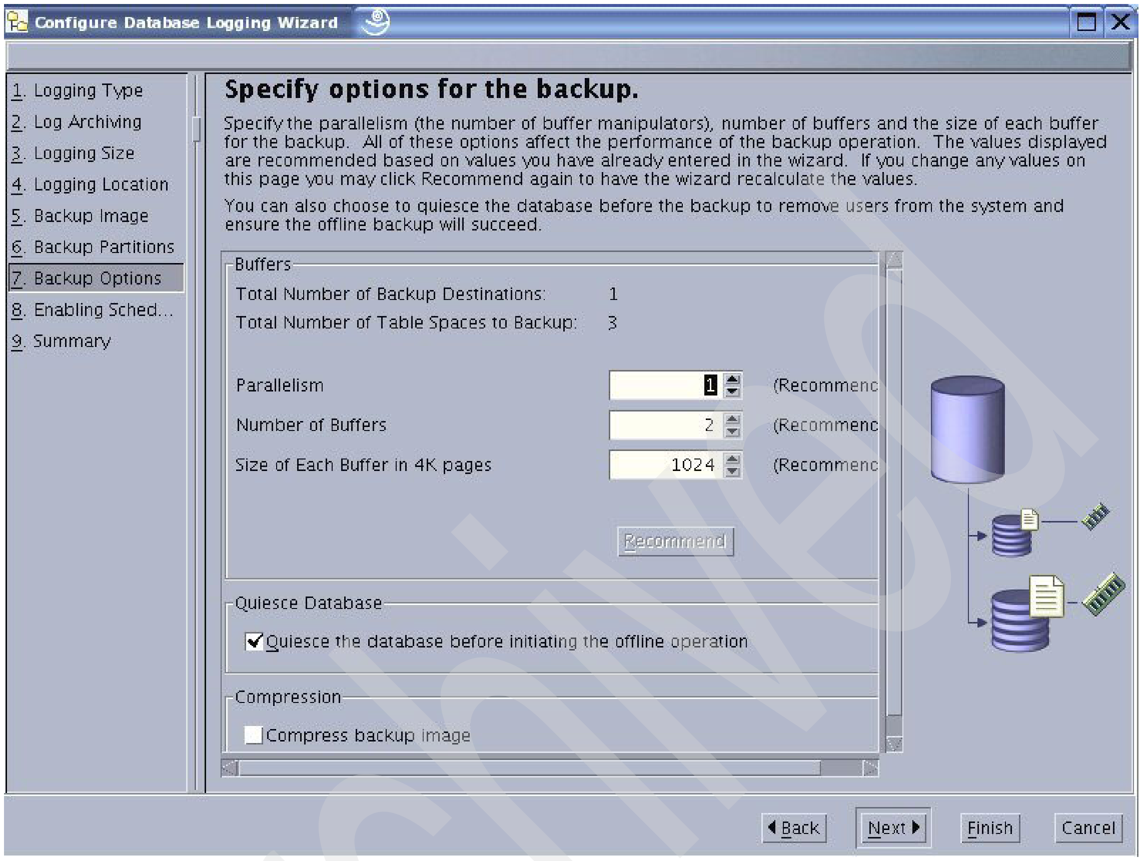
Partitions

Order	Partition number	Host name	Port number	Switch name
1	2	pfue.torola...	1	pfue
2	1	pfue.torola...	0	pfue

Figure 5-71 Archival logging - Backup partitions

Go on with the configuration by clicking the **Next** button. We are now on the *Backup Options* page, as shown in Figure 5-72 on page 264. The wizard will provide you with recommendations based on the number of backup destinations and the number of table spaces in your database for parallelism, number of buffers, and size of the buffers. You can change these and get back again to the recommendations by using the Recommend button. The Quiesce Database option is very useful if you want to prevent users from accessing your database. You cannot take an offline backup while your database is activated, and the quiesce of the database prevents that.

Tip: Only use the Compression option when you are using a device that does not have hardware compression, for example, tape drives. First of all the hardware compression costs you no CPU time, and secondly, it is more efficient in most cases.



Specify options for the backup.

Specify the parallelism (the number of buffer manipulators), number of buffers and the size of each buffer for the backup. All of these options affect the performance of the backup operation. The values displayed are recommended based on values you have already entered in the wizard. If you change any values on this page you may click Recommend again to have the wizard recalculate the values.

You can also choose to quiesce the database before the backup to remove users from the system and ensure the offline backup will succeed.

Buffers

Total Number of Backup Destinations: 1

Total Number of Table Spaces to Backup: 3

Parallelism 1 (Recommenc

Number of Buffers 2 (Recommenc

Size of Each Buffer in 4K pages 1024 (Recommenc

Quiesce Database

[✓] Quiesce the database before initiating the offline operation

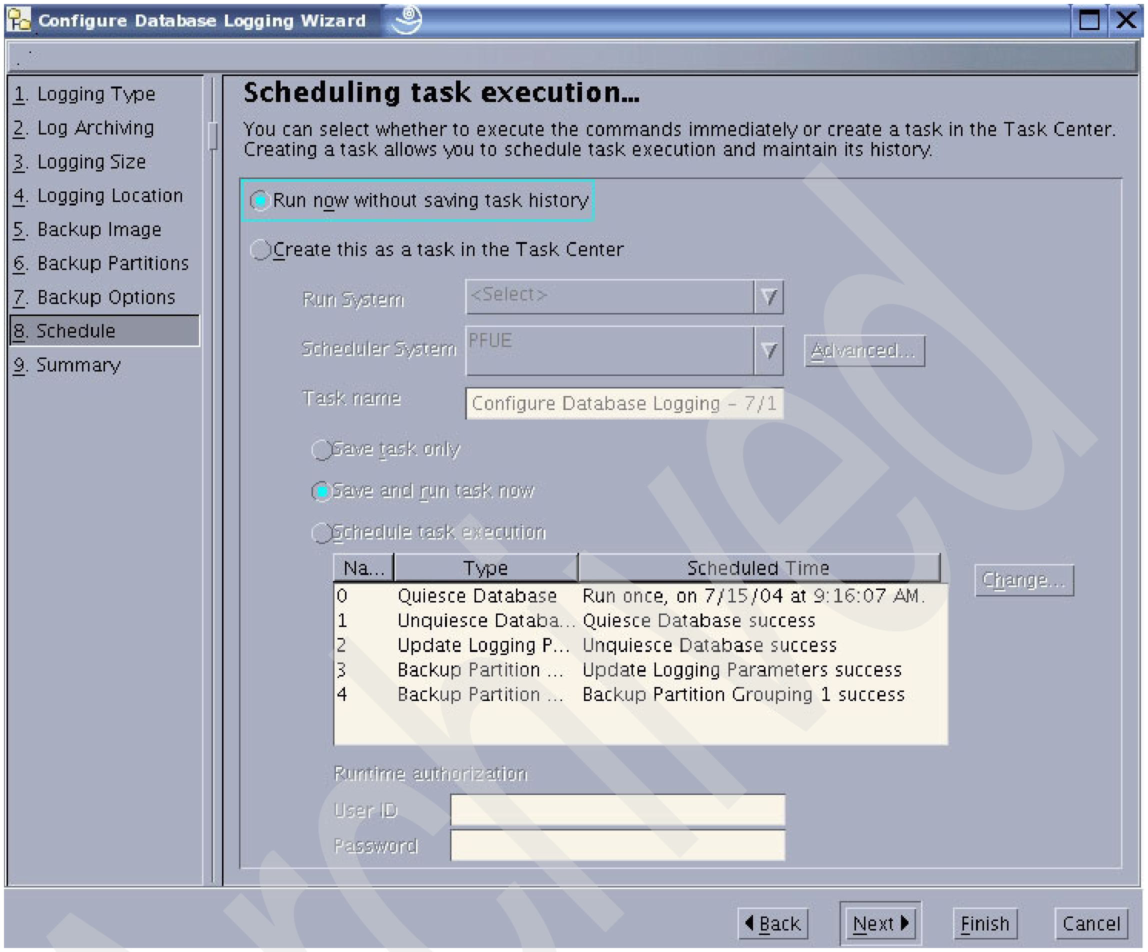
Compression

[ ] Compress backup image

Figure 5-72 Archival logging - Backup options

We get to the next page, the Schedule, by clicking **Next**. This page is shown in Figure 5-73 on page 265. Select the mode you want to run (now or scheduled) and click the **Next** button to go to the Summary page shown in Figure 5-74 on page 266. Review the settings you entered and when everything is correct complete the wizard by clicking the **Finish** button.

Because we are doing an offline backup, DB2 displays the warning message (as shown in Figure 5-75 on page 266), which you have to confirm by clicking **OK**. If you click **Cancel** you go back to the Summary page of the database logging configuration wizard.



Scheduling task execution...

You can select whether to execute the commands immediately or create a task in the Task Center. Creating a task allows you to schedule task execution and maintain its history.

[✓ Run now without saving task history]

Create this as a task in the Task Center

Run System: <Select>

Scheduler System: PFUE

Task name: Configure Database Logging - 7/1

Save task only

[✓ Save and run task now]

Schedule task execution

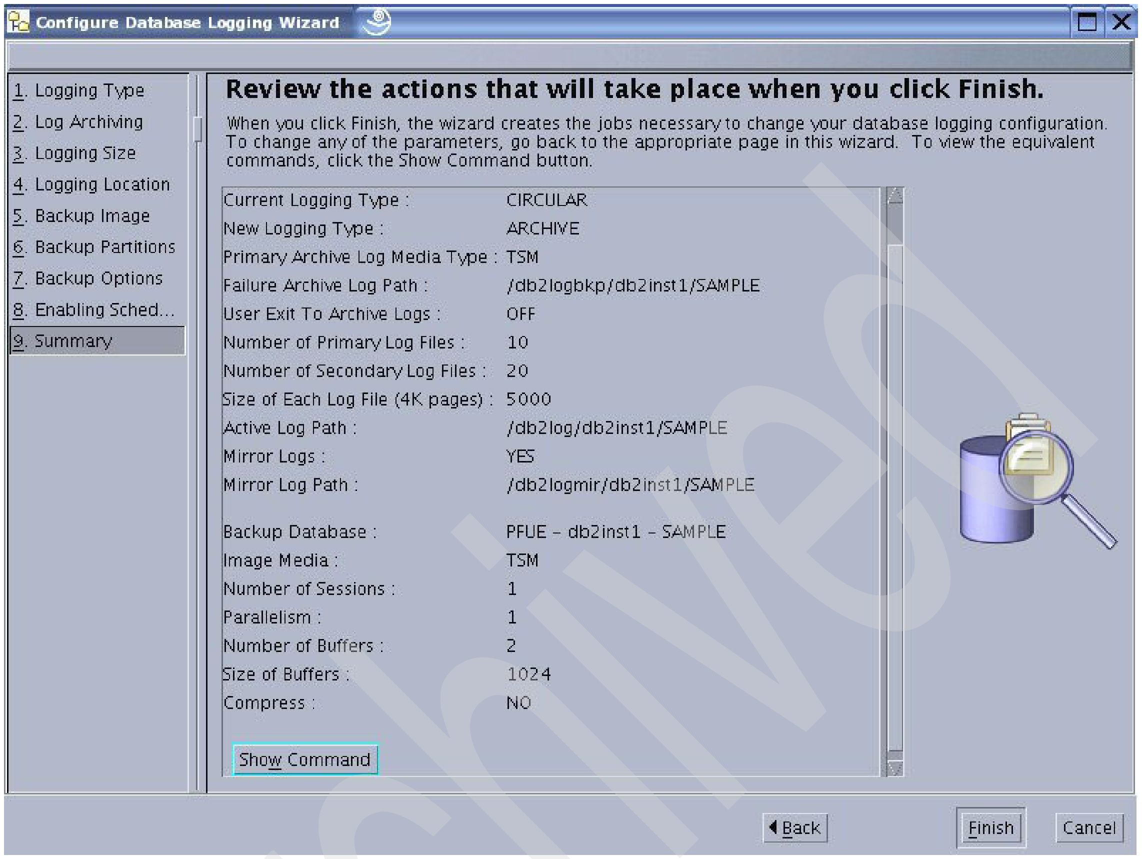
Na...	Type	Scheduled Time
0	Quiesce Database	Run once, on 7/15/04 at 9:16:07 AM.
1	Unquiesce Databa...	Quiesce Database success
2	Update Logging P...	Unquiesce Database success
3	Backup Partition ...	Update Logging Parameters success
4	Backup Partition ...	Backup Partition Grouping 1 success

Runtime authorization

User ID:

Password:

Figure 5-73 Archival logging - Enabling scheduling

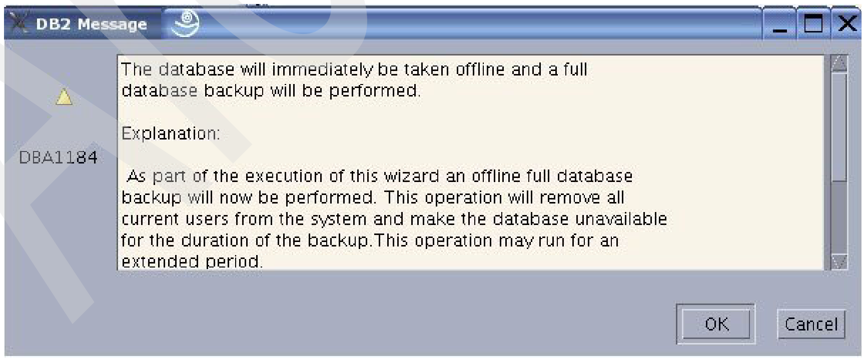


Review the actions that will take place when you click Finish.

When you click Finish, the wizard creates the jobs necessary to change your database logging configuration. To change any of the parameters, go back to the appropriate page in this wizard. To view the equivalent commands, click the Show Command button.

Current Logging Type :	CIRCULAR
New Logging Type :	ARCHIVE
Primary Archive Log Media Type :	TSM
Failure Archive Log Path :	/db2logbkp/db2inst1/SAMPLE
User Exit To Archive Logs :	OFF
Number of Primary Log Files :	10
Number of Secondary Log Files :	20
Size of Each Log File (4K pages) :	5000
Active Log Path :	/db2log/db2inst1/SAMPLE
Mirror Logs :	YES
Mirror Log Path :	/db2logmir/db2inst1/SAMPLE
Backup Database :	PFUE - db2inst1 - SAMPLE
Image Media :	TSM
Number of Sessions :	1
Parallelism :	1
Number of Buffers :	2
Size of Buffers :	1024
Compress :	NO

Figure 5-74 Archival logging - Summary



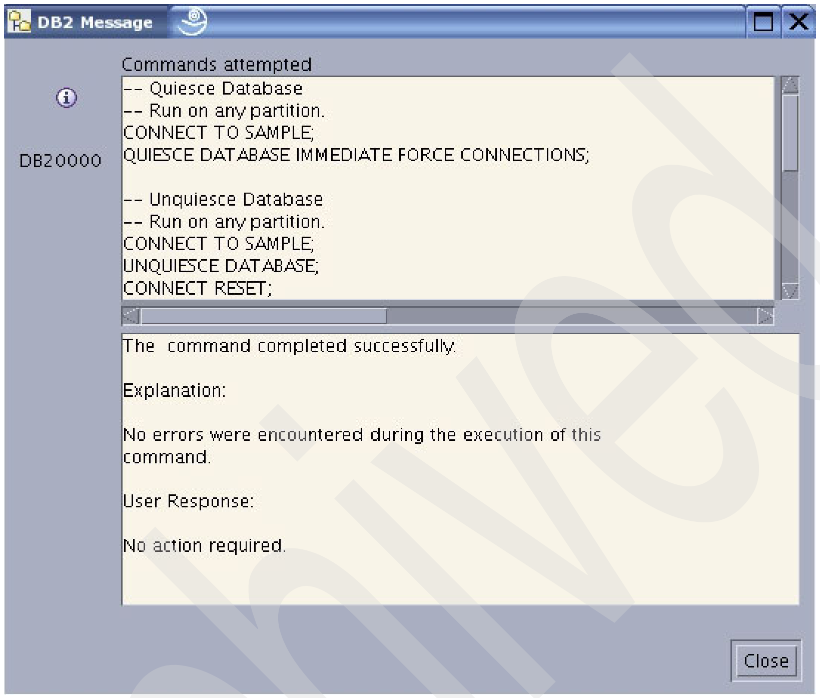
The database will immediately be taken offline and a full database backup will be performed.

Explanation:

As part of the execution of this wizard an offline full database backup will now be performed. This operation will remove all current users from the system and make the database unavailable for the duration of the backup.This operation may run for an extended period.

Figure 5-75 Archival logging - Backup message

When all tasks run successfully you get the message shown in Figure 5-76. If there was a problem you get a similar screen showing the error message. Shut the window with the **Close** button.



```
-- Quiesce Database
-- Run on any partition.
CONNECT TO SAMPLE;
QUIESCE DATABASE IMMEDIATE FORCE CONNECTIONS;

-- Unquiesce Database
-- Run on any partition.
CONNECT TO SAMPLE;
UNQUIESCE DATABASE;
CONNECT RESET;
```

```
The  command completed successfully.

Explanation:

No errors were encountered during the execution of this
command.

User Response:

No action required.
```

Figure 5-76 Archival logging - Success message

5.1.3 Self-tuning for backup and restore

DB2 UDB V8.2 will now automatically choose the number of buffers, the buffer size, and the parallelism settings for both backup and restore operations. The values chosen are based on the amount of memory available, the number of processors available, and the database configuration. The objective is to minimize the amount of time required for backup and restore operations to complete. The BACKUP DATABASE and RESTORE DATABASE commands will automatically choose an optimal value for the following parameters whenever they are not explicitly specified:

- WITH num-buffers BUFFERS
- PARALLELISM n
- BUFFER buffer-size

For restore database operations, a multiple of the buffer size used for the backup operation will always be used.

The values specified by the database manager configuration parameters BACKBUFSZ and RESTBUFSZ are ignored. If you want to use these values, you must explicitly specify them when you issue the BACKUP DATABASE or RESTORE DATABASE command.

5.1.4 Recover database

The new RECOVER DATABASE command combines the functionality of the RESTORE DATABASE and ROLLFORWARD DATABASE commands. When you use this command, you specify the point-in-time to which you want the database recovered. You do not need to indicate which database backup image must be restored or which log files are required to reach the specified point-in-time. A point-in-time restore affects all partitions that are listed in the db2nodes.cfg file. The RECOVER DATABASE command also supports recovery operations to the end of the log files. In this case only the partitions specified are affected. If no partitions are specified, it affects all partitions that are listed in the db2nodes.cfg file.

The recover database command will first restore the catalog partition and then in parallel the remaining partitions. Then it will initiate the rollforward in parallel on all partitions.

An example for a point-in-time recovery is:

```
db2 RECOVER DATABASE sample TO 20040801121212 USING LOCAL TIME
```

An example for an end-of-logs recovery is:

```
db2 RECOVER DATABASE sample TO END OF LOGS
```

5.1.5 Automatic setting of prefetch size

The prefetch size for a table space determines the number of pages read from a table space when prefetching is being performed. Because prefetching is a means to improve query performance, setting the correct prefetch size is an important step in performance tuning.

When the prefetch size is not specified for a table space, then DB2 uses the value for the DFT_PREFETCH_SZ configuration parameter as the default. This parameter can, since DB2 UDB V8.2, be set to AUTOMATIC, which allows DB2 to calculate an appropriate prefetch size for a table space based on the extent size, the number of containers, and the number of physical spindles per container. This frees the user from having to determine the appropriate value for the table space prefetch size and from having to remember to reset this value when any containers are added to or removed from a table space.

An example of setting DFT_PREFETCH_SZ to AUTOMATIC is:

```
db2 UPDATE DB CFG FOR sample USING DFT_PERFETCH_SZ AUTOMATIC
```

5.2 Monitoring

Monitoring is an important part of keeping your database alive and performing well. A good monitoring will help prevent disruptions to your database, because you will be informed of arising problems and can take counteractive measures.

In this section we talk about a few DB2 monitoring instruments, how you can monitor DB2 using them, and about some tools in Linux to monitor the system.

5.2.1 DB2 instruments

DB2 UDB provides you with a rich set of database and application activity monitor elements, such as:

- Locks and deadlocks monitor elements
- Lock wait information monitor elements
- Rollforward monitoring monitor elements
- Table space activity monitor elements
- Table activity monitor elements
- Table reorganization monitor elements
- SQL cursors monitor elements
- SQL statement activity monitor elements
- SQL statement details monitor elements
- Subsection details monitor elements
- Dynamic SQL monitor elements
- Intra-query parallelism monitor elements
- CPU usage monitor elements
- Snapshot monitoring monitor elements
- Event monitoring monitor elements

These topics can all be found in the *IBM DB2 UDB System Monitor Guide and Reference*, SC09-4847. In this redbook we cover only the snapshot monitoring from this list. We also talk about other tools in DB2, which are presented on the next few pages.

We divided the DB2 tools section into two parts: Command line and graphical tools.

Command-line tools

Command-line tools are very useful, particularly in large environments and if you want to code your own scripts that call some monitoring functions.

Snapshot function

The spectra of information you can get with the snapshot function reaches from information about the database manager over applications to buffer pools. To get some of the information, you need to enable the corresponding monitor switches. For example, to get information about the buffer pools you need to have enabled the buffer pool monitor switch. You can do this globally for the buffer pools by issuing the following command:

```
db2 UPDATE DBM CFG USING dft_mon_bufpool ON
```

Or you can also do it by using the update monitor switch command. This setting will only be effective until the application that turned the switch on terminates or a db2stop is performed and it affects only the partition you are connected with, unless you specify the global option. After a db2stop or a terminate of the application, the switches will be updated with the default settings from the database manager configuration. An example of setting the monitor switches globally is:

```
db2 UPDATE MONITOR SWITCH USING bufferpool ON GLOBAL
```

Note: Every switch you turn on will use system resources, so only turn on the switches you really need, for example, the switches for things you are watching.

The syntax of the snapshot command looks like this:

```
GET SNAPSHOT FOR {DATABASE MANAGER | ALL [DCS] DATABASES | ALL [DCS]
APPLICATIONS | ALL BUFFERPOOLS | [DCS] APPLICATION {APPLID appl-id | AGENTID
appl-handle} | FCM FOR ALL DBPARTITIONNUMS | LOCKS FOR APPLICATION {APPLID
appl-id | AGENTID appl-handle} | {ALL | [DCS] DATABASE | [DCS] APPLICATIONS |
TABLES | TABLESPACES | LOCKS | BUFFERPOOLS | DYNAMIC SQL [write-to-file]} ON
database-alias} [AT DBPARTITIONNUM db-partition-number | GLOBAL]
```

The output of a buffer pool snapshot would look like Example 5-2.

Example 5-2 Snapshot for buffer pools

```
[db2inst1@DB2RB01 db2inst1]$ db2 get snapshot for bufferpools on tpch

             Bufferpool Snapshot

Bufferpool name                            = IBMDEFAULTBP
Database name                              = TPCH
Database path                              =
/db2icedata/dbdir/db2inst1/NODE0001/SQL00002/
Input database alias                       = TPCH
Snapshot timestamp                         = 2004-07-13 13.19.56.657079

Buffer pool data logical reads             = 396
Buffer pool data physical reads            = 0
Buffer pool temporary data logical reads   = 0
Buffer pool temporary data physical reads  = 0
Buffer pool data writes                    = 0
Buffer pool index logical reads            = 792
Buffer pool index physical reads           = 0
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Total buffer pool read time (ms)           = 0
Total buffer pool write time (ms)          = 0
Asynchronous pool data page reads          = 0
Asynchronous pool data page writes         = 0
Buffer pool index writes                   = 0
Asynchronous pool index page reads         = 0
Asynchronous pool index page writes        = 0
Total elapsed asynchronous read time       = 0
Total elapsed asynchronous write time      = 0
Asynchronous data read requests            = 0
Asynchronous index read requests           = 0
No victim buffers available                = 0
Direct reads                               = 1188
Direct writes                              = 0
Direct read requests                       = 198
Direct write requests                      = 0
Direct reads elapsed time (ms)             = 19
Direct write elapsed time (ms)             = 0
Database files closed                      = 0
Data pages copied to extended storage      = 0
Index pages copied to extended storage     = 0
Data pages copied from extended storage    = 0
Index pages copied from extended storage   = 0
Unread prefetch pages                      = 0
Vectored IOs                               = 0
Pages from vectored IOs                    = 0
Block IOs                                  = 0
Pages from block IOs                       = 0
Physical page maps                         = 0

Node number                                = 1
Tablespaces using bufferpool               = 3
Alter bufferpool information:
 Pages left to remove                      = 0
 Current size                              = 1000
```

```
Post-alter size                           = 1000
```

With the data you find in the snapshot you can, for instance, calculate the overall buffer pool hit ratio as:

```
   1 - ((Buffer pool data physical reads + Buffer pool index physical reads) /
   (Buffer pool data logical reads + Buffer pool index logical reads))
```

The higher the number the better. Good values are around a 90 percent hit ratio. You have to reconsider your buffer pool when the hit ratio is much lower than 90 percent. Normally, increasing the size of the buffer pool will lead to a better hit ratio. But you will get to a certain point where you are not earning much more in respect to the hit ratio with what you have to give to the buffer pool in terms of memory.

An important snapshot is the application snapshot where you can see what applications do. If the application is using dynamic SQL you can also see the SQL it uses. You can see the sorts it did, the sort time, and so on. Example 5-3 shows you a snapshot for all applications.

Example 5-3   Get snapshot for applications

```
[db2inst1@DB2RB01 db2inst1]$ db2 get snapshot for all applications on tpch

Application Snapshot

Application handle                         = 65591
Application status                         = UOW Waiting
Status change time                         =
Application code page                      = 1208
Application country/region code            = 2
DUOW correlation token                     = *N1.db2inst1.040713154045
Application name                           = db2bp
Application ID                             = *N1.db2inst1.040713154045
Sequence number                            = 0003
TP Monitor client user ID                  =
TP Monitor client workstation name         =
TP Monitor client application name         =
TP Monitor client accounting string        =

Connection request start timestamp         = 2004-07-13 11.40.45.359732
Connect request completion timestamp       = 2004-07-13 11.40.45.360165
Application idle time                      =
CONNECT Authorization ID                   = DB2INST1
Client login ID                            = db2inst1
Configuration NNAME of client              =
Client database manager product ID         = SQL08020
Process ID of client application           = 11942
```

```
Platform of client application              = LINUX
Communication protocol of client            = Local Client

Inbound communication address               = *N1.db2inst1

Database name                               = TPCH
Database path                               =
/db2icedata/dbdir/db2inst1/NODE0001
/SQL00002/
Client database alias                       = TPCH
Input database alias                        =
Last reset timestamp                        =
Snapshot timestamp                          = 2004-07-13 14.27.44.714032
The highest authority level granted         =
        Direct DBADM authority
        Direct CREATETAB authority
        Direct BINDADD authority
        Direct CONNECT authority
        Direct CREATE_NOT_FENC authority
        Direct LOAD authority
Direct IMPLICIT_SCHEMA authority
        Direct CREATE_EXT_RT authority
        Direct QUIESCE_CONN authority
        Indirect SYSADM authority
        Indirect CREATETAB authority
        Indirect BINDADD authority
        Indirect CONNECT authority
        Indirect IMPLICIT_SCHEMA authority
Coordinating database partition number      = 1
Current database partition number           = 1
Coordinator agent process or thread ID      = 11891
Agents stolen                               = 0
Agents waiting on locks                     = 0
Maximum associated agents                   = 1
Priority at which application agents work   = 0
Priority type                               = Dynamic

Lock timeout (seconds)                      = -1
Locks held by application                   = 0
Lock waits since connect                    = 0
Time application waited on locks (ms)       = 0
Deadlocks detected                          = 0
Lock escalations                            = 0
Exclusive lock escalations                  = 0
Number of Lock Timeouts since connected     = 0
Total time UOW waited on locks (ms)         = 0

Total sorts                                 = 0
Total sort time (ms)                        = 0
Total sort overflows                       = 0

Data pages copied to extended storage      = 0
Index pages copied to extended storage     = 0
Data pages copied from extended storage    = 0
Index pages copied from extended storage   = 0
Buffer pool data logical reads             = 0
Buffer pool data physical reads            = 0
Buffer pool temporary data logical reads   = 0
Buffer pool temporary data physical reads  = 0
Buffer pool data writes                    = 0
Buffer pool index logical reads            = 0
Buffer pool index physical reads           = 0
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Buffer pool index writes                   = 0
Total buffer pool read time (ms)           = 0
Total buffer pool write time (ms)          = 0
Time waited for prefetch (ms)              = 0
Unread prefetch pages                      = 0
Direct reads                               = 0
Direct writes                              = 0
Direct read requests                       = 0
Direct write requests                      = 0
Direct reads elapsed time (ms)             = 0
Direct write elapsed time (ms)             = 0

Number of SQL requests since last commit   = 0
Commit statements                          = 2
Rollback statements                        = 0
Dynamic SQL statements attempted           = 9
Static SQL statements attempted            = 2
Failed statement operations                = 0
Select SQL statements executed             = 2
Update/Insert/Delete statements executed   = 0
DDL statements executed                    = 0
Internal automatic rebinds                 = 0
Internal rows deleted                      = 0
Internal rows inserted                     = 0
Internal rows updated                      = 0
Internal commits                           = 1
Internal rollbacks                         = 0
Internal rollbacks due to deadlock         = 0
Binds/precompiles attempted                = 0
Rows deleted                               = 0
Rows inserted                              = 0
Rows updated                               = 0
Rows selected                              = 8
Rows read                                  = 9
Rows written                               = 0

UOW log space used (Bytes)                 = 0
Previous UOW completion timestamp          =
Elapsed time of last completed uow (sec.ms)= 0.000000
UOW start timestamp                        =
UOW stop timestamp                         =
UOW completion status                      =

Open remote cursors                        = 0
Open remote cursors with blocking          = 0
Rejected Block Remote Cursor requests      = 0
Accepted Block Remote Cursor requests      = 2
Open local cursors                         = 0
Open local cursors with blocking           = 0
Total User CPU Time used by agent (s)      = 0.109989
Total System CPU Time used by agent (s)    = 0.139986
Host execution elapsed time                = 0.000000

Package cache lookups                      = 5
Package cache inserts                      = 2
Application section lookups                = 9
Application section inserts                = 2
Catalog cache lookups                      = 8
Catalog cache inserts                      = 1
Catalog cache overflows                    = 0
Catalog cache high water mark              = 0

Workspace Information

 Shared high water mark                    = 371839
 Total shared overflows                    = 0
 Total shared section inserts              = 2
 Total shared section lookups              = 2
 Private high water mark                   = 0
 Total private overflows                   = 0
 Total private section inserts             = 0
 Total private section lookups             = 0

Most recent operation                      = Static Commit
Most recent operation start timestamp      =
Most recent operation stop timestamp       =
Agents associated with the application     = 1
Number of hash joins                       = 0
Number of hash loops                       = 0
Number of hash join overflows              = 0
Number of small hash join overflows        = 0

Statement type                             =
Statement                                  = Static Commit
Section number                             = 0
Application creator                        =
Package name                               =
Consistency Token                          =
Cursor name                                =
Statement database partition number        = 0
Statement start timestamp                  =
Statement stop timestamp                   =
Elapsed time of last completed stmt(sec.ms)= 0.000000
Total Statement user CPU time              = 0.000000
Total Statement system CPU time            = 0.000000
SQL compiler cost estimate in timerons     = 0
SQL compiler cardinality estimate          = 0
Degree of parallelism requested            = 0
Number of agents working on statement      = 0
Number of subagents created for statement  = 0
Statement sorts                            = 0
Total sort time                            = 0
Sort overflows                             = 0
Rows read                                  = 0
Rows written                               = 0
Rows deleted                               = 0
Rows updated                               = 0
Rows inserted                              = 0
Rows fetched                               = 0
Buffer pool data logical reads             = 0
Buffer pool data physical reads            = 0
Buffer pool temporary data logical reads   = 0
Buffer pool temporary data physical reads  = 0
Buffer pool index logical reads            = 0
Buffer pool index physical reads           = 0
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Blocking cursor                            = NO

Agent process/thread ID                    = 11891
  Database partition number                = 1
  Agent Lock timeout (seconds)             = -1
  Memory usage for agent:

    Memory Pool Type                       = Application Heap
       Current size (bytes)                = 147456
       High water mark (bytes)             = 147456
       Configured size (bytes)             = 491520
```

You see that you can find a lot of information in the application snapshot. It is a very useful tool to monitor the application activities and to resolve problems of applications or to narrow down in which field you have to investigate further.

db2mtrk

db2mtrk provides a complete report of memory status for instances, databases and agents. This command outputs the following memory pool allocation information:

- Current size
- Maximum size (hard limit)
- Largest size (high water mark)
- Type (identifier indicating function for which memory will be used)
- Agent who allocated pool (only if the pool is private)

The same information is also available from the Snapshot monitor.

In a partitioned database environment, this command can be invoked from any database partition defined in the db2nodes.cfg file. It returns information only for that partition. This command does not return information for remote servers.

Usage:

```
db2mtrk -i | -d | -p [-m | -w] [-v] [-r interval [count]] [-h]
```

The options are:

-i		Display instance level memory usage.
-d		Display database level memory usage.
-p		Display agent private memory usage.
-m		Display maximum usage information.
-w		Display watermark usage information.
-v		Display verbose memory usage information.
-r		Run in repeat mode.
	interval	Amount of seconds to wait between reports
	count	Number of reports to generate before quitting
-h		Display this help screen.

Running db2mtrk with the instance and database option in verbose mode gives you the output of Example 5-4.

Example 5-4 db2mtrk

```
[db2inst1@DB2RB01 db2inst1]$ db2mtrk -i -d -v
```

```
Tracking Memory on: 2004/07/13 at 14:44:13

Memory for instance

   Database Monitor Heap is of size 180224 bytes
   Other Memory is of size 3244032 bytes
   FCMBP Heap is of size 17432576 bytes
   Total: 20856832 bytes

Memory for database: TPCH

   Backup/Restore/Util Heap is of size 16384 bytes
   Package Cache is of size 917504 bytes
   Catalog Cache Heap is of size 131072 bytes
   Buffer Pool Heap is of size 21659648 bytes
   Buffer Pool Heap is of size 4325376 bytes
   Buffer Pool Heap is of size 655360 bytes
   Buffer Pool Heap is of size 393216 bytes
   Buffer Pool Heap is of size 262144 bytes
   Buffer Pool Heap is of size 196608 bytes
   Lock Manager Heap is of size 491520 bytes
   Database Heap is of size 3571712 bytes
   Other Memory is of size 49152 bytes
   Application Control Heap is of size 16384 bytes
   Application Control Heap is of size 16384 bytes
   Application Control Heap is of size 16384 bytes
   Application Control Heap is of size 16384 bytes
   Application Control Heap is of size 327680 bytes
   Application Group Shared Heap is of size 57344000 bytes
   Application Control Heap is of size 32768 bytes
   Application Control Heap is of size 16384 bytes
   Application Control Heap is of size 32768 bytes
   Application Control Heap is of size 32768 bytes
   Application Control Heap is of size 327680 bytes
   Application Group Shared Heap is of size 57344000 bytes
   Total: 148193280 bytes
```

When you use the -m option (maximum) and then compare with the -w option (watermark), you can find hints about memory heaps that could be too big, for example, not fully used.

SQL

It is always a good idea to have one or a few SQL commands that reflect your normal workload to test the performance of your database. This is especially useful if you plan to change some configuration parameters. You can run your query, measure the time, do your configuration change, run the query again, and measure. So in most cases you get a useful hint if your change had a positive impact. Or at least you realize when your database will slow down.

You can also use it to see if a scale up or scale out was successful, for example, if you get any performance benefits from it.

db2diag

db2diag is a new tool to analyze the db2diag.log. It can help you find out what the ZRC codes mean, it can scan the db2diag.log for a list of errors, or it can just show you ongoing the last few entries of the db2diag.log. Example 5-5 shows a **db2diag -h**, which gives you all the options of db2diag. You can even get more detailed help with **db2diag -h all**.

