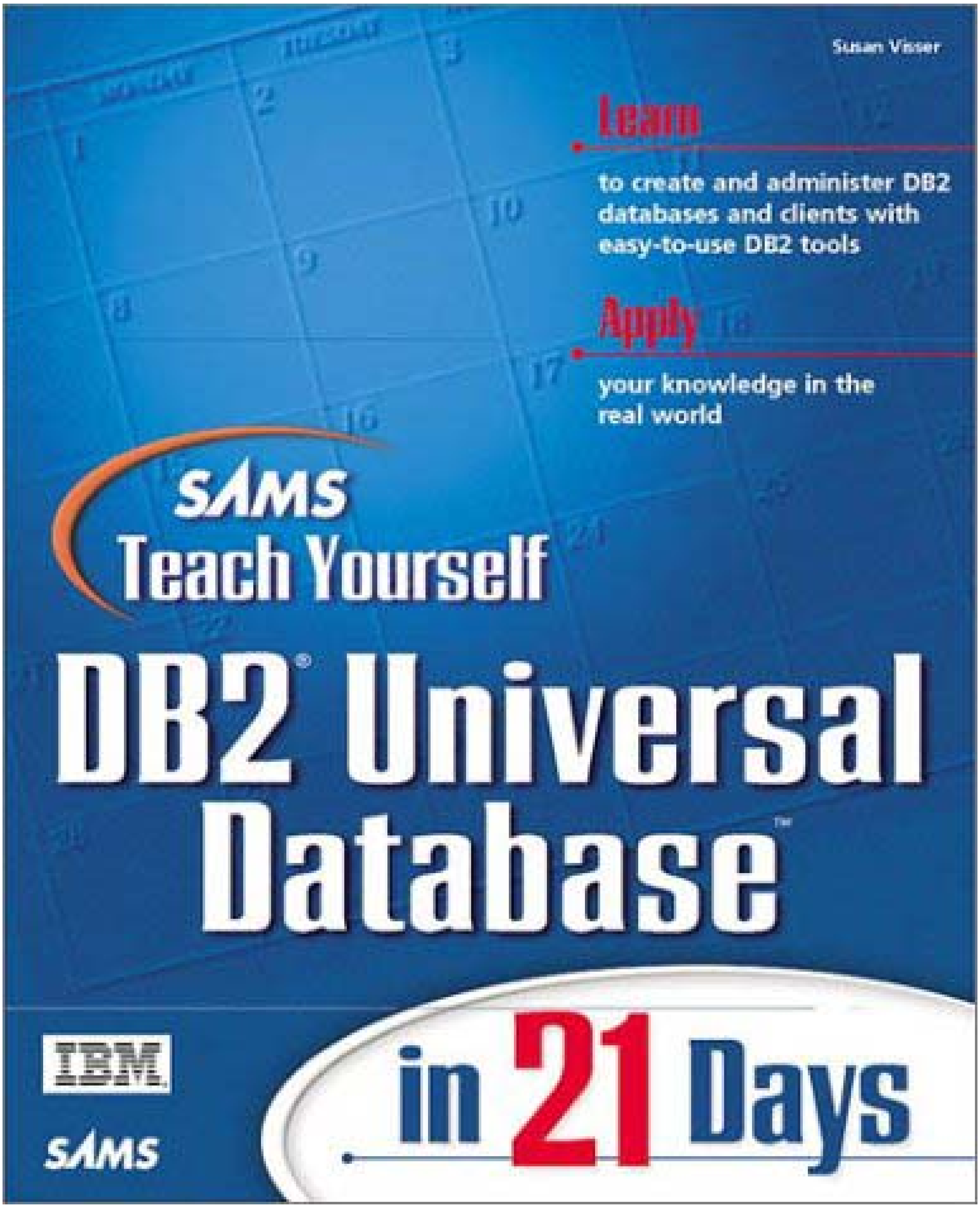

Susan Visser
Learn
to create and administer DB2 databases and clients with easy-to-use DB2 tools
Apply
your knowledge in the real world
SAMS Teach Yourself
DB2® Universal Database™
in 21 Days
IBM
SAMS

Release Team[oR] 2001
[x] Database

Sams Teach Yourself DB2 Universal Database in 21 Days
by Susan Visser

ISBN: 0672312786

Sams © 1998, 501 pages

This in-depth guide offers a hands-on, 21-day course in DB2 database administration.

Table of Contents	Colleague Comments
Back Cover

Synopsis by Dean Andrews

DB2 development team member Susan Visser guides you through the administering of a DB2 Universal Database system in this in-depth 21-day course. Although DB2 runs on UNIX, OS/2, and Windows NT, this book's examples focus primarily on Windows NT. However, most of the course's content -- which includes configuring a DB2 client/server setup, creating databases, using admin tools, backup and recovery methods, performance tuning, and more -- describes the techniques used for any DB2 version 5 product. As with other "in 21 days" books, this volume comes complete with quizzes, tips, lesson plans, and answers in the back.

Table of Contents

Back Cover

This step-by-step, hands-on guide teaches you all the new feature to DB2 Universal Database Version 5. Learn the basics of this database in 21 easy-to-follow lessons. You'll:

- Learn how DB2 can improve your productivity and profitability with better organization
- Master shortcuts to instantly increase your productivity
- Find out more about parallelism, replication, and Web access.
- Take advantage of helpful tips from the experts.
- Learn to build secure distributed databases
- Build replication databases into a network environment

About the Author

Susan Visser is a member of the IBM's DB2 Universal Database development team and the author of numerous books on the product.

Sams Teach Yourself DB2 Universal Database in 21Days

Susan M. Visser

Sams
A Division of Macmillan Computer Publishing
201 West 103rd St., Indianapolis, Indiana, 46290 USA

Executive Editor

Bryan Gambrel

Acquisitions Editor

Angela C. Kozlowski

Development Editors

Susan Shaw Dunn
Rich Alvey

Managing Editor

Jodi Jensen

Project Editor

Maureen A. McDaniel

Copy Editors

Kate Talbot
Michael Brumitt
San Dee Phillips

Indexer

Erika Millen

Technical Editors

IBM
Jim Hobuss
Willie Favero
Rich Yevich

Software Development Specialist

Andrea Duvall

Production

Michael Henry
Linda Knose
Timothy Osborn
Staci Somers
Mark Walchle

International Standard Book Number: 0-672-31278-6

About the Author

Susan M. Visser has been an employee of IBM Canada for 8 1/2 years. For six of these years, she worked as a technical writer assigned to the DB2 Universal Database product. As part of her job, she has written more than 15 technical manuals over the five versions of the product. She helped set the information strategy and provided input to the look and feel of the product. You can reach Susan via email at `suenkarl@interlog.com`.

Acknowledgments

Thanks to the many people who helped make this book a reality. Special thanks go to Susan Dunn and Angela Kozlowski at Macmillan Publishing for helping get me over the hurdles of this, my first publication.

Thanks also to my management and colleagues: Gary Sampson, Kathryn Fryer, Mark Ryan, Jasna Krmpotic, Sheila Richardson, David Sky, David Ready, Richard Hedges, and Dale McInnis. Thanks also to everyone else I deal with on a regular basis who may have helped without even knowing they helped.

Thanks also to my husband, Karl, for helping me test the information and the samples and for looking after all the things I ignored.

Introduction

Overview

Welcome to *Teach Yourself DB2 Universal Database in 21 Days*! This book explores how to install and use version 5 of the DB2 Universal Database product.

DB2 Universal Database is a powerful relational database that can help you and your business become more successful. This book will show you how to exploit DB2's features to create world-class applications that can put your business ahead of your competition. Throughout the book we have highlighted successful applications that work with DB2's most lucrative features, including integration with the Internet, storing and manipulating multimedia objects, DB2 parallelism, exploitation of the Windows NT environment and electronic business.

The focus is on using the administration tools provided with the product. During a three-week period, you will be exposed to many aspects of database administration, including installing the product, setting up a server to accept requests from clients, designing and creating a database, designing and creating the tables and relationships that make up a relational database, exploiting table spaces, and using the various tools to tune performance.

You also will learn how to use the tools provided with DB2 Universal Database to simplify the task of administering your database.

In addition to the 21 lessons are three appendixes: one to provide the answers to the quiz questions, one that explains how to uninstall the DB2 product, and one that walks you through the massive library included with the product.

Assumptions

You'll get the most out of this book if you have the following level of experience:

- Some knowledge of the Windows NT operating environment
- Previous experience with relational databases

This book steps you through installing the DB2 Universal Database version 5 products provided on the CD-ROM included with the book. The DB2 Workgroup Edition and DB2 Personal Edition are included on the CD-ROM. These editions enable you to use the product on a trial basis and allow you to install clients to set up a client/server environment on the Windows NT platform. The DB2 Personal Edition is also available on a trial basis and enables you to have a standalone database system on either Windows NT or Windows 95.

How to Use This Book

This book has been designed as a three-week teach-yourself course, complete with chapter quizzes, exercises, and examples that you can try on your own. It's expected that you complete a chapter a day for a period of 21 days. Of course, you should work at your own pace, so if you can complete more than one chapter in a day, this is fine. Also, if you think you need more time to complete a chapter, take all the time that you need.

Each week begins with a "Week at a Glance" section and ends with a "Week in Review" section. Each lesson ends with a question-and-answer section related to that day's lesson. There is also a quiz at the end of the day to test your grasp of the day's concepts and skills. We urge you to complete these sections to reinforce your new knowledge.

Conventions Used in This Book

Menu names are separated from menu options by a vertical bar (|). For example, File | Open means that you should choose Open from the File menu.

Input All code and other elements that you type in (input) appears in **`boldface monospace`**. Also, such inputted elements are accompanied by a special Input icon in the book's margin.

Output Commands and other code that you don't necessarily type in appear in regular `monospace`. The result (output) of any input that you type in is also in monospace, and is accompanied by a special Output icon.

Syntax In syntax, words that appear in *`italic monospace`* are considered placeholders—rather than actually use those words when using the command, replace the words with the appropriate value. Any syntax provided is also accompanied by a special Syntax margin icon.

Note Notes present tengential information that can help you understand DB2 concepts and techniques.

Tip Tips are little pieces of information to begin to help you in real-world situations. They often offer shortcuts or information that makes a task easier or faster.

Caution Pay careful attention to Cautions: They provide information about detrimental performance issues or dangerous errors.

About the CD-ROM

As mentioned earlier, the CD-ROM that accompanies this book includes a try-and-buy version of DB2 Universal Database Workgroup Edition version 5, DB2 Universal Database Personal Edition, DB2 Software Developer's Kit, Net.Data, Netscape, Lotus SmartSuite, and VisualAge for Java. It also includes all the sample data used in the examples in the book.

Week 1

At a Glance

During Week 1, you'll learn many of the concepts of DB2 Universal Database. Later in the week, you'll see how to install a server and a client and how to set up the communications between the two separate machines. The last lesson of the week teaches you how to logically design a relational database by outlining the sample database used throughout the book. Here's a day-by-day summary of Week 1:

- Day 1, "What Can DB2 Do For You?" focuses on the many products that make up DB2 Universal Database version 5. The concepts of server and client are defined, and the role of the administration tools is discussed. You'll also see how you can use the Internet, ODBC, or Java applications to access data in DB2 databases.
- Day 2, "Exploring the Capabilities of DB2 Universal Server," introduces the many concepts of a relational database and how these concepts relate to DB2 Universal Database.
- During Day 3, "Installing and Configuring DB2 Server," you'll install and configure the DB2 Universal Database Enterprise Edition product on a Windows NT system. This product acts as a server.
- When the product is installed, use the instructions in Day 4, "Getting Started," to learn how to log on, work with passwords, start or stop DB2, and understand the objects that DB2 has added to your desktop.
- Day 5, "Configuring Server Communications with the Control Center," discusses how you can use DB2's administration tools to set up or modify the communications protocols used for client/server communications.
- When your DB2 server is installed and configured, use the instructions in Day 6, "Installing DB2 Clients," to set up a client workstation on Windows NT or Windows 95.
- Day 7, "Designing a CDLIB Database," introduces the database used throughout the rest of the book to demonstrate the concepts and features of the DB2 Universal Database product. As the sample database is introduced, the theory of logically designing a database is explained.

Day 1: What Can DB2 Do For You?

Overview

In business environments today, the demand for information access is increasing while the volume and complexity of data grows rapidly along with the variety of applications. Users once satisfied with simple file management programs now need powerful database management tools and application programmers require the features of systems once available only on larger machines. New applications need to be developed and historical data needs to be preserved.

More and more, critical business applications are being moved to client/server LANs because of the advances in hardware capacity, software function, and performance. Data may now be located across the office, across the country, or around the world. Decisions must be made on where best to store data, how to access it quickly, and how to set up production databases and applications on various platforms so that they can best interact.

This book focuses on setting up and using a DB2 server on a Windows NT system and a DB2 Client Application Enabler on a Windows NT or Windows 95 system. DB2 Universal Database is a powerful database management system that helps you manage your data in these complex and rapidly changing environments. This relational database management system (RDBMS) allows users to create, update, and control relational databases by using Structured Query Language (SQL).

Designed to meet the information needs of small and large businesses alike, DB2 is available on various platforms, including large systems such as MVS/ESA, VM, and VSE; mid-sized systems such as OS/400, AIX, and other UNIX-based systems; and single or LAN-based systems such as OS/2, Windows 95, and Windows NT.

Data managed by DB2 database servers can be accessed and manipulated by applications on PC workstations running popular operating systems such as OS/2, Windows 95, Windows NT, Windows 3.1, and Macintosh, and by applications developed for UNIX workstations from IBM, HP, SUN, Silicon Graphics, the Santa Cruz Organization, and Siemans Nixdorf.

In today's lesson, you'll learn about the various DB2 products available and which one is right for you. You'll learn the basic features available in the products and how these features can help you organize your data.

What Is DB2 Universal Database?

The DB2 products that run on OS/2, Windows NT, UNIX, and related platforms are collectively known as *DB2 Universal Database*. This is to distinguish that these products run on similar platforms and share the same source code.

DB2 is a Web-enabled relational database management system that supports Java. It can be scaled from single processors to symmetric multiprocessors to massively parallel processors and is multimedia-capable with image, audio, video, and text support.

DB2's Various Flavors

DB2 comes with various features: DB2 Workgroup Edition, DB2 Enterprise Edition, DB2 Client Application Enabler, DB2 Personal Developer's Edition, DB2 Universal Developer's Edition, DB2 Personal Edition, DB2 Connect Enterprise Edition, DB2 Connect Personal Edition, and DB2 Enterprise–Extended Edition.

Note DB2 Workgroup Edition, DB2 Enterprise Edition, DB2 Personal Edition, and DB2 Enterprise–Extended Edition are commonly referred to as DB2 servers throughout this book.DB2 Workgroup Edition

DB2 Workgroup Edition

DB2 Workgroup Edition is licensed on a per-user basis. It's available for the Windows NT, OS/2, AIX, HP-UX, and Solaris platforms.