Example 5-5 db2diag

```
db2inst2@db2rb09:~> db2diag -h

db2diag - The db2diag.log Analysis Tool

 db2diag is a tool for filtering and formatting the db2diag.log file

 Command syntax:
                   .------------.  .--------------.
                   V            |  V              |
      >>--db2diag--+------------+--+--------------+-->< 
                    |          |    |            |
                     --option--      --filename--

 Command parameters:

 filename          - one or more space-separated path names of diagnostic logs
 -help   , -h , ?  - help information. To get help on help, try "db2diag -h h"
 -filter , -g      - case-sensitive search for a list of field-pattern pairs
 -gi               - case-insensitive search for a list of field-pattern pairs
 -gv               - case-sensitive invert matching
 -gvi    , -giv    - case-insensitive invert matching
 -invert , -v      - invert the sense of matching for all filtering options
 -exist            - record field must exist in order to be processed
 -pid              - find all records for a list of process IDs
 -tid              - find all records for a list of thread IDs
 -node   , -n      - find all records for a list of nodes
 -error  , -e      - find all records for a list of errors
 -level  , -l      - find all records for a list of severity levels
 -history, -H      - display the history of logged records for a time interval
 -time   , -t      - display all the records within a particular time interval
 -count  , -c      - display a count of matching records
 -verbose, -V      - display all record fields whether they contain data or not
 -strict           - display records using one "field: value" pair per line
```

```
-cbe              - display records in the Common Base Event (CBE) format
-fmt              - format tool's output using a format string
-output , -o      - save output into a file
-follow , -f      - continuously display appended records as the file grows
-archive, -A      - archive a diagnostic log file
-rc               - display descriptions of DB2 error return codes, ZRC or ECF

"db2diag -h <option1[,option2[,option3...]]>" - displays additional help and
 usage examples for one or more options specified in the options list

"db2diag -h brief"    - displays help for all options without examples

"db2diag -h examples" - displays a few typical examples to get started

"db2diag -h tutorial" - displays more advanced examples covering all features

"db2diag -h notes"    - displays usage notes and restrictions that apply

"db2diag -h all"      - displays help in the most complete form with detailed
                        information about all options and usage examples
```

db2pd

db2pd is a new utility that can be used to retrieve statistics from a running DB2 instance or database. It is similar to the onstat utility for Informix.

The tool can provide a wide range of information useful for troubleshooting and problem determination, performance improvements, and application development design, including:

- Locks
- Buffer pools
- Table spaces
- Containers
- Dynamic SQL statements
- Agents
- Applications
- Memory pools and sets
- Transactions
- Logs
- And others

The tool collects this information without acquiring any latches or using any engine resources. It is therefore possible (and expected) to retrieve information that is changing while db2pd is collecting information; hence, the data may not be completely accurate. However, two benefits to collecting information without latching include faster retrieval and no competition for engine resources.

You can pass the options directly on the command line or you can enter in the interactive mode. We show you in Example 5-6 one example of passing the options to db2pd.

Example 5-6 db2pd

```
[db2inst1@DB2RB01 db2inst1]$ db2pd -fcm

Database Partition 1 -- Active -- Up 0 days 22:39:22

FCM:
Address    Bufs       BufLWM     Anchors    AnchLWM    Entries    EntryLWM
RQBs       RQBLWM
0x100D8040 4090       3992       1533       1523       1536       1524
2008       1995

DBP        TotBufSnt  TotBufRcv  Status
0          0          0          Undefined
1          0          0          Active
2          4825       4058       Active
3          4657       3893       Active
4          3845       3733       Active
5          3845       3733       Active
6          3845       3733       Active
```

A complete list of the command options can be found in the *IBM DB2 UDB Command Reference V8*, SC09-4828.

Graphical tools

Graphical tools help you to simplify the monitoring, and some have the ability to visualize the monitored data. These tools can sometimes be very useful to better understand and solve a problem. Besides the three tools we are talking about here there are a lot more tools on the market, for example, the IBM DB2 Performance Expert.

We show on the following pages a selection of three tools you get with your DB2.

Activity monitor

To start the Activity monitor right-click in the Control Center (db2cc) the database you want the activity monitor to run on and select the **Activity Monitor** option, as indicated with the red arrow in Figure 5-77 on page 282.

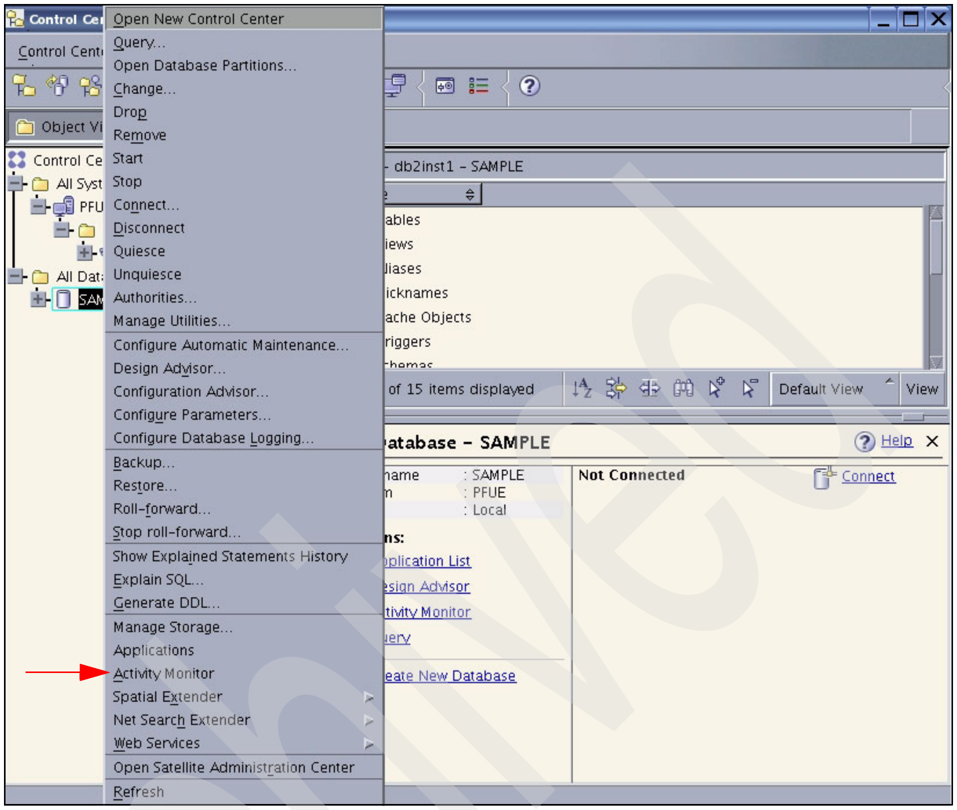
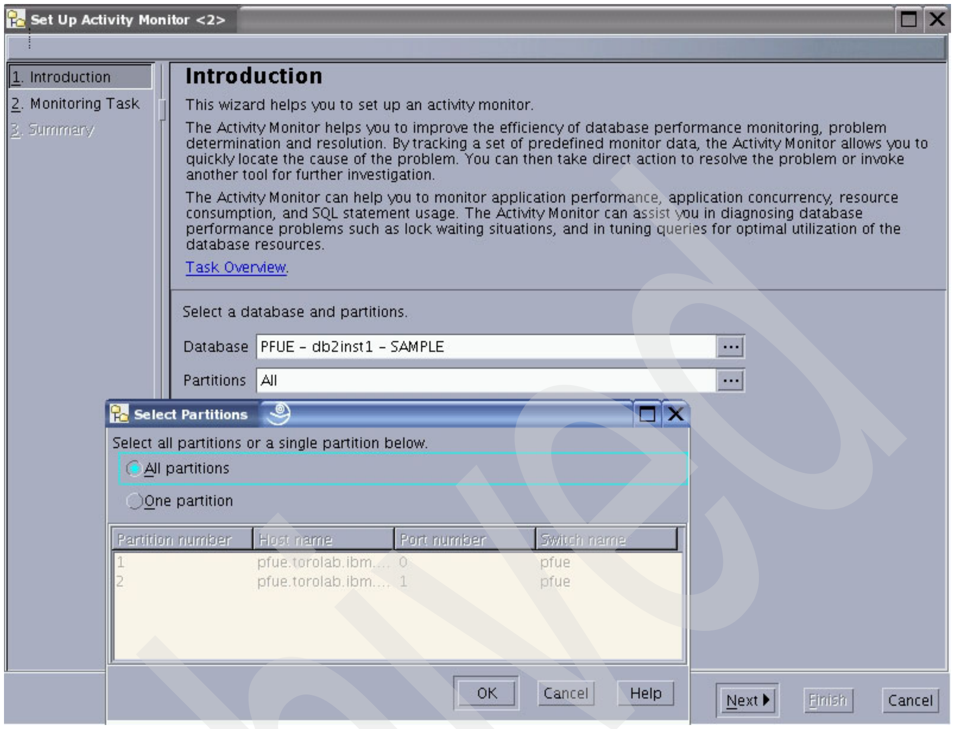


Figure 5-77 Activity monitor - Start

You are taken into the setup dialog, where on the first page you can select the database and the partitions you want your activity monitor to run on (Figure 5-78 on page 283). The Select Partitions dialog is only shown when you click the button on the right side of the Partitions box. When you have set the settings for the database and partitions continue by clicking **Next**.



Set Up Activity Monitor <2>

1. Introduction
2. Monitoring Task
3. Summary

Introduction

This wizard helps you to set up an activity monitor.

The Activity Monitor helps you to improve the efficiency of database performance monitoring, problem determination and resolution. By tracking a set of predefined monitor data, the Activity Monitor allows you to quickly locate the cause of the problem. You can then take direct action to resolve the problem or invoke another tool for further investigation.

The Activity Monitor can help you to monitor application performance, application concurrency, resource consumption, and SQL statement usage. The Activity Monitor can assist you in diagnosing database performance problems such as lock waiting situations, and in tuning queries for optimal utilization of the database resources.

Task Overview.

Select a database and partitions.

Database: PFUE - db2inst1 - SAMPLE

Partitions: All

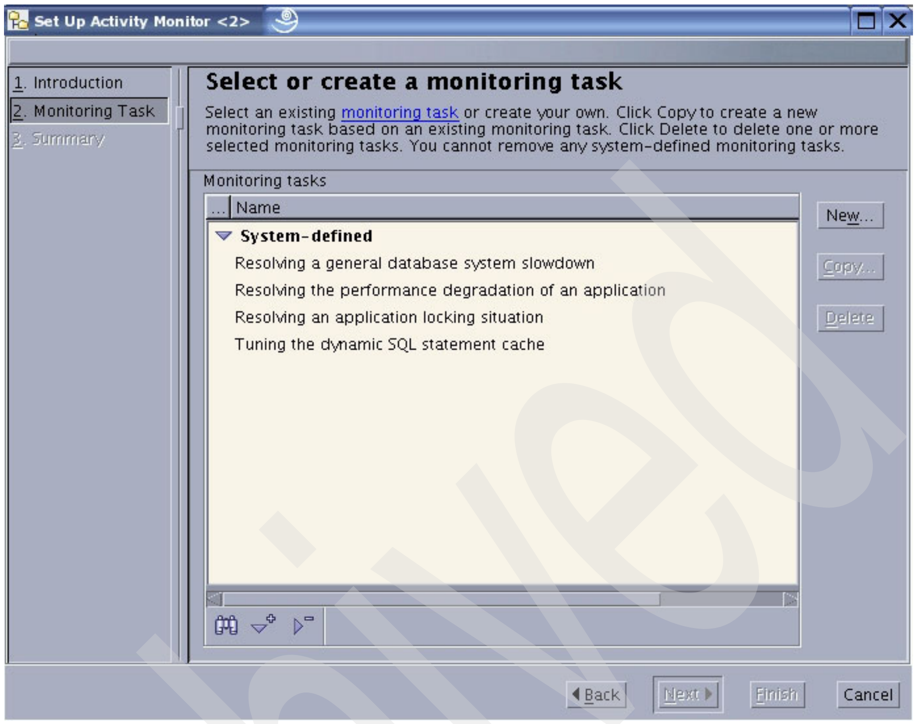
Select Partitions

Select all partitions or a single partition below.

[✓ All partitions] One partition

Partition number	Host name	Port number	Switch name
1	pfue.torolab.ibm....	0	pfue
2	pfue.torolab.ibm....	1	pfue

Figure 5-78 Activity monitor set up - Introduction



Select or create a monitoring task

Select an existing monitoring task or create your own. Click Copy to create a new monitoring task based on an existing monitoring task. Click Delete to delete one or more selected monitoring tasks. You cannot remove any system-defined monitoring tasks.

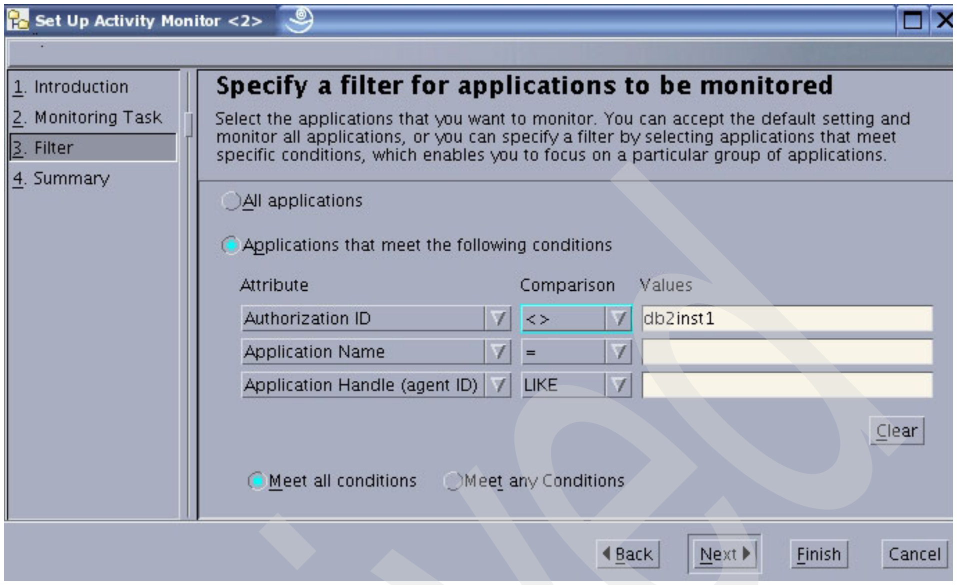
Monitoring tasks

...	Name
	System-defined
	Resolving a general database system slowdown
	Resolving the performance degradation of an application
	Resolving an application locking situation
	Tuning the dynamic SQL statement cache

Figure 5-79 Activity monitor set up - Monitoring task

The Monitoring Task page shown in Figure 5-79 has several system-defined monitoring tasks to choose from. We first go through a system-defined monitoring task and then show how to create a user-defined task.

As an example for a system-defined task we choose “Resolving a general database system slowdown”. Do so by selecting the task in the Monitor tasks window, and then click the **Next** button. We then get to the Filter page, as shown in Figure 5-80 on page 285 where we can define if we want to monitor all applications or if we want to narrow it down to some conditions that the application has to meet. Click **Next** to get to the Summary page shown in Figure 5-81 on page 286.


Set Up Activity Monitor <2>

1. Introduction
2. Monitoring Task
3. Filter
4. Summary

Specify a filter for applications to be monitored

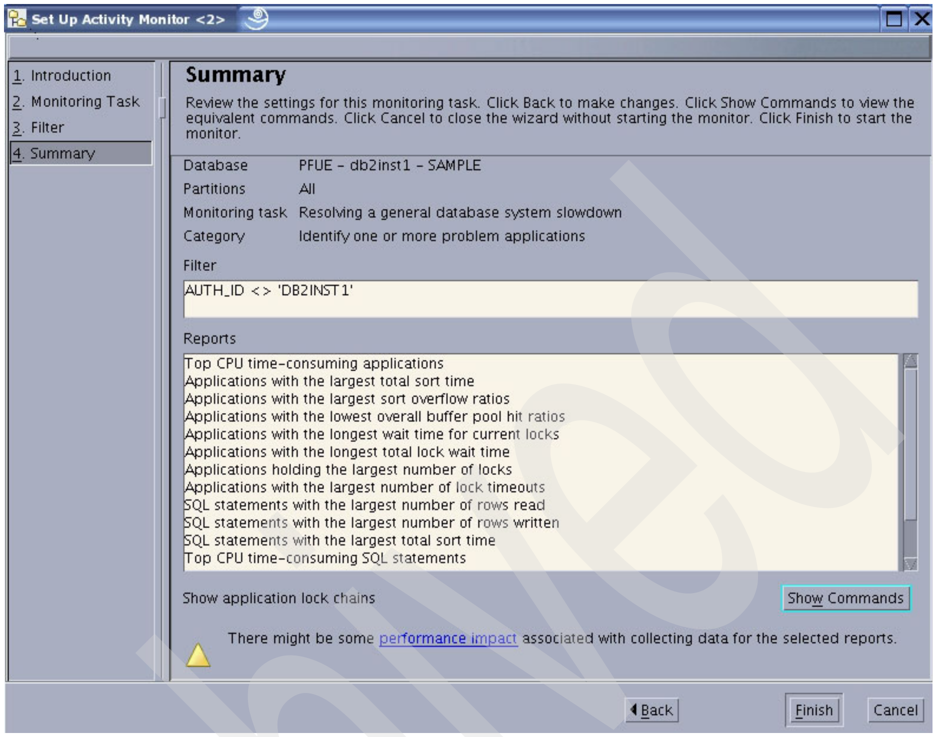
Select the applications that you want to monitor. You can accept the default setting and monitor all applications, or you can specify a filter by selecting applications that meet specific conditions, which enables you to focus on a particular group of applications.

All applications [✓ Applications that meet the following conditions]

Attribute	Comparison	Values
Authorization ID	<>	db2inst1
Application Name	=	
Application Handle (agent ID)	LIKE	

[✓ Meet all conditions] Meet any Conditions

Figure 5-80 Activity monitor set up - Filter


Set Up Activity Monitor <2>

1. Introduction
2. Monitoring Task
3. Filter
4. Summary

Summary

Review the settings for this monitoring task. Click Back to make changes. Click Show Commands to view the equivalent commands. Click Cancel to close the wizard without starting the monitor. Click Finish to start the monitor.

Database	PFUE - db2inst1 - SAMPLE
Partitions	All
Monitoring task	Resolving a general database system slowdown
Category	Identify one or more problem applications

Filter

```
AUTH_ID <> 'DB2INST1'
```

Reports

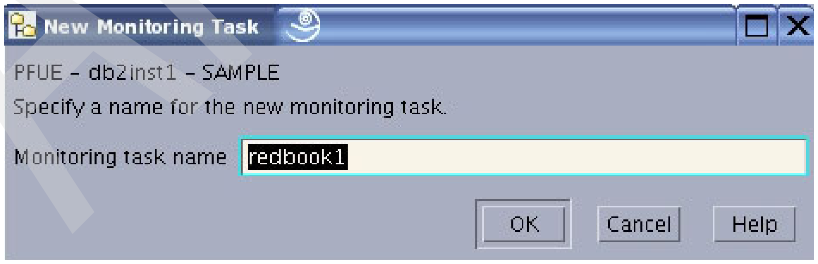
```
Top CPU time-consuming applications
Applications with the largest total sort time
Applications with the largest sort overflow ratios
Applications with the lowest overall buffer pool hit ratios
Applications with the longest wait time for current locks
Applications with the longest total lock wait time
Applications holding the largest number of locks
Applications with the largest number of lock timeouts
SQL statements with the largest number of rows read
SQL statements with the largest number of rows written
SQL statements with the largest total sort time
Top CPU time-consuming SQL statements
```

Show application lock chains

There might be some performance impact associated with collecting data for the selected reports.

Figure 5-81 Activity monitor set up - Summary

The Summary page shows the filter you selected and the reports that will be available. Before we finish we want to go through the creation of a user-defined monitoring task. We create a new task with the **New** button on the Monitoring Task page that was shown in Figure 5-79 on page 284. The New Monitoring Task screen comes up and we must enter a name for the monitoring task in the corresponding text box (shown in Figure 5-82). Click **OK** when done.


New Monitoring Task

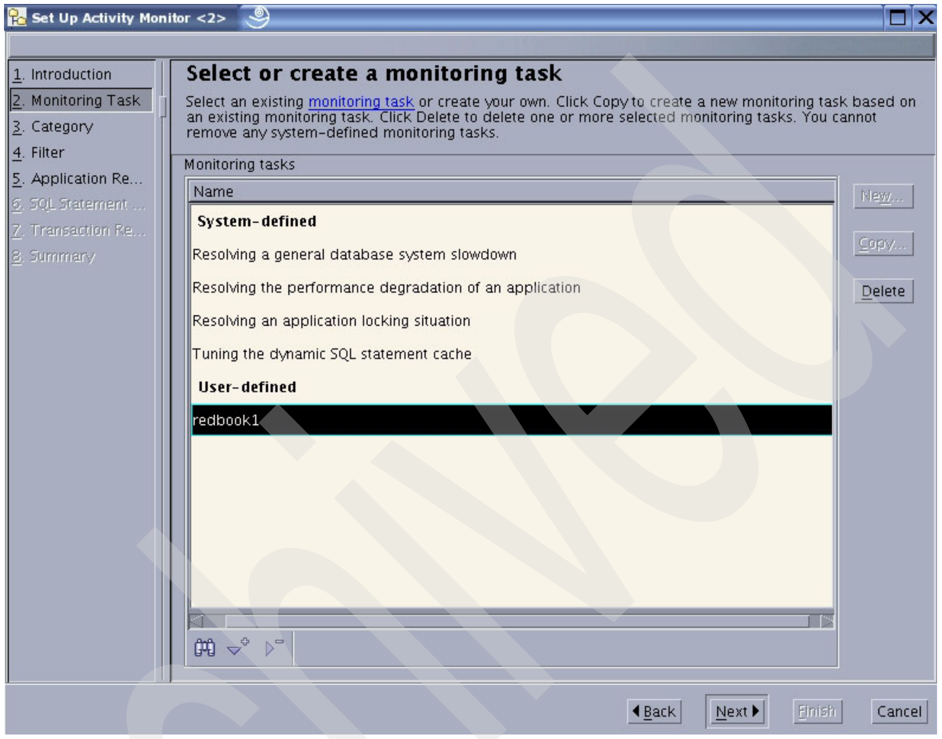
PFUE - db2inst1 - SAMPLE

Specify a name for the new monitoring task.

Monitoring task name: redbook1

Figure 5-82 Activity monitor - New monitoring task

We are now back on the Monitoring Task page, which is updated with the name of the user-defined monitoring task, shown in Figure 5-83. Click the **Next** button to get to the next page.



Select or create a monitoring task

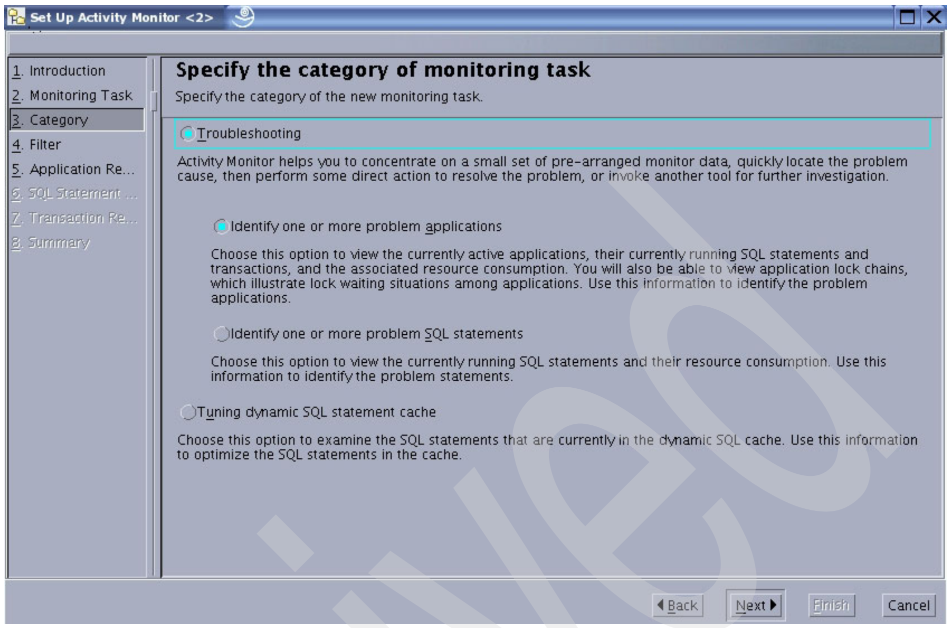
Select an existing monitoring task or create your own. Click Copy to create a new monitoring task based on an existing monitoring task. Click Delete to delete one or more selected monitoring tasks. You cannot remove any system-defined monitoring tasks.

Monitoring tasks

Name
System-defined
Resolving a general database system slowdown
Resolving the performance degradation of an application
Resolving an application locking situation
Tuning the dynamic SQL statement cache
User-defined
redbook1

Figure 5-83 Activity monitor set up - User-defined monitoring task

On the Category page shown in Figure 5-84 on page 288, we define in which direction we want to monitor. We chose, as an example, the “Identify one or more problem applications” category. Click **Next** to get to the Filter page, shown in Figure 5-85 on page 288. The Filter page gives you the same options as for the system-defined tasks shown in Figure 5-80 on page 285. Click **Next** when you have adjusted the filter to your needs.


Set Up Activity Monitor <2>

1. Introduction
2. Monitoring Task
3. Category
4. Filter
5. Application Re...
6. SQL Statement ...
7. Transaction Re...
8. Summary

Specify the category of monitoring task

Specify the category of the new monitoring task.

Troubleshooting

Activity Monitor helps you to concentrate on a small set of pre-arranged monitor data, quickly locate the problem cause, then perform some direct action to resolve the problem, or invoke another tool for further investigation.

Identify one or more problem applications

Choose this option to view the currently active applications, their currently running SQL statements and transactions, and the associated resource consumption. You will also be able to view application lock chains, which illustrate lock waiting situations among applications. Use this information to identify the problem applications.

Identify one or more problem SQL statements

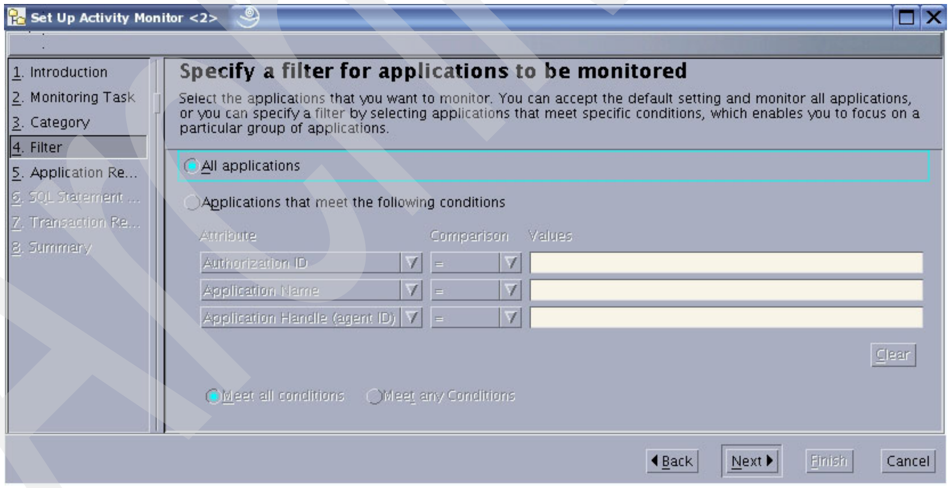
Choose this option to view the currently running SQL statements and their resource consumption. Use this information to identify the problem statements.

Tuning dynamic SQL statement cache

Choose this option to examine the SQL statements that are currently in the dynamic SQL cache. Use this information to optimize the SQL statements in the cache.

Back | Next | Finish | Cancel

Figure 5-84 Activity monitor set up - Category for user-defined task


Set Up Activity Monitor <2>

1. Introduction
2. Monitoring Task
3. Category
4. Filter
5. Application Re...
6. SQL Statement ...
7. Transaction Re...
8. Summary

Specify a filter for applications to be monitored

Select the applications that you want to monitor. You can accept the default setting and monitor all applications, or you can specify a filter by selecting applications that meet specific conditions, which enables you to focus on a particular group of applications.

All applications

Applications that meet the following conditions

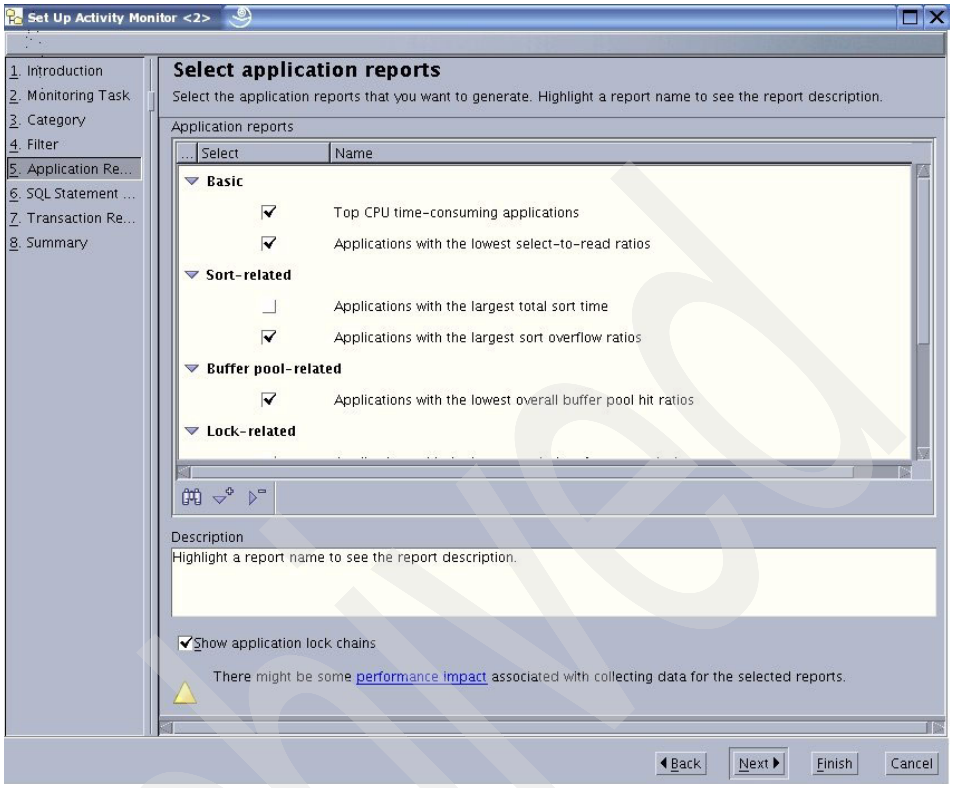
Attribute	Comparison	Values
Authorization ID	=	
Application Name	=	
Application Handle (agent ID)	=	

Clear

Meet all conditions   Meet any Conditions

Back | Next | Finish | Cancel

Figure 5-85 Activity monitor set up - Filter for user-defined task


Set Up Activity Monitor <2>

1. Introduction
2. Monitoring Task
3. Category
4. Filter
5. Application Re...
6. SQL Statement ...
7. Transaction Re...
8. Summary

Select application reports

Select the application reports that you want to generate. Highlight a report name to see the report description.

Application reports

Select	Name
Basic	
✓	Top CPU time-consuming applications
✓	Applications with the lowest select-to-read ratios
Sort-related	
☐	Applications with the largest total sort time
✓	Applications with the largest sort overflow ratios
Buffer pool-related	
✓	Applications with the lowest overall buffer pool hit ratios
Lock-related	

Description

Highlight a report name to see the report description.

✓ Show application lock chains

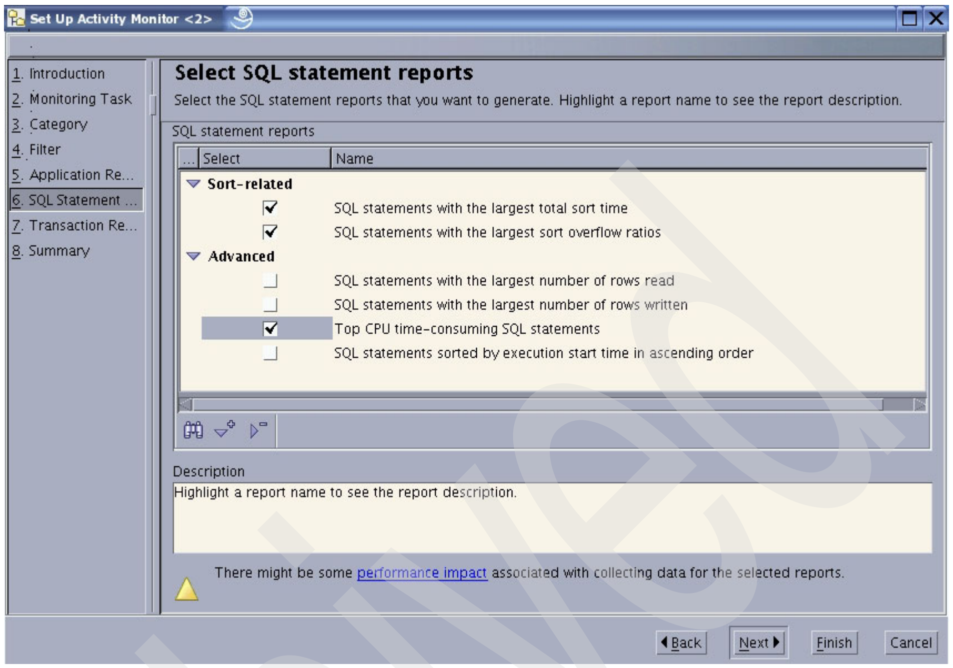
There might be some performance impact associated with collecting data for the selected reports.

Figure 5-86   Activity monitor set up - Application report for user-defined task

At the Application Reports page you can select the items you want to collect data for.

Attention: Most of the reports will have a performance impact associated with the collection of the data (not only the ones for the applications).

When you finish your selection click **Next** to get to the SQL Statement page shown in Figure 5-87 on page 290. There you can select from a few reports for SQLs. When you have selected the desired reports continue by clicking the **Next** button.


Set Up Activity Monitor <2>

1. Introduction
2. Monitoring Task
3. Category
4. Filter
5. Application Re...
6. SQL Statement ...
7. Transaction Re...
8. Summary

Select SQL statement reports

Select the SQL statement reports that you want to generate. Highlight a report name to see the report description.

SQL statement reports

Select	Name
Sort-related	
✓	SQL statements with the largest total sort time
✓	SQL statements with the largest sort overflow ratios
Advanced	
☐	SQL statements with the largest number of rows read
☐	SQL statements with the largest number of rows written
✓	Top CPU time-consuming SQL statements
☐	SQL statements sorted by execution start time in ascending order

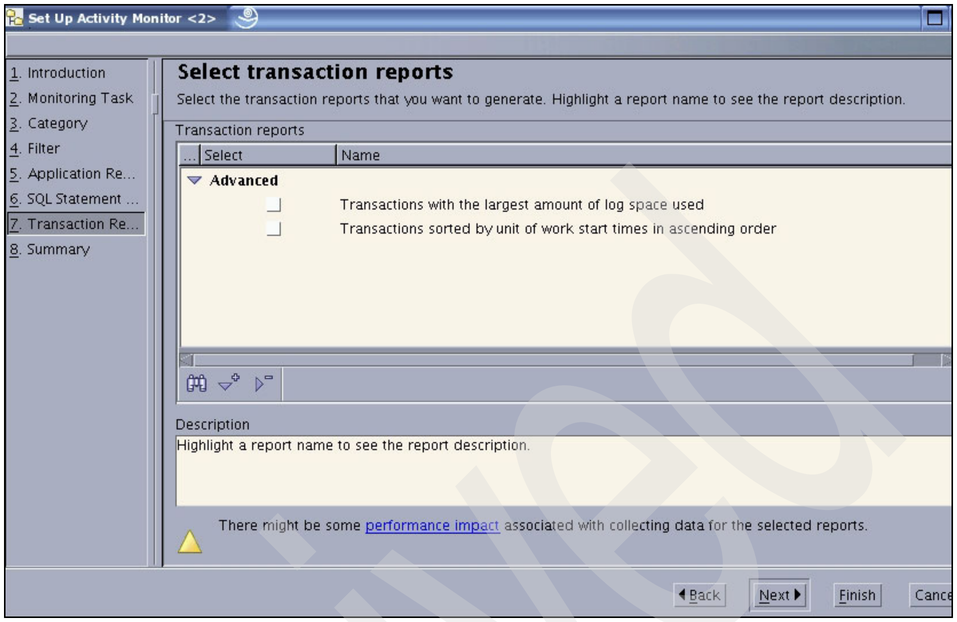
Description

Highlight a report name to see the report description.

There might be some performance impact associated with collecting data for the selected reports.

Figure 5-87 Activity monitor set up - SQL report for user-defined task

The Transaction Reports page also makes you some reports available, as shown in Figure 5-88 on page 291. Select the reports you need and click **Next**.


Set Up Activity Monitor <2>

1. Introduction
2. Monitoring Task
3. Category
4. Filter
5. Application Re...
6. SQL Statement ...
7. Transaction Re...
8. Summary

Select transaction reports

Select the transaction reports that you want to generate. Highlight a report name to see the report description.

Transaction reports

Select	Name
Advanced	
☐	Transactions with the largest amount of log space used
☐	Transactions sorted by unit of work start times in ascending order

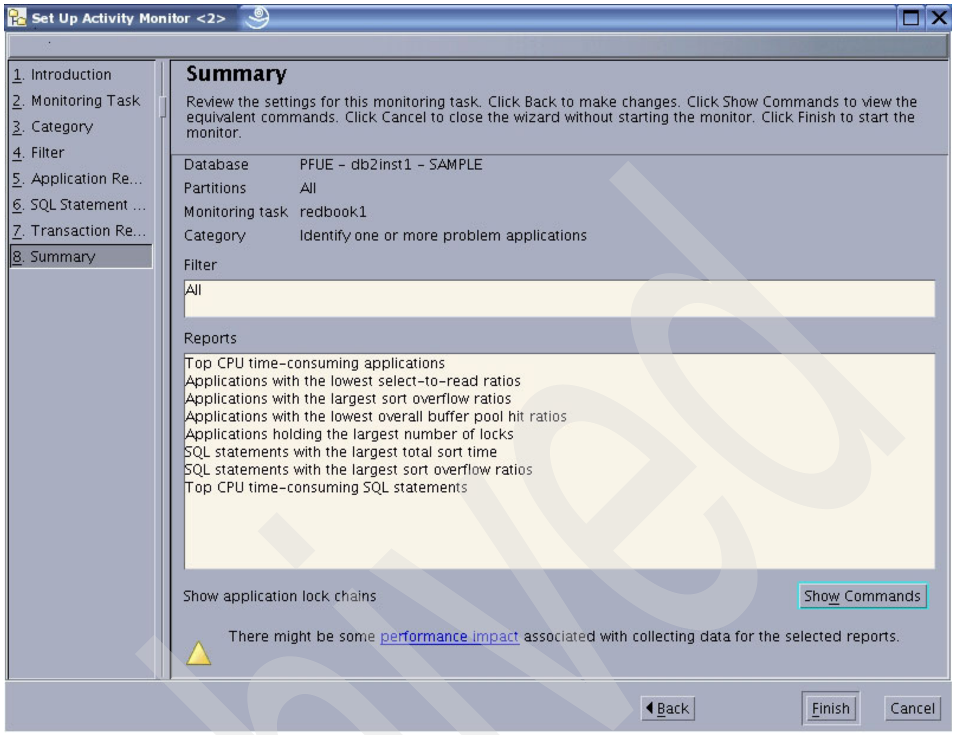
Description

Highlight a report name to see the report description.

There might be some performance impact associated with collecting data for the selected reports.

Figure 5-88 Activity monitor set up - Transaction report for user-defined task

The Summary page, shown in Figure 5-89 on page 292, lists all selected reports and shows the filter we choose. Once you have reviewed all information, click **Finish** to complete the setup and go to the Activity Monitor, shown in Figure 5-90 on page 293.



Summary

Review the settings for this monitoring task. Click Back to make changes. Click Show Commands to view the equivalent commands. Click Cancel to close the wizard without starting the monitor. Click Finish to start the monitor.

Database	PFUE - db2inst1 - SAMPLE
Partitions	All
Monitoring task	redbook1
Category	Identify one or more problem applications

Filter

All

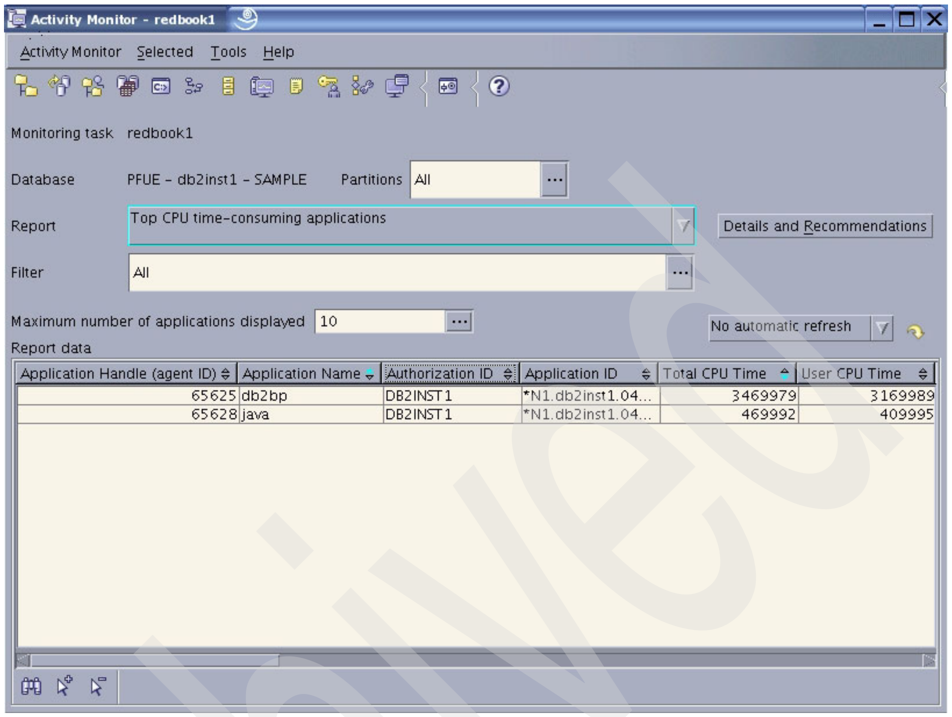
Reports

Top CPU time-consuming applications
Applications with the lowest select-to-read ratios
Applications with the largest sort overflow ratios
Applications with the lowest overall buffer pool hit ratios
Applications holding the largest number of locks
SQL statements with the largest total sort time
SQL statements with the largest sort overflow ratios
Top CPU time-consuming SQL statements

Show application lock chains

There might be some performance impact associated with collecting data for the selected reports.

Figure 5-89 Activity monitor set up - Summary for user-defined task



Monitoring task redbook1

Database PFUE - db2inst1 - SAMPLE Partitions All

Report Top CPU time-consuming applications

Filter All

Maximum number of applications displayed 10

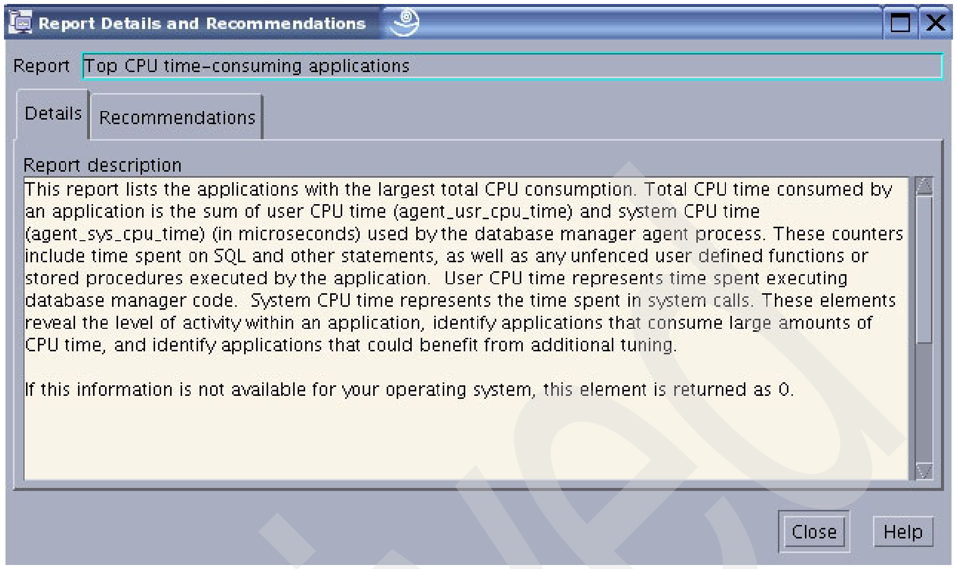
Report data

Application Handle (agent ID)	Application Name	Authorization ID	Application ID	Total CPU Time	User CPU Time
65625	db2bp	DB2INST1	*N1.db2inst1.04...	3469979	3169989
65628	java	DB2INST1	*N1.db2inst1.04...	469992	409995

Figure 5-90 Activity monitor - Top CPU time

At the activity monitor you must choose one of the reports with the pull-down menu. You can choose to automatically refresh your report data with the pull-down menu right next to the No automatic refresh box or click the arrow right beside the No automatic refresh box to get a one-time refresh of your report data.

By clicking the **Details and Recommendations** button you can get detailed information about your report, as shown in Figure 5-91 on page 294. When you go to the Recommendations tab, shown in Figure 5-92 on page 294, you get a recommendation regarding your report. In our case the recommendation is to optimize the queries that use the most CPU time with the design advisor. Click **Close** to get back to the Activity monitor.



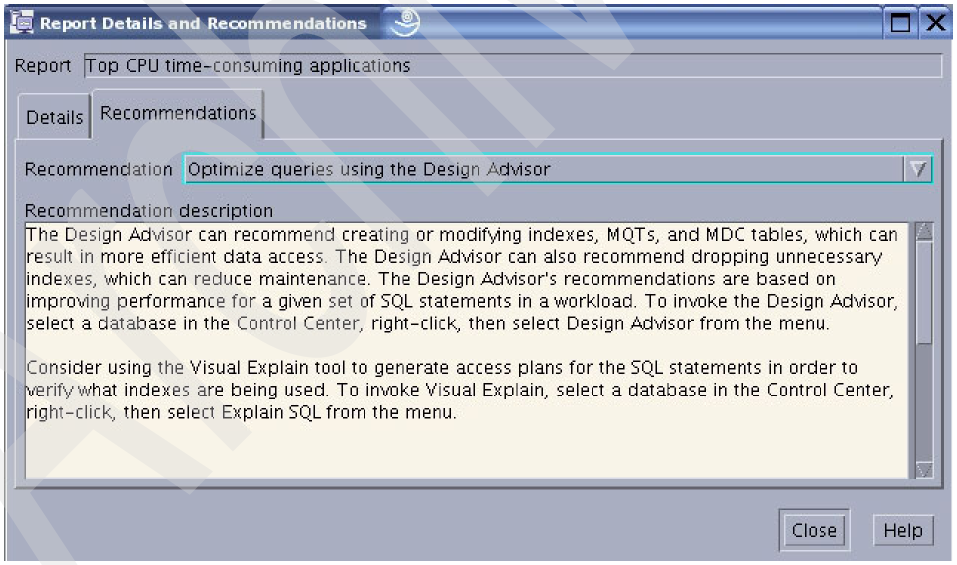
Report: Top CPU time-consuming applications

Report description

This report lists the applications with the largest total CPU consumption. Total CPU time consumed by an application is the sum of user CPU time (agent_usr_cpu_time) and system CPU time (agent_sys_cpu_time) (in microseconds) used by the database manager agent process. These counters include time spent on SQL and other statements, as well as any unfenced user defined functions or stored procedures executed by the application. User CPU time represents time spent executing database manager code. System CPU time represents the time spent in system calls. These elements reveal the level of activity within an application, identify applications that consume large amounts of CPU time, and identify applications that could benefit from additional tuning.

If this information is not available for your operating system, this element is returned as 0.

Figure 5-91 Activity monitor - Report details



Report: Top CPU time-consuming applications

Recommendation: Optimize queries using the Design Advisor

Recommendation description

The Design Advisor can recommend creating or modifying indexes, MQTs, and MDC tables, which can result in more efficient data access. The Design Advisor can also recommend dropping unnecessary indexes, which can reduce maintenance. The Design Advisor's recommendations are based on improving performance for a given set of SQL statements in a workload. To invoke the Design Advisor, select a database in the Control Center, right-click, then select Design Advisor from the menu.

Consider using the Visual Explain tool to generate access plans for the SQL statements in order to verify what indexes are being used. To invoke Visual Explain, select a database in the Control Center, right-click, then select Explain SQL from the menu.

Figure 5-92 Activity monitor - Report recommendations

Health monitor

We talked a lot about the configuration of the Health monitor in 5.1.1, “Health Center enhancements” on page 217, especially in the Health indicator configuration launchpad part. Here we want only to show how the Health Center works as a whole.

The Health Monitor will monitor the database and will notify the people in the notification list about the warnings and alert according to the monitoring setting you defined. Once the warning or alert condition arises, you can start the Recommendation advisor from the Health Center, as shown in "Recommendation advisor" on page 230, to resolve the problem.

In summary we could say that the Health Center is a monitoring and notification GUI tool that also has the ability to take immediate, predefined actions on some exceeded fixed limits.

Memory Visualizer

To start the Memory Visualizer you can type `db2memvis` in a command window or right-click an instance in the Control Center and select **View memory usage**, as shown in Figure 5-93.

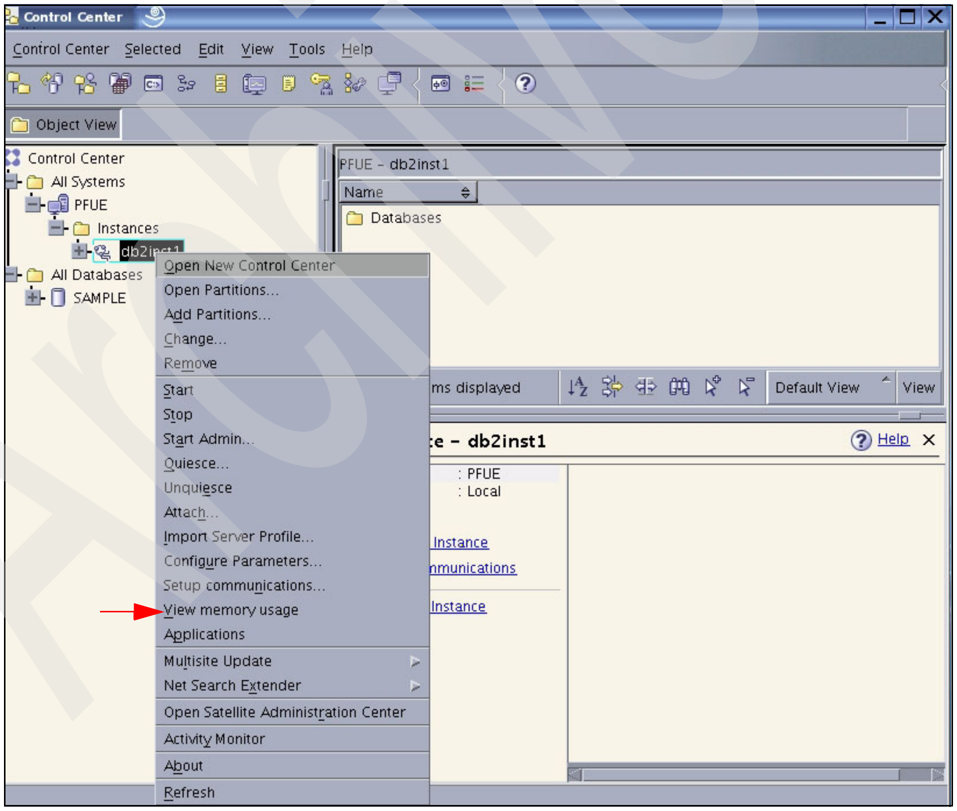
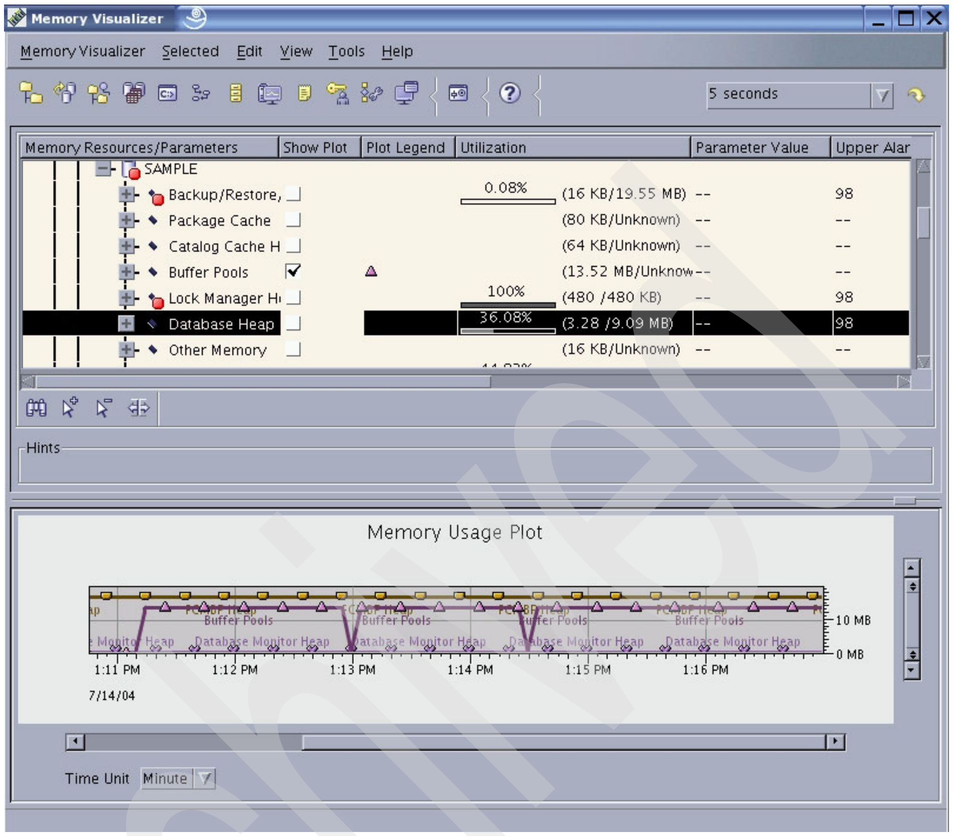


Figure 5-93 Memory Visualizer - Startup



Memory Resources/Parameters	Show Plot	Plot Legend	Utilization		Parameter Value	Upper Alar
SAMPLE						
Backup/Restore,	☐		0.08%	(16 KB/19.55 MB)	--	98
Package Cache	☐			(80 KB/Unknown)	--	--
Catalog Cache H	☐			(64 KB/Unknown)	--	--
Buffer Pools	✓	△		(13.52 MB/Unknow	--	--
Lock Manager H	☐		100%	(480 /480 KB)	--	98
Database Heap	☐		36.08%	(3.28 /9.09 MB)	--	98
Other Memory	☐			(16 KB/Unknown)	--	--

Figure 5-94 Memory Visualizer

The Memory Visualizer (Figure 5-94) shows you (for each partition) a wide variety of memory usage parameters, such as for the instance and databases. You can have the values plotted to see how the trend is for a parameter.

Memory Visualizer is a graphical representation of db2mtrk data. db2mtrk data is reported in the snapshot monitor stream in the Memory Heap section. In the Memory Visualizer you also see to what extend your heaps are used, which gives you the ability to detect the big allocated heaps.

5.2.2 Linux tools

All linux distributions come with a wide range of monitoring tools. We present some of the most common tools delivered with Linux, and also some free, available tools.

Vmstat

Vmstat reports virtual memory statistics, which include, in the given order, the following information, shown in Example 5-7:

- r: Number of processes waiting for run time
- b: Number of processes in uninterruptible sleep
- swpd: Amount of virtual memory used
- free: Amount of idle memory
- buff: Amount of memory used as buffers
- cache: Amount of memory used as cache
- si: Memory swapped in from disk
- so: Memory swapped to disk
- bi: Blocks received
- bo: Blocks sent
- in: Interrupts (#)
- cs: Context switches (#)
- us: User time (%)
- sy: System time (%)
- id: Idle time (%)
- wa: I/O-waits (%)

You can run **vmstat** with a counter and a delay, for example, if you want vmstats to collect the statistics four times every two seconds you use the following command:

```
vmstat 2 4
```

The output of that command is shown in Example 5-7.

Example 5-7 vmstat

```
[root@DB2RB01 root]# vmstat 2 4
procs                      memory      swap          io     system         cpu
 r  b   swpd   free   buff  cache   si   so    bi    bo   in    cs us sy id wa
 3  0      0 1098400 199052 2393244    0    0     4    36    0    61 39  1 58  2
 3  0      0 1097660 199052 2393244    0    0     0     0  202   677 51  1 48  0
 3  0      0 1096556 199052 2393244    0    0     0    60  136   522 51  3 45  1
 3  0      0 1096556 199052 2393244    0    0     0     0  433  1337 50  1 49  0
```

This command is ideal for monitoring the CPU usage (us, sy, id, wa) and the paging rate, which can be found under the memory in (si) and memory out (so) columns. Other important columns are the amount of allocated virtual storage (swpd) and free virtual storage (free).

This command is useful for determining if something is suspended or just taking a long time. Additional information about vmstat can be found in the man pages of vmstat (**man vmstat**).

iostat

Iostats is used to show the throughput on the devices and partitions, and also reports the CPU usage. The program gives you the possibility to run it multiple times in a specified interval. Iostat is not installed by default (SuSE, Red Hat); you have to install the sysstat package to get the **iostat** command.

The syntax of the command is:

```
iostat [ options... ] [ <interval> [ <count> ] ]
```

An example output of **iostat 5 5** is shown in Example 5-8.

Example 5-8 iostat

```
[root@DB2RB01 root]# iostat 5 5
Linux 2.4.21-15.ELsmp (DB2RB01.torolab.ibm.com)        23/07/04

avg-cpu:  %user   %nice    %sys %iowait   %idle
          39.22    0.12    1.48    1.64   57.55

Device:            tps   Blk_read/s   Blk_wrtn/s   Blk_read   Blk_wrtn
sda               1.80         9.60        74.57    2451176   19033094
sda1              0.00         0.00         0.00        238         70
sda2              1.80         9.60        74.57    2450314   19033024
sda3              0.00         0.00         0.00        176          0
sdb               0.00         0.00         0.00        538        464
sdb1              0.00         0.00         0.00        338        464

avg-cpu:  %user   %nice    %sys %iowait   %idle
          50.70    0.00    0.00    0.40   48.90

Device:            tps   Blk_read/s   Blk_wrtn/s   Blk_read   Blk_wrtn
sda               0.60         0.00        43.29          0        216
sda1              0.00         0.00         0.00          0          0
sda2              0.60         0.00        43.29          0        216
sda3              0.00         0.00         0.00          0          0
sdb               0.00         0.00         0.00          0          0
sdb1              0.00         0.00         0.00          0          0

avg-cpu:  %user   %nice    %sys %iowait   %idle
          52.00    0.00    1.10    0.00   46.90

Device:            tps   Blk_read/s   Blk_wrtn/s   Blk_read   Blk_wrtn
sda               0.00         0.00         0.00          0          0
sda1              0.00         0.00         0.00          0          0
sda2              0.00         0.00         0.00          0          0
sda3              0.00         0.00         0.00          0          0
sdb               0.00         0.00         0.00          0          0
sdb1              0.00         0.00         0.00          0          0
```

This command is useful for monitoring I/O activities. You can use the read and write rate to estimate the amount of time required for certain SQL operations (if they are the only activity on the system).

This command is also useful for determining if something is suspended or just taking a long time.

The reports produced can also be used to identify disks that are heavily used. If you also see disks that are almost idle over a long period, you may want to reconsider your storage partitioning.

Top

Top lists the running tasks and also gives some information about the system. The data is refreshed in a configurable time interval. Top also gives you limited process manipulation capabilities like renice and kill. Additional information about options and the configuration of top can be found in the man pages (**man top**).

An example output of top is shown in Example 5-9.

Example 5-9 top

```
[root@DB2RB01 root]# top
 17:11:16  up 2 days, 23:01,  4 users,  load average: 1.24, 1.12, 1.45
222 processes: 218 sleeping, 4 running, 0 zombie, 0 stopped
CPU states:  cpu    user    nice  system    irq  softirq  iowait    idle
           total   50.1%    0.0%    0.7%   0.0%     0.0%    0.8%   48.0%
           cpu00   21.3%    0.0%    1.3%   0.0%     0.0%    1.7%   75.4%
           cpu01   79.0%    0.0%    0.1%   0.1%     0.0%    0.0%   20.5%
Mem:  4123072k av, 3023028k used, 1100044k free,       0k shrd,  199084k buff
                    634544k actv, 1533528k in_d,     264k in_c
Swap: 2040244k av,       0k used, 2040244k free                 2393148k cached

  PID USER     PRI  NI  SIZE  RSS SHARE STAT %CPU %MEM   TIME CPU COMMAND
29047 db2fenc1  25   0  9096 9088  5192 R    49.6  0.2 118:39   1 db2fmp
 2943 root      15   0  1380 1380   896 R     0.3  0.0   0:00   0 top
    1 root      15   0   520  520   452 S     0.0  0.0   0:06   0 init
    2 root      RT   0     0    0     0 SW    0.0  0.0   0:00   0 migration/0
    3 root      RT   0     0    0     0 SW    0.0  0.0   0:00   1 migration/1
    4 root      15   0     0    0     0 SW    0.0  0.0   0:00   1 keventd
    5 root      34  19     0    0     0 SWN   0.0  0.0   0:00   0 ksoftirqd/0
    6 root      34  19     0    0     0 SWN   0.0  0.0   0:00   1 ksoftirqd/1
    9 root      15   0     0    0     0 SW    0.0  0.0   0:00   1 bdflush
    7 root      15   0     0    0     0 SW    0.0  0.0   0:02   0 kswapd
    8 root      15   0     0    0     0 SW    0.0  0.0   0:04   0 kscand
   10 root      15   0     0    0     0 SW    0.0  0.0   0:05   0 kupdated
   11 root      25   0     0    0     0 SW    0.0  0.0   0:00   0 mdrecoveryd
   20 root      15   0     0    0     0 SW    0.0  0.0   0:16   0 kjournald
```

ps

The **ps** command is a UNIX-based system command that returns process status information about active processes. Use it to look for discrepancies between DB2 processes that are running and DB2 processes that you expect to be there. Flags control the types of information displayed for each active process, and may be applied simultaneously to give a cumulative effect.

To get all processes running under instance owner db2inst2 you use the **ps -fu db2inst2** command, as shown in Example 5-10.

Example 5-10 ps -fu db2inst2

```
db2inst2@db2rb09:~> ps -fu db2inst2
UID        PID  PPID  C STIME TTY          TIME CMD
db2inst2 24522 22731  0 Jul13 pts/5    00:00:00 su - db2inst2
db2inst2 24523 24522  0 Jul13 pts/5    00:00:00 -bash
db2inst2 26403 25157  0 Jul13 pts/5    00:00:00 su - db2inst2
db2inst2 26404 26403  0 Jul13 pts/5    00:00:00 -bash
db2inst2 14523 14522  0 Jul14 ?        00:00:00 db2sysc
db2inst2 14528 14523  0 Jul14 ?        00:00:00 db2fcmdm
db2inst2 14529 14523  0 Jul14 ?        00:00:00 db2pdbc
db2inst2 14530 14523  0 Jul14 ?        00:00:00 db2ipccm
db2inst2 14531 14523  0 Jul14 ?        00:00:00 db2tcpcm
db2inst2 14532 14523  0 Jul14 ?        00:00:00 db2tcpcm
db2inst2 14534 14523  0 Jul14 ?        00:00:00 db2resync
db2inst2 14535 14527  0 Jul14 ?        00:00:00 db2srvlst
db2inst2 14537 14523  0 Jul14 ?        00:00:00 db2panic (idle)
db2inst2 14538 14523  0 Jul14 ?        00:00:00 db2hmon
,0,0,0,1,0,1,1,1e014,2,0
db2inst2 14550 14532  0 Jul14 ?        00:00:03 db2agent (idle)
db2inst2 14705 14527  0 Jul14 ?        00:00:01 db2agent (idle)
db2inst2  5666  5639  0 11:59 pts/2    00:00:00 su - db2inst2
db2inst2  5667  5666  0 11:59 pts/2    00:00:00 -bash
db2inst2 10645 14530  0 13:07 ?        00:00:00 db2agent (idle)
db2inst2 17652  5667  0 15:14 pts/2    00:00:00 ps -fu db2inst2
```

You can access information on the command syntax and flag options by using the **man ps** command on a system command prompt.

ping/traceroute

The **ping** command is used to test the communication between two computers. An example output is shown in Example 5-11.

Example 5-11 ping

```
db2inst2@db2rb09:~> ping -c 3 db2rb10
PING db2rb10.torolab.ibm.com (9.26.162.63) 56(84) bytes of data.
```

```
64 bytes from db2rb10.torolab.ibm.com (9.26.162.63): icmp_seq=1 ttl=64
time=0.236 ms
64 bytes from db2rb10.torolab.ibm.com (9.26.162.63): icmp_seq=2 ttl=64
time=0.205 ms
64 bytes from db2rb10.torolab.ibm.com (9.26.162.63): icmp_seq=3 ttl=64
time=0.103 ms

--- db2rb10.torolab.ibm.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.103/0.181/0.236/0.057 ms
```

Additional information about the **ping** command can be found in the man pages (**man ping**).

The **traceroute** command is used to show what path a package is taking from the sender host to the receiver host (see Example 5-12).

Example 5-12 traceroute

```
db2rb09:~ # traceroute proxy.ch.ibm.com
traceroute to proxy.ch.ibm.com (9.139.253.30), 30 hops max, 40 byte packets
 1  rsb1162.torolab.ibm.com (9.26.162.2)  12.702 ms   10.644 ms   8.748 ms
 2  tlb-bso-1a.torolab.ibm.com (9.26.67.21)  7.191 ms   5.339 ms   3.405 ms
 3  tlb-ld-a-ge0-11-v45.torolab.ibm.com (9.26.114.146)  1.959 ms   17.422 ms
11.896 ms
 4  9.26.0.106  11.888 ms   10.160 ms   19.262 ms
 5  9.26.0.124  17.982 ms   16.011 ms   14.554 ms
 6  ibm-ce-lab1.torolab.ibm.com (9.26.0.6)  11.988 ms   14.450 ms   13.364 ms
 7  * * *
 8  9.29.254.194  46.414 ms   54.528 ms   52.544 ms
 9  9.64.12.133  51.179 ms   49.193 ms   47.195 ms
10  USCT009C12008R01-USCT009D6509R04.wan.ibm.com (9.64.12.5)  55.948 ms
53.950 ms   51.967 ms
11  USNY023C12008R01-SRP1-0.wan.ibm.com (9.64.1.4)  48.241 ms   46.254 ms
44.259 ms
12  USNY023D6509R03-USNY023C12008R01.wan.ibm.com (9.64.6.2)  52.634 ms   50.659
ms   48.666 ms
13  usny023d12008r05-usny023d6509r03.wan.ibm.com (9.64.6.26)  48.016 ms
45.762 ms   52.086 ms
14  usn54ny001er1-usny023d12008r05.wan.ibm.com (9.64.11.70)  50.091 ms   48.104
ms   46.117 ms
15  IBP9CHEZH03A7300R01-ATM3-0-100.wan.ibm.com (9.64.136.197)  153.704 ms
151.768 ms   159.135 ms
16  9.64.144.226  159.659 ms   157.672 ms   155.692 ms
17  ns.bi.ch.ibm.com (9.139.253.30)  153.645 ms   151.683 ms   160.420 ms
```

Additional information about the **traceroute** command can be found in the man pages (**man traceroute**).

Netstat

This command lets you know the network traffic on each node, and the number of error packets encountered. It is useful for isolating network problems. In Example 5-13 we show the output of a **`netstat -s`**. For additional information consult the man pages (**`man netstat`**).

Example 5-13 netstat -s

```
db2rb09:~ # netstat -s
Ip:
    60547165 total packets received
    0 forwarded
    0 incoming packets discarded
    60543528 incoming packets delivered
    72945962 requests sent out
    2312238 fragments received ok
Icmp:
    292 ICMP messages received
    19 input ICMP message failed.
    ICMP input histogram:
        destination unreachable: 37
        timeout in transit: 50
        echo requests: 199
        echo replies: 6
    1054 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
        destination unreachable: 855
        echo replies: 199
Tcp:
    20840 active connections openings
    3626 passive connection openings
    5 failed connection attempts
    369 connection resets received
    15 connections established
    57720406 segments received
    70613066 segments send out
    1536 segments retransmited
    1 bad segments received.
    17289 resets sent
Udp:
    2494734 packets received
    846 packets to unknown port received.
    0 packet receive errors
    2316570 packets sent
TcpExt:
    8 invalid SYN cookies received
    17 packets pruned from receive queue because of socket buffer overrun
    ArpFilter: 0
```

```
3433 TCP sockets finished time wait in fast timer
143365 delayed acks sent
33 delayed acks further delayed because of locked socket
Quick ack mode was activated 367 times
22606 packets directly queued to recvmsg prequeue.
50453348 packets directly received from backlog
8680126 packets directly received from prequeue
31100456 packets header predicted
44716 packets header predicted and directly queued to user
TCPPureAcks: 781027
TCPHPAcks: 42653677
TCPRenoRecovery: 0
TCPSackRecovery: 276
TCPSACKReneging: 0
TCPFACKReorder: 3
TCPSACKReorder: 11
TCPRenoReorder: 0
TCPTSReorder: 0
TCPFullUndo: 0
TCPPartialUndo: 0
TCPDSACKUndo: 0
TCPLossUndo: 39
TCPLoss: 423
TCPLostRetransmit: 0
TCPRenoFailures: 0
TCPSackFailures: 47
TCPLossFailures: 0
TCPFastRetrans: 512
TCPForwardRetrans: 136
TCPSlowStartRetrans: 517
TCPTimeouts: 192
TCPRenoRecoveryFail: 0
TCPSackRecoveryFail: 65
TCPSchedulerFailed: 0
TCPRcvCollapsed: 1666
TCPDSACKOldSent: 410
TCPDSACKOfoSent: 0
TCPDSACKRecv: 38
TCPDSACKOfoRecv: 0
TCPAbortOnSyn: 0
TCPAbortOnData: 3
TCPAbortOnClose: 326
TCPAbortOnMemory: 0
TCPAbortOnTimeout: 18
TCPAbortOnLinger: 0
TCPAbortFailed: 0
TCPMemoryPressures: 0
```

Lsof

This command lists the open files. You can, for example, find out what processes are accessing a certain directory with **`lsof <directory>`**, as shown in Example 5-14.

Example 5-14 lsof

```
db2rb09:/db2home # lsof /db2home/db2inst2
COMMAND   PID     USER   FD   TYPE DEVICE SIZE NODE NAME
su      24522 db2inst2  cwd    DIR    9,1  872  379 /db2home/db2inst2
bash    24523 db2inst2  cwd    DIR    9,1  872  379 /db2home/db2inst2
su      26403 db2inst2  cwd    DIR    9,1  872  379 /db2home/db2inst2
```

If you have a file sharing violation, this is the tool to find out who is accessing that specific file. Additional information about lsof can be found in the man pages (**`man lsof`**).

Free/watch

Free is a tool to show the free and used physical, as well as swap, memory in the system. Furthermore, it reports information about the used kernel buffers.

The watch tool can execute a program periodically. The output of a **`watch free`** is shown in Example 5-15, and a **`watch --differences free`** is shown in Example 5-16. The --differences option highlights changes in the output.

Example 5-15 watch free

```
[root@DB2RB01 root]# watch free
Every 2s: free                                          Fri Jul 23 16:37:51
2004

             total       used       free     shared    buffers     cached
Mem:       4123072    3019888    1103184          0     198980    2390776
-/+ buffers/cache:     430132    3692940
Swap:      2040244          0    2040244
```

Example 5-16 watch --differences free

```
[root@DB2RB01 root]# watch --differences free
Every 2s: free                                               Fri Jul 23
16:49:25 2004

             total       used       free     shared    buffers     cached
Mem:       4123072    3026552    1096520          0     199016    2393020
-/+ buffers/cache:     434516    3688556
Swap:      2040244          0    2040244
```

Additional information is available in the man pages (**`man free`**, **`man watch`**).