The following products come with DB2 Workgroup Edition:

- The *DB2 Universal Database server* (referred to as a *DB2 server*) allows local and remote clients and applications to create, update, control, and manage relational databases by using Structured Query Language (SQL), Open Database Connectivity (ODBC), or Call Level Interface (CLI).

- The *DB2 Client Pack* CD-ROM contains all the latest DB2 Client Application Enablers. With DB2 Client Application Enabler, clients from various platforms can connect to any DB2 Universal Database, DB2 Connect, or DataJoiner multiuser product.

- The *DB2 Net.Data* CD-ROM contains all supported DB2 Net.Data (formerly known as DB2 World Wide Web Connection) products. DB2 Net.Data allows application developers to create Internet applications that access data from DB2 databases.

- *Domino Go Webserver* provides a high-performance Web server that runs on a broad range of platforms.

- *Lotus Approach* is a front-end database-access tool for Windows 3.1, Windows 95, Windows NT, and OS/2 clients.

DB2 Enterprise Edition

DB2 Enterprise Edition includes all the products provided in DB2 Workgroup Edition, plus support for host connectivity, providing users with access to DB2 databases residing on host systems such as MVS/ESA, OS/390, AS/400, VM, and VSE. It also supports unlimited LAN database access. DB2 Enterprise Edition is available for the Windows NT, OS/2, AIX, HP-UX, and Solaris platforms.

DB2 Client Application Enabler

DB2 Client Application Enabler (referred to as a *DB2 client*) allows a client workstation to access the DB2 server. DB2 Client Application Enablers are available for the OS/2, Windows 95, Windows NT, Windows 3.1, DOS, Macintosh, AIX, HP-UX, Solaris, SINIX, Silicon Graphics, and SCO OpenServer platforms.

DB2 Personal Developer's Edition

DB2 Personal Developer's Edition comes with DB2 Universal Database Personal Edition, DB2 Connect Personal Edition, and DB2 Software Developer's Kits for OS/2, Windows NT, Windows 95, and Windows 3.1 platforms.

The DB2 Personal Developer's Edition gives you all the tools you need to create multimedia database applications that can run on OS/2, Windows NT, Windows 95, or Windows 3.1 platforms and can connect to any DB2 server. The edition includes the following complementary products: DB2 Extenders, VisualAge for Basic, and VisualAge for Java Professional Edition.

DB2 Universal Developer's Edition

DB2 Universal Developer's Edition comes with DB2 Universal Database Workgroup Edition, DB2 Connect Enterprise Edition, DB2 Software Developer's Kits, network support, Net.Data, Domino Go Webserver, and application development tools for all supported platforms.

This edition gives you all the tools you need to create multimedia database applications that can run on a variety of platforms and can connect to any DB2 server. It includes the following complementary products: DB2 Extenders, VisualAge for Basic, and VisualAge for Java Professional Edition.

DB2 Personal Edition

DB2 Personal Edition allows you to create and use local databases and access remote databases if they're available. DB2 Personal Edition is available for the OS/2, Windows NT, and Windows 95 platforms.

DB2 Connect Enterprise Edition

DB2 Connect Enterprise Edition (formerly known as DDCS Multi-User Gateway) provides access from network clients to DB2 databases residing on host systems such as MVS/ESA, OS/390, OS/400, VM, and VSE. DB2 Connect Enterprise Edition is available for the OS/2, Windows NT, AIX, HP-UX, and Solaris platforms.

DB2 Connect Personal Edition

DB2 Connect Personal Edition (formerly known as DDCS Single-User) provides access from a single workstation to DB2 databases residing on host systems such as MVS/ESA, OS/390, OS/400, VM, and VSE, as well as access to DB2 Universal Databases. DB2 Connect Personal Edition is available for the OS/2, Windows 3.1x, Windows NT, and Windows 95 platforms.

DB2 Enterprise-Extended Edition

DB2 Enterprise–Extended Edition (formerly known as DB2 Parallel Edition) allows databases to be partitioned across multiple independent computers of a common platform. To end users and application developers, it still appears as a single database on a single computer.

This fully scalable database system allows applications to use a database that's simply too large for a single computer to handle efficiently. SQL operations and utilities can operate in parallel on the individual database partitions, thereby speeding up the execution time of a single query or utility.

DB2 Enterprise–Extended Edition is now available for the AIX platform, with Solaris and Windows NT versions coming soon.

DB2's Architecture

All members of the DB2 family have the same basic architecture as the original MVS/ESA version and use many of the same key algorithms. It's important to understand, however, that these later products aren't just a port from MVS/ESA to other operating systems: Their internal components have been optimized to exploit each platform's specific features.

DB2 is an open system. In addition to client platforms provided by IBM, all DB2 database servers allow for open access from any product that supports the Distributed Relational Database Architecture (DRDA) protocol. This support eliminates the need for expensive add-on components and gateways. IBM also offers a facility to access any other RDBMS that implements the DRDA application server specification. This support is offered by a companion product known as *DB2 Connect*.

In addition to its data management functions, DB2 includes tools that let you create customized applications for accessing and working with data. Support is available for the development of multimedia, object-oriented, Internet, and ODBC applications.

DB2 and Its Companion Products

If your organization has data spread across multiple databases, remote relational access can represent an important advantage in the way data can be designed, managed, and used. DB2 makes it possible for organizations to distribute and access data across a network of systems.

Users can query, add, delete, or update data in remote databases, letting you focus on the design of your database and the problems to be solved rather than on the complexities of gaining access to the data. Data is requested at one location and provided by another. The database serving the request maintains authorizations for remote requests in the same way as for local requests.

To understand how data is distributed, you must understand the components that make up such an environment. The key components include a *database server* and one or more *database clients*. The server controls one or more databases and handles requests from clients that want to access those databases. Each component is described in the following sections.

The DB2 Server

DB2 Universal Database is available in four versions: Personal Edition, Workgroup Edition, Enterprise Edition, and Enterprise Edition–Extended. The database engine in each version is identical. The engine is a full-function, robust database management system that includes optimized SQL support based on actual database usage and tools to help manage the data. The difference between these products is the capability to support remote clients, the licensing considerations, and the number of database partitions supported.

The Personal Edition can be accessed only from local applications; it can't act as a server supporting multiple clients. The Personal Edition supports any local applications that run on the same computer where the database resides. Figure 1.1 shows an example of a DB2 Personal Edition setup with several local applications.

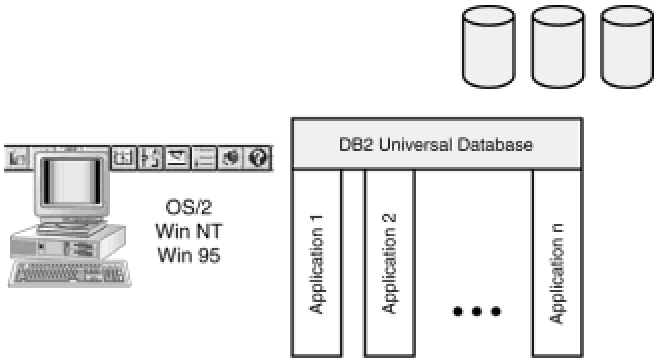


Figure 1.1: A sample setup of IBM's DB2 Universal Database Personal Edition.