Nmon

Nmon is a tool that brings a big variety of information on the screen. Instead of using different tools to collect the data you have them all in one tool. The tool has the capability to write the information to a csv-file, which can then be used for performance analysis.

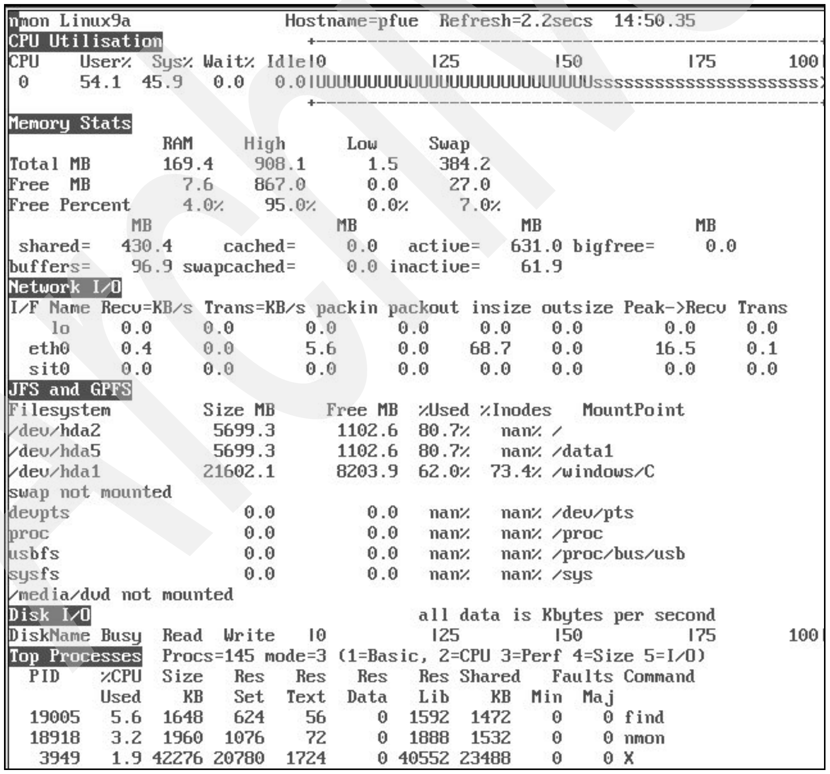
The nmon tool is *not officially supported* and no warranty and no help by IBM are given. More information about the tools is given in the following Web article:

`http://www.ibm.com/developerworks/eserver/articles/analyze_aix/`

You can download the tool from this Web site:

`http://www.ibm.com/developerworks/eserver/articles/analyze_aix/agree_down.html`

A screenshot with some of the available information is shown in Figure 5-95.



```
nmon Linux9a                Hostname=pfue  Refresh=2.2secs  14:50.35
CPU Utilisation              +-----------------------------------------------------+
CPU    User%  Sys% Wait% Idle|0          |25         |50          |75        100|
 0     54.1  45.9   0.0   0.0|UUUUUUUUUUUUUUUUUUUUUUUUUUUsssssssssssssssssssssss>
                             +-----------------------------------------------------+
Memory Stats
               RAM     High     Low     Swap
Total MB       169.4    908.1     1.5    384.2
Free  MB         7.6    867.0     0.0     27.0
Free Percent     4.0%    95.0%    0.0%     7.0%
            MB                  MB                 MB                 MB
 shared=   430.4     cached=     0.0   active=   631.0 bigfree=     0.0
buffers=    96.9 swapcached=     0.0 inactive=    61.9
Network I/O
I/F Name Recv=KB/s Trans=KB/s packin packout insize outsize Peak->Recv Trans
    lo     0.0      0.0        0.0      0.0      0.0    0.0        0.0     0.0
  eth0     0.4      0.0        5.6      0.0     68.7    0.0       16.5     0.1
  sit0     0.0      0.0        0.0      0.0      0.0    0.0        0.0     0.0
JFS and GPFS
Filesystem         Size MB      Free MB  %Used %Inodes   MountPoint
/dev/hda2           5699.3      1102.6  80.7%    nan% /
/dev/hda5           5699.3      1102.6  80.7%    nan% /data1
/dev/hda1          21602.1      8203.9  62.0%  73.4% /windows/C
swap not mounted
devpts                 0.0         0.0    nan%    nan% /dev/pts
proc                   0.0         0.0    nan%    nan% /proc
usbfs                  0.0         0.0    nan%    nan% /proc/bus/usb
sysfs                  0.0         0.0    nan%    nan% /sys
/media/dvd not mounted
Disk I/O                                 all data is Kbytes per second
DiskName Busy  Read  Write   |0          |25         |50          |75        100|
Top Processes  Procs=145 mode=3 (1=Basic, 2=CPU 3=Perf 4=Size 5=I/O)
  PID     %CPU  Size    Res    Res    Res    Res Shared   Faults Command
          Used    KB    Set   Text   Data    Lib     KB  Min  Maj
  19005    5.6  1648    624     56      0   1592   1472    0     0 find
  18918    3.2  1960   1076     72      0   1888   1532    0     0 nmon
   3949    1.9 42276  20780   1724      0  40552  23488    0     0 X
```

Figure 5-95 nmon

Sar

With sar you can collect and report system information. A good initial point is the following command:

```
sar -o data.file interval count >/dev/null 2>&1 &
```

In our specific case we used the following:

```
sar -o /tmp/db2rbrun1 5  24  >/dev/null  2>&1 &
```

The data is stored in a binary format, so to view the information use the **`sar -f data.file`** (or in our case the **`sar -f /tmp/db2rbrun1`** command). The output is shown in Example 5-17.

Example 5-17 sar

```
db2rb09:~ # sar -f /tmp/db2rbrun1
Linux 2.6.5-7.79-smp (db2rb09)  07/15/04

17:13:51         CPU     %user     %nice   %system   %iowait     %idle
17:13:56         all      0.10      0.00      0.30      0.60     99.00
17:14:01         all      0.70      0.00      0.70      2.00     96.59
17:14:06         all      0.10      0.00      0.20      0.80     98.90
17:14:11         all      1.20      0.00      0.70      2.00     96.10
17:14:16         all      0.00      0.00      0.20      0.90     98.90
17:14:21         all      0.80      0.00      1.00      2.30     95.91
17:14:26         all      0.00      0.00      0.00      0.80     99.20
17:14:31         all      0.70      0.00      0.70      1.90     96.70
17:14:36         all      0.00      0.00      0.10      1.10     98.80
17:14:41         all      0.70      0.00      0.70      0.80     97.80
17:14:46         all      0.00      0.00      0.10      1.90     98.00
17:14:51         all      0.70      0.00      0.70      0.50     98.10
17:14:56         all      0.00      0.00      0.10      1.90     98.00
17:15:01         all      0.80      0.00      1.10      0.80     97.30
17:15:06         all      0.10      0.00      0.20      0.80     98.90
17:15:11         all      0.70      0.00      0.70      0.80     97.80
17:15:16         all      0.00      0.00      0.20      1.90     97.90
17:15:21         all      2.30      0.00      1.90      6.59     89.21
17:15:26         all      0.00      0.00      0.10      1.00     98.90
17:15:31         all      0.70      0.00      0.80      1.90     96.60
17:15:36         all      0.00      0.00      0.10      0.80     99.10
17:15:41         all      0.80      0.00      0.70      2.30     96.20
17:15:46         all      0.00      0.00      0.10      0.90     99.00
17:15:51         all      0.80      0.00      0.60      1.80     96.80
Average:         all      0.47      0.00      0.50      1.55     97.49
```

For additional information about sar have a look at the man pages (**`man sar`**).

dd

The **dd** command copies a file (from standard input to standard output, by default) with a changeable I/O block size, while optionally performing conversions on it. The command syntax looks like:

```
time dd options ...
```

Options:

bs=BYTES	Force ibs=BYTES and obs=BYTES.
cbs=BYTES	Convert BYTES bytes at a time.
conv=KEYWORDS	Convert the file as per the comma-separated keyword list.
count=BLOCKS	Copy only BLOCKS input blocks.
ibs=BYTES	Read BYTES bytes at a time.
if=FILE	Read from FILE instead of stdin.
obs=BYTES	Write BYTES bytes at a time.
of=FILE	Write to FILE instead of stdout.
seek=BLOCKS	Skip BLOCKS obs-sized blocks at start of output.
skip=BLOCKS	Skip BLOCKS ibs-sized blocks at start of input.
--help	Display this help and exit.
--version	Output version information and exit.

You can use dd combined with time to measure the read/write rate for your raw disk or your file system. Here are several examples:

- File system write rate

  Suppose we have a file system called /db2fs1p1; we use the following command to measure the write rate:

  ```
  # time dd if=/dev/zero of=/db2fs1p1/test.dat bs=64k count=1024
  1024+0 records in
  1024+0 records out

  real    0m0.236s
  user    0m0.000s
  sys     0m0.230s
  ```

  We focused on real time, so the data write rate is 64 MB/0.236=271 MB/S.

- File system read rate

  We can use the following command to test the read rate:

  ```
  # time dd if=/db2fs1p1/test.dat of=/dev/null bs=64K count=1024
  ```

```
1024+0 records in
1024+0 records out

real    0m0.084s
user    0m0.000s
sys     0m0.080s
```

So the data read rate is 64 M/0.084=761 MB/s

For a raw device just replace the file to the raw device path; for example:

```
#time dd if=/dev/hda1 of=/dev/null bs=64k count=1024
```

The larger the file size, the more reliable the result. Normally, it is faster when you use /dev/zero or /dev/null as the input/output device.

6

High availability

DB2 Integrated Cluster Environment is a highly available solution. In this chapter we described components and possible configurations of a high-availability solution. We also give a detailed description of implementation of failover protection in a DB2 Integrated Cluster Environment.

This chapter contains the following:

- Introduction to high availability
- TSA implementation

6.1 Introduction

Many times the definition of high availability has been reduced to failover protection. This chapter is no different. We focus on describing high availability of a DB2 Integrated Cluster Environment from a failover protection point of view. But we would like to at least mention that a highly available solution is more than the ability to fail over to a backup server in case of a hardware or software failure.

A highly available solution has hardware as well as software components. To prevent a hardware failure, it should be considered to have redundant resources in servers, such as network adapters, storage adapters, and power adapters. Various levels of RAID protection should be applied to storage systems, to ensure their availability.

DB2 software itself has many features that increase its availability:

- Online utilities: Table Load, Table Reorg, Index Reorg—enable access to table while respective utilities are being executed.
- Online system reconfiguration: More than 50 database and instance-level parameters can be updated online.
- Throttle utilities: Resource-intensive utilities, such as BACKUP, REORG, REBALANCE could be throttled, so that running DB2 applications have only a small impact on performance.
- Dual Logging: Enables DB2 log files to be written to two locations, thus increasing availability, in case one of the logging paths becomes unavailable.

There are many more features and functions within DB2 that make it highly available.

But as suggested earlier, let us now cover the failover characteristic of a highly available DB2 Integrated Cluster Environment.

6.1.1 What is high availability

High availability (HA) is the term that is used to describe systems that run and are available to customers more or less all the time. For this to occur:

- Transactions must be processed efficiently, without appreciable performance degradations (or even loss of availability) during peak operating periods. In a partitioned database environment DB2 can take advantage of both intrapartition and interpartition parallelism to process transactions efficiently. *Intrapartition parallelism* can be used in an SMP environment to process the various components of a complex SQL statement simultaneously. *Interpartition parallelism* in a partitioned database environment, on the other hand, refers to the simultaneous processing of a query on all participating nodes; each node processes a subset of the rows in the table.

- Systems must be able to recover quickly when hardware or software failures occur, or when disaster strikes. DB2 has an advanced continuous checkpointing system and a parallel recovery capability that allow for extremely fast crash recovery.
- Software that powers the enterprise databases must be continuously running and available for transaction processing. To keep the database manager running, you must ensure that another database manager can take over if it fails. This is called failover. *Failover* capability allows for the automatic transfer of workload from one system to another when there is a hardware failure.

6.1.2 Types of high availability

There are two types of high availability that we want to discuss: *Continuous availability* and *failover availability.*

Continuous availability

Continuous availability requires that the database engine be available for processing SQL transactions without any downtime. Such availability is usually implemented in the mission-critical business applications. For this to happen total redundancy is required, which means that you must have two systems that are fully independent of one another, both in hardware and software.

In this case SQL transactions take place on both systems at the same time. The failure of one system will cause no interruption of transaction processing on its partner. In order to achieve this, the application must be aware of both systems and implement each transaction as a *distributed unit of work* (DUOW) across both systems. A DUOW is a series of SQL statements executed as a single transaction that is coordinated between both systems. The application will submit a transaction to both systems and receive return codes from both systems upon success or failure. If a failure takes place where one database system is no longer able to function, the application can continue to process its workload with the remaining system with no interruption.

Such a solution provides 100 percent availability to the application, but on the other hand requires duplicate hardware, and also the application itself has to be aware of the DUOW transactions.

Failover availability

Failover availability is differentiated from continuous availability by the fact that for some period of time, however small, the database engine is not available for transaction processing. The essential elements for this type of solution are:

- Primary and secondary systems
- Failure detection
- Data source movement

The two systems have copies of the database data, and when a failure is detected, a failover takes place. In the failover process, the data source is moved from the primary to the secondary system.

There are two types of failover availability: *Synchronous* and *asynchronous*.

Asynchronous failover protection can be achieved by keeping a copy of your database on another machine that is perpetually rolling the log files forward. *Log shipping* is the process of copying whole log files to a standby machine, either from an archive device, or through a user exit program running against the primary database. With this approach, the primary database is restored to the standby machine, using either the DB2 restore utility or the split mirror function. You can also use suspended I/O support to quickly initialize the new database. The standby database on the standby machine continuously rolls the log files forward. If the primary database fails, any remaining log files are copied over to the standby machine. After a rollforward to the end of the logs and stop operation, all clients are reconnected to the standby database on the standby machine.

Synchronous failover protection guarantees that the data sources on primary and secondary systems are identical, and complete continuity is maintained after failover. Such failover support is typically provided through platform-specific software such as Tivoli System Automation for Linux. We discuss these software options in more detail later in this chapter.

The two most common synchronous failover strategies on the market are known as *idle standby* and *mutual takeover.*

Idle standby

In this configuration, one system is used to run a DB2 instance, and the second system is “idle”, or in standby mode, ready to take over the instance if there is an operating system or hardware failure involving the first system. Overall system performance is not impacted, because the standby system is idle until needed.

Mutual takeover

In this configuration, each system is designated backup for another system. Overall system performance may be impacted, because the backup system must do extra work following a failover: It must do its own work plus the work that was being done by the failed system.

6.1.3 High availability in clusters

A *cluster* consists of a combination of hardware and software. One of its main design points is to provide high availability—the ability of applications to restart execution on another server when the server they originally ran on failed (or perhaps became overloaded). Cluster disk devices are physically accessible to two or more servers. They would most likely be disk subsystems or arrays. Depending on the cluster server and software that uses its services, the data on disk may be shared among servers (that is, accessible to more than one server at the same time), or it can be owned by only one server at the time, with ownership transferring to another server in case the first server fails. Cluster software not only restarts the application on the backup server, but it is also able to transfer the network identity to the backup server, thus enabling remote application to use the services without the need to reconfigure the addresses. In the case where there are separate applications (for example, a Web server and a database or two separate databases), it provides the opportunity to offload the busy server to the less busy server or to split the workload across both servers as long as each client request can be serviced by a single server.

Figure 6-1 is a generic view of a high-availability cluster.

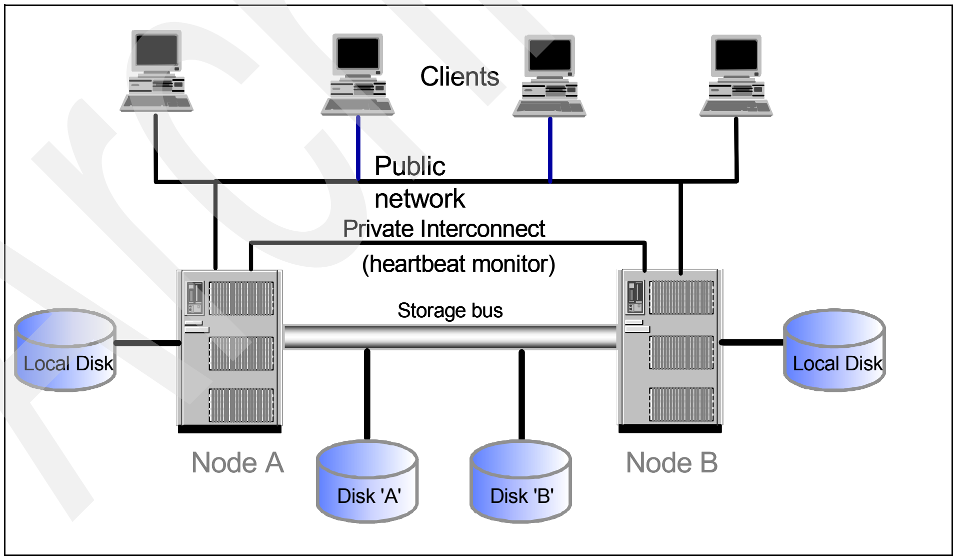


Figure 6-1 Sample high-availability cluster

Let us describe components of the high-availability cluster. Node A acts as a primary server, which is active. Node B is a secondary server, which can either be active or passive. Application data is stored on disks that can be taken over (disks A and B in this diagram). They are usually in RAID arrays for additional level of availability. The application itself is typically stored on local disks. Failover software uses *heartbeat monitoring* between systems to confirm availability. Heartbeat monitoring involves system services that maintain constant communication between all the nodes in the cluster. If a heartbeat is not detected, failover to a secondary system starts.

Clients are connected to the primary server using the public network interface. Now let us discuss what happens when failover occurs in a highly available cluster configuration.

6.1.4 Failover in a highly available cluster

In case there is a failure (hardware or software) on the primary server, the high-availability software moves all the necessary resources to the secondary system. These resources include the disk resources of the physical database, the network resources, and the database server resources.

In a highly available cluster solution, a single copy of the physical database is stored on a shared storage system. In the DB2 environment only one system can “own” the storage array at a time. When a failure is detected, ownership of the storage is moved from the primary system to the secondary system. The network resources are moved as well, which is called *IP address takeover.* This is basically the ability to transfer a server IP address from one machine to another when a server goes down. To a client application, the two machines appear at different times to be the same server.

Finally the database server resources are started on the secondary system and the database is made available.

Since there is only a single copy of the database, it is always in sync. The time for the failover and restart of the database engine depends on several factors:

- The time needed to detect a failure
- The length of time necessary to move database resources (storage array, networking resources, etc.)
- The time required for the DB2 engine to perform crash recovery

DB2 always performs crash recovery when the database is not shut down properly. Crash recovery is the processing of the log files, making sure all committed transactions are written to disk and uncommitted transactions are rolled back. The time required to perform this operation depends upon the amount of “open” work in the database logs at the time of failure. The entire failover could take just a few seconds, or longer if a large workload needs to be processed from the log files.

One advantage of this type of availability solution is that it does not require that any changes be made to the application or to the client configuration directories. The high-availability software provides a virtual IP address resource for database connections. The IP address will fail over when a failure is detected, and the same connect statement can be used by the application that it used before. When a failover takes place, all applications are disconnected, and the client returns a communication error condition to the application. Once the database server is running on the secondary system, the application can simply reissue the connect statement and continue to process work as before.

6.1.5 High-availability clustering configurations

Similar to the different RAID configurations, several clustering configurations are possible, each with its own cost and redundancy characteristics. Idle standby and mutual takeover were discussed in 6.1.2, “Types of high availability” on page 311. In an active standby configuration one of the active nodes is dedicated to act as a failover node to the rest of the nodes in the cluster. In the “others” configuration, a DB2 node with several logical database partitions is configured such that in case of a failure, its database partitions will be recreated on all other nodes in the cluster.

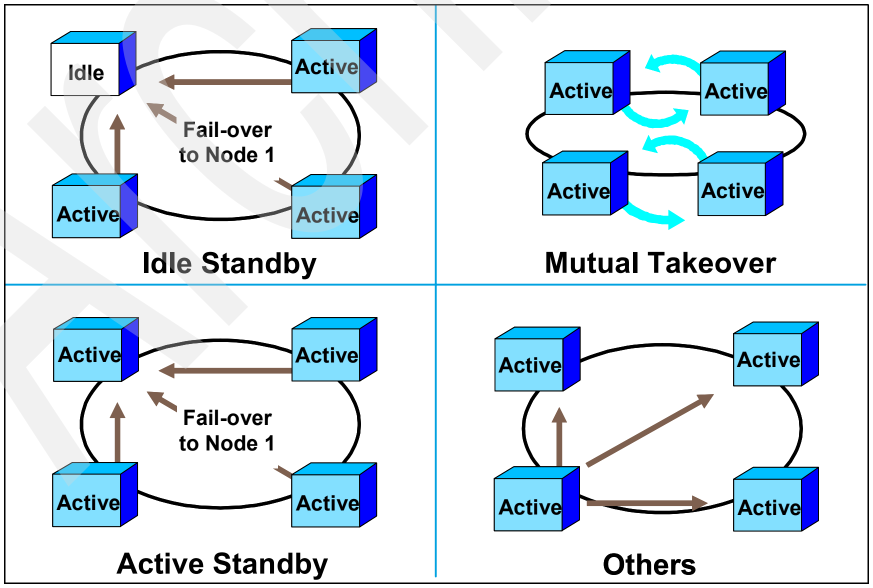


Figure 6-2 High-availability clustering configurations

The DB2 Integrated Cluster Environment supports all the configuration shown in Figure 6-2 on page 315. Specific examples are shown in the following pages.

Mutual takeover clustering configuration

Figure 6-3 is an example of a partitioned database consisting of four database partitions, each running on a separate server. The server hardware consists of four nodes connected with a high-speed network. Additional cluster software such as Tivoli System Automation is installed. The four nodes are separated into two pairs. Each pair is defined as a cluster to the cluster software. Each node acts as the failover server for the other node in the cluster.

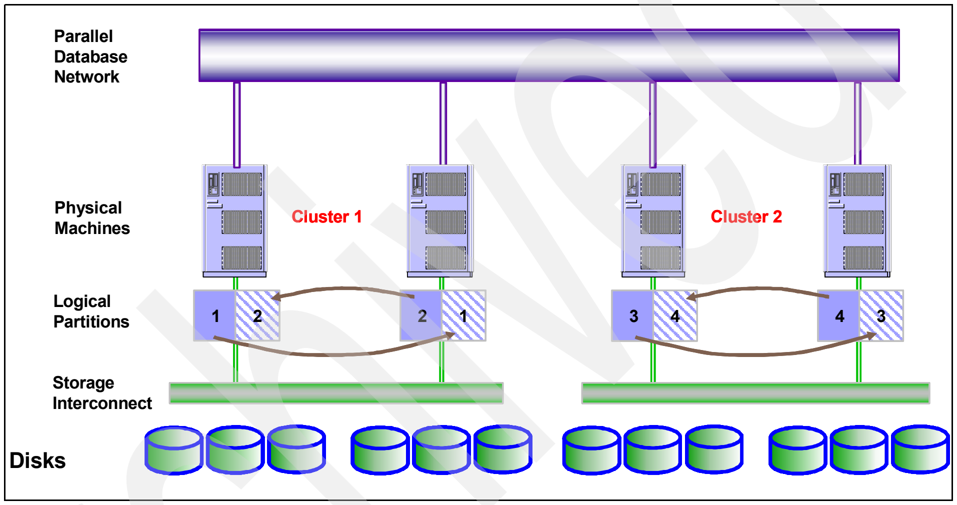


Figure 6-3 Mutual takeover

This is a mutual takeover configuration, meaning that a failure would result in that node's partition being recovered and its workload being taken over from the other node in the cluster that is already running a workload.

The cluster software typically allows the invocation of failover scripts when a failover occurs. These scripts can be used to adjust various operating parameters when a failover or failback occurs. For example, in this example, these scripts may be used to shrink the buffer pools of both affected logical partitions when a failover occurs, and restore them to normal on failback. Several sample takeover scripts, tailored for the various cluster software environments, are provided with DB2.

Idle standby clustering configuration

Figure 6-4 is an "idle standby" example. The cluster includes all four nodes, and one node is kept idle, standing by to take over the workload of any node that fails.

Although this configuration is more costly, it does have the advantage that there is no performance degradation after a failover.

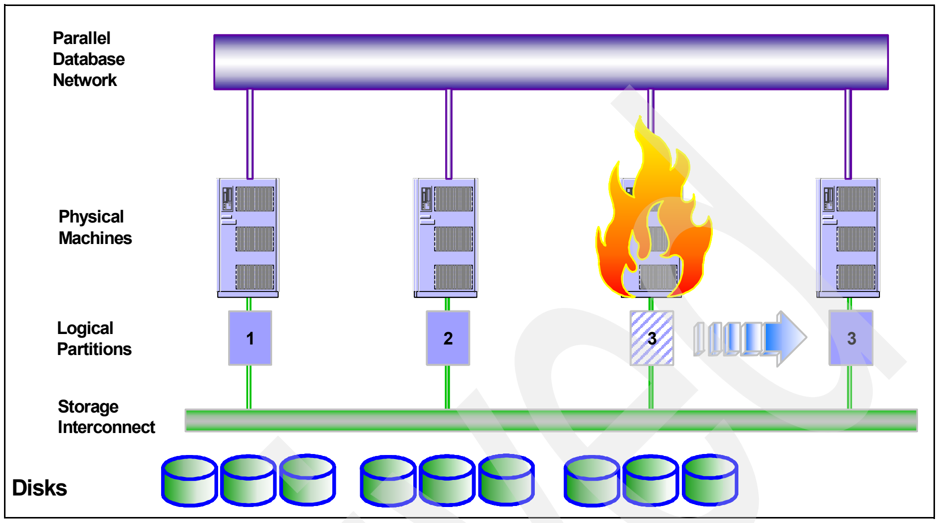


Figure 6-4 Idle standby

Balanced mutual takeover

This is a balanced mutual takeover scenario (Figure 6-5 on page 318) that provides balanced degradation of performance when a failure occurs. It makes use of DB2's ability to define multiple logical partitions on the same physical node.

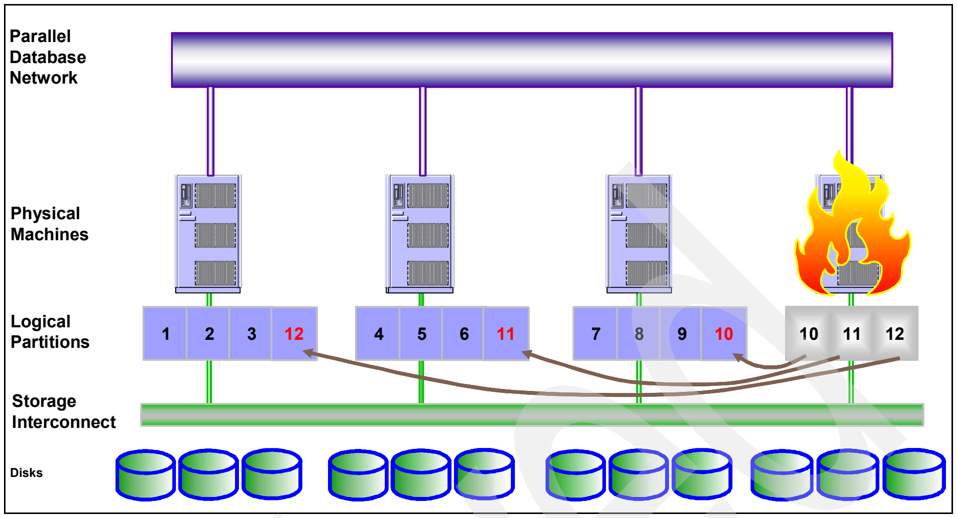


Figure 6-5 Balanced mutual takeover

6.1.6 High-availability software solutions

There are a number of software solutions (commercial or Open Source) that can provide failover protection to a DB2 Integrated Cluster Environment. In this section we want to give a general overview of the following products:

- IBM Tivoli System Automation for Multiplatforms
- LifeKeeper for Linux from Steeleye Technologies Inc.
- Veritas Cluster Server from Veritas
- Open Source High-Availability Linux Project

Further on in this chapter we describe in great detail the actual implementation of a failover solution for a DB2 Integrated Cluster Environment using IBM Tivoli System Automation for Multiplatforms.

IBM Tivoli System Automation for Multiplatforms

IBM Tivoli System Automation for Multiplatforms is a product that provides high availability by automating the control of resources such as processes, file systems, IP addresses, and other resources in Linux-based clusters. It facilitates the automatic switching of users, applications, and data from one system to another in the cluster after a hardware or software failure.

The infrastructure of the product detects the improper operation of a system, transactions, and processes, and initiates corrective action without disrupting users. IBM Tivoli System Automation offers “mainframe-like” high availability by using fast detection of outages and sophisticated knowledge about application components and their relationships. It provides quick and consistent recovery of failed resources and whole applications either in place or on another system of a Linux cluster without any operator intervention. Thus it relieves operators from manual monitoring, remembering application components and relationships, and therefore eliminates operator errors.

IBM Tivoli System Automation allows you to configure high-availability systems through the use of policies that define the relationships among the various components. These policies can be applied to existing applications with minor modifications. Once the relationships are established, IBM Tivoli System Automation will assume responsibility for managing the applications on the specified nodes as configured. This reduces implementation time and the need for complex coding of applications. In addition, systems can be added without modifying scripts, and resources can be easily added, too.

DB2 UDB V8.2 comes with a *DB2 for TSA agent*. The agent consists of the set of scripts necessary for the control of DB2 instances, DAS servers, and file system mount points, as well as utilities that simplify management of the cluster.

More information about IBM Tivoli System Automation for Multiplatforms can be found at:

`http://www.ibm.com/software/tivoli/products/sys-auto-linux`

LifeKeeper for Linux by Steeleye Technologies Inc.

Steeleye's LifeKeeper for Linux is a software application that ensures the continuous availability of applications by maintaining system uptime. LifeKeeper maintains the high availability of clustered Linux systems by monitoring system and application health, maintaining client connectivity, and providing uninterrupted access regardless of where clients reside—on the corporate Internet, intranet or extranet.

With LifeKeeper, hardware component or application faults are detected in advance of a full system failure through multiple fault-detection mechanisms. LifeKeeper monitors Linux clusters using intelligent processes and multiple LAN heartbeats. By sending redundant signals between server nodes to determine system and application health, LifeKeeper confirms a system's status before taking action. This reduces the risk of a single point of failure and minimizes false failovers. LifeKeeper also limits unnecessary failovers by recovering failed applications, without a full failover to another server, if the hardware is still active.

LifeKeeper for Linux provides protection for Linux environments to support disaster tolerance, multiple system failures, or faster recovery, including:

- Multi-directional recovery
- Shared data support

- Cascading failover

Steeleye offers LifeKeeper Application Recovery Kits that include tools and utilities that allow LifeKeeper to manage and control a specific application. The LifeKeeper DB2 Recovery Kit provides protection for DB2 database instances. It is a prepackaged solution that decreases the time to prepare DB2 instances for failover protection.

More information on the LifeKeeper for Linux solution can be found at:

http://www.steeleye.com

Veritas Cluster Server from Veritas

Veritas Cluster Server provides off-the-shelf support for a wide range of applications, including DB2. Veritas Cluster Server Enterprise Agent for DB2 works to make sure that DB2 is highly available, regardless of what happens to the database. The DB2 enterprise agent monitors DB2 database instances while they are up and running on a given system. If the system fails, the agent detects the failure and takes the DB2 instance offline. Veritas Cluster Server conducts failover to another system in the cluster, where the agent brings DB2 instances online.

More information about Veritas Cluster Server can be found at:

http://www.veritas.com

Open Source High-Availability Linux Project

This is a failover solution that comes from the Open Source Community. Its goal is to provide a high-availability (clustering) solution for Linux that promotes reliability, availability, and serviceability (RAS) through a community development effort.

It is packaged with enterprise versions of the Linux distributions. This solution has two main components:

- Heartbeat - Which implements serial, UDP, and PPP/UDP heartbeats together with IP address takeover including a resource model including resource groups.
- DRBD - This component provides software remote monitoring capabilities.

More information about the Open Source High Availability Linux Project can be found at:

http://www.linux-ha.org

6.1.7 Automatic Client Reroute

Up until now we were focusing on the availability of the DB2 server. In this section we want to describe a new availability option for DB2 client applications.

Whenever a DB2 server crashes, each client that is connected to that server receives a communication error that terminates the connection, resulting in an application error. In a high-availability environment it is important to implement either a redundant (stand-by) database or the ability to fail the server over to a standby node. In either case, the DB2 client code attempts to re-establish the connection to the original server, which may be running on a failover node, or to a new server running a stand-by database.

When the connection is re-established, the application receives an error message that informs it of the transaction failure, but the application can continue with the next transaction.

The main goal of the Automatic Client Reroute feature is to enable a DB2 client application to recover from a loss of communication so that the application can continue its work with minimal interruption. But rerouting is only possible when there is an alternate location that is identified to the client connection.

6.2 TSA implementation

This section describes how to implement a highly available DB2 instance with IBM Tivoli System Automation (TSA).

The topics discussed in this chapter are as follows:

- TSA fundamentals
- TSA installation
- TSA resource planning for an HA DB2 instance
- Basic implementation with the **`regdb2salin`** script
- Advanced implementation
- Scaling out HA DB2 clusters

First, we introduce TSA fundamental concepts, followed by the installation steps for TSA. Then we discuss how to implement a HA DB2 cluster with TSA and describe the detailed implementation steps. At the end, we address scaling out issues on HA DB2 clusters.

6.2.1 TSA fundamentals

TSA manages the availability of applications in Linux systems or clusters on xSeries, zSeries, and AIX systems or clusters. It provides the following features:

- High availability and resource monitoring
- Policy-based automation
- Automatic recovery
- Automatic movement of applications
- Resource grouping

The following terms describe the fundamental concepts of TSA.

Cluster/peer domain

A cluster, or a peer domain, is the group of host systems on which TSA manages resources. A cluster can consist of one or more systems or nodes.

Resource

A resource is any piece of hardware or software that can be defined to TSA. These resources can either be defined manually by the administrator using the **mkrsrc** (make resource) command or through “harvesting” functionality of the cluster infrastructure, whereby resources are automatically detected and prepared for use. All resources are controlled through various resource managers provided by TSA. Resources have attributes that can be defined. For example, when considering an IP address as a resource, attributes would include the IP address itself and the net mask. There are two types of resources:

- A *fixed resource* is a resource that runs only on a single node in the cluster.
- A *floating resource* is a resource that can run on several nodes in the cluster.

Resource class

A *resource class* is a collection of resources of the same type. Resource classes allow you to define the common characteristics among the resources. For example, if an application is a resource, the resource class can define identifying characteristics, such as the name of the application; and varying characteristics, such as whether the application is running or not.

Resource group

A *resource group* is a logical container for a collection of resources. This container allows you to control multiple resources as a single logical entity. Resource groups are the primary mechanism for operations within TSA. They can be nested; that is, a resource group can belong to another resource group as its part.

Equivalency

An *equivalency* is a collection of resources that provides the same functionality. For example, equivalencies are used by selecting network adapters that should host an IP address. If one network adapter goes offline, TSA selects another network adapter to host the IP address.

Nominal states

The *nominal state* of a resource group indicates to TSA whether the resources in the group should be online or offline at this point in time. So setting the nominal state of a group to "offline" indicates that you want TSA to stop the resources in the group. You can notify TSA of the desired nominal state of a resource group by changing the NominalState resource group attribute.

Relationships

TSA allows you to define relationships between resources in a cluster. There are two types of relationship:

- *Start/stop relationships* are used to define start and stop dependencies between resources. There are StartAfter, StopAfter, DependsOn, DependsOnAny, and ForcedDownBy relationships in this type.
- *Location relationships* are used when resources must, or should if possible, be started on the same or a different node in a cluster. There are Collocation, AntiCollocation, Affinity, AntiAffinity, and IsStartable relationships in this type.

Quorum

The *quorum* operation is a mechanism to protect critical resources and to keep data consistency. To modify the cluster definition or perform certain cluster operations, active nodes must achieve a quorum in number, more than half of the nodes in the cluster. There are two types of quorum:

- *Configuration quorum* determines when configuration changes in a cluster will be accepted.
- *Operational quorum* determines whether resources can be safely activated without creating conflicts with other resources.

Tie breaker

In case of a tie in which a cluster has been split into two subclusters with an equal number of nodes, a tie breaker is used to determine which subcluster will have an operational quorum.

TSA provides system automation

With resources and resource groups defined, TSA automates the deployment and execution of those resources, based on relationships among resources and resource groups. For example, when TSA finds a resource that goes offline, it automatically tries to restart the resource on the same node or other nodes, depending on the nodes' availability. And if the resource is restarted on another node, the resources depending on that failed resource are also moved to the node, if necessary.

For further details, see Chapter 1, "Introduction", in *IBM Tivoli System Automation for Multiplatforms Guide and Reference Version 1.2,* SC33-8217.

6.2.2 TSA installation

This section describes the installation procedures for TSA.

Supported platforms and prerequisites

TSA V1.2 supports Linux on the zSeries, xSeries, pSeries®, iSeries, and AIX 5.2 platforms.

Before starting the installation, check the following prerequisites:

- The Public Domain Ksh (dpksh) is required.
- Perl is required.
- NFS agnt is required. NFS agent can be downloaded from the TSA Web site:

  `http://www.ibm.com/software/tivoli/producst/sys-auto-linux/downloads.html`
- The Linux environment variable CT_MANAGEMENT_SCOPE must be set to 2.
- At least 100 MB of free space under /usr/sbin.

The following Web site provides you with up-to-date information about the supported platforms and prerequisites:

`http://www.ibm.com/software/tivoli/products/sys-auto-linux/requirements.html`

Installation steps

For each node, repeat the following steps to install TSA. Only the root user can install TSA.

1. Prepare the TSA installation media.

   TSA is shipped with DB2 UDB V8.2 on a CD labelled Additional Features for Linux. To install TSA, prepare the TSA installation medium for each node.
2. Change your working directory.

Change your working directory to <path-to-installation-medium>/SAM12/<platform>. For example, if you mount the TSA installation medium under /mnt/cdrom, type in the following command:

```
/mnt/cdrom/SAM12/i386
```

3. Install TSA.

   Run the installation script **installSAM**.

   ```
   ./installSAM
   ```

   Before installation starts, the license information is displayed. You can scroll forward line-by-line using the Enter key, and page by page using the spacebar. Once you have scrolled to the bottom of the License Information file and you want to accept the license information, type y.

4. Set the Linux environmental variable CT_MANAGEMENT_SCOPE.

   Ensure that the Linux environmental variable CT_MANAGEMENT_SCOPE is set to 2 for all possible TSA administrative users. You can set the variable permanently if you set it in the profile.

   Note: The installation script automatically detects whether CSM is already installed or not. If CSM is installed, the script tries to share some base RPM packages with CSM. You may encounter version conflicts between CSM and TSM. Consult your IBM contacts for the case.

   For further details, see Chapter 2, "Installing IBM Tivoli System Automation", in *IBM Tivoli System Automation for Multiplatforms Guide and Reference Version 1.2,* SC33-8217.

6.2.3 TSA resource planning for an HA DB2 cluster

This section describes how to design resources and resource groups for DB2.

Considerations on implementation methods

To implement a highly available DB2 instance, you must define quite a few resources for:

- DB2 partitions
- File systems
- IP addresses, etc.

We have to organize these into resource groups and define relationships among resources.

DB2 UDB V8.2 provides the **regdb2salin** script as a sample to automate the resource definition for a highly available DB2 instance. At the time of writing, the **regdb2salin** script can handle a partitioned database that:

- Has only one highly available IP address for the whole instance. The IP address is possibly used for a node that is dedicated to coordinator agents.
- Needs at most one file system for each database partition. Each file system is used for a partition's database directory.

If your database configuration matches these criteria, the **regdb2salin** script does most of the work for you to implement a highly available DB2 instance. Otherwise you need to define resources manually. In this case, begin with designing resource schematics for your database.

The **regdb2salin** and other sample scripts can be found under the installed directory /opt/IBM/db2/v8.1/TSA.

Designing resources and resource groups

To design resource schematics, identify the candidates for resources first, and then find dependencies among them. This process gives you a good idea of what a whole resource structure should look like. Though you can arbitrarily define resources and resource groups, there are general steps you can follow:

1. Identify the resources in your cluster. The candidates are:
   - DB2 partitions
   - File systems that must be mounted for partitions
   - Service IP addresses for clients to connect to nodes
   - Network Interface Cards (NICs)
   - Volume groups where the file systems reside (if you are using LVM)
2. Group related resources.

   Begin with grouping a partition and its depending resources. The candidates for related resources are:
   - Service IP address for the partition
   - File systems and volume groups the partition depends on

   These related resources can be grouped with the partition only if they are dedicated to it. Otherwise they must be in a separated resource group to be shared by other resource groups.

   > **Note:** Exclude the file system for the instance owner's home directory from the candidates. It should be grouped with NFS server resources. An NIC is also a resource the partition indirectly depends on; however, it must be grouped into an equivalency with other NICs that host the same set of IP addresses.

3. Identify nodes to deploy resources.

   Identify the nodes where each resource group could run. The list of those nodes are a candidate value for the NodeList attribute of each resource in a group. Check that all the resources in a group can be deployed on the same set of nodes. If not, grouping may be incorrect. Reconsider grouping.

4. Define relationships among resource groups.

   Find dependencies among resource groups. Begin with applying the following constituents:

   – Partitions must be started one-by-one in sequential order. Use *StartAfter* relationships among the partitions.

   – Each partition depends on an NFS server running somewhere in the cluster. Use *DependsOnAny* relationships between the partitions and the NFS server.

   To designate which partitions can collocate, use the following relationships.

   – The *Collocated* relationship between two resources ensures they are located on the same node.

   – The *AntiColloated* relationship between two resources ensures they are located on different nodes.

   – The *Affinity* relationship between resources A and B encourages locating A on the same node where B is already running.

   – The *AntiAffinity* relationship between resource A and B encourages locating A on different nodes from the one B is already running on.

6.2.4 Basic implementation with the regdb2salin script

This section describes the implementation steps using the **`regdb2salin`** script. The implementation steps are summarized below:

- Creating a peer domain
- Setting up a highly available NFS server
- Checking the mount points for file systems
- Creating an equivalency for network interfaces
- Running the **`db2regsarin`** script

Sample configuration

We use the configuration in Figure 6-6 on page 328 as in example in the rest of this section.

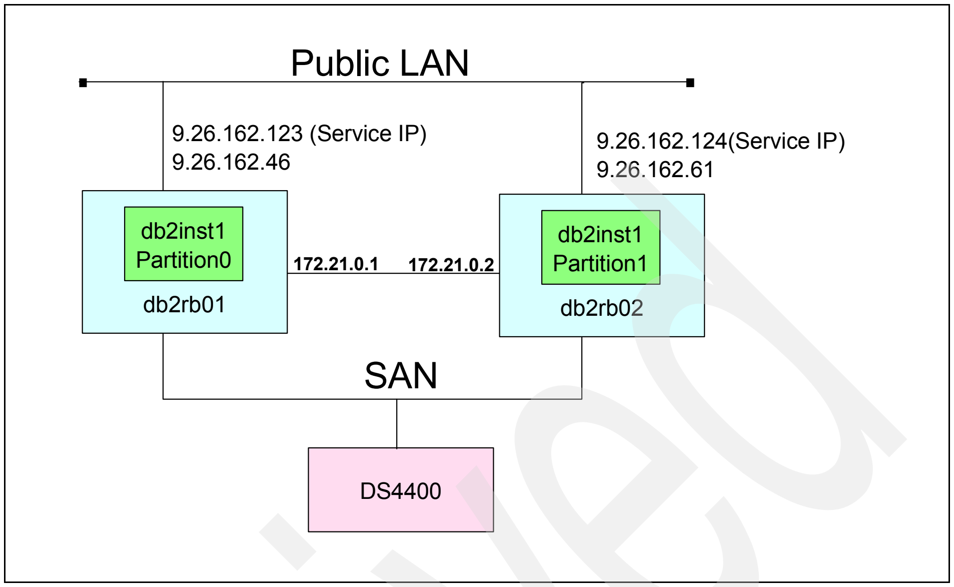


Figure 6-6 A sample configuration for basic implementation

This example is a variant of our lab configuration tailored to be used with the **regdb2sarin** script. The file systems in Table 6-1 reside in DS4400.

Table 6-1 File system used in the sample configuration

Block device	**Mount point**	**Usage**
/dev/sdb1	/db2sharedhome	The instance owner's home directory
/dev/sdb2	/varlibnfs	The NFS-related files
/dev/sdc1	/db2db/db2inst1/NODE0000	The database directory for partition 0
/dev/sdc2	/db2db/db2inst1/NODE0001	The database directory for partition 1
/dev/sde	N/A	The tie-breaker disk

In this example, we also assume that:

- The instance name is db2inst1.
- The service IP address for database clients is 9.26.162.123.
- LVM is not used.
- /db2sharedhome is shared via NFS and mounted on /home/db2inst1.
- Use one SCSI disk (/dev/sde) as a tie breaker.

Creating a peer domain

Before defining resources, you must prepare a peer domain where your resources are deployed. Follow the steps below.

1. Prepare participating nodes for secure communication.

   For nodes to join the cluster, they must be prepared for secure communication within the cluster. Use the **preprpnode** command to enable nodes' secure communication.

   ```
   preprpnode hostname1 hostname2 ....
   ```

   hostnames are the host name of nodes to be enabled. For our sample configuration, we have to enable two nodes, db2rb01 and db2rb02.

   ```
   # preprpnode db2rb01 db2rb02
   ```

2. Create a peer domain.

   To create a peer domain, use the **mkrpdomain** command.

   ```
   mkrpdomain domainname hostname hostname ....
   ```

   domainname is the name of a domain you create, and *hostnames* are the host names of the participating nodes. For our sample configuration, we create a peer domain named SA_Domain.

   ```
   # mkrpdomain SA_Domain db2rb01 db2rb02
   ```

3. Start the peer domain.

   Once you create the domain, start it with the **startrpdomain** command.

   ```
   # startrpdomain SA_domain
   ```

   It takes a while for the domain to be online. Use the **lsrpdomain** command to see the running status of the domain.

   ```
   # lsrpdomain
   Name      OpState RSCTActiveVersion MixedVersions TSPort GSPort
   SA_Domain Online  2.3.3.1           No            12347  12348
   ```

   Note: You cannot define resources before the domain gets online. Make sure that the domain is online before you keep on.

4. Create a tie breaker (optional).

   In a two-nodes only cluster a tie breaker is required. A tie breaker exclusively uses one SCSI disk for cluster quorum duties. The disk can be a very small logical disk of a disk array, but it must be dedicated to the tie breaker.

   To create a tie breaker with a SCSI disk, you have to have four parameters for the disk to identify it: HOST, CHANNEL, ID, and LUN. With the block device name for the disk handy, use the **dmesg** command to check these parameters. The following is an example in our sample configuration.

```
# dmesg | grep "Attached scsi disk"
Attached scsi disk sda at scsi0, channel 0, id 0, lun 0
Attached scsi disk sdb at scsi0, channel 0, id 1, lun 0
Attached scsi disk sdc at scsi1, channel 0, id 0, lun 1
Attached scsi disk sdd at scsi1, channel 0, id 0, lun 2
Attached scsi disk sde at scsi1, channel 0, id 0, lun 3
```

For each attached scsi disk, scsi, channel, id, and lun are its values for HOST, CHANNEL, ID, and LUN. So our tie breaker disk /dev/sde is identified by the values of HOST=0, CHANNEL=0, ID=0, and LUN=3. With these parameters, we can create a tie breaker with the **mkrsrc** command.

```
# mkrsrc IBM.TieBreaker Name="tb" Type="SCSI" \
    DeviceInfo="HOST=1 CHAN=0 ID=0 LUN=3" \
    HeartbeatPeriod = 5
```

In our sample configuration, we create the tie breaker named 'tb'. We must set the tie breaker to the domain with the **chrsrc** command.

```
# chrsrc -c IBM.PeerNode OpQuorumTieBreaker="tb"
```

Setting up a highly available NFS server

A partitioned DB2 instance relies on NFS to share its home directory among partitions, so you also have to make the NFS service highly available to implement a highly available DB2 instance. The steps below are a brief overview of highly available NFS service implementation. For further details, see "Highly available NFS server with Tivoli System Automation for Linux" at the Web site:

http://www.ibm.com/db2/linux/papers.html

1. Install the additional RPM package.

   Several sample NFS scripts are provided as an add-on package for TSA. Download the RPM package 'sam-policies' from the URL below.

   http://www.ibm.com/software/tivoli/producst/sys-auto-linux/downloads.html

   Install the downloaded RPM package with the **rpm** command on each node as root user.

   Important: This RPM package depends on the TSA RPM packages. Do not forget to uninstall this RPM package before you uninstall TSA.

2. Stop NFS.

While doing the implementation, make sure that the NFS server is stopped on all nodes. And also make sure that the NFS server does not start automatically at system boot time. Let TSA handle it.

3. Prepare the NFS-related files in a shared storage.

   An NFS server creates several files under /var/lib/nfs and uses them to store various state information. To make the takeover of a failed NFS server transparent to its clients, these files must also be taken over.

   First, create a file system in shared storage for these files and prepare the mount point /varlibnfs on each node.

   Then, copy the files to the /varlibnfs from /var/lib/nfs on either node.

   ```
   # mount /varlibnfs
   # cp /var/lib/nfs/* /varlibnfs
   # unmount /varlibnfs
   ```

   Then create a symbolic link on each node.

   ```
   # cd /var/lib
   # mv nfs nfs.org
   # ln -s /varlibnfs nfs
   ```

4. Edit /etc/exports and /etc/fstab.

   On each node, edit /etc/exports to include the entry for the shared instance home directory. In our sample configuration, /etc/exports has the entry below.

   ```
   /db2sharedhome db2rb*(rw,sync,no_root_squash)
   ```

   Also edit /etc/fstab to add the entries for:

   – The local mount point for /varlibnfs
   – The local mount point for the instance home directory

   In our sample configuration, these entries look like:

   ```
   /dev/sdb1   /db2sharedhome  ext3  noauto 1 0
   /dev/sdb2   /varlibnfs      ext3  noauto 1 0
   ```

   Make sure that these files are identical across all nodes.

5. Assign the service IP address for the NFS server.

   Prepare one service IP address for the NFS server. In our sample configuration, we use 9.26.162.36 for the service IP address.

6. Edit the configuration file.

   TSA provides a utility script to automate the resource definition for an NFS server. This script takes a configuration file as its input and defines the necessary resources automatically.

Edit these configuration file named *sa-nfsserver-conf* located under /usr/sbin/rsct/sapolicies/nfsserver. You must set appropriate values to the configuration parameters in Table 6-2.

Table 6-2 Parameters to set

Parameter	Description
nodes	The nodes in the domain where the NFS server can run
ip_1	The service IP and net mask
nieq_1	The name of an equivalency for the network interfaces
data_work	The mount point for the shared data

For our sample configuration, these parameters are set as below:

```
nodes=”db2rb01 db2rb02”
ip_1=”9.26.162.36,255.255.255.0”
nieq_1=”eth0:db2rb01,eth0:db2rb02”
detractor=”/db2sharedhome”
```

7. Run the script.

   Run the **cfgnfsserver** script to define the resources for the NFS server.

   ```
   # /usr/sbin/rsct/sapolicies/nfsserver/cfgnfsserver -p
   ```

8. Make it online.

   Now the resource group for the NFS server is created and ready to get online. Use the **chrg** command to change the nominal state of the resource group.

   ```
   # chrg -o online SA-nfsserver-rg
   ```

Checking the mount points for file systems

Make sure that /etc/fstab of all nodes has the entries for:

- The local mount points for the file systems in a shared storage.
  - Database directories for each node (if you place the database directory somewhere other than default place)
  - The instance home directory
  - Table space containers (directory and file containers)
- The NFS mount point for the instance home directory.

Important: For the NFS mount point, use the service IP assigned for the NFS server.

Creating an equivalency for network interfaces

IP address resources depend on network interfaces. To fail over an IP address from one network interface to another, you must group the network interfaces into an equivalency, which hosts the same IP address. Create an equivalency with the **mkequ** command.

```
mkequ equivalencyname \
  IBM.NetworkInterface:nicname1:hostname1,nicname2:hostname2, ....
```

equivalencyname is the name of an equivalency you create. *nicname* is the name of the network interface that belongs to the equivalency, and *hostname* is the host name of the node the interface resides. In our sample configuration, we have one service IP address and two nodes that can host the IP address.

```
mkequ virpubnic IBM.NetworkInterface:eth0:db2rb01,eth0:db2rb02
```

Important: The **regdb2salin** script assumes that the service IP address depends on the equivalency named *virpubnic*. You must create the equivalency with this name when you use the script.

Mounting client NFS mount points

Before your DB2 instance gets online, each participating computer must mount the instance home directory. You can use automounter to do this for you, or manually mount the home directory. In the case you choose to mount it manually, it is the responsibility of the system administrator to mount the client NFS mount point prior to use of your DB2 instance. To use automounter, see "Ensure NFS is running" on page 147.

Important: The **regdb2salin** script makes a DB2 instance go online at the end of configuration process. Be sure to mount the home directory on participating computers before you run the script.

Running the regdb2salin script

Now you are ready to run the **regdb2salin** script. The script is located under /opt/IBM/db2/V8.1/ha/salinux.

```
regdb2salin -a instancename -i serviceip
```

instancename is the name of instance and *serviceip* is the service IP address for the first partition. Example 6-1 shows the case for our sample configuration.

Example 6-1 Running the regdb2salin script

```
# cd /opt/IBM/db2/V8.1/ha/salinux
# ./regdb2salin -a db2inst1 -i 9.26.162.123
```

```
About to register db2inst1 with TSA

Checking cluster validity...
Checking configuration ...
Checking db2rb01 ...
Checking db2rb02 ...

Making resource group  db2_db2inst1_0-rg  ...
Making IP resource  db2_db2inst1_0-rs_ip ...
Making DB2 resource  db2_db2inst1_0-rs  ...
Making resource group  db2_db2inst1_1-rg  ...
Making DB2 resource  db2_db2inst1_1-rs  ...

Online resource group db2_db2inst1_0-rg ...
Online resource group db2_db2inst1_1-rg ...

db2_db2inst1_0-rg is now online
db2_db2inst1_1-rg is now online

Done, instance is now HA
```

As you notice from the example, the resource groups become online after the resources are created. To verify this, use the **getstatus** script to see the current operational state of the resources, as shown in Example 6-2.

Example 6-2 Display the current status

```
# /opt/IBM/db2/V8.1/ha/salinux/getstatus

-- Resource Groups and Resources --

                       Group Name                Resources
                       ----------                ---------
                db2_db2inst1_0-rg        db2_db2inst1_0-rs
                db2_db2inst1_0-rg   db2_db2inst1_0-rs_mount
                db2_db2inst1_0-rg      db2_db2inst1_0-rs_ip
                                -                        -
                db2_db2inst1_1-rg        db2_db2inst1_1-rs
                db2_db2inst1_1-rg   db2_db2inst1_1-rs_mount
                                -                        -
                  SA-nfsserver-rg      SA-nfsserver-server
                  SA-nfsserver-rg        SA-nfsserver-ip-1
                  SA-nfsserver-rg SA-nfsserver-data-varlibnfs
                  SA-nfsserver-rg    SA-nfsserver-data-work
                                -                        -


-- Resources --
```

```
                Resource Name                    Node Name              State
                -------------                    ---------              -----
            db2_db2inst1_0-rs                      db2rb01             Online
            db2_db2inst1_0-rs                      db2rb02            Offline
                            -                            -                  -
      db2_db2inst1_0-rs_mount                      db2rb01             Online
      db2_db2inst1_0-rs_mount                      db2rb02            Offline
                            -                            -                  -
         db2_db2inst1_0-rs_ip                      db2rb01             Online
         db2_db2inst1_0-rs_ip                      db2rb02            Offline
                            -                            -                  -
            db2_db2inst1_1-rs                      db2rb01            Offline
            db2_db2inst1_1-rs                      db2rb02             Online
                            -                            -                  -
      db2_db2inst1_1-rs_mount                      db2rb01            Offline
      db2_db2inst1_1-rs_mount                      db2rb02             Online
                            -                            -                  -
          SA-nfsserver-server                      db2rb01             Online
          SA-nfsserver-server                      db2rb02            Offline
                            -                            -                  -
            SA-nfsserver-ip-1                      db2rb01             Online
            SA-nfsserver-ip-1                      db2rb02            Offline
                            -                            -                  -
 SA-nfsserver-data-varlibnfs                       db2rb01             Online
 SA-nfsserver-data-varlibnfs                       db2rb02            Offline
                            -                            -                  -
       SA-nfsserver-data-work                      db2rb01             Online
       SA-nfsserver-data-work                      db2rb02            Offline
                            -                            -                  -
```

You can control the nominal state of resource groups with the **chrg** command. To make a resource group offline, type the command below.

```
chrg -o offline resourcegroupname
```

Type in the command below to make a resource group online.

```
chrg -o online resourcegroupname
```

Note: You can remove the resource definitions with the **unregdb2salin** script. Be sure to make the resources offline before you run the script.

6.2.5 Advanced implementation

This section describes the advanced implementation steps, which do not use the **regdb2salin** script. The implementation steps are summarized below:

1. Creating a peer domain

2. Setting up a highly available NFS server
3. Checking the mount points for file systems
4. Creating an equivalency for network interfaces
5. Creating resources
6. Creating resource groups
7. Defining relationships among resources
8. Configuring DB2 for high availability

Sample configuration

We use the configuration in Figure 6-7 as an example in the rest of this section.

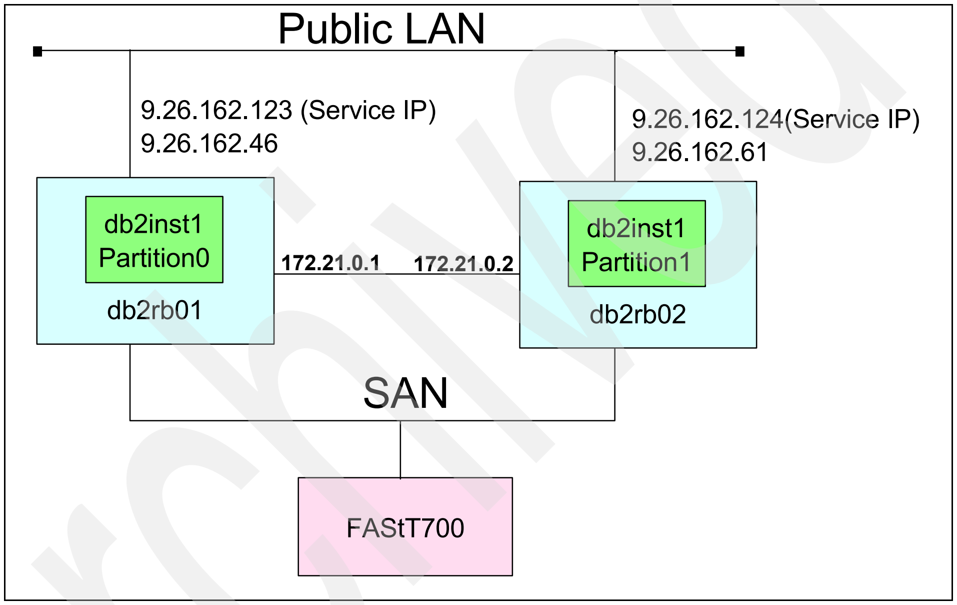


Figure 6-7 A sample configuration for advanced implementation

This example is almost the same as the example used in the previous section, except for the following:

- Each node is assigned its own service IP address.
- Each partition uses more than one file system.

The file systems used and service IP addresses assigned are shown in Table 6-3 on page 337 and Table 6-4 on page 337.

Table 6-3 File systems used in the sample configuration

Block device	Mount point	Usage
/dev/sdb1	/db2sharedhome	The instance owner's home directory
/dev/sdb2	/varlibnfs	The NFS-related files
/dev/sdc1	/db2db/db2inst1/NODE0000	The database directory for partition 0
/dev/sdc2	/db2fs1p0	The regular table spaces for partition 0
/dev/sdc3	/db2fs2p0	The temporary table spaces for partition 0
/dev/sdc4	/db2db/db2inst1/NODE0001	The database directory for partition 1
/dev/sdc5	/db2fs1p1	The regular table spaces for partition 1
/dev/sdc6	/db2fs2p1	The temporary table spaces for partition 1
/dev/sde	N/A	The tie breaker disk

Table 6-4 Service IP addresses used in the sample configuration

Partition	Service IP address
0	9.26.162.123
1	9.26.162.124

Creating a peer domain

You need to create a peer domain. The procedures are same as described in the previous section. See "Creating a peer domain" on page 329 for details.

Setting up a highly available NFS server

You need to set up a highly available NFS server. The procedures are the same as described in the previous section. See "Setting up a highly available NFS server" on page 330 for details.

Checking the mount points for file systems

Make sure that /etc/fstab on each node includes the necessary entries. The check points are almost the same as described in "Checking the mount points for file systems" on page 332. In addition, you must check for the entries in /etc/fstab for the file systems below:

- File or directory containers for table spaces
- Log directories

- Any other file systems to be mounted for the instance

Creating an equivalency for network interfaces

As we described in the previous section, you need to group NICs into equivalencies that can host the same set of IP addresses. See “Creating an equivalency for network interfaces” on page 333 for details.

Creating resources

Every application needs to be defined as a resource in order to be managed and automated with TSA. We also regard a file system as an application, which just mounts and unmounts the file system. Application resources are usually defined in the generic resource class IBM.Application. In this resource class there are several attributes that define a resource, but at least three of them are application-specific:

- StartCommand
- StopCommand
- MonitorCommand

These commands may be scripts or binary executables. There are some requirements for these commands, and usually you have to write the scripts for your application and test them well. But as for DB2, the sample scripts are provided and you do not have to write them yourself. You just have to define the resource with these sample scripts according to your resource schematics. To define a resource, use the **mkrsrc** command.

```
mkrsrc classname \
  Name="resourcename" \
  attribute1=value1 \
  attribute2=value2 \
      .
      .
```

classname is the name of the resource class, and usually you just need two classes, as below:

- IBM.Application is the class for an application resource.
- IBM.ServiceIP is the class for a service IP address resource.

resourcename is the name of the resource you create. You must give it a unique resource name in the domain. *attributes* are the names of resource attributes, and *values* are the values for the attributes. For IBM.Application resources, you should define the attributes in Table 6-5 on page 339.

Note: Technically, *name* is also an attribute, which is used to identify the resource.

And for IBM.ServiceIP resources, the attributes you should define are shown in Table 6-6.

Table 6-5 Resource attributes for IBM.Application

Name	Description	Default
ProtectionMode	Indicates whether the resource is critical or not. Takes 0 (non-critical) or 1 (critical).	0
ResourceType	Indicates whether the resource is fixed or floating. Takes 0 (fixed) or 1 (floating).	1
StartCommand	The exact command and arguments used to start the application.	N/A
StartCommandTimeout	The timeout for the start command.	
StopCommand	The exact command and arguments used to stop the application.	N/A
StopCommandTimeout	The timeout for the stop command.	
MonitorCommand	The exact command and arguments used to check the running status of the application.	N/A
MonitorCommandPeriod	The interval between monitor command invocations.	5
MonitorCommandTimeout	The timeout for the monitor command.	5
UserName	The name of the user ID used to run the start, stop, and monitor scripts.	N/A
NodeNameList	Indicates on which nodes the application is allowed to run.	N/A

Table 6-6 Resource attributes for IBM.ServiceIP

Name	Description	Default
ProtectionMode	Indicates whether the resource is critical or not. Takes 0 (non-critical) or 1 (critical).	0
ResourceType	Indicates whether the resource is fixed or floating. Takes 0 (fixed) or 1 (floating).	1
IPAddress	The IP address.	N/A
NetMask	The NetMask for the IP address.	N/A
NodeNameList	Indicates on which nodes the IP address is hosted.	N/A

According to your resource schematics, create the resources with the **mkrsrc** command. Figure 6-8 shows our resource schematics.

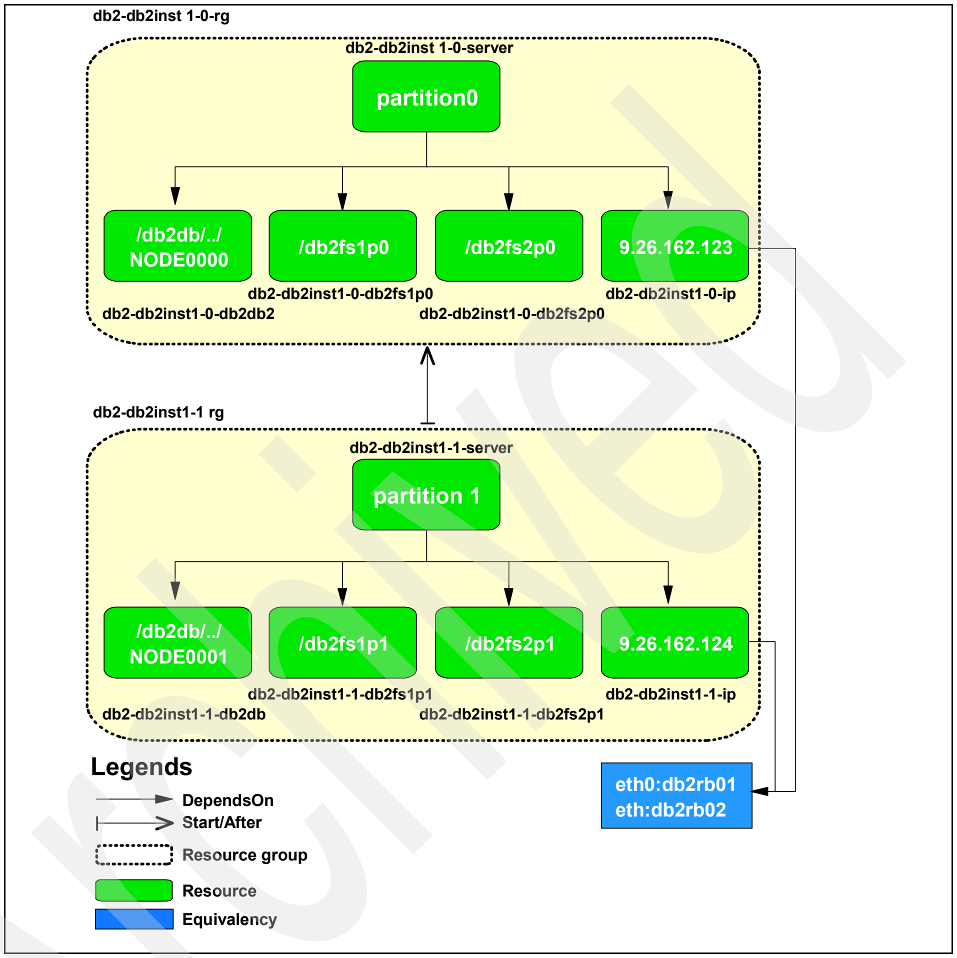


Figure 6-8 Resource schematics for the sample configuration

As you notice, we have two resource groups, one for each partition. And each resource group has five resources. One for the partition, three for the file systems, and one for the service IP address. We also define an equivalency named “virpubnic”, which groups network interfaces on the nodes.

Note: We have omitted the resource group for the NFS server. Assume that each partition resource group has a “DependsOnAny” relationship to the NFS server resource group.

Example 6-3 shows our resource definition script for the partition 0, which belongs to the resource group “db2-db2inst1-0-rg”.

Example 6-3 Resource definition for the partition 0

```
mkrsrc IBM.Application \
  Name="db2-db2inst1-0-server" \
  StartCommand="/opt/IBM/db2/V8.1/ha/salinux/db2_start.ksh db2inst1 0" \
  StartCommandTimeout=240 \
  StopCommand="/opt/IBM/db2/V8.1/ha/salinux/db2_stop.ksh db2inst1 0" \
  StopCommandTimeout=160 \
  MonitorCommand="/opt/IBM/db2/V8.1/ha/salinux/db2_monitor.ksh db2inst1 0" \
  MonitorCommandPeriod=12 \
  MonitorCommandTimeout=7 \
  UserName=root \
  NodeNameList={'db2rb01','db2rb02'}

mkrsrc IBM.Application \
  Name="db2-db2inst1-0-db2db" \
  ProtectionMode=1 \
  StartCommand="/opt/IBM/db2/V8.1/ha/salinux/mount_start.ksh
/db2db/db2inst1/NODE0000" \
  StartCommandTimeout=120 \
  StopCommand="/opt/IBM/db2/V8.1/ha/salinux/mount_stop.ksh
/db2db/db2inst1/NODE0000" \
  StopCommandTimeout=300 \
  MonitorCommand="/opt/IBM/db2/V8.1/ha/salinux/mount_monitor.ksh
/db2db/db2inst1/NODE0000" \
  MonitorCommandPeriod=6 \
  MonitorCommandTimeout=4 \
  UserName=root \
  NodeNameList={'db2rb01','db2rb02'}

mkrsrc IBM.Application \
  Name="db2-db2inst1-0-db2fs1p0" \
  ProtectionMode=1 \
  StartCommand="/opt/IBM/db2/V8.1/ha/salinux/mount_start.ksh /db2fs1p0" \
  StartCommandTimeout=120 \
  StopCommand="/opt/IBM/db2/V8.1/ha/salinux/mount_stop.ksh /db2fs1p0" \
  StopCommandTimeout=300 \
  MonitorCommand="/opt/IBM/db2/V8.1/ha/salinux/mount_monitor.ksh /db2fs1p0" \
  MonitorCommandPeriod=6 \
  MonitorCommandTimeout=4 \
  UserName=root \
  NodeNameList={'db2rb01','db2rb02'}

mkrsrc IBM.Application \
  Name="db2-db2inst1-0-db2fs2p0" \
  ProtectionMode=1 \
  StartCommand="/opt/IBM/db2/V8.1/ha/salinux/mount_start.ksh /db2fs2p0" \
  StartCommandTimeout=120 \
  StopCommand="/opt/IBM/db2/V8.1/ha/salinux/mount_stop.ksh /db2fs2p0" \
  StopCommandTimeout=300 \
  MonitorCommand="/opt/IBM/db2/V8.1/ha/salinux/mount_monitor.ksh /db2fs2p0" \
  MonitorCommandPeriod=6 \
  MonitorCommandTimeout=4 \
  UserName=root \
  NodeNameList={'db2rb01','db2rb02'}

mkrsrc IBM.ServiceIP \
  Name="db2-db2inst1-0-ip" \
  ProtectionMode=1 \
  IPAddress=9.26.162.123 \
  NetMask=255.255.255.0 \
  NodeNameList={'db2rb01','db2rb02'}
```

As described before, each IBM.Application resource needs the specific start, stop, and monitor command. DB2 UDB V8.2 provides several scripts for this purpose.

- **db2_start.ksh** is the start command script for partition resources.
- **db2_stop.ksh** is the stop command script for partition resources.
- **db2_monitor.ksh** is the monitor command script for partition resources.
- **mount_start.ksh** is the start command script for file system resources.
- **mount_stop.ksh** is the stop command script for file system resources.
- **mount_monitor.ksh** is the monitor command script for file system resources.

These scripts are located under /opt/IBM/db2/V8.1/ha/salinux.

Important: You must carefully set the time-out and interval periods for partition resources; otherwise your resources will not work. The time needed to start and stop an instance increases almost linearly in proportion to the number of partitions. For our sample configuration, we use the following formulas.

```
Start-command-timeout = n * 30 + 180 where n is the number of partitions
Stop-command-timeout = n * 20 + 120
Monitor-command-interval = n + 10
Monitor-command-timeout = n + 5
```

These formulas are very specific to our test lab environment, and you should never use them blindly. Starting with these formulas is not a problem, but you have to carefully test them prior to use in your production environment.

We also have to create the resources for partition 1, but we omit the sample listing here for the sake of simplicity. It is almost the same as Example 6-3 on page 341, except for the minor differences in resource names and values.

Creating resource groups

After you create the resources, group them into resource groups. First you have to create resource groups with the **mkrg** command.

```
mkrg groupname
```

Then add member resources to the resource group with the **addrgmbr** command.

```
addrgmbr -g groupname resourceclass:resourcename
```

For our sample configuration we form the resource group for partition 0, as shown in Example 6-4.