The Personal Edition includes the database engine plus the DB2 tools for administrative tasks such as configuring the system, replicating data, tuning performance, backing up and recovering the system, and managing media and the DB2 Client Application Enabler component for access to remote servers. This environment is ideal if you want a simple standalone system or if you perform database administration tasks and need to have local databases to prototype applications. If you want to use an application development environment, you should consider the DB2 Developer's Editions.

The Workgroup and Enterprise editions include functions that allow DB2 to be accessed by local and remote clients. Remote clients must have the DB2 Client Application Enabler component installed to access a database server.

The Workgroup and Enterprise editions also include the database engine and the DB2 tools for performing administrative tasks, as well as the DB2 Client Application Enabler component for access to remote database servers.

DB2 Enterprise Edition–Extended provides the capability of the database to be partitioned across multiple, independent computers by a LAN. It contains the same DB2 Tools and the DB2 Client Application Enabler as the other DB2 servers but can handle extremely large databases and can improve performance by adding more processing power to a given database operation.

DB2 Enterprise Edition–Extended exploits large-scale SMP and multinode (MPP/Cluster) configurations. Figure 1.2 shows an example of a possible configuration.

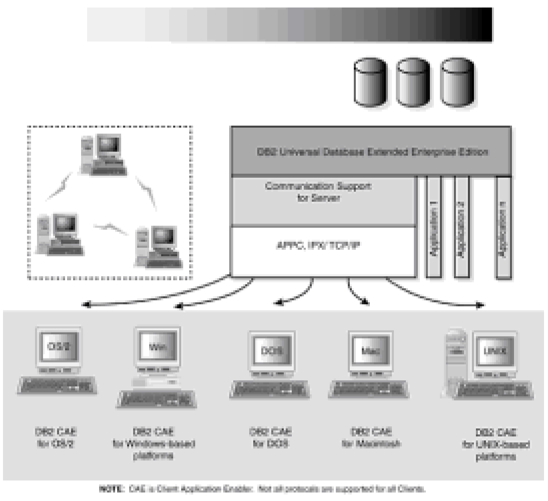

DB2 Universal Database Extended Enterprise Edition
Communication Support for Server
APPC, IPX/ TCP/IP
Application 1
Application 2
Application n
OS/2
Win
DOS
Mac
UNIX
DB2 CAE for OS/2
DB2 CAE for Windows-based platforms
DB2 CAE for DOS
DB2 CAE for Macintosh
DB2 CAE for UNIX-based platforms
NOTE: CAE is Client Application Enabler. Not all protocols are supported for all Clients.

Figure 1.2: The DB2 Enterprise Edition–Extended environment, with remote client support.

The DB2 Clients

The DB2 server can run applications locally and can be accessed by applications running on remote clients with the DB2 Client Application Enabler installed. The DB2 Client Application Enabler provides a runtime environment that allows client applications to access one or more remote databases. Software on the DB2 Client Pack CD-ROM provides support for clients on the following operating systems:

- OS/2
- Windows NT, Windows 95, and Windows 3.x
- UNIX (AIX, HP-UX, Solaris, SINIX, Silicon Graphics IRIX, SCO OpenServer)
- Macintosh

The Client Application Enabler for DOS version 1.2 is available only from the Web. See Day 6, "Installing DB2 Clients," for information on how to download the DOS client.These DB2 Client Application Enablers can be copied to the appropriate workstations for each remote client that will be connecting to the server.

Figure 1.3 shows a DB2 server with local applications and applications running on remote clients. DB2 Workgroup Edition or Enterprise Edition is installed on the server and contains your databases. Any applications running on the server are known as *local applications*. The client systems need one of the DB2 Client Application Enabler products installed, which allows applications to access the data on the remote server system.

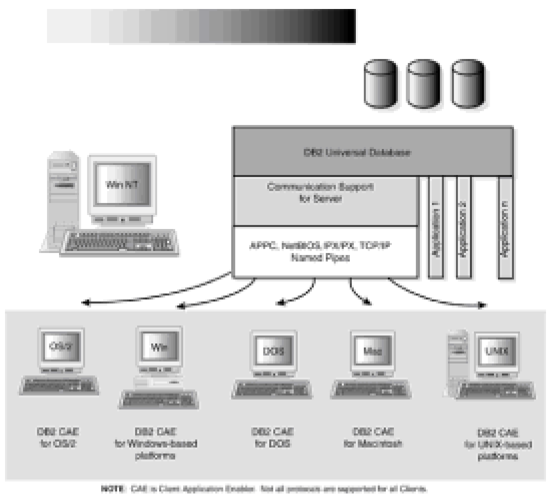

DB2 Universal Database
Communication Support for Server
APPC, NetBIOS, IPX/PX, TCP/IP Named Pipes
Application 1
Application 2
Application n
Win NT
OS/2
Win
DOS
Mac
UNIX
DB2 CAE for OS/2
DB2 CAE for Windows-based platforms
DB2 CAE for DOS
DB2 CAE for Macintosh
DB2 CAE for UNIX-based platforms
NOTE: CAE is Client Application Enabler. Not all protocols are supported for all Clients.

Figure 1.3: DB2 with local applications and remote clients.

The DB2 Client Application Enabler component is built into each DB2 Universal Database product. A workstation with DB2 Client Application Enabler installed can access any DB2 server with any of a number of supported communication protocols. For example, DB2 Client Application Enabler for Windows NT supports the APPC, IPX/SPX, Named Pipes, NetBIOS, and TCP/IP protocols. (The same protocols are supported on DB2 for Windows NT servers.) This heterogeneous support protects existing investment in client workstations and allows you to select a server machine that's most appropriate for your database environment.

In addition to accepting requests from the clients listed earlier, DB2 Universal Database has the DRDA Application Server feature built in. It accepts requests from MVS, OS/400, VM, and other DRDA clients.

Understanding How DB2 Universal Database Works with Data

As well as providing a relational database to store your data, DB2 lets you administer requests to query, update, insert, or delete data from local or remote client applications.

DB2 Universal Database includes graphical tools that allow you to tune performance, access remote DB2 servers, manage all servers from a single site, develop powerful applications, and process SQL queries. These tools are described later in the section "DB2 Tools for Administering Databases."

When a network is operational and protocols are functional on the workstations, LAN-to-LAN connections between DB2 servers and clients require no additional software. For example, Figure 1.4 shows a DB2 server on a Windows NT workstation that's connected to a LAN in Montreal, and another DB2 server on an OS/2 workstation connected to a LAN located in Toronto. As long as a connection exists between the two LANs, clients on either network can access either server.

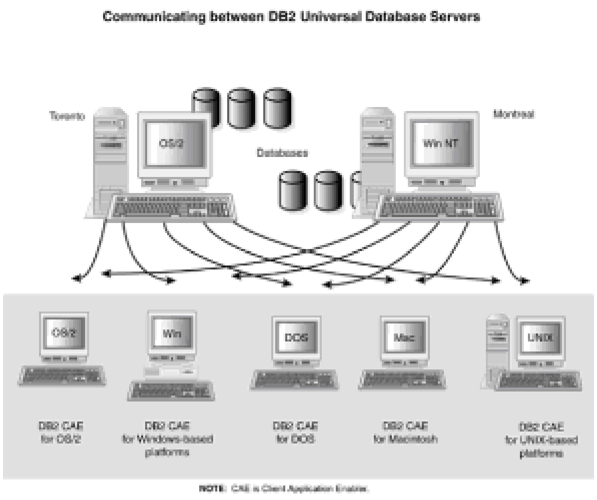

Communicating between DB2 Universal Database Servers
Toronto
OS/2
Databases
Win NT
Montreal
OS/2
Win
DOS
Mac
UNIX
DB2 CAE for OS/2
DB2 CAE for Windows-based platforms
DB2 CAE for DOS
DB2 CAE for Macintosh
DB2 CAE for UNIX-based platforms
NOTE: CAE is Client Application Enabler.