Example 6-4 Resource group definition for partition 0

```
mkrg db2-db2inst1-0-rg

addrgmbr -g db2-db2inst1-0-rg IBM.Application:db2-db2inst1-0-server
addrgmbr -g db2-db2inst1-0-rg IBM.Application:db2-db2inst1-0-db2db
addrgmbr -g db2-db2inst1-0-rg IBM.Application:db2-db2inst1-0-db2fs1p0
addrgmbr -g db2-db2inst1-0-rg IBM.Application:db2-db2inst1-0-db2fs2p0
addrgmbr -g db2-db2inst1-0-rg IBM.ServiceIP:db2-db2inst1-0-ip
```

For the resources in a resource group, define the relationships among them with the **mkrel** command.

```
mkrel -p relationshiptype -S sourcename -G targetname relationshipname
```

relationshipname is the name of the relationship you create, and *relationshiptype* is the type of the relationship described in “Relationships” on page 323. *sourcename* and *targetname* are the names of the resource or resource group that form this relationship.

Note: A relationship has a direction from the source to the target. Be careful about the direction when you specify the source and target.

According to our resource schematics, we define the relationships as shown in Example 6-5.

Example 6-5 Dependency definition for partition 0

```
mkrel -p DependsOn \
  -S IBM.Application:db2-db2inst1-0-server \
  -G IBM.Application:db2-db2inst1-0-db2db \
  db2-db2inst1-0-server-dependson-db2-db2inst1-0-db2db

mkrel -p DependsOn \
  -S IBM.Application:db2-db2inst1-0-server \
  -G IBM.Application:db2-db2inst1-0-db2fs1p0 \
  db2-db2inst1-0-server-dependson-db2-db2inst1-0-db2fs1p0

mkrel -p DependsOn \
  -S IBM.Application:db2-db2inst1-0-server \
  -G IBM.Application:db2-db2inst1-0-db2fs2p0 \
  db2-db2inst1-0-server-dependson-db2-db2inst1-0-db2fs2p0

mkrel -p DependsOn \
  -S IBM.Application:db2-db2inst1-0-server \
  -G IBM.ServiceIP:db2-db2inst1-0-ip \
  db2-db2inst1-0-server-dependson-db2-db2inst1-0-ip

mkrel -p DependsOn \
  -S IBM.ServiceIP:db2-db2inst1-0-ip \
  -G IBM.Equivalency:virpubnic \
  db2-db2inst1-0-ip-dependson-virpubnic
```

Again, we omit the sample listing for partition 1 for the sake of simplicity.

Defining relationships among resource groups

As we described in "Designing resources and resource groups" on page 326, you have to define a few relationships among resource groups.

- Use StartAfter among the partitions to serialize their startup.
- Use DependsOnAny between the partitions and the NFS server.

Example 6-6 is the example for our sample configuration.

Example 6-6 Relationship definition

```
mkrel -p DependsOnAny \
  -S IBM.ResourceGroup:db2-db2inst1-0-rg \
  -G IBM.Application:SA-nfsserver-server \
  db2-db2inst1-0-rg-dependsonany-SA-nfsserver-server

mkrel -p DependsOnAny \
  -S IBM.ResourceGroup:db2-db2inst1-1-rg \
  -G IBM.Application:SA-nfsserver-server \
  db2-db2inst1-1-rg-dependsonany-SA-nfsserver-server

mkrel -p StartAfter \
  -S IBM.ResourceGroup:db2-db2inst1-1-rg \
  -G IBM.ResourceGroup:db2-db2inst1-0-rg \
db2-db2inst1-1-rg-startafter-db2-db2inst1-0-rg
```

Configuring DB2 for high availability

As the final touch, set the DB2_NUM_FAILOVER_NODES variable. This variable tells DB2 how many partitions the FCM expects to fail over on the node. Use the **db2set** command to update the variable.

```
# db2set DB2_NUM_FAILOVER_NODES 2
```

For our sample environment, we simply set the variable to the number of nodes.

Now you are ready to make the DB2 instance go online.

```
# chrg -o online db2-db2inst1-0-rg
# chrg -o online db2-db2inst1-1-rg
```

Use the **getstatus** script under /opt/IBM/db2/V8.1/ha/salinux to see the running status of these resources. For further information about TSA operations, see *IBM Tivoli System Automation for Multiplatforms Guide and Reference Version 1.2,* SC33-8217.

6.2.6 Scaling out your HA DB2 clusters

When you add nodes and/or partitions to your cluster to scale out the capacity, you have to reconfigure your TSA implementation. Though the procedures for this reconfiguration vary implementation by implementation, there are a few hints for the task:

- The simplest, but not easiest approach is to drop all resources and resource groups, and then redo the configuration steps from resource planning.
- If the number of partitions is the same and a few new nodes are added to the cluster, you just have to change the NodeList attributes of the resources, so that they can be located on the new nodes.
- If new partitions are added, create a new resource group for each newly added partition. Then add relationships to ensure each that partition starts sequentially. You also have to modify the NodeList attribute of each resource to arrange partition deployments.

Before you make any changes to the resource configuration, do not forget to make the resource groups offline. For example, assume that we have a configuration, where four partitions are deployed on two nodes, as described in the db2nodes.cfg file shown below.

```
0 db2rb01 0
1 db2rb01 1
2 db2rb02 0
```

```
3 db2rb02 1
```

If we add two new nodes, db2rb03 and db2rb04, to the cluster, the db2nodes.cfg file should be modified as below.

```
0 db2rb01 0
1 db2rb02 0
2 db2rb03 0
3 db2rb04 0
```

In this case, we have to modify the NodeList attributes of the resources that corresponds to the partitions. Use the **chrsrc** command to change the resource attributes.

```
chrsrc -s 'Name like "db2-db2inst1-%" && ResourceType=1' \
  IBM.Application \
  NodeNameList="{'db2rb01','db2rb02','db2rb03','db2rb04'}"
```

In our example, we assume that each partition resource has a name like 'db2-db2inst1-*number*'. You also have to change the NodeList attributes of the service IP resource.

```
chrsrc -s 'Name like "%" && ResourceType=1' \
  IBM.ServiceIP \
  NodeNameList="{'db2rb01','db2rb02','db2rb03','db2rb04'}"
```

6.2.7 Notes on Logical Volume Manager

If you would like to use Logical Volume Manager, you have to define a volume group as a resource.

Writing scripts for volume groups

You have to write your own scripts for volume groups. Though we do not have any samples here, there are a few hints for you.

- Start and Stop command

  Use the **vgchange** command to activate or deactivate a volume group.

- Monitor command

  Use the **vgdisplay** command to monitor whether a volume group is activated or not.

In the scripts, do not forget to check the input parameters to the scripts. At least you have to check whether the specified volume group exists or not. And in case you cannot find the volume group, you may want to run the **vgscan** command and recheck before exiting with an error code.

Add volume group resources to resource groups

You can create a separate resource group for a volume group resource. If a volume group is dedicated to resources in a single resource group, you may add the volume group resource to that group, rather than separate the volume group in another group. In either way, define DependsOn relationships from file system resources to the volume group resource. For example, if the partitions in our sample configuration (shown in Figure 6-7 on page 336) use LVM to store files systems, the resource schematics should look like the one shown in Figure 6-9.

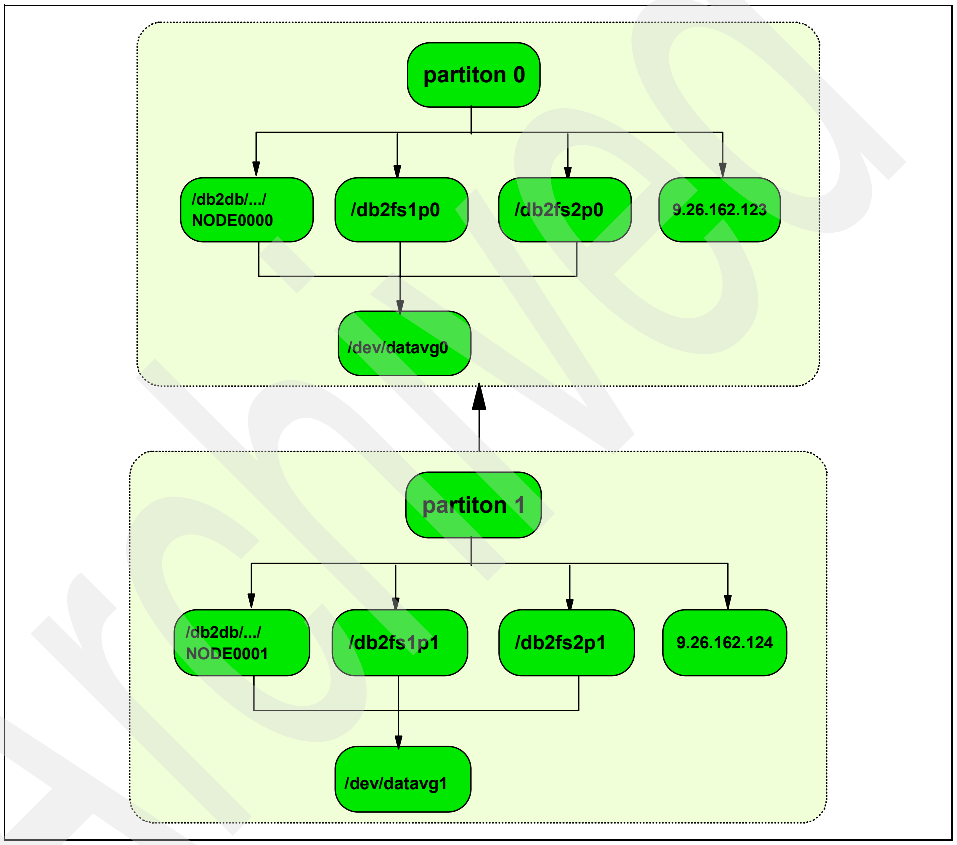


Figure 6-9 A sample resource schematics using LVM

In this example, we assume each partition uses a volume group /dev/datavg*x*, where *x* is the partition number.

7

Scaling

In this chapter we discuss scalability in the DB2 Integrated Cluster Environment. Much of this topic has broader application to any DB2 partitioned or clustered environment, but the DB2 Integrated Cluster Environment architectures lend themselves particularly well to quickly building high-performance clustered database systems with a fast return on investment.

We begin with a general discussion of scalability, and provide some guidance to assist you in planning for the growth of your business and the growth of your database.

We then examine a number of scaling strategies in more detail with an emphasis on horizontal scaling (scale-out).

Finally we walk you through the steps involved in scaling your database.

This chapter contains the following sections:

- Introduction to scalability
- Scaling up
- Scaling out

7.1 Introduction to scalability

Scalability is defined as a measure of a system's ability to maintain a given level of performance as system resources increase. In an ideal situation, the level of system performance will increase in direct proportion to the increase in resources. In other words, a doubling of system resources will result in a doubling of performance. A directly proportionate increase such as this is called "linear scalability" and is the goal for any database management system.

A system's scalability can be plotted on a graph, as shown in Figure 7-1.

In this fictitious example, System A (the top-most line) achieved "better than linear" scalability, meaning that for every 100 percent increase in system resources the system achieved greater than a 100 percent increase in performance. System B (the middle line) achieved exact linear scalability, achieving a 100 percent increase in performance for every 100 percent increase in system resources. System C (represented by the bottom-most line) achieved "less than linear" scalability, meaning that for every 100 percent increase in system resources the system achieved less than a 100 percent increase in performance.

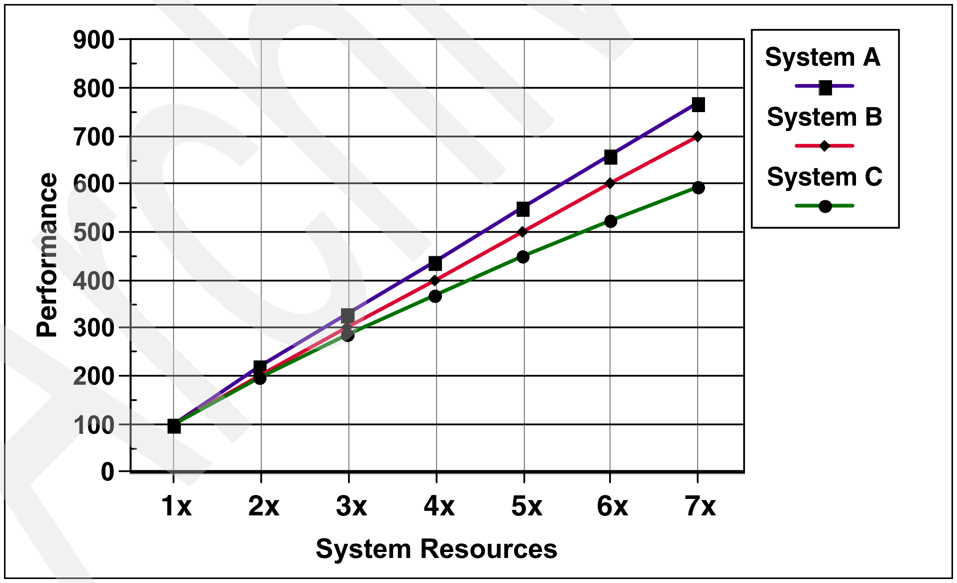


Figure 7-1 Examples of scalability curves

When you examine the scalability of your own systems you will likely find that some areas scale better than others. For example, you might reach 97 percent scalability on your batch workload and 105 jpercent scalability on your end-user queries.

7.1.1 Scale-up vs. scale-out

When we need to increase the capacity of a system we can choose to scale in one of two directions, vertical or horizontal.

Vertical scaling

Vertical scaling, which is also known as “scale up”, is the process of adding resources (for example, memory, processors, storage) to an existing database partition.

The benefit to scaling up is that it is relatively easy. In general it requires only that the software be designed to take advantage of additional resources. For example, your database server may start out on a 2-way system with 4 GB of memory and six hard drives. As the database grows in size or the number of users increase, you can easily scale up by adding additional processors, memory, and disk resources to maintain the same level of performance.

The drawback to scaling up is that it may not provide linear scalability. As you add resources there is some overhead in resource management that limits the scalability of single systems. Scaling up may introduce a diminished return, as the system cost increases proportionally to the amount of resources, but does not provide an equally proportional return in performance.

In general, the DB2 Integrated Cluster Environment configurations are less prone to the diminishing return of scaling up because they are built with smaller building blocks (2-way and 4-way servers) as opposed to larger SMP machines.

Tip: If you choose to scale up, adding resources in a balanced manner (for example, adding processors, memory, storage, and I/O bandwidth) gives you the best chance for achieving linear scalability.

Horizontal scaling

Horizontal scaling, also known as “scale-out”, is the process of adding additional servers to an existing server or cluster. DB2 UDB’s shared-nothing architecture is well matched to the shared-nothing parallel model of today’s hardware architectures. Each server node has its own copy of the operating systems, memory resources, own I/O subsystem, storage resources, and so forth. Because clusters avoid the memory contention problems inherent in SMP systems, users are better able to expand power and improve performance by incrementally adding commodity-priced nodes to the cluster and balancing the workload among them. The result is a full node’s worth of performance for each added node. For example, in one of our lab environments we begin with two 2-way servers with 6 GB of memory and seven disk drives. We scaled-out using two additional, identical servers.

Although it is not a requirement that all servers in a cluster be identical, it does reduce the complexity of scaling out and load balancing if the servers are as close to identical as possible. The DB2 optimizer will take into account variances in system capabilities when it chooses plans, but keep in mind that you will strive to have your data evenly distributed across your database partitions. This will more evenly distribute your workload. If your database servers differ drastically in performance capacity, your “stronger” servers will be forced to wait for your “weaker” servers. If you are in this position, we recommend that you redistribute your workload using a customized partition scheme to balance the workload according to the server’s capabilities. This can be accomplished by using a using a DISTFILE during your data redistribution.

Note: The fact that the DB2 database is partitioned does not imply that the system cannot be implemented on shared storage. This is a common misconception of a shared-nothing architecture. In reality, multiple database partitions are often created on a single server or on multiple servers using a shared storage network (for example, ESS, SAN). This is done to more effectively leverage system resources, make scale-out easier, or provide for high availability.

Up or out - How about both

Scale direction does not have to be a binary choice. With a little bit of planing you can effectively use scale-up and scale-out together. For example, assume that you are building a system that was expecting to double in size after the first year and double again after the second year. Your implementation might look like this:

- This year - Start with two “building blocks”.
  - Two IBM eServer x365 servers
  - Two processors with 8 GB of memory each
  - Two database partitions per server
  - Appropriate amount of storage and I/O bandwidth for workload
- Next year - Scale up.
  - Upgrade to four processors and 16 GB of memory on both servers.
  - Add storage and I/O bandwidth.
  - Add two database partitions to each server and redistribute data via the REDISTRIBUTE command or one of the other methods described in 7.3, “Scaling out” on page 364.

Note: The addition of partitions in this step is not required. You may or may not add database partitions when you scale up. In this scenario we suggest it in this step to simplify our scale out in the following year.

- The following year - Scale out.
  - Add two building blocks.
  - Two IBM eServer x365 servers.
  - Four processors with 16 GB of memory each.
  - Adjust and balance storage and I/O bandwidth across all servers as required by workload.
  - Move two database partitions from each of the old servers to the new servers. A detailed example of this procedure is shown in 7.3.2, “Moving database partitions without data redistribution” on page 365. Briefly, the steps are:

    i. Stop DB2.
    ii. Unmount DB2 file systems from old servers and mount to new severs.
    iii. Update the db2nodes.cfg file.
    iv. Start DB2.

7.1.2 Scalability dimensions

Let us take a look at some examples of linear scalability:

- If a workload (for example, a query stream) executes in 10 seconds, a doubling of system resources will result in an execution time of five seconds for that same workload against that same amount of data.
- If a workload executes in 10 seconds, a doubling of system resources and a doubling in the amount of user data will maintain a 10-second execution time.
- If 20 concurrent streams of a workload execute in an average of 15 seconds, a doubling of system resources will maintain a 15-second execution time average under a 40 concurrent stream workload.
- If the average response time for 50 user transactions is one second, a tripling of system resources will support 150 users while maintaining a one-second response time.

Notice that all of the examples are defined using three common dimensions.

- Workload - The set of tasks (for example, queries, load jobs, transactions) to be executed. These are meant to mimic the expected load on the system.

- Scale factors - A combination of data size and/or system resource levels. In any scaling exercise, adjust either or both of these in order to maintain or increase performance.
- Performance metrics - The methods used to measure the performance of the workload tests.

These three dimensions help us to arrive data points at various scale levels. Scalability rates are simply the ratio of these data points. Let us look at each in further detail.

Workload

The key to success in predicting system performance under growth scenarios is an accurately modeled benchmark workload. All major benchmarks, including the Transaction Processing Council's (http://www.tpc.org) TPC-H and TPC-C, as well as the major ISV software benchmarks (PeopleSoft, SAP), utilize well-known standardized workloads in an attempt to facilitate "apples-to-apples" comparisons of systems. The same holds true for your own benchmark workloads. Your model should be comprised of a combination of:

- Batch processing (for example, data loads, data archival, reporting)
- Queries that are representative of those run by end users
- Queries that are representative of those run by applications
- Regularly scheduled database maintenance (for example, backup, reorg, runstats)

The workload should be developed with strong involvement from the application development and end-user interface teams who should provide the majority of the benchmark workload from these perspectives. The DBA team should be responsible for providing the benchmark workload with the database maintenance perspective.

There are a number of benefits to maintaining a defined benchmark workload. It:

- Allows for more accurate and clearly defined system testing
- Provides a reusable, clearly defined test suite for repeatable, predictable testing
- Provides a common frame of reference for interested parties (database administrators, application developers, and systems administrators)
- Provides a baseline so that tuning and scaling test comparisons are "apples-to-apples"

The workload should be periodically reviewed and tasks added or removed to ensure that the workload remains reflective of the current "real world" workload.

Additionally, the scalability tests should be compared to the actual results of the scalability observed in the production environment and adjustments made so that future tests remain predictive.

Scale factor

Scale factor can be thought of in two dimensions. The first dimension is database size. At face value this is a simple concept. Everyone can plainly see that a 2-TB database is twice the size of a 1-TB database. However, database size scale factors cannot be used to compare scalability between systems unless the workload remains constant. In other words, when we are concerned about scalability, 2 TB is only twice the size of 1 TB if the workload remains constant, making use of the same queries and processes.

The second dimension of scale factor is the system itself and the capabilities of the various subsystem components. Some examples are:

- Disk performance in MB/second transfer rate and I/Os/second
- Disk controller bandwidth
- Communications bandwidth
- CPU power and cache
- Memory

All of these factors must be taken into consideration when considering the scale factor between two systems. In order to do accurate scalability testing, each of these components must be scaled in proportion. For example, in order for system A to be declared 2x the scale of system B, it must have twice the amount of *each* of these components.

Consider the following scenario. Let us assume that we have reached 90 percent of capacity on our disk storage. Our CPU utilization is only at 30 percent, so we feel we have sufficient processing power for at least 2x growth. Our systems group upgrades us from 36-GB drives to the 72-GB models. We grow another 25 percent and performance begins to deteriorate. Our mistake was not examining each of the subsystem components. In doing so, we failed to realize that we had also reached transfer rate capacity of our disks. The new disks have the same transfer rates as the smaller disks. Although we have relieved a disk space capacity problem, we have created a disk throughput capacity problem.

Performance metrics

Although there are a number of standard performance metrics when we look at scalability, they generally measure some aspect of the same thing. In the world of databases, speed has always been king. Here we look at three common metrics used in benchmark tests.

Power

The term *power* metric is borrowed from the TPC benchmarks and represents a *single user stream* workload. This type of metric is a good choice when you are concerned with performance of a batch-type workload. The TPC formula uses a geometric mean ("nth" root of the product of "n" numbers) of the single user run times. The use of a geometric mean as opposed to an arithmetic mean (simple average) will serve to reduce the effect of outliers. Rather than go into a lot of mathematical theory, we suggest using the GEOMEAN formula in your favorite spreadsheet program (OpenOffice, Lotus 123, Microsoft Excel).

Throughput

The term *throughput* is used here to describe a *level of workload* that can be maintained or accomplished *over a period of time*. Transactional systems, such as the TPC-C OLTP benchmark, are often measured in transactions per second. The TPC-H benchmark uses queries per hour as one of its measurements. Throughput workloads usually consist of *parallel streams of workload*. It tests the system's ability to load balance under multi-user conditions.

A system with good scalability would be able to achieve the same amount of throughput (total queries processed in an hour) regardless if it was 1000 queries submitted by 10 users (100 queries each) or 1000 queries submitted by 100 users (10 queries each).

Response rime

A *response time* metric is generally used to measure the elapsed time to satisfy an end-user transaction or request. Response times are difficult to isolate because they are often impacted by the performance of the network. For this reason response time metrics are good candidates for use of a geometric mean. In order to isolate database performance from network performance it may be necessary to run response time tests on the database server in addition to across the network. Response time tests must also be scaled (run at various user levels) in order to simulate the expected number of users on the system.

Keys to developing good performance metrics:

- Figure out what you care about and find a way to measure it consistently.

  If you care about getting a workload complete every night in a short period of time, use a *power* metric against that workload. If you care about not having to pay people overtime to process insurance claims using your online system, use a response time metric. If you care about 100 power users doing ad-hoc analysis against your data warehouse, use a throughput metric. The point is to ensure that you are using a metric that represents the business demands placed upon your system.

- Measure what you intend to measure.

  For example, if you are interested in end-user response time at the edge of your network, do not rely solely on measurements captured at the database server. Or, if your users average 10 long-running complex queries per day, do not benchmark using simple queries that run in under a minute.
- Make the metrics meaningful to all interested parties.

  Involve the application development and end-user groups in developing your metrics. Leverage the expertise of your own systems groups.

7.2 Scaling up

Although DB2 UDB V8.1 Enterprise Server Edition can scale up to utilize all CPUs in a SMP environment, we find that most Linux customers do not exceed four CPUs per server. As stated in Chapter 2, “Selecting your cluster” on page 33, the cost of one 8-way server is substantially more expensive than two 4-way servers. In a Linux environment it is more cost effective to scale out than it is to scale up. However, it is common for customers to start off with a 2-way server and then scale up to a 4-way server or add more memory. This section discusses how to scale up your system on DB2 Integrated Cluster Environment.

7.2.1 Adding additional processors

Whether you started with a uniprocessor (x335, x345, e325 or HS20) or you started with the x365 2-way, you may reach a point in which you have maximized your CPU utilization. At this time you can add one or more CUPs to your existing server to help relieve the work load. Because of DB2’s shared-nothing architecture, you will essentially be doubling your CPU throughput. This section covers how to take advantage of the additional CPUs with DB2 UDB. A section has also been included to measure the performance impact the additional CPUs have had on your system.

7.2.2 Taking advantage of greater parallelism

The DB2 UDB optimizer will transparently take advantage of the additional CPUs in a number of different ways. The DB2 UDB query optimizer will make use of any additional CPUs and capacity that you give it. The degree of parallelism available for a system can speed up the performance of large complex queries in two main ways.

Inter-query parallelism

Inter-query parallelism allows multiple queries to run in parallel on a database system. This is one of the fundamental components of any database system. Increasing the number of CPUs available on the system will not change how DB2 UDB schedules the execution of the queries, since the distribution of CPU cycles is handled by the operating system. Your workload will, however, run faster since each query will have more CPU cycles assigned to it in a smaller time period.

If your workload was running poorly due to a CPU bottleneck, then additional CPUs or power will alleviate this problem. Throwing extra CPU power at a problem is not always the best solution. You should take a look at your database and determine if a few queries or transactions have been poorly designed and are using much more time and CPU power then they actually require. The performance of queries can be easily observed using a DB2 snapshot on dynamic SQL. The snapshot will give you a summary of all executions of the dynamic SQL queries on the system, with their user and system CPU usage displayed separately. The access plans of the longest running, and most active CPU using SQL queries should be examined to see if there is any indexing or issues.

Adjustments

With the addition of extra CPUs you should be able to squeeze more out of your system. If you were near the limit of your previous system then you may have to put restrictions on the amount of work, and applications, that could use the database. The list below summarizes the parameters that can be increased to allow a greater amount of inter-query parallelism and throughput.

- MAXAGENTS: Maximum number of agents
  Configuration parameter level: Instance

  The parameter controls the maximum number of database agents that can be running for a DB2 instance. This parameter is often used to cap the workload on a system due to CPU limitations. For large data warehousing systems that have a large number of queries that use multiple agents, putting a lower limit on this parameter can cap the resources used by DB2.

  If the number of CPUs is increased, then the parameter should be increased too in order to allow for DB2 to use more of the additional resources.

- MAX_CONNECTIONS: Maximum number of connections
  Configuration parameter level: Instance

  This parameter controls the maximum number of connections into all of the databases contained in the instance. This parameter is a very effective method to control the amount of work on the system if the CPU usage is constrained. The parameter should be increased as more CPU power is available.

- MAX_COORDAGENTS: Maximum number of coordinating agents
  Configuration parameter level: Instance

  Each connection to a database requires a coordinating agent. For queries that use intra-query parallelism, and the INTRA_PARALLEL parameter is set to YES, then the coordinating agent is also used to control the combining of information from all the agents that are processing a query or data set in parallel. By controlling the number of coordinating agents, you limit the number of queries that can be active at one time.
- MAXCAGENTS: Maximum number of concurrently executing coordinator agents
  Configuration parameter level: Instance

  This parameter was new for DB2 UDB V8.1 and is used by the new Connection Concentrator functionality. If this parameter is less then the MAX_COORDAGENTS, then only MAXCAGENTS will be created. A query requires that an coordinating agent service it for it to be active. The available coordinating agents will be allocated to the individual connections in a round robin-fashion, as they require to be actively run in the database engine. This reduces the number of agents that are spawned and the amount of memory required for the agents. The connection concentrator can be viewed as a connection multiplexer.

  This parameter will also have to be increased to allow DB2 to fully utilize the extra CPU power of a machine if the parameter had previously been used to control the amount of system resources used by DB2.

Intra-query parallelism

Intra-query parallelism is when DB2 takes a complex query and breaks it into sub-components that can be run separately, and then brings the results together to complete the query. This query optimization is automatically used by the optimizer, but having additional CPUs available could allow DB2 to consider more effective query execution plans.

Example 7-1 shows a sample query that will be able to take advantage of DB2's intra-query parallelism feature. DB2 could have one agent query the population table and have another agent query the bankrecords table in parallel. The coordinating agent would then bring the results of these two queries together and complete the query.

Example 7-1   Query that could make use of DB2 intra-query parallelism

```
select pop.lastName, bnk.bankAccount
from
   (select idNumber,lastName
    from population
    where city = 'Toronto'
   ) as pop,
   (select idNumber, bankAccount
    from bankrecords
    where accountBal > 1000000
   ) as bnk
where pop.idNumber = bnk.idNumber
```

Adjustments

The parameters listed below will help better utilize the available CPUs for intra-query parallelism.

- MAXAGENTS: Maximum number of agents
  Configuration parameter level: Instance

  If the maximum number of agents is kept low, then DB2 may not have the resources to fully parallel the query. Increasing this value will allow more agents to be available for the instance.

- MAXAPPLS: Maximum number of applications on the system
  Configuration parameter level: Database

  The maximum number of applications is controlled at the database level. This control is similar to the maximum number of connections set at the instance level. If you used this to control the maximum amount of activity on your database then you should increase the parameter as more CPUs are added.

Intra-partition parallelism

Intra-partition parallelism is when DB2 has multiple agents execute the same query on different sections of the data on the same partition (Figure 7-2 on page 361). This is useful when you are working with very large tables and it would take too long for one agent to scan the entire table.

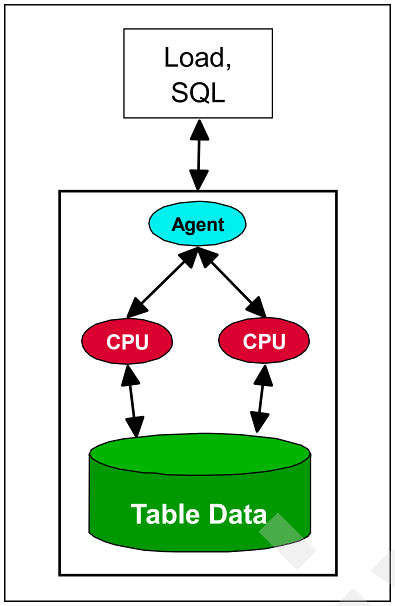


Figure 7-2 Intra-partition parallelism

Adjustments

The parameters listed below help better utilize the available CPUs for intra-query parallelism.

- INTRA_PARALLEL: Intra-parallel parallelism
  Configuration parameter level: Instance

  The default for intra-partition parallelism is to have it set to NO. This is good for OLTP systems since the extra work required to coordinate running a query in parallel and the extra compilation time are often worse than the performance gained for short queries. For DSS and warehouse systems, however, this parameter can cause a dramatic improvement. The improvement comes at a CPU cost since extra database agents are used to process the query in parallel. If the CPU of your system is being used too heavily, then turning this parameter off will reduce the workload, but will also slow down your query processing. The more complex queries could end up greatly increasing the amount of CPU usage since they could be highly parallelized by DB2.

- MAX_QUERYDEGREE: Maximum degree of parallelism
  Configuration parameter level: Instance

  The maximum degree of parallelism controls the maximum degree of intra-partition parallelism. This is a good way to throttle how heavily the query is run in parallel and therefore how many agents are run. As you increase the number of CPUs on your system, you should scale this value up as well if there will be extra processing capacity available.

- NUM_QUANTILES: Number of quantiles
Configuration parameter level: Database

  The number of quantiles determines how many quantiles are captured when the runstats statistics tool is run. Having a greater number of quantiles will give DB2 a better idea of how many agents are needed to effectively scan a table or index range and the agents will then be better allocated.

7.2.3 Adding additional memory

A lack of memory can be a bottlenecks for an application that uses databases heavily. This is especially true with sort and aggregate intensive applications and applications that support large numbers of concurrent users. With Linux, the 32-bit operating system addressability limited the amount of memory available to a database partition to around 1.75 GB. DB2 UDB can utilize up to approximately 2.5 gigabytes of memory on certain Linux distributions by changing the way DB2 loads its shared libraries. As of DB2 UDB V8.2, DB2 automatically does this to take advantage of more memory.

With the availability of 64-bit processors and development of Linux 64-bit operating systems such memory limits are no longer a concern; however, attention must still be paid to tuning memory, regardless of size.

Physically adding more memory

Physically adding more memory to your system is a simple process. Memory is generally added when the system is down, but some modern systems also allow for hot-swappable memory. DB2 can then be configured to take advantage of the greater amount of memory on the system.

Prior to performing a memory upgrade, you should analyze the memory usage of your system under its current workload. Some tools available to you are listed below. For a complete overview of the DB2 Memory Visualizer please see 5.2.1, “DB2 instruments” on page 269, and 5.2.2, “Linux tools” on page 296.

- DB2 Memory Visualizer.
- DB2 Snapshots.
- DB2 Snapshot table functions.
- **`db2mtrk`** command.

- The Linux **ps** command can provide information on the amount of memory processes are using. For example, the following command will list the all processes on the system and sort by memory used:

  ```
  ps -eo "pid,ppid,ruser,rgroup,vsz=VIRTUAL,args"|sort -k 5 -n -r
  ```

- Other useful Linux commands:
  - vmstat
  - sar
  - top
- New in DB2 V8.2 is the **db2bp** command, which can give information about memory usage by the different memory pools within DB2.
- The DB2 Configuration Advisor is a tool that will calculate suggested configuration parameters. The Configuration Advisor is launched from the DB2 Control Center by right-clicking the database and selecting **Configuration Advisor**. The advisor will ask you a series of questions about your environment, gather system information, and make configuration parameter suggestions.

  The Configuration Advisor is also available in command line mode via the AUTOCONFIGURE command. An example is shown below.

Example 7-2 DB2 AUTOCONFIGURE

```
AUTOCONFIGURE USING
  mem_percent     85
  workload_type   mixed
  num_stmts       1
  tpm             60
  admin_priority  performance
  is_populated    yes
  num_local_apps  0
  num_remote_apps 10
  isolation       CS
  bp_resizeable   yes
APPLY NONE

Current and Recommended Values for Database Configuration

 Description Parameter   Current Value       Recommended Value
 ---------------------------------------------------------------------------------------
 Database heap (4KB)                            (DBHEAP) = 10000             3159
 Number of primary log files                (LOGPRIMARY) = 4                 3
 Number of secondary log files               (LOGSECOND) = 1                 0
 Max number of active applications            (MAXAPPLS) = 80                40
 Percent. of lock lists per application       (MAXLOCKS) = 20                60
 Sort list heap (4KB)                         (SORTHEAP) = 2000              3974
 SQL statement heap (4KB)                     (STMTHEAP) = 10000             4096
Statistics heap size (4KB)               (STAT_HEAP_SZ) = 4384             4384
 Utilities heap size (4KB)                (UTIL_HEAP_SZ) = 5000              279910
```

```
      Current and Recommended Values for Bufferpool(s)

Description                                 Parameter   Current Value       Recommended Value
-------------------------------------------------------------------------------------------------
BP32K                                Bufferpool size = 44274               93069
IBMDEFAULTBP                         Bufferpool size = 44274               93069
```

7.3 Scaling out

To scale out on DB2 UDB we add servers to an existing DB2 system of one or more servers. In this section we look at the methods available to scale DB2 out to additional servers for increased performance.

In order to leverage the additional resources of the new servers, we first add them to the cluster from the system point of view and then make DB2 aware of them. With DB2's shared-nothing architecture, the new servers will take control over a portion of the data. To accomplish this we can move existing database partitions to the new servers. DB2 automatically redistributes the data or moves the data to the new partitions manually.

When a database environment is migrated from a single server to a multi-server partitioned environment, the database design should be reviewed carefully in order to define which groups of tables should be partitioned and how many partitions should exist per table or group of tables.

Adding servers and partitions will result in increased internal communication and task coordination work for the database. However, this choice provides the advantage of balancing data and user access across more than one system.

You can add partitions while the database manager system is running or while it is stopped. If you add partitions while the system is running, you must stop and restart teh system before the new partitions become active.

When you scale your system by changing the environment, you should be aware of the impact that such a change can have on your database procedures, such as loading data, backing up the database, and restoring the database.

DB2 UDB applies the concept of function shipping to improve performance on partitioned environments, which reduces network traffic. Whenever possible, DB2 will ship processes, such as SQL queries, rather than data. During the query optimization process DB2 analyzes the SQL and sends all or part of the it to the partition where the data resides. Relational operators are executed on the node partition containing the data whenever possible. Function shipping is well suited to the shared-nothing architecture model.

The following sections outline the options available and describe the processes to accomplish them.

- Preparing the new servers
- Moving database partitions without data redistribution
- Adding database partitions and redistributing data
  - Data redistribution via the REDISTRIBUTE command
  - Data redistribution via INSERT/SELECT
  - Data redistribution via UNLOAD/LOAD

Important: Although DB2's database partitioning capability is built into the DB2 UDB Enterprise Server Edition (ESE) engine, the use of this feature requires that you be licensed for DB2's Data Partitioning Feature (DPF). If you have questions about using DPF please contact your local IBM software sales representative, your IBM reseller, or IBM Software Sales at 888-SHOP-IBM.

7.3.1 Preparing the new servers

In order to prepare your newly added server to be a member of your DB2 cluster, you will need to install DB2 on the server and add it to the DB2 instance. The first step in this process is to install DB2 on the newly added servers.

The preferred method to accomplish this is to use the **`db2setup`** command with a response file generated on the server on which you first installed DB2.

If you do not have a response file, you can generate one by running **`db2setup`** on your first DB2 server. Proceed as if you were going to install DB2, but on the "Installation action" panel deselect the "Install DB2 UDB Enterprise Server Edition on this computer" check box and select the "Save your settings in a response file" check box. On the Summary panel verify that the space required is 0 MB and save your response file.

A final alternative is to install DB2 via the **`db2seup`** GUI installation from the source media. The only problem with this method is that you may end up with different components installed on your servers.

For a complete overview of the following steps please refer to 4.3, "DB2 partitioned database setup" on page 153.

7.3.2 Moving database partitions without data redistribution

This option is available if you have multiple partitions on your existing servers and have planned ahead with regard to your file systems and naming conventions. The advantage of this method is that it requires no data redistribution and very little system down time. For example, assume you have two 2-way servers and have defined two data partitions on each. You can easily move one data partition off of each old server to a new server. The process involves the following six steps:

1. Stop the DB2 instance.
2. Unmount the file systems associated with the partitions you want to move from the old servers.
3. Mount those file systems on their new servers.
4. Move the database directories (if necessary).
5. Update the *db2nodes.cfg* file to reflect the new location of the moved partitions.
6. Start the DB2 instance.

The disadvantage of this method is that you will most likely be able use it only one time. With this method the number of database partitions does not change, they are simply moved. It is unlikely that you will have enough partitions in your original configuration to "feed" multiple iterations of this approach and maintain a consistent ration between CPU power and data. This is especially true in the typical Integrated Cluster Environment "small-SMP" configurations. With DB2 Integrated Cluster Environment you would likely start with no more than four partitions per server

The advantage of this method is that it is very fast. It can be accomplished in a couple of minutes or less.

Below we demonstrate this approach in our lab environment. The lab hardware configuration is described in 4.1, "Lab environment" on page 126. Configuration 3 is used to demonstrate the scaling-out process. We started with two database servers, db2rb09 and db2rb10, in the cluster. Each has two logical database partitions. We then added two database servers, db2rb11 and db2rb12, to the cluster. In this example we move two of our logical data partitions to their own servers. We move the following partitions:

- Partition 2 will be moved from db2rb09 to db2rb11.
- Partition 3 will be moved from db2rb10 to db2rb12.

In our initial configuration we create two database partitions on each of our 2-way e325 servers, as shown in Table 7-1 on page 367.

Table 7-1 Original partition layout

Partition number	Server	Comments
0	db2rb09	Two logical partitions
2	db2rb09	
1	db2rb10	Two logical partitions
3	db2rb10	

Table 7-2 and Table 7-3 show the original file systems on server db2rb09 and db2rb10.

Table 7-2 Original file systems for server DB2RB09

File system	Mount Point	Purpose
/dev/md1	/db2home	Instance home (shared across both servers via NFS)
/dev/md2	/db2db	Database directory
/dev/md0	/db2fs0p0	Partition 0 - Catalog
/dev/md4	/db2fs1p0	Partition 0 - Data
/dev/md5	/db2fs2p0	Partition 0 - Temp
/dev/md3	/db2log1p0	Partition 0 - Logs
/dev/md12	/db2fs1p2	Partition 2 - Data
/dev/md13	/db2fs2p2	Partition 2 - Temp
/dev/md11	/db2log1p2	Partition 2 - Logs

Table 7-3 Original file systems for server DB2RB10

File system	Mount Point	Purpose
db2rb09: /db2home	/db2home	Instance home (shared across both servers via NFS)
/dev/md6	/db2db	Database directory
/dev/md4	/db2fs1p1	Partition 1 - Data
/dev/md5	/db2fs2p1	Partition 1 - Temp
/dev/md3	/db2log1p1	Partition 1 - Logs
/dev/md16	/db2fs1p3	Partition 3- Data

File system	Mount Point	Purpose
/dev/md17	/db2fs2p3	Partition 3- Temp
/dev/md15	/db2log1p3	Partition 3- Logs

Step 1 - Stop the DB2 instance

Issue the **db2stop** command to stop the database instance (Example 7-3).

Example 7-3 Stop the DB2 instance

```
db2inst2@db2rb09:/> db2stop
07/27/2004 12:15:14     0   0   SQL1064N  DB2STOP processing was successful.
07/27/2004 12:15:14     2   0   SQL1064N  DB2STOP processing was successful.
07/27/2004 12:15:15     3   0   SQL1064N  DB2STOP processing was successful.
07/27/2004 12:15:17     1   0   SQL1064N  DB2STOP processing was successful.
SQL1064N  DB2STOP processing was successful.
```

Step 2 - Unmount the file systems from the original servers

Example 7-4 has the unmount commands we used to unmount file systems in db2rb09 and db2rb10.

Example 7-4 Unmount file systems from db2rb09 and db2rb10

```
db2rb09:~ # umount /db2log1p2
db2rb09:~ # umount /db2fs1p2
db2rb09:~ # umount /db2fs2p2

db2rb10:~ # umount /db2log1p3
db2rb10:~ # umount /db2fs1p3
db2rb10:~ # umount /db2fs2p3
```

Step 3 - Mount those file systems on their new servers

Example 7-5 shows how to mount the file system to new servers.

Example 7-5 Mount file system on the new servers (db2rb11 and db2rb12)

```
db2rb11:~ # mount /dev/md11 /db2log1p2
db2rb11:~ # mount /dev/md12 /db2fs1p2
db2rb11:~ # mount /dev/md13 /db2fs2p2

db2rb12:~ # mount /dev/md15 /db2log1p3
db2rb12:~ # mount /dev/md16 /db2fs1p3
db2rb12:~ # mount /dev/md17 /db2fs2p3
```

Step 4 - Move the database directories (if necessary)

This step would only be necessary if you have a single mount point that contains subdirectories for multiple database directories.

In a non-HA environment (for example, data warehouse, development, or test systems), you might have created multiple instances and/or databases on your original servers and shared a single file system for them. If this is the case, you probably mounted your database directory at a higher level (for example, /db2db) and created subdirectories for you database directories (for example, /db2db/db2ins1/mydb). In this scenario, you would move the database directories to their new servers.

In an HA environment, you would most likely create a different mount point and file system for each of your database partition's database directories. In the event of a failover, the backup server would simply mount those file systems and take control of the partition.

Since the non-HA environment involves this extra step, it is shown below. We do this using the secure copy (scp) program, as shown in Example 7-6.

Example 7-6 Copy DB2 database directories from original servers to new servers

```
db2rb09:~ # scp -pr /db2db/db2inst2/NODE0002 db2rb11:/db2db/db2inst2/
db2rb10:~ # scp -rp /db2db/db2inst2/NODE0003 db2rb12:/db2db/db2inst2/
```

Step 5 - Update the db2nodes.cfg file

In this step we updated the *db2nodes.cfg* file to indicate the new locations of our partitions. The db2nodes.cfg file resides in the ~/sqllib of the instance home directory. For example, on our system this file is in /db2home/db2inst2/sqllib/db2nodes.cfg.

We made the following changes:

- Update the host name (column two) to indicate the server name (as defined in our /etc/hosts file) to indicate the new server names.
- Update the logical port (column three) to indicate that each server now has only one logical partition.

Example 7-7 shows the original db2nodes.cfg file. The updated content is shown in Example 7-8 on page 370.

Example 7-7 Original db2nodes.cfg file

```
0 db2rb09 0
1 db2rb10 0
2 db2rb09 1
```

```
3 db2rb10 1
```

Example 7-8 Updated db2nodes.cfg file

```
0 db2rb09 0
1 db2rb10 0
2 db2rb11 0
3 db2rb12 0
```

Step 6 - Start the DB2 instance and verify the change

Once the DB2 instance is started with the **db2start** command, you can verify your change via a SELECT across all data partitions. See Example 7-9.

Example 7-9 Verify moving database partition

```
db2inst2@db2rb09:~> db2start
07/27/2004 15:10:10     0   0   SQL1063N  DB2START processing was successful.
07/27/2004 15:10:10     1   0   SQL1063N  DB2START processing was successful.
07/27/2004 15:10:10     3   0   SQL1063N  DB2START processing was successful.
07/27/2004 15:10:10     2   0   SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.

db2inst2@db2rb09:~> db2 "select count(*) from tpch.supplier"
1
-----------
400000

  1 record(s) selected.

db2inst2@db2rb09:~>
```

The new configuration

In our new configuration we have one database partition on each of our 2-way e325 servers, as shown in Table 7-4.

Table 7-4 New partition layout

Partition number	Server	Comments
0	db2rb09	One logical partition
1	db2rb10	One logical partition
2	db2rb11	One logical partition
3	db2rb12	One logical partition

Example 7-5, Example 7-6, Example 7-7, and Example 7-8 on page 372 show the new layout of the file systems in each database servers.

Table 7-5 Updated file systems for server DB2RB09

File system	Mount point	Purpose
/dev/md1	/db2home	Instance home (shared across all servers via NFS)
/dev/md2	/db2db	Database directory
/dev/md0	/db2fs0p0	Partition 0 - Catalog
/dev/md4	/db2fs1p0	Partition 0 - Data
/dev/md5	/db2fs2p0	Partition 0 - Temp
/dev/md3	/db2log1p0	Partition 0 - Logs

Table 7-6 Updated file systems for server DB2RB10

File system	Mount point	Purpose
db2rb09: /db2home	/db2home	Instance home (shared across both servers via NFS)
/dev/md6	/db2db	Database directory
/dev/md4	/db2fs1p1	Partition 1 - Data
/dev/md5	/db2fs2p1	Partition 1 - Temp
/dev/md3	/db2log1p1	Partition 1 - Logs

Table 7-7 File systems for new server DB2RB11

File system	Mount point	Purpose
db2rb09: /db2home	/db2home	Instance home (shared across both servers via NFS)
/dev/md10	/db2db	Database directory
/dev/md12	/db2fs1p2	Partition 2 - Data
/dev/md13	/db2fs2p2	Partition 2 - Temp
/dev/md11	/db2log1p2	Partition 2 - Logs

Table 7-8 File systems for new server DB2RB12

File system	Mount point	Purpose
db2rb09: /db2home	/db2home	Instance home (shared across both servers via NFS)
/dev/md14	/db2db	Database directory
/dev/md16	/db2fs1p3	Partition 3- Data
/dev/md17	/db2fs2p3	Partition 3- Temp
/dev/md15	/db2log1p3	Partition 3- Logs

7.3.3 Adding database partitions

Another way to scale out is to add database partitions to the new server(s). You can add partitions while the database manager system is running or while it is stopped. If you add partitions while the system is running, you must stop and restart the system before the new partitions become active.

Adding partitions

Once DB2 is installed on the new server you can add database partitions to your instance. To add partitions you can use the Add Partitions wizard, which is accessed from the DB2 Control Center. The procedure is outlined in the steps below.

Step 1 - Launch the Add Partitions wizard

From the DB2 Control Center:

1. Right-click the DB2 instance or left click the DB2 instance and the choose **Selected** from the menu.
2. Select **Add Partitions** from the menu, as shown in Figure 7-3 on page 373.

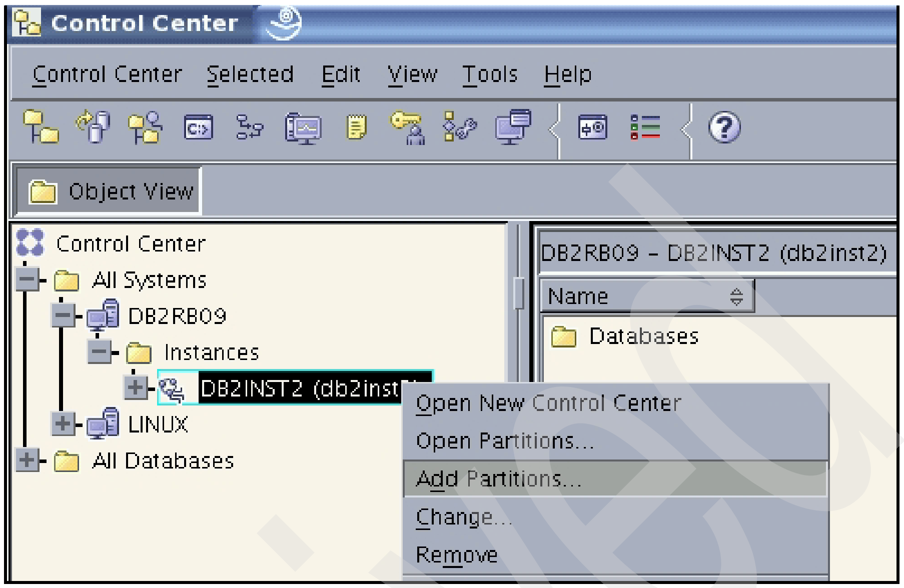
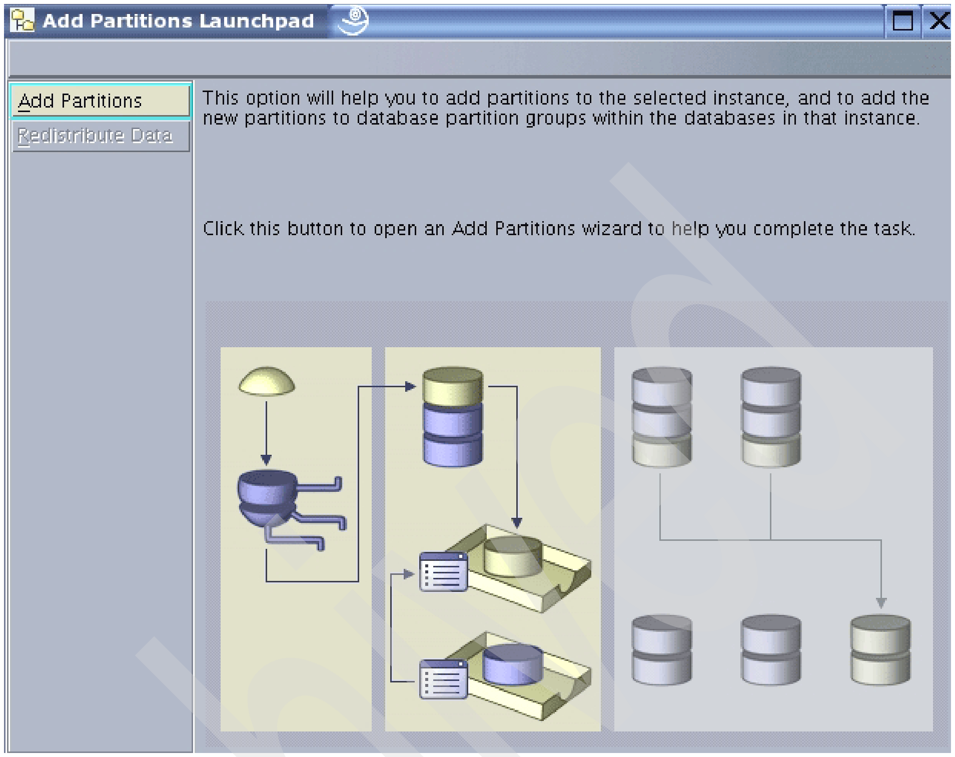


Figure 7-3 Adding a partition to the DB2 instance

You will be presented with the Add Partitions Launchpad, as shown in Figure 7-4 on page 374.


Add Partitions Launchpad

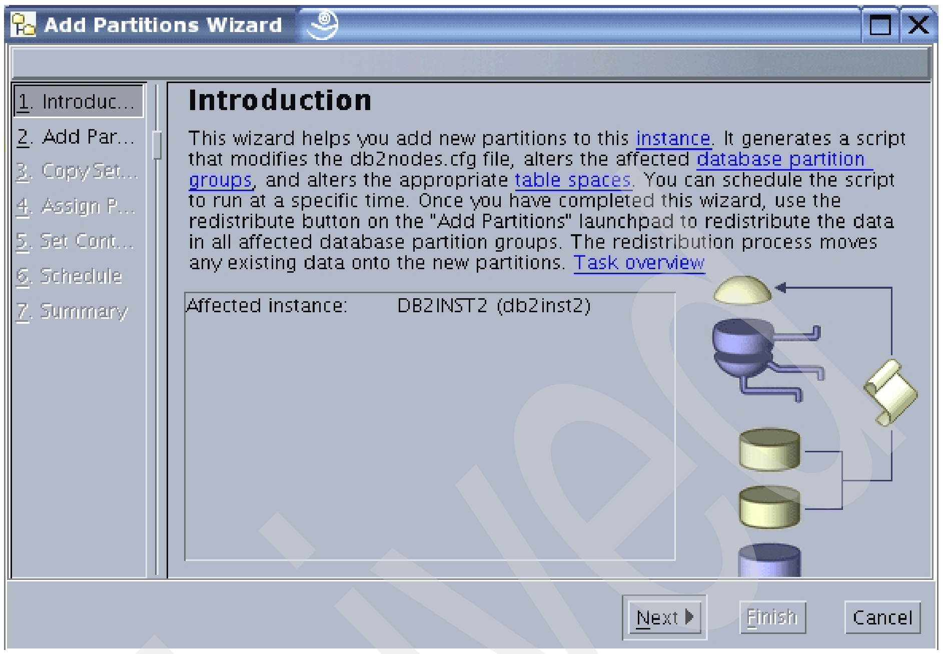
This option will help you to add partitions to the selected instance, and to add the new partitions to database partition groups within the databases in that instance.

Click this button to open an Add Partitions wizard to help you complete the task.

Figure 7-4 Add Partitions Launchpad

Step 2 - Add Partitions Launchpad

From the Add Partitions Launchpad, click the **Add Partitions** button. You will be presented with the Add Partitions Wizard, as shown on tFigure 7-5 on page 375.



Introduction

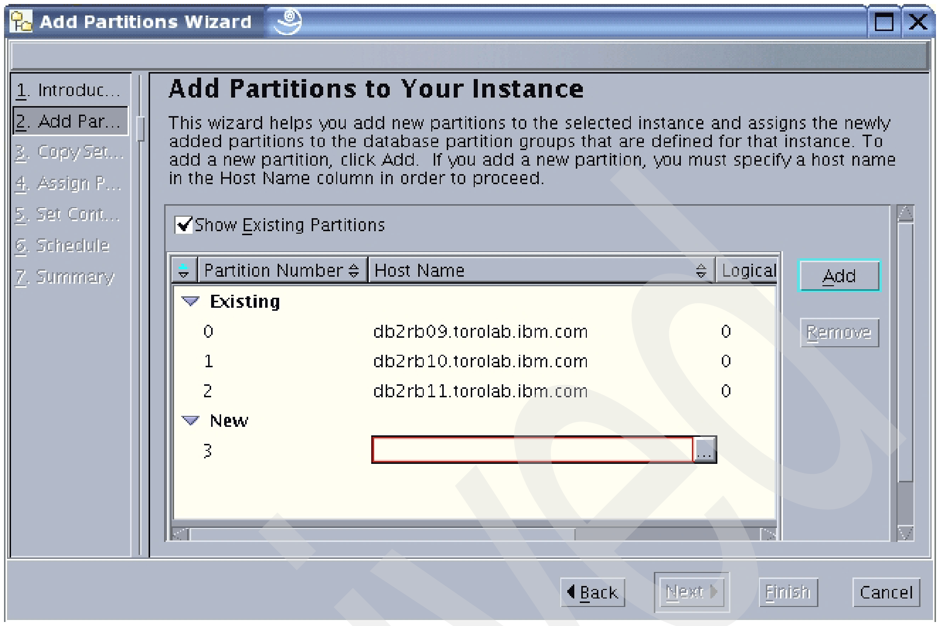
This wizard helps you add new partitions to this instance. It generates a script that modifies the db2nodes.cfg file, alters the affected database partition groups, and alters the appropriate table spaces. You can schedule the script to run at a specific time. Once you have completed this wizard, use the redistribute button on the "Add Partitions" launchpad to redistribute the data in all affected database partition groups. The redistribution process moves any existing data onto the new partitions. Task overview

Affected instance: DB2INST2 (db2inst2)

Figure 7-5 Add Partitions Wizard introduction

Step 3 - Add Partitions Introduction

From the Add Partitions Wizard Introduction panel, click **Next**. the Add Partitions to Your Instance dialog shown (Figure 7-6 on page 376).


Add Partitions Wizard

1. Introduc...
2. Add Par...
3. Copy Set...
4. Assign P...
5. Set Cont...
6. Schedule
7. Summary

Add Partitions to Your Instance

This wizard helps you add new partitions to the selected instance and assigns the newly added partitions to the database partition groups that are defined for that instance. To add a new partition, click Add. If you add a new partition, you must specify a host name in the Host Name column in order to proceed.

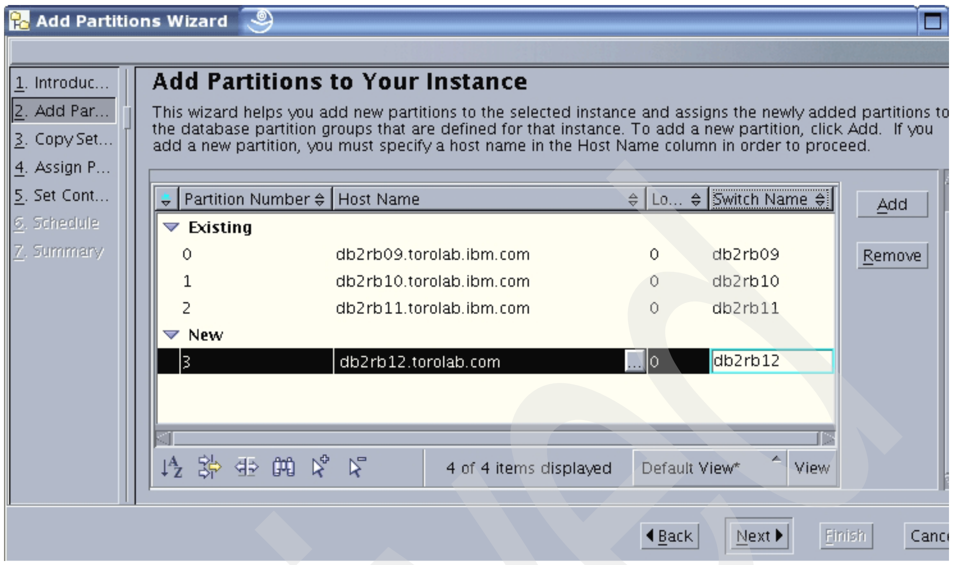
☑ Show Existing Partitions

Partition Number	Host Name	Logical
Existing		
0	db2rb09.torolab.ibm.com	0
1	db2rb10.torolab.ibm.com	0
2	db2rb11.torolab.ibm.com	0
New		
3		

Figure 7-6 Add Partitions Wizard - Add Partitions panel

Step 4 - Add a database partition

From Add Partitions, click the **Add** button. You will be presented with an empty New partition entry, as shown in Figure 7-7 on page 377.



Add Partitions to Your Instance

This wizard helps you add new partitions to the selected instance and assigns the newly added partitions to the database partition groups that are defined for that instance. To add a new partition, click Add. If you add a new partition, you must specify a host name in the Host Name column in order to proceed.

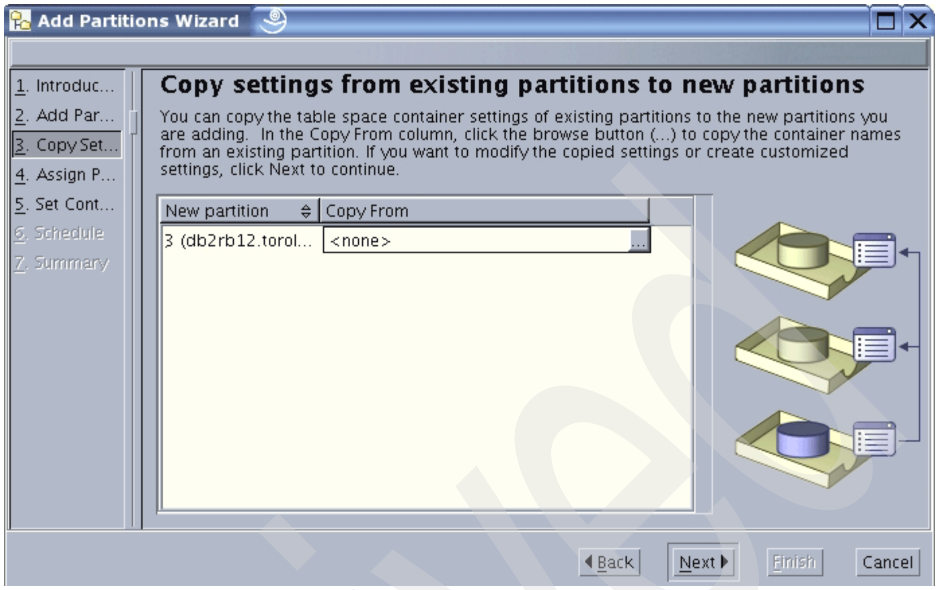
Partition Number	Host Name	Lo...	Switch Name
Existing			
0	db2rb09.torolab.ibm.com	0	db2rb09
1	db2rb10.torolab.ibm.com	0	db2rb10
2	db2rb11.torolab.ibm.com	0	db2rb11
New			
3	db2rb12.torolab.com	0	db2rb12

Figure 7-7 Add Partitions Wizard - Enter server information for new partition

Step 5 - Enter host information

From the Add Partitions panel:

1. Enter the host name and switch name in the New partition entry. If you are not using a high-speed switch, set the switch name to the server name in your /etc/hosts file.
2. If you are creating a second database partition on the server, increment the Logical Partition entry. On each physical server the database partitions are numbered 0,1, 2, etc. See “Updating the node configuration file” in the *Quick Beginnings for DB2 Servers,* GC09-4836, manual for more information.
3. Click **Next**. You will be presented with the Copy Settings panel (Figure 7-8 on page 378).


Add Partitions Wizard

1. Introduc...
2. Add Par...
3. Copy Set...
4. Assign P...
5. Set Cont...
6. Schedule
7. Summary

Copy settings from existing partitions to new partitions

You can copy the table space container settings of existing partitions to the new partitions you are adding. In the Copy From column, click the browse button (...) to copy the container names from an existing partition. If you want to modify the copied settings or create customized settings, click Next to continue.

New partition	Copy From
3 (db2rb12.torol...	<none>

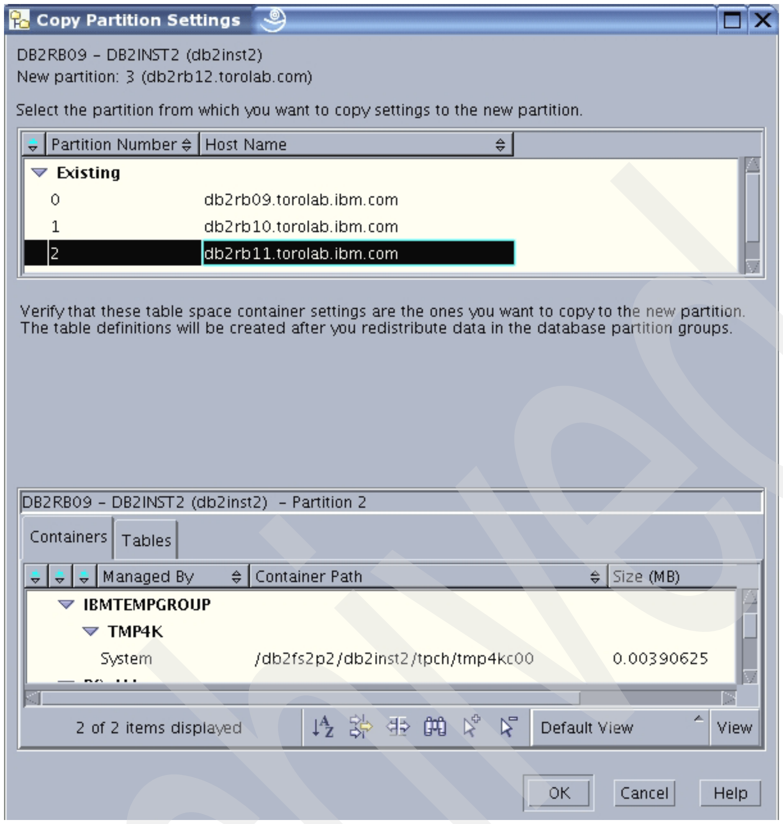
Back · Next · Finish · Cancel

Figure 7-8 Add Partitions Wizard - Copy settings panel

Step 6 - Copy partition settings from existing partition

From the Copy Settings panel:

1. Click the ... button in the Copy From column.
2. Click **Next**.
3. You will be presented with the Copy partition settings panel, as shown in Figure 7-9 on page 379.


Copy Partition Settings

DB2RB09 - DB2INST2 (db2inst2)
New partition: 3 (db2rb12.torolab.com)

Select the partition from which you want to copy settings to the new partition.

Partition Number	Host Name
Existing	
0	db2rb09.torolab.ibm.com
1	db2rb10.torolab.ibm.com
2	db2rb11.torolab.ibm.com

Verify that these table space container settings are the ones you want to copy to the new partition.
The table definitions will be created after you redistribute data in the database partition groups.

DB2RB09 - DB2INST2 (db2inst2) - Partition 2

Managed By	Container Path	Size (MB)
IBMTEMPGROUP		
TMP4K		
System	/db2fs2p2/db2inst2/tpch/tmp4kc00	0.00390625

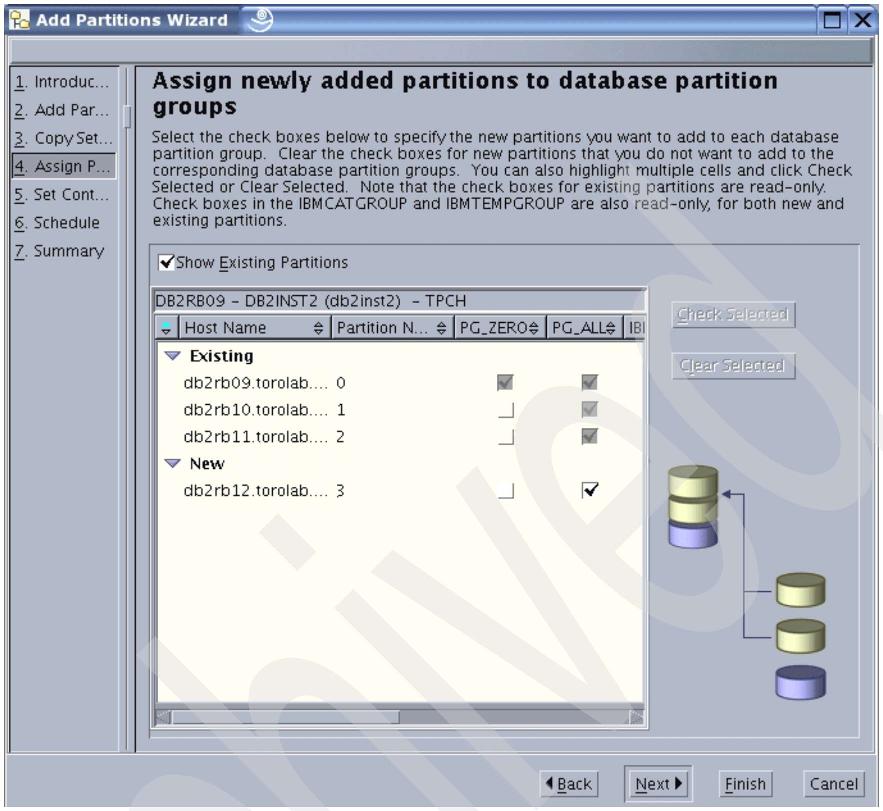
2 of 2 items displayed

Figure 7-9 Add Partitions Wizard - Copy Partition Settings

Step 7 - Select a partition to copy

From the Copy Partition Settings panel:

1. In the top window of the panel, select the partition you wish to use as a model for your new partition.
2. After a moment the partition group and tablespace information will appear in the lower window.
3. You can select and view any of the partitions. When you are satisfied, click **OK**.
4. You will be presented with the Assign Partition Groups panel, as shown on Figure 7-10 on page 380.


Add Partitions Wizard

1. Introduc...
2. Add Par...
3. Copy Set...
4. Assign P...
5. Set Cont...
6. Schedule
7. Summary

Assign newly added partitions to database partition groups

Select the check boxes below to specify the new partitions you want to add to each database partition group. Clear the check boxes for new partitions that you do not want to add to the corresponding database partition groups. You can also highlight multiple cells and click Check Selected or Clear Selected. Note that the check boxes for existing partitions are read-only. Check boxes in the IBMCATGROUP and IBMTEMPGROUP are also read-only, for both new and existing partitions.

✓ Show Existing Partitions

DB2RB09 - DB2INST2 (db2inst2) - TPCH

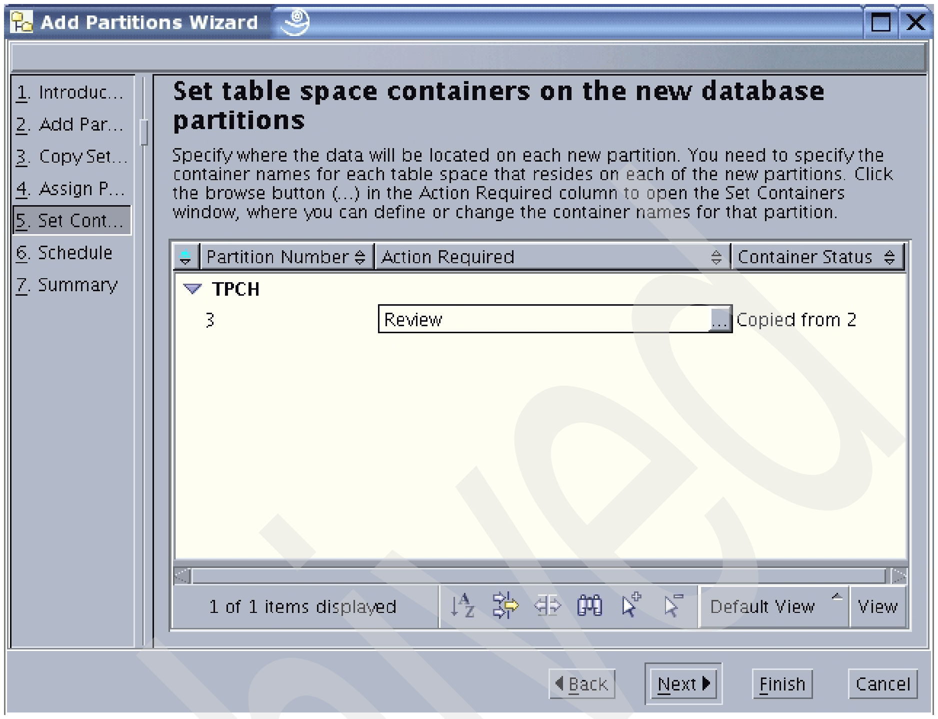
Host Name	Partition N...	PG_ZERO	PG_ALL
Existing			
db2rb09.torolab....	0	✓	✓
db2rb10.torolab....	1		✓
db2rb11.torolab....	2		✓
New			
db2rb12.torolab....	3		✓

Figure 7-10 Add Partitions Wizard - Assign partition groups

Step 8 - Include new partition in existing partitions groups

From the Assign Partitions panel:

1. Scroll across the panel and check the boxes associated with the database partition groups that should include your new partition. The existing partitions and partitions groups are listed for your reference.
2. Click **Next**. You will be presented with the Set Containers panel, as shown in Figure 7-11 on page 381.