Figure 1.4: Accessing data on multiple servers.

Within a single transaction, databases on both servers are accessed and updated, and the integrity of the data on both servers is maintained. This is commonly known as a *two-phase commit* or *distributed-unit-of-work access*.

DB2 Tools for Administering Databases

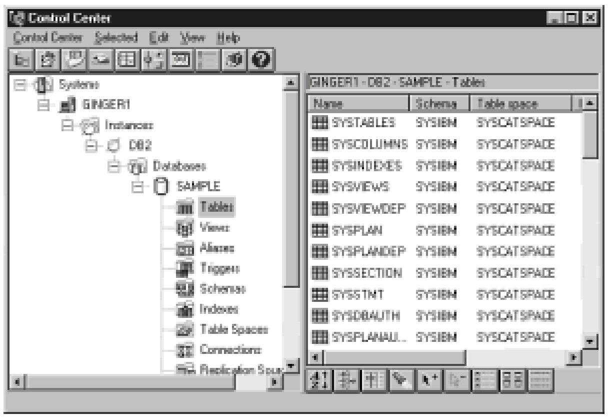
You can perform database administration tasks locally or remotely with the DB2 Administration Tools. Use the Control Center to graphically perform server administrative tasks such as configuring, backing up and recovering data, managing directories, propagating data, scheduling jobs, and managing media. Use the Command Center to access and manipulate databases from a graphical interface.

The Server Administration folder contains the Control Center, from which you can access tools to help you administer DB2 servers. The tools are installed by default on DB2 servers and optionally installable on any DB2 client running Windows 95, Windows NT, or OS/2.

If you want to have a dedicated database administrator (DBA) system, which allows you to administer remote DB2 databases, you can install the tools on a client system or on DB2 Personal Edition.

Managing Databases with the Control Center

The Control Center displays database objects (such as databases, tables, and packages) and their relationships to each other. It contains tools for performing common database administration tasks. With the Control Center, you can manage a local database server or multiple remote database servers (including databases in a parallel environment) and the database objects within them, all from a single point of control. It provides seamless integration of the DB2 Administration Tools, gives you a clear view of all managed systems, lets you manage databases remotely, and provides step-by-step assistance for some tasks. Figure 1.5 shows the main Control Center window.



Name	Schema	Table space
SYSTABLES	SYSIBM	SYSCATSPACE
SYSCOLUMNS	SYSIBM	SYSCATSPACE
SYSINDEXES	SYSIBM	SYSCATSPACE
SYSVIEWS	SYSIBM	SYSCATSPACE
SYSVIEWDEP	SYSIBM	SYSCATSPACE
SYSPLAN	SYSIBM	SYSCATSPACE
SYSPLANDEP	SYSIBM	SYSCATSPACE
SYSSECTION	SYSIBM	SYSCATSPACE
SYSSTMT	SYSIBM	SYSCATSPACE
SYSDBAUTH	SYSIBM	SYSCATSPACE
SYSPLANAU...	SYSIBM	SYSCATSPACE

Figure 1.5: The Control Center's main window.

From the Control Center, you can perform the following tasks on database objects:

- Create and drop a database.
- Create, alter, and drop a table space or table.
- Create, alter, and drop an index.
- Backup and recover a database or table space.
- Define the replication sources and subscriptions to replicate data between systems.

You can also manage database manager environments (known as *DB2 instances*) asfollows:

- Maintain the communications protocols.
- Set database manager configuration values that affect performance.

SmartGuides are provided to help you perform complex tasks. For example, a SmartGuide is available to tune the performance of your system. See the "SmartGuides" section in Appendix C, "Road Map to DB2 Information," for descriptions of the various SmartGuides and how to invoke them.

The Control Center provides additional facilities to help you manage your DB2 servers. You can run these facilities from the Control Center toolbar or from icons in the Administration Tools folder:

- The *Command Center*, to enter DB2 commands and SQL statements.
- The *Script Center*, to create mini-applications known as *scripts*, which you can store and invoke later.
- The *Journal*, to view all available information about jobs that are pending execution, are executing, or have completed execution as well as the recovery history log, the alerts log, and the messages log.
- The *Alert Center*, to monitor your system for early warnings of potential problems or to automate actions to correct problems discovered.

- The *Tools Setting*, to change the settings for the Control Center, Alert Center, and Replication.

- *DB2 Performance Monitor*, to monitor the performance of your DB2 system.

- *Visual Explain*, to graphically analyze and tune SQL statements and to analyze query-access plans.

The Administration Server is used as a service by the DB2 Administration Tools to satisfy requests. Implemented as a DB2 instance, it has interfaces to start, stop, catalog, and configure itself. The Administration Server is required in order to use any of the administration tools described earlier.

Managing Communications on the Server

The Control Center's Setup Communications option allows you to view, update, and reset the server protocol settings. This tool helps database administrators do the following:

- Configure database manager communications parameters for a new instance or maintain the communication configuration of an existing instance. Many required steps are automated. You simply need to select the communication protocols you want supported on the server instance.

- Generate database information in a profile that can be used to configure clients.

Managing Connections to Databases withthe Client Configuration Assistant

The Client Configuration Assistant helps you manage your database connections to remote database servers. It leads you through the necessary steps to configure andmanage DB2 clients, while at the same time hides many of the steps required by automating them. With the Client Configuration Assistant, you can do the following:

- Define database connections in three ways so that applications can use thedatabases:

- Search the network for available databases and select one. Client access is automatically set up for that database.

- Use a database access profile provided by a database administrator to automatically define connections.

- Manually configure a connection to a database by entering a few required connection parameters.

- Remove cataloged databases or modify the properties of a cataloged database.

- Test connections to local or remote databases identified on your system to ensure that you can connect to the server you need.

- Bind applications to a database by selecting utilities or bind files from a list.

- Tune the client configuration parameters on your system. Parameters are logically grouped and hints are provided on the interface as parameters are selected.

- Set up a connection to a DRDA server if DB2 Connect is installed.

- Export the values used for your client to another client, so you can duplicate the setup from one client to another.

- Register the database as an ODBC data source and customize the settings for the ODBC application you're using.

Accessing Host Data from the Desktop

DB2 Connect gives clients on your LAN access to data stored on host systems. It provides applications with transparent access to host data through a standard architecture for managing distributed data. This standard, known as *Distributed Relational Database Architecture* (*DRDA*), allows your applications to establish a fast connection to databases on MVS, OS/390, OS/400, VM, and VSE hosts.

DB2 manages a great deal of the data in large organizations on systems such as OS/390, MVS, AS/400, and VSE and VM. Applications that run on any of the supported platforms can work with this data transparently, as if a local database server managed it.

The following tools and products can access host data easily by using DB2 Connect:

- Spreadsheets such as Lotus 1-2-3 and Microsoft Excel to analyze real-time data without the cost and complexity of data extract procedures and import procedures.

- Decision support tools such as Business Objects, Intersolv Q+E Database Editor, and Crystal Reports to provide real-time information.

- Personal database products such as Lotus Approach and Microsoft Access.

- Development tools such as IBM VisualAge, PowerBuilder, Microsoft Visual Basic, and Borland Delphi to create client/server solutions.

DB2 Connect comes in two versions: *DB2 Connect Personal Edition* and *DB2 Connect Enterprise Edition*. With DB2 Connect Personal Edition, only the local clients on the DB2 Connect workstation can access the host. Figure 1.6 shows an example of the DB2 Connect Personal Edition.

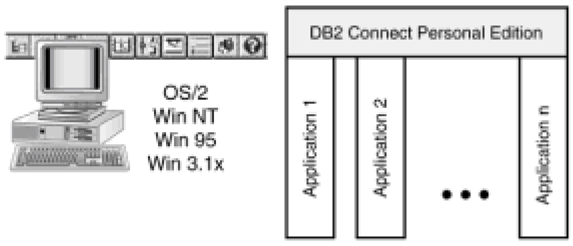


Figure 1.6: A sample DB2 Connect Personal Edition setup.

DB2 Connect Enterprise Edition allows multiple clients to connect to host data and can significantly reduce the effort required to establish and maintain access to enterprise data. Figure 1.7 shows an example of clients connecting to host databases through DB2 Connect Enterprise Edition.

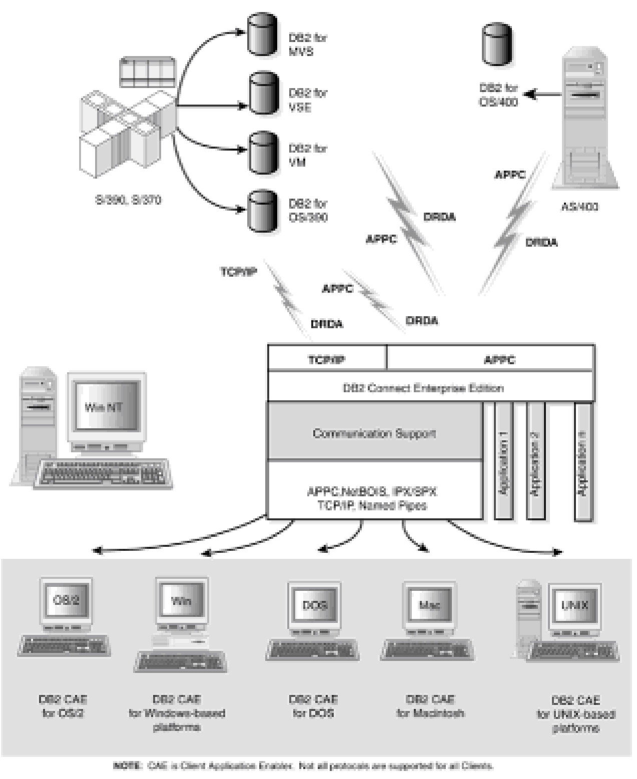


Figure 1.7: A sample DB2 Connect Enterprise Edition setup.

Also, you can use IBM's Data Replication tools to propagate data among database servers on the network. This allows users and application programs to have access to data for informational or prototyping purposes without interfering with operational systems.

Developing Applications with theDB2 Software Developer's Kit

The DB2 development editions give you all the tools you need to create database applications that run on various platforms and connect to any DB2 database. The DB2 development editions include the following products:

- DB2 Universal Database
- DB2 Software Developer's Kit
- DB2 Extenders Development Toolset
- VisualAge for Basic
- Net.Data
- Lotus Approach

The DB2 Software Developer's Kit (DB2 SDK) is a collection of tools designed to meet the needs of database application developers. It includes libraries, header files, documented APIs, and sample programs to build character-based, multimedia, or object-oriented applications. You can install the DB2 SDK on the server with DB2 Universal

Database or on a remote workstation. If you install it on the server, the DB2 SDK can access remote and local databases; if you install it on a client, the DB2 SDK can access only remote databases.

A platform-specific version of the DB2 SDK is available for each supported operating system. The DB2 Universal Developer's Edition provides the DB2 SDK for all supported platforms, the DB2 Extender development toolset, and VisualAge for Basic. Applications developed with the DB2 SDK run on any platform where the equivalent DB2 Client Application Enabler component is installed. Through the DB2 Client Application Enabler, these applications can access all DB2 servers and, by using DB2 Connect, can also access other database servers that support DRDA.

The DB2 SDK allows you to develop applications that use the following interfaces:

- Embedded SQL
- Call Level Interface (CLI) development environment (compatible with ODBC from Microsoft)
- Java Database Connectivity (JDBC)
- DB2 application programming interfaces to access database utilities

DB2 SDK supports several programming languages (including COBOL, C, and C++) for application development and provides precompilers for the supported languages.

Accessing DB2 Data from the Web

The popularity of the Internet and the World Wide Web has created a demand for Web access to enterprise data. Java Database Connectivity (JDBC) and Net.Data are provided with DB2 to allow you to create Web-based applications that access data in DB2 databases.

Using Java Database Connectivity

Use JDBC to create applications or applets that access data in DB2 databases. You can run JDBC applications from any system that has the DB2 Client Application Enabler software installed; a Web browser and a Web server aren't required.

You can run JDBC applets inside HTML Web pages on any system with a Java-enabled browser, regardless of your client's platform. No additional software is required on your client system beyond this browser. The processing of JDBC applets is shared between the client and the server.

The JDBC server and the DB2 Client Application Enabler must reside on the same machine as the Web server. The JDBC server calls in to the DB2 Client Application Enabler to connect to local or remote databases, or into MVS/ESA to connect to host databases. When a connection to a DB2 database is requested by the applet, the JDBC client opens a TCP/IP socket to the JDBC server on the machine where the Web server is running. Figure 1.8 shows an example of Java-enabled browsers remotely accessing data from DB2 databases.

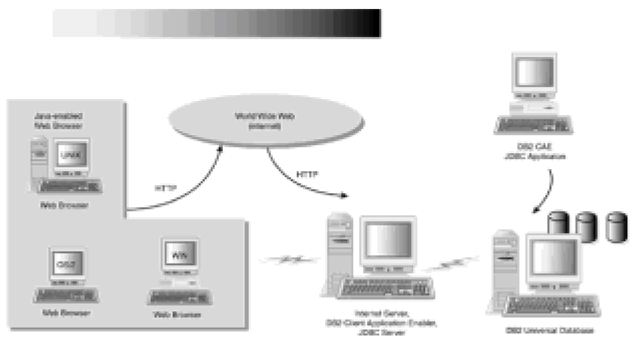


Figure 1.8: Using JDBC to access Internet data stored on DB2.

Using Net.Data

Use Net.Data to create applications that are stored on a Web server and can be viewed from any Web browser. While viewing these documents, users can select automated queries or define new ones that retrieve the specified information directly from a DB2 database.

Automated queries don't require user input; they are links in the HTML document and, when selected, trigger existing SQL queries and return the results from the DB2 database. These links can be triggered repeatedly to access current DB2 data.

Customized queries require user input. Users define the search criteria by selecting options from list boxes or by typing values in entry fields. They submit the search by clicking a pushbutton. Net.Data uses the user-supplied information to dynamically build a complete SQL statement and then sends the query to the database.

Tip For a demonstration of Net.Data applications, check out the Data Management link on IBM's software Web page (`http://www.software.ibm.com`).

You can install Net.Data with the server, DB2 Universal Database, to allow local access to databases. Net.Data can be installed with DB2 Client Application Enabler software to allow remote access to databases. In both cases, Net.Data and a Web server must be installed on the same system. Figure 1.9 shows an example of Net.Data remotely accessing data from DB2 databases.

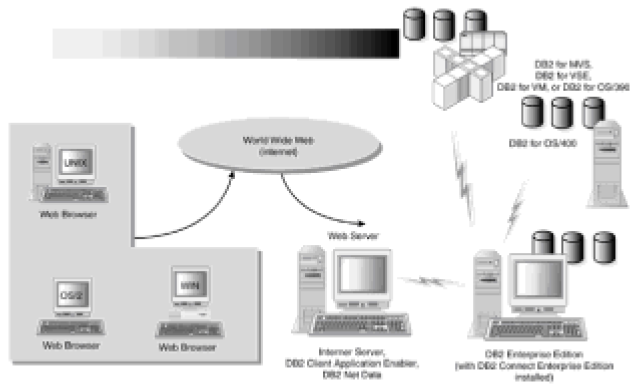

DB2 for MVS, DB2 for VSE, DB2 for VM, or DB2 for OS/390
DB2 for OS/400
World Wide Web (Internet)
UNIX
Web Browser
Web Server
OS/2
Web Browser
WIN
Web Browser
Internet Server, DB2 Client Application Enabler, DB2 Net.Data
DB2 Enterprise Edition (with DB2 Connect Enterprise Edition installed)

Figure 1.9: Using Net.Data to access Internet data stored on DB2.

Using DB2 Parallelism

DB2 extends the database manager to the parallel, multinode environment. A *database partition* is part of a database that consists of its own data, indexes, configuration files, and transaction logs. A database partition is sometimes called a *node* or *database node*.

Because data is divided across database partitions, you can use the power of multiple processors on multiple physical nodes to satisfy requests for information. Data retrieval and update requests are decomposed automatically into subrequests and executed inparallel among the applicable database partitions. That databases are split across database partitions is transparent to SQL users.

DB2 offers several parallel features to enhance the performance and efficiency of your databases. By using DB2 Universal Database Enterprise Edition–Extended, you can partition the database across multiple, independent computers connected by a LAN. This way, an application can use a database that's simply too large for a single computer to handle efficiently.

By using DB2 in a symmetric multiprocessor (SMP) environment, you can speed up the execution of a single SQL query by exploiting multiple processors' disks and memory. This way, workload can be divided more evenly among processors, thereby achieving better scalability. This is possible because all processors have the choice to work on all data, which is analogous to a single queue serviced by multiple bank tellers.

Overview of the DB2 Enterprise Servers

This section presents a brief overview of the enterprise server members of the DB2 family: DB2 for OS/390, DB2 for AS/400, and DB2 for VSE and VM. All DB2 Client Application Enablers described in this book provide access to these products with the help of DB2 Connect. This extends DB2 Call Level Interface, ODBC, and JDBC support to all DB2 platforms.

Note Much of the discussion of the DB2 Universal Database products applies to the other DB2 products as well, but not all. For more specific information about each product, refer to the documentation available with them.

These products are covered in the following sections:

- DB2 for OS/390
- DB2 for AS/400
- DB2 for VSE and VM

DB2 for OS/390

Since its introduction, DB2 for OS/390 (initially known simply as *DB2* before DB2 was available on other platforms) has been the database of choice for large companies and organizations. DB2 for OS/390 specializes in providing high-performance service for large-scale online transaction processing systems, complex decision support systems, data warehousing, and business intelligence needs. DB2 for OS/390 also gives you support for client/server processing, stored procedures, object-oriented programming, and support for large objects. Highlights of DB2 for OS/390 are as follows:

- It supports Net.Data, from which you can set up e-commerce applications.
- It can manage terabytes of data on behalf of thousands of users.
- It enables data sharing, which makes it possible for a group of DB2 subsystems to have concurrent access to the same data without replication and keeps data available

during outages.

- It extends network support to include native TCP/IP, which allows you to connect various DRDA clients without using an SNA gateway machine.
- It supports powerful desktop tools such as VisualAge, PowerBuilder, Visual Basic, and Lotus Approach to access data managed by DB2 for OS/390.
- It can keep data available 24 hours a day, seven days a week. Many activities that usually require a database manager to stop running don't affect its operation.
- It achieves excellent performance through use of a cost-based optimizer, large buffer pools for efficient I/O processing, parallel query processing, and concurrency when running utilities and SQL applications.
- It uses sophisticated locking and logging methods and flexible authorization techniques to keep your data valid, accurate, and consistent across systems.
- It's enabled with year 2000 support.

If you have large amounts of enterprise data that must be accessed from all your corporate locations, use DB2 for OS/390 on a System/390.

DB2 for AS/400

Formerly known as the AS/400 database manager, DB2 for AS/400 integrates with the AS/400 hardware and operating system for optimal performance in running centralized or distributed commercial database applications on single or multiple systems. This product is compatible with the entire DB2 product line, from DB2 on mainframes to DB2 for Windows NT. It enables an AS/400 host to operate equally well with centralized database applications, or to act as a database server in a heterogeneous client/server network.

Some features of DB2 for AS/400 include the following:

- **World Wide Web accessibility.** With a standard browser and a Web connection, your company and customers can easily tap into your data warehouse.
- **Parallel database.** DB2 symmetric multiprocessing, DB2 multisystem, and parallel I/O are some of the choices available on an AS/400 to improve performance of very large databases.
- **Standard features.** All standard DB2 functions are available, including declarative referential integrity, data replication, advanced SQL optimizer, and enablement with year 2000 support.

With state-of-the-art database functions, reliable performance, and conformance to open standards, DB2 for AS/400 provides stable and mature support in a client/server environment.

DB2 for VSE and VM

Like MVS/ESA and AS/400, the VSE and VM database platforms let you store large amounts of enterprise data. DB2 for VSE and VM is easy to use, conforms with standards across multiple platforms, offers flexible application development, and provides excellent performance. It can be used as a server with a number of clients on the workstation platforms.

Some of its other benefits are that it

- Implements DRDA2 Distributed Unit of Work (DUOW) Application Server support. This way, you can read and update in multiple locations within a single unit of work. Integrity of data is protected with a two-phase commitment.

- Supports DataPropagator-Relational capture. You can manage and distribute data across multiple database platforms throughout your enterprise with DataPropagator-Relational capture support. This enables you to maintain consistent and synchronized information throughout your enterprise, even if the majority of your staff is mobile.

- Includes an uncommitted read option that improves productivity for database users. Multiple users can access data in read-only mode, thus improving their access time. All data can be accessed in read-only mode, including data that may have been changed.

- Improves database administration by automating many routine database administration tasks through the IBM SQL Master for VSE and VM program.

- Ships with a copy of the DB2 development editions to enable application development on common workstation and Internet platforms.

- Is enabled with Year 2000 support.

You can achieve high performance and exploit the ESA technology with support for VM data spaces on the VM/ESA platform and the virtual disk feature on the VSE/ESA platform. For more information on the DB2 for VSE and VM product, see the Web site at `http://www.software.ibm.com/data/db2/vse-vm`.

Summary

In today's lesson, you saw that DB2 Universal Database is a relational database management system that supports a variety of client and server platforms and communication protocols. DB2 Universal Database is made of several administration tools, such as the Control Center and the Command Center, that simplify the task of managing your environment and data.

Data stored in enterprise database management systems such as DB2 for MVS, DB2 for OS/390, and DB2 for AS/400 can be accessed directly from a DB2 client workstation through the DB2 Connect product.

You can create or use many different types of applications to access DB2 data. You can write your own applications by using languages such as C, C++, or COBOL, create Web-based applications by using Java or Net.Data, or use existing applications such as Microsoft Access or Lotus Approach.

What Comes Next?

On Day 2, "Exploring the Capabilities of DB2 Universal Server," you learn about the relational concepts that make up DB2 Universal Database.

Q&A

Q On what platforms can I run DB2 Universal Database?

A The server edition of DB2 Universal Database Version 5 runs on AIX, HP-UX, OS/2, Solaris, and Windows NT. The client edition of DB2 runs the same platforms the server, as well as DOS, Macintosh, Windows 95, SCO OpenServer, SINIX, SGI IRIX, and Windows 3.1. The personal edition of DB2 runs on OS/2, Windows NT, and

Windows 95. The parallel edition of DB2 runs on AIX, Solaris, and Windows NT.

Q What tools are available to help me administer and manage DB2 databases?

A The main tool available with DB2 is the Control Center, from which you can launch the Command Center, the Script Center, the Journal, the Alert Center, Tools Setting, Performance Monitor, and Visual Explain. Also available is the Client Configuration Assistant.

Q I need to access data on enterprise systems. What software is required?

A You need one of the DB2 Connect products to access data on enterprise systems including DB2 for MVS/ESA, DB2 for OS/390, DB2 for VSE and VM, and DB2 for AS/400. You can use the DB2 Connect Personal Edition product to directly access the enterprise data from your workstation. You can use the DB2 Connect Enterprise Edition as a gateway to have many clients in your network access enterprise data.

Workshop

The purpose of the Workshop is to allow you to test your knowledge of the material covered in the lesson. See if you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson. The answers appear in Appendix A, "Answers to Quiz Questions."

Quiz

1. What's a local application? What's a remote application?
2. What are the two ways to use Java Database Connectivity to access DB2 data?
3. What's the name of the DB2 product that provides a parallel, multinode database environment?
4. Name the interfaces that you can use when creating applications with the DB2 Software Developer's Edition.
5. DB2's Workgoup and Enterprise Editions are largely the same, except for two differences. What are the differences? When would you use one edition over the other?

Day 2: Exploring the Capabilities of DB2 Universal Server

Overview

The major components of the DB2 Universal Database relational database management system include the database engine and facilities to access data, such as the Command Center, command-line processor, the Control Center, and application interfaces. A rich suite of tools also is available for the DB2 environment. Figure 2.1 gives an overview of the major components.

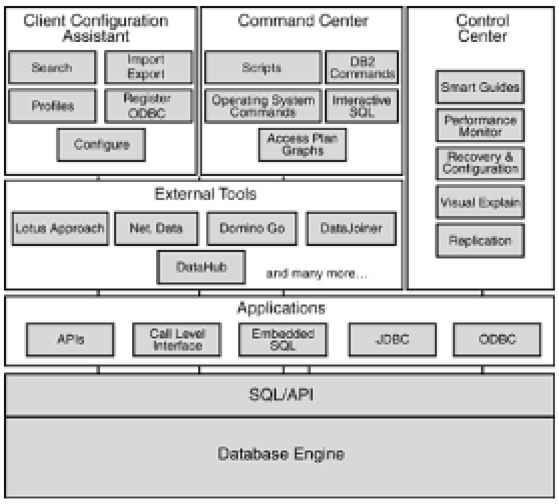


Figure 2.1: Components of DB2 Universal Database.

The database engine provides the base functions of the DB2 relational database management system. It manages the data, controls all access to it, generates packages, generates optimized paths, provides transaction management, ensures data integrity and data protection, and provides concurrency control. All data access takes place through the SQL interface.

This lesson presents the basic elements of a relational database and introduces various database objects, the development environment, and the database administration tools. The basic elements of the database engine are database objects, system catalogs, directories, and configuration files. These elements are described in the next section.

The Components of DB2 Universal Database

You can manage the database and access data through several facilities:

- The *Control Center* provides a graphical interface to functions such as recovering a database, defining directories, configuring the system, managing media, Visual Explain, and Performance Monitor. See Day 4, "Getting Started," for details.

- The *Command Center* provides a graphical interface to the command-line processor, enabling access to data through the use of database commands and interactive SQL. See Day 10, "Accessing the Data," for more information.

- The *Client Configuration Assistant* (*CCA*) automates protocol configuration for clients accessing remote database servers. See Day 16, "Administering Clients with the CCA," for more information.

- A series of *SmartGuides* provide step-by-step guidance to functions such as creating a table or tuning performance. See Day 8, "Creating Databases," for instructions on using the Create Database SmartGuide. The other SmartGuides are covered throughout this book.

- *Applications* access data by using embedded SQL, the Call Level Interface, Open Database Connectivity (ODBC), Java Database Connectivity (JDBC), or application programming interfaces (APIs). DB2 Universal Database provides many APIs to manipulate the database environment. External tools supported by DB2 Universal Database also use these APIs. These interfaces are described later in the section "Application Programming Interfaces (APIs)."

- *External tools* provide a variety of additional functions. For an example of accessing data through Lotus Approach, see Day 10, "Accessing the Data."

The server version of DB2 contains communication support that enables access from remote workstations in a LAN environment.

Day 2: Exploring the Capabilities of DB2 Universal Server

Overview

The major components of the DB2 Universal Database relational database management system include the database engine and facilities to access data, such as the Command Center, command-line processor, the Control Center, and application interfaces. A rich suite of tools also is available for the DB2 environment. Figure 2.1 gives an overview of the major components.

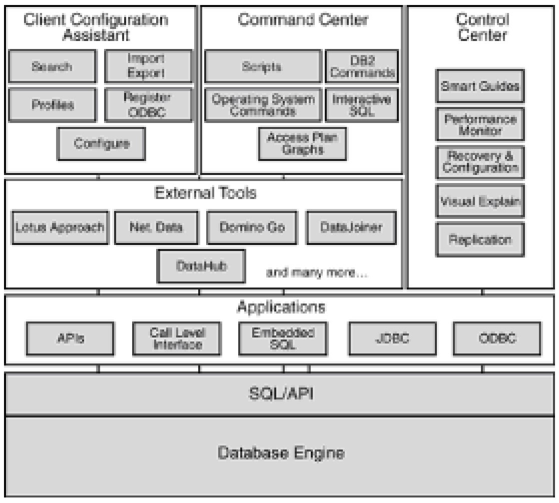


Figure 2.1: Components of DB2 Universal Database.

The database engine provides the base functions of the DB2 relational database management system. It manages the data, controls all access to it, generates packages, generates optimized paths, provides transaction management, ensures data integrity and data protection, and provides concurrency control. All data access takes place through the SQL interface.

This lesson presents the basic elements of a relational database and introduces various database objects, the development environment, and the database administration tools. The basic elements of the database engine are database objects, system catalogs, directories, and configuration files. These elements are described in the next section.

The Components of DB2 Universal Database

You can manage the database and access data through several facilities:

- The *Control Center* provides a graphical interface to functions such as recovering a database, defining directories, configuring the system, managing media, Visual Explain, and Performance Monitor. See Day 4, "Getting Started," for details.

- The *Command Center* provides a graphical interface to the command-line processor, enabling access to data through the use of database commands and interactive SQL.

See Day 10, "Accessing the Data," for more information.

- The *Client Configuration Assistant* (*CCA*) automates protocol configuration for clients accessing remote database servers. See Day 16, "Administering Clients with the CCA," for more information.

- A series of *SmartGuides* provide step-by-step guidance to functions such as creating a table or tuning performance. See Day 8, "Creating Databases," for instructions on using the Create Database SmartGuide. The other SmartGuides are covered throughout this book.

- *Applications* access data by using embedded SQL, the Call Level Interface, Open Database Connectivity (ODBC), Java Database Connectivity (JDBC), or application programming interfaces (APIs