Add Partitions Wizard

1. Introduc...
2. Add Par...
3. Copy Set...
4. Assign P...
5. Set Cont...
6. Schedule
7. Summary

Set table space containers on the new database partitions

Specify where the data will be located on each new partition. You need to specify the container names for each table space that resides on each of the new partitions. Click the browse button (...) in the Action Required column to open the Set Containers window, where you can define or change the container names for that partition.

Partition Number	Action Required	Container Status
TPCH		
3	Review	Copied from 2

1 of 1 items displayed

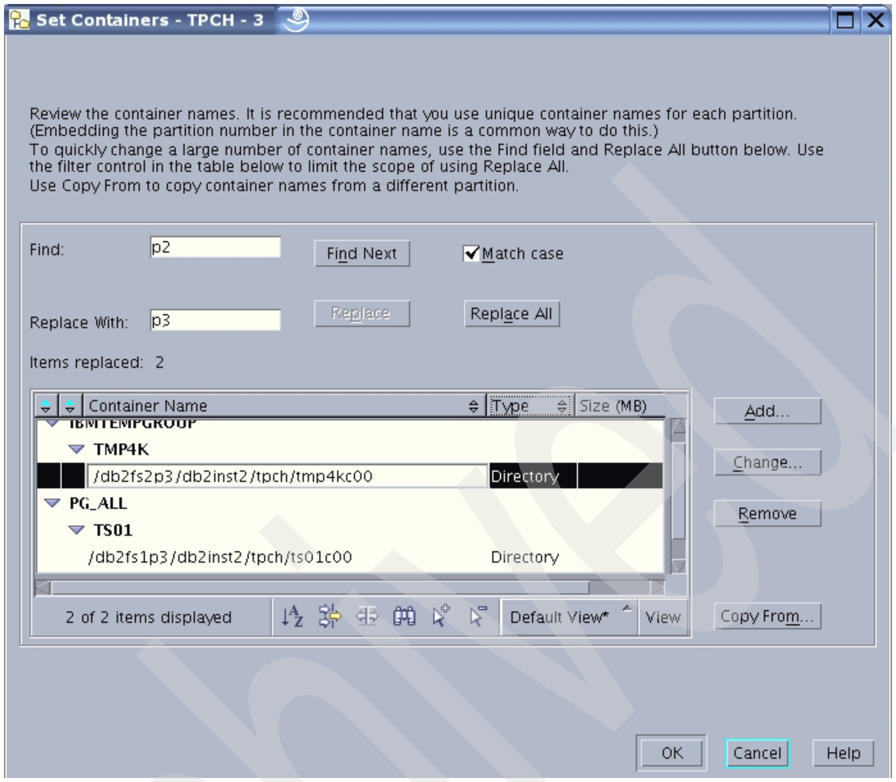
Figure 7-11 Add Partitions Wizard - Set containers for new partition

Step 9 - Review container paths for new data partition

The previous step used the partition you selected as the model for the container paths (that is, the files, directories, or devices used by the table space containers in that partition) for you new partition. You need to review and update those paths as appropriate for the new partition.

From the Set Containers panel:

1. You will note that the Action Required column indicates that you need to Review the container paths that were copied in the previous step.
2. Click the ... button to the right of Review, then click **Next**. You will be presented with the Set Containers dialog, as shown on Figure 7-12 on page 382.


Set Containers - TPCH - 3

Review the container names. It is recommended that you use unique container names for each partition. (Embedding the partition number in the container name is a common way to do this.)
To quickly change a large number of container names, use the Find field and Replace All button below. Use the filter control in the table below to limit the scope of using Replace All.
Use Copy From to copy container names from a different partition.

Find: p2

Replace With: p3

Match case: ✓

Items replaced: 2

Container Name	Type	Size (MB)
IBMTEMPGROUP		
TMP4K		
/db2fs2p3/db2inst2/tpch/tmp4kc00	Directory	
PG_ALL		
TS01		
/db2fs1p3/db2inst2/tpch/ts01c00	Directory	

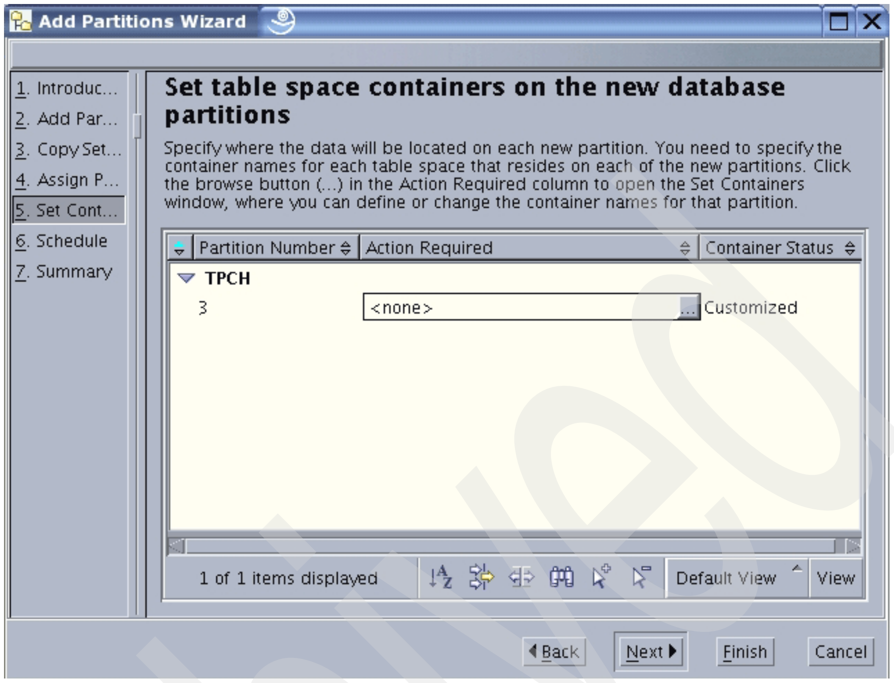
2 of 2 items displayed

Figure 7-12 Add Partitions Wizard - Set Containers

Step 10 - Review and update container paths for new data partition

From the Set Containers dialog box:

1. You can use the File Browse feature to add or change the paths for the containers or edit them directly.
2. If you have used a good naming convention, the easiest way will be to simply use the Edit/Replace feature of the dialog box. For example, we did a global change of all occurrences of p2 to p3.
3. Repeat the process as needed for each table space container.
4. When you are satisfied, click **OK**. You will be returned to the previous panel, as shown on Figure 7-13 on page 383.


Add Partitions Wizard

Set table space containers on the new database partitions

Specify where the data will be located on each new partition. You need to specify the container names for each table space that resides on each of the new partitions. Click the browse button (...) in the Action Required column to open the Set Containers window, where you can define or change the container names for that partition.

Partition Number	Action Required	Container Status
TPCH		
3	<none>	Customized

1 of 1 items displayed

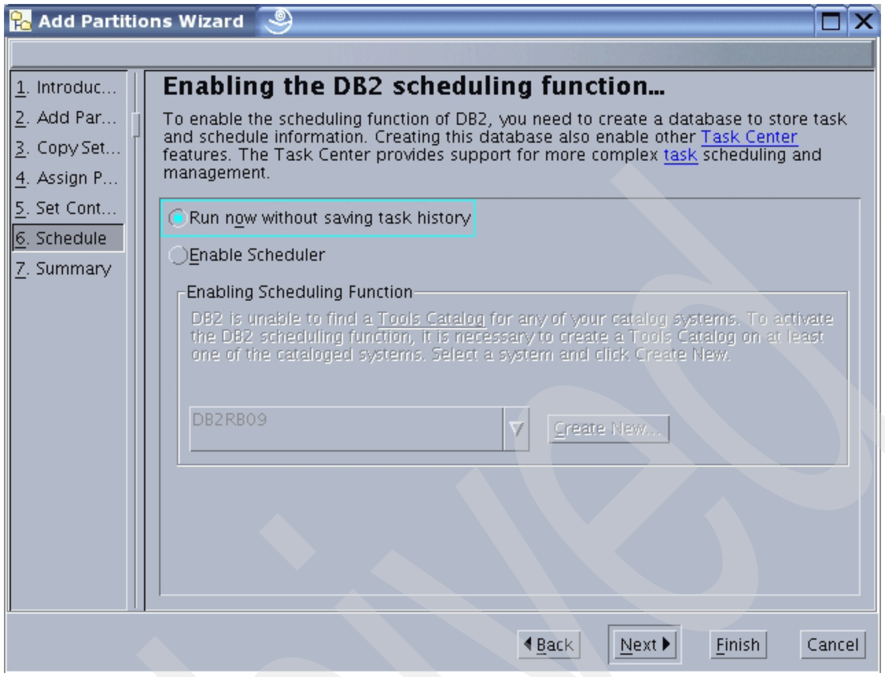
Figure 7-13 Add Partitions Wizard - Set containers completed

Step 11 - Ready to run

DB2 now has all of the information it needs to create the new partition. This process includes the following tasks:

1. Create the new data database partition and add it to the DB2 instance.
2. Restart the DB2 instance.
3. Add a TEMPSPACE container to the new database partition.
4. Add the new partition to the database partition group.
5. Add containers to the table spaces that reside in database partition groups that span the new partition.

Click **Next** and you will be presented with the Scheduling panel, as shown on Figure 7-14 on page 384.



Enabling the DB2 scheduling function...

To enable the scheduling function of DB2, you need to create a database to store task and schedule information. Creating this database also enable other Task Center features. The Task Center provides support for more complex task scheduling and management.

[✓ Run now without saving task history]

Enable Scheduler

Enabling Scheduling Function

DB2 is unable to find a Tools Catalog for any of your catalog systems. To activate the DB2 scheduling function, it is necessary to create a Tools Catalog on at least one of the cataloged systems. Select a system and click Create New.

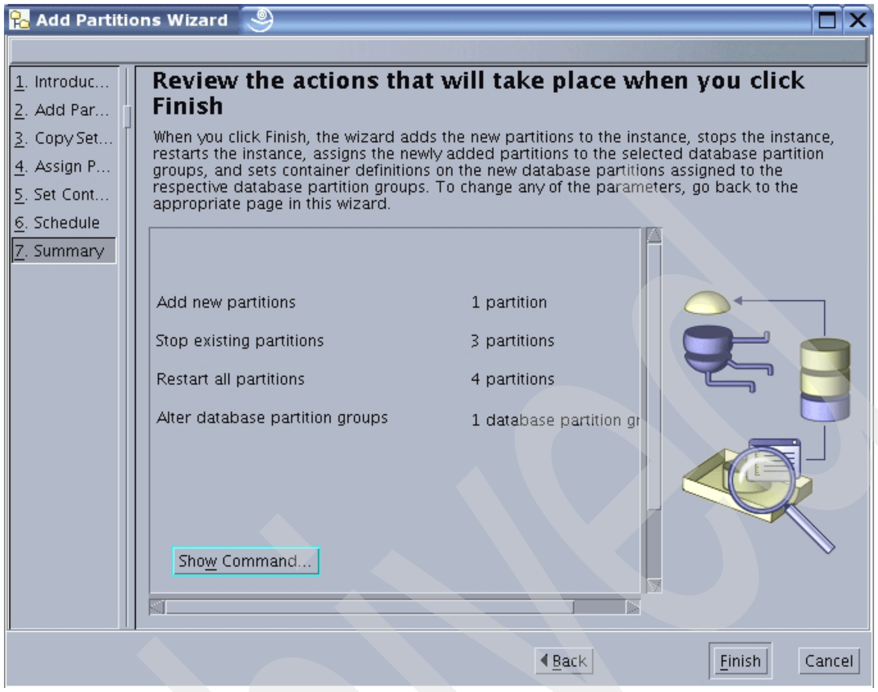
DB2RB09 | Create New...

Figure 7-14 Add Partitions Wizard - Scheduling

Step 12 - Schedule execution

On the Schedule panel you can immediately run the task or schedule it for execution in the future via the DB2 Task Center. We recommend saving the command script. This can be done via the Summary panel or via the Task Center. Both methods are shown below.

Click the **Next** button. The Summary panel will be presented, as shown on Figure 7-15 on page 385.



Review the actions that will take place when you click Finish

When you click Finish, the wizard adds the new partitions to the instance, stops the instance, restarts the instance, assigns the newly added partitions to the selected database partition groups, and sets container definitions on the new database partitions assigned to the respective database partition groups. To change any of the parameters, go back to the appropriate page in this wizard.

Add new partitions	1 partition
Stop existing partitions	3 partitions
Restart all partitions	4 partitions
Alter database partition groups	1 database partition gr

Figure 7-15 Add Partitions Wizard - Summary

Step 13 - Results

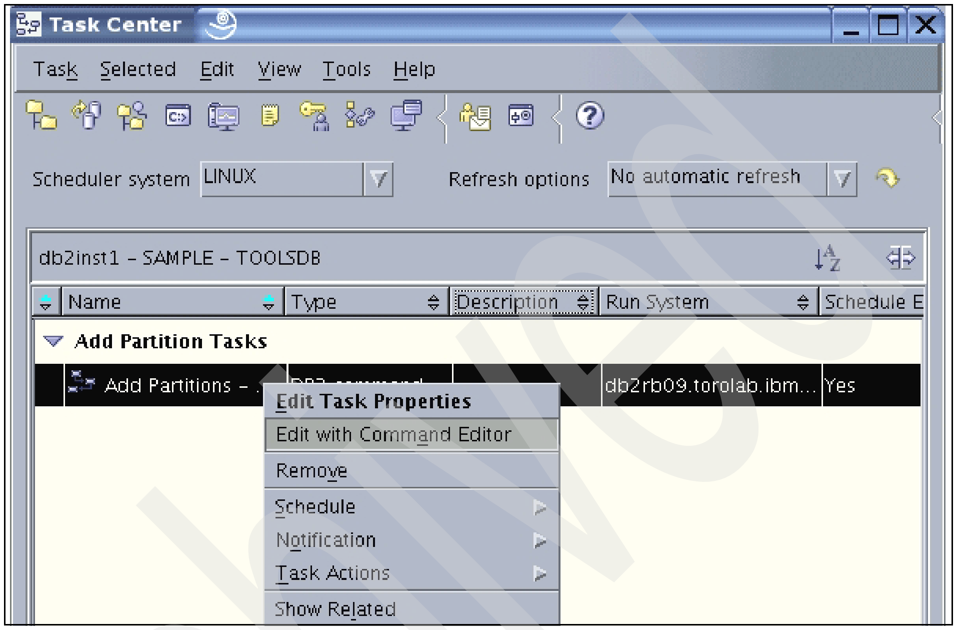
This panel will allow you to see the command sequence that the Add Partition Wizard constructed to create the new database partition. Click the **Show Command** button. The script is as shown in Example 7-10.

Example 7-10 Add partition script

```
db2start dbpartitionnum 3 ADD DBPARTITIONNUM
HOSTNAME "db2rb12.torolab.com" PORT 0 NETNAME db2rb12
WITHOUT TABLESPACES;
db2stop force;
db2start;
connect to tpch;
ALTER TABLESPACE TMP4K ADD ('/db2fs2p3/db2inst2/tpch/tmp4kc00')
ON DBPARTITIONNUM (3);
ALTER DATABASE PARTITION GROUP PG_ALL ADD DBPARTITIONNUM (3)
WITHOUT TABLESPACES;
ALTER TABLESPACE TS01 ADD ('/db2fs1p3/db2inst2/tpch/ts01c00')
ON DBPARTITIONNUM (3);
connect reset;
```

Step 14 - Task Center (optional)

If you chose to save the command script in the DB2 Task Center, you can return later to schedule, execute, and/or edit the command script. To access the DB2 Task Center, click the Task Center icon ( ).



Scheduler system LINUX

Refresh options No automatic refresh

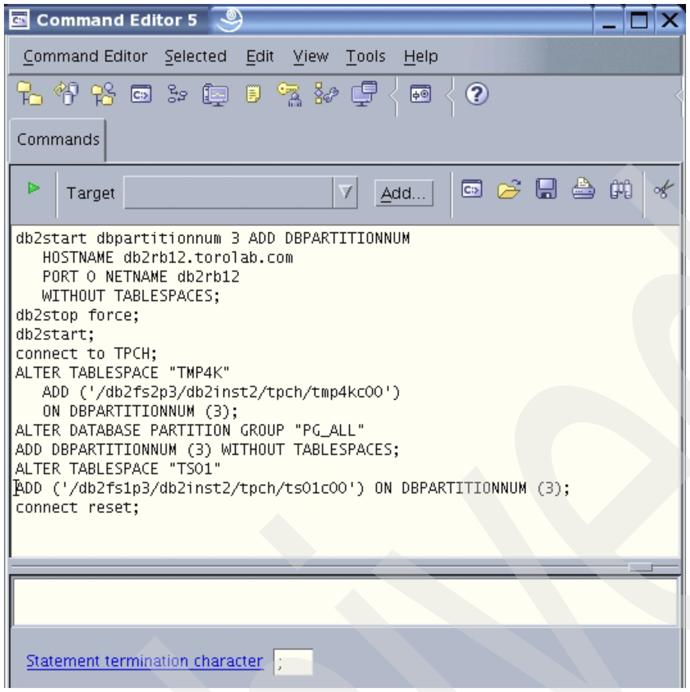
db2inst1 – SAMPLE – TOOLSDB

Add Partition Tasks

Name	Type	Description	Run System	Schedule E
Add Partitions – .	DB2 command		db2rb09.torolab.ibm...	Yes

Figure 7-16 Task Center

From this panel (Figure 7-16) you can edit, schedule, or execute the script immediately. Right-click the saved task and select **Edit with Command Editor** to bring up the script in the DB2 Command Editor, as shown on Figure 7-17 on page 387.


Command Editor 5
Command Editor Selected Edit View Tools Help
Commands
Target
Add...
Statement termination character

```
db2start dbpartitionnum 3 ADD DBPARTITIONNUM
   HOSTNAME db2rb12.torolab.com
   PORT 0 NETNAME db2rb12
   WITHOUT TABLESPACES;
db2stop force;
db2start;
connect to TPCH;
ALTER TABLESPACE "TMP4K"
   ADD ('/db2fs2p3/db2inst2/tpch/tmp4kc00')
   ON DBPARTITIONNUM (3);
ALTER DATABASE PARTITION GROUP "PG_ALL"
ADD DBPARTITIONNUM (3) WITHOUT TABLESPACES;
ALTER TABLESPACE "TS01"
ADD ('/db2fs1p3/db2inst2/tpch/ts01c00') ON DBPARTITIONNUM (3);
connect reset;
```

Figure 7-17 DB2 Command Editor

Step 15 - The DB2 Command Editor (optional)

In DB2 Version 8.2 the Command Center is replaced by the Command Editor. The Command Editor includes all of the functions previously available with the Command Center, but provides these functions in a simplified interface.

Use the Command Editor to generate, edit, execute, and manipulate SQL statements and DB2 commands; to work with the resulting output; and to view a graphical representation of the access plan for explained SQL statements.

Step 16 - Adding the database partition

Example 7-11 is the output from the execution of the script commands.

Example 7-11 Executing the commands to add the database partition

```
db2inst2@db2rb09:~> db2start dbpartitionnum 3 ADD DBPARTITIONNUM HOSTNAME
db2rb12 PORT 0 NETNAME db2rb12 WITHOUT TABLESPACES

07/29/2004 09:56:51     3   0   SQL6075W  The Start Database Manager operation
successfully added the node. The node is not active until all nodes are stopped
and started again.
```

```
db2inst2@db2rb09:~> db2stop force
07/29/2004 10:00:38     0   0   SQL1064N  DB2STOP processing was successful.
07/29/2004 10:00:44     1   0   SQL1064N  DB2STOP processing was successful.
07/29/2004 10:00:44     3   0   SQL1064N  DB2STOP processing was successful.
SQL1064N  DB2STOP processing was successful.

db2inst2@db2rb09:~> db2start
07/29/2004 10:00:50     1   0   SQL1063N  DB2START processing was successful.
07/29/2004 10:00:50     3   0   SQL1063N  DB2START processing was successful.
07/29/2004 10:00:50     2   0   SQL1063N  DB2START processing was successful.
07/29/2004 10:00:54     0   0   SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.

db2inst2@db2rb09:~> db2 connect to tpch

   Database Connection Information

 Database server        = DB2/LINUXX8664 8.2.0
 SQL authorization ID   = DB2INST2
 Local database alias   = TPCH

db2inst2@db2rb09:~> db2 "ALTER TABLESPACE TMP4K
ADD ('/db2fs2p3/db2inst2/tpch/tmp4kc00') ON DBPARTITIONNUM (3)"
DB20000I  The SQL command completed successfully.

db2inst2@db2rb09:~> db2 "ALTER DATABASE PARTITION GROUP PG_ALL
ADD DBPARTITIONNUM (3) WITHOUT TABLESPACES"
SQL20189W  The buffer pool operation (CREATE/ALTER) will not take effect until
the next database startup due to insufficient memory.  SQLSTATE=01657

db2inst2@db2rb09:~> db2 "ALTER TABLESPACE TS01
ADD ('/db2fs1p3/db2inst2/tpch/ts01c00') ON DBPARTITIONNUM (3)"
SQL1759W  Redistribute nodegroup is required to change data partitioning for
objects in nodegroup "PG_ALL" to include some added nodes or exclude some
dropped nodes.  SQLSTATE=01618
```

Note that there was not enough memory to create the buffer pools on the new database partition. This is a normal warning. The buffer pools should allocate successfully after the database is restarted (for example, upon and ACTIVATE database command or first connection after all connections have terminated).

Also note that DB2 issued a warning (SQL1759W) when a table space container was added to table space TS01. Redistribution is now required for any table affected by the addition of the new partition. In other words, any table that resides in a table space created in the affected partition group will need to be redistributed. In this example, all table spaces in PG_ALL need to be redistributed.

7.3.4 Data redistribution via REDISTRIBUTE command

In order to redistribute table data among the partitions in a partitioned database, you use the REDISTRIBUTE DATABASE PARTITION GROUP command.

The redistribute command performs the following steps:

1. Obtains a new partitioning map ID for the target partitioning map.
2. Performs necessary catalog updates.
3. Creates database files for all new database partitions.
4. Redistributes the data on a table-by-table basis for every table in the database partitioning group, in the following steps:
   a. Locks the row for the table in the SYSTABLES catalog table.
   b. Invalidates all packages that involve this table.
   c. Locks the table in exclusive mode.
   d. Uses DELETEs and INSERT processing to move rows from their old partition to their new partition.
   e. If the redistribution operation succeeds, it issues a COMMIT for the table and continues with the next table in the database partitioning group. If the operation fails before the table is fully redistributed, the utility issues a ROLLBACK on updates to the table, ends the entire redistribution operation, and returns an error.
5. Deletes the old partitioning map from the SYSCAT.PARTITIONMAPS catalog view.
6. Does a COMMIT for all changes.

In Example 7-12, a redistribution moved a small amount of data to a fourth partition. Before the redistribution each partition held approximately 33-K rows. After the redistribution each held approximately 25-K rows.

Example 7-12 Sample redistribute before and after

```
-- COUNT ROWS BEFORE
SELECT dbpartitionnum(s_suppkey), count(*)
FROM tpch.supplier
GROUP BY dbpartitionnum(s_suppkey)
---------- -----------
         1       33338
         0       33269
         2       33393

-- RUN THE REDISTRIBUTE
REDISTRIBUTE DATABASE PARTITION GROUP PG_ALL UNIFORM
```

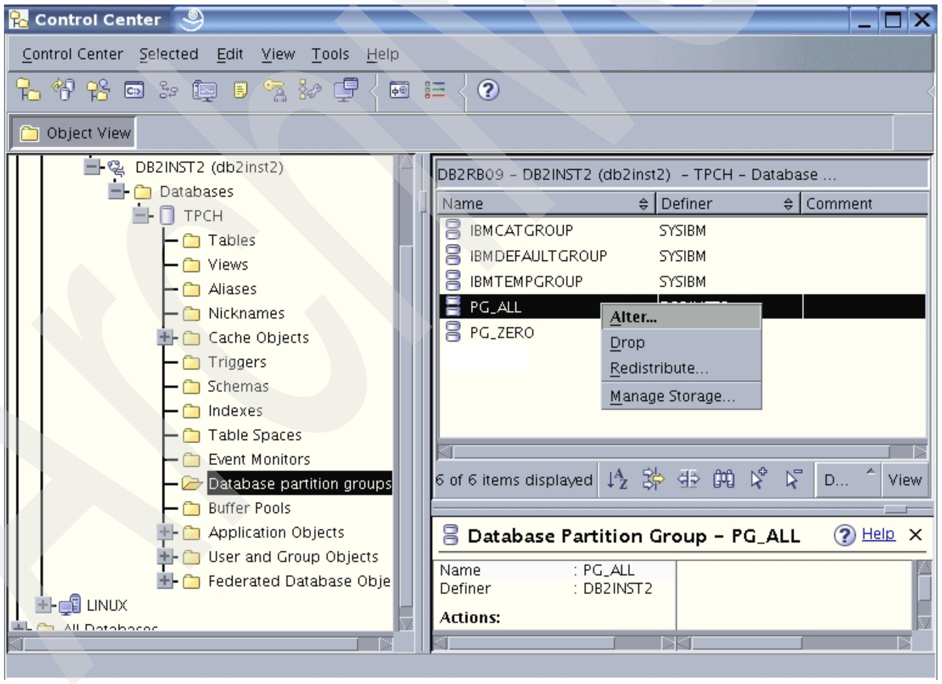
```
DB20000I  The REDISTRIBUTE NODEGROUP command completed successfully.

-- COUNT ROWS AFTER
SELECT dbpartitionnum(s_suppkey), count(*)
FROM tpch.supplier
GROUP BY dbpartitionnum(s_suppkey)
---------- -----------
         0       24891
         2       25080
         3       24981
         1       25048
```

The Redistribute Data wizard

New in DB2 UDB V8.2 is the Redistribute Data wizard, which builds and optionally executes a redistribution strategy for you.

To launch the Redistribute Data wizard, from the Control Center right click on the database partition group and select **Redistribute**, as shown in Figure 7-18.



DB2RB09 – DB2INST2 (db2inst2) – TPCH – Database ...

Name	Definer	Comment
IBMCATGROUP	SYSIBM	
IBMDEFAULTGROUP	SYSIBM	
IBMTEMPGROUP	SYSIBM	
PG_ALL		
PG_ZERO		

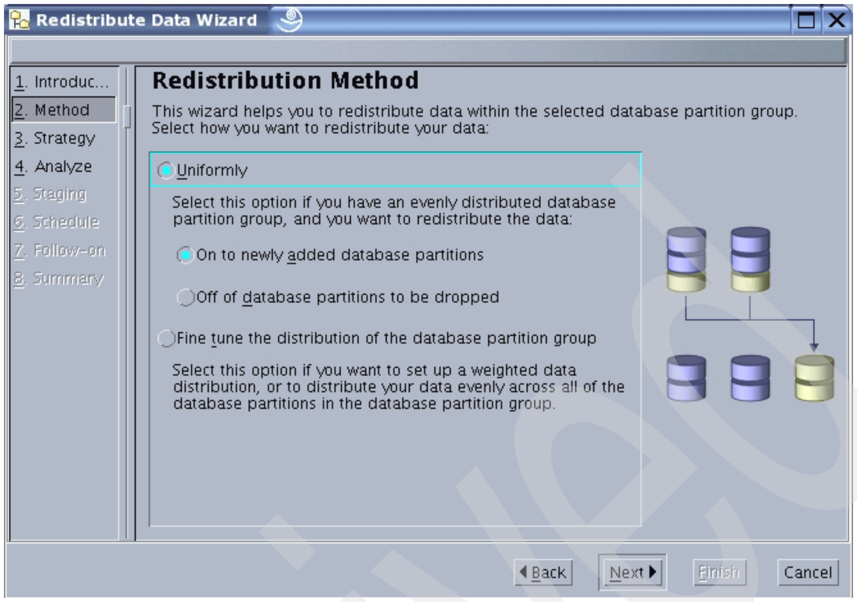
Database Partition Group – PG_ALL

Name : PG_ALL
Definer : DB2INST2

Actions:

Figure 7-18   Launch the Redistribute Data wizard

After the Introduction step, The Redistribution Method panel is shown (Figure 7-19 on page 391).


Redistribute Data Wizard

1. Introduc...
2. Method
3. Strategy
4. Analyze
5. Staging
6. Schedule
7. Follow-on
8. Summary

Redistribution Method

This wizard helps you to redistribute data within the selected database partition group.
Select how you want to redistribute your data:

[✓ Uniformly]

Select this option if you have an evenly distributed database partition group, and you want to redistribute the data:

[✓ On to newly added database partitions]

Off of database partitions to be dropped

Fine tune the distribution of the database partition group

Select this option if you want to set up a weighted data distribution, or to distribute your data evenly across all of the database partitions in the database partition group.

Figure 7-19 Redistribute Data wizard

In this panel, the Redistribute Data wizard allows you to select which direction you wish to redistribute data. You will usually be redistributing data on to new partitions or off of partitions you wish to delete. You may also "fine tune" your redistribution to correct skew or allow for systems of different power, but this is rare.

The subsequent panels allow you to:

- Build a redistribution strategy manually or have the wizard suggest a plan based on the amount of data to be redistributed, log file size, and other system information.
- Optionally regenerate statistics and rebind affected plans.
- Execute the redistribution immediately or save it in the Task Center for execution at a later time.

Redistribution tips

Following are a few tips for data redistribution operation.

Drop indexes on large tables before redistribution

Your overall redistribution will complete faster if you drop the indexes off of your large tables before you initiate the redistribution.

Log space requirements for the redistribution

The INSERT and DELETE operations at each database partition where data is being redistributed will be logged. Logging will be heaviest on the database partition that will lose the most data, or on the database partition that will gain the most data. For example, If you are adding a single partition to a four-partition database, approximately twenty percent of the data in the four original database partitions will be moved to the new database partition. This means that twenty percent of the DELETE operations will occur on each of the four original database partitions, and all of the INSERT operations will occur on the new database partition.

Assuming that you had 1,000,000 evenly distributed rows in the original table, you would expect to move 50,000 rows from each of the original partitions to the new partition. Therefore, you would need enough log space to log 50,000 DELETE operations for each of the original partitions and enough log space for 200,000 INSERT operations on the target data partition.

In planning for log space, consider how much data will be added or deleted from each of the tables rather than table size. A smaller table might require more log space to redistribute than a larger table if it is not as evenly distributed. You may also want to consider setting the *logsecond* database configuration parameter to -1 to avoid most log space problems.

Drop replicated materialized query tables (MQTs)

If the data in a database partition group contains replicated materialized query tables, you must drop these tables before you redistribute the data. After data is redistributed, you can recreate the materialized query tables.

Redistribution restrictions

The following operations on database objects that are in the partition group being redistributed *are permitted* while the redistribution is executing. However, DB2 takes an exclusive lock on the table being redistributed. Therefore, you cannot do them against the table that is currently being redistributed.

- Create indexes on other tables.
- Drop other tables.
- Drop indexes on other tables.
- Query other tables.
- Update other tables.
- Create new tables in a table space defined in the database partition group.
- Create table spaces in the database partition group.

The following operations *are not permitted* while the utility is running:

- Start another redistribution operation on the database partition group.

- Execute an ALTER TABLE statement on any table in the database partition group.
- Drop the database partition group.
- Alter the database partition group.

7.3.5 Alternate methods of data redistribution

The redistribution process will take an exclusive lock on the table currently being redistributed. Depending on your online availability requirements, available disk space, and amount of data being redistributed, you may want to consider using a less-intrusive method to redistribute your data. Two other common methods are via parallel INSERT/SELECT and via EXPORT/LOAD.

Data redistribution via INSERT/SELECT

Another method for data redistribution is to create a new table with the same structure as the old table and populate it via SQL.

The advantage of this approach is that the old table is still available during the process. The new table created by the old table is dropped and the new table is renamed to the old name. This rename takes only a few seconds, so the impact to end users or online availability is minimized. This method also uses less log space, as the new table can be built with NOT LOGGED INITIALLY enabled.

The disadvantage of this approach is that it requires additional disk space equivalent to the size of the old table. It also requires that you be able to manage the build of the new table so that it is synchronized with the old table at the time of cutover. If you are building a large table, you must be able to isolate the updates that will have occurred while the build process runs. Referential Integrity (RI) between tables must also be considered with this approach.

You may choose to use this option on a large fact table and let DB2 automatically redistribute the smaller tables.

This procedure involves the following steps:

1. Create a new database partition group that includes the new partition.
2. Create a new table space in the database partition group from step 1.
3. Run **`db2look`** to generate DDL for the old table. Use the -x option to get the authorization statements.
4. Edit the DDL to create a new table in the new table space. Specify the NOT LOGGED INITIALLY option on the table.

5. In a single unit of work (use the +c option on the DB2 Command Line Processor and add COMMIT statements to your DDL/SQL):
   a. Alter the new table using the ACTIVATE NOT LOGGED INITIALLY option.
   b. Build the new table via SQL using an INSERT INTO *newtable* SELECT * FROM *oldtable*.
   c. Commit.
6. Use the DDL from step 3 to:
   a. Recreate authorizations and views using the DDL from step 3.
   b. Recreate indexes using the DDL from step 3.
7. Runstats on the new table.
8. Rebind.
9. Drop or rename the old table and rename the new table to a production name.

Data redistribution via EXPORT/LOAD

If you are redistributing large tables and the redistribution will result in the majority of the table moving (for example, you are scaling from one server to three servers) you might want to consider using an unload and reload approach.

The advantages of this method are that it uses very little log space and allows you to unload directly to tape, which will save disk space. To speed the unload time, run DB2 EXPORT in parallel on each partition. This can be done by running an EXPORT on each partition, and specifying the predicate WHERE NODENUMBER (colname) = CURRENT NODE. Another option is to use `db2batch` with the -p option. You could also use the DB2 High Performance Unload (a separately packaged product) to reduce the unload time.

This procedure involves the following steps:

1. Unload the data via DB2 EXPORT Utility or High Performance Unload.
2. Consider dropping indexes, especially on large tables or using INDEXING MODE DEFERRED on the LOAD.
3. Prepare the new tables via one of the following methods:
   – Truncate the tables. This can be done with a LOAD with the REPLACE option and using /dev/null as the input file. Add the partition group using the ALTER DATABASE PARTITION GROUP command or the DB2 Command Center GUI. Allow DB2 to perform the redistribution on the empty tables. Load the tables using the exported data.
   – Create a new database partition group that includes the new partition. Drop and recreate the table inthe desired partition group. Load the table using the exported data.

- Create a new database partition group that includes the new partition. Create new tables in the desired partition group as done in the INSERT/SELECT scenario above. After the tables have been reloaded, drop the production tables and rename the new tables. Consider disk space requirements for this scenario.

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 400. Note that some of the documents referenced here may be available in softcopy only.

- *Up and Running with DB2 for Linux,* SG24-6899
- *Up and Running with DB2 UDB ESE, Partitioning for Performance in an e-Business Intelligence World,* SG24-6917

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

- 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-O_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

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

IBM

Redbooks

DB2 Integrated Cluster Environment Deployment Guide

DB2 Integrated Cluster Environment Deployment Guide

Integrated solution using powerful IBM DB2 UDB and IBM @server

Detailed environtment design and implementation guide

Configuration examples and scenarios

The IBM DB2 Integrated Cluster Environment for Linux is a completely integrated, high-performance, and pre-tested solution that incorporates best-of-breed software, hardware, and services. This IBM Redbook provides you with the technical details of the DB2 Integrated Cluster Environment.

We begin with a general overview of DB2 Integrated Cluster Environment and the fundamentals of DB2 in a Linux cluster environment, followed by a discussion of hardware options for cluster ingredients that constitute DB2 Integrated Cluster Environment and considerations for their selection. We cover the planning and database design considerations for building a DB2 partitioned database that is both flexible and scalable and provide the implementation details of DB2 Linux clusters.

This publication also cover the new autonomic features of DB2 UDB V8.2 and system tools monitoring. It describes components and possible configurations of a high-availability solution and provides implementation details of a failover protection in an DB2 Integrated Cluster Environment. At the end, we discuss scalability in the DB2 Integrated Cluster Environment, including some guidance in planning for the growth of a business and database. We also examine a number of scaling strategies in more detail and provide the steps involved in scaling a database.

SG24-6376-00 ISBN 0738490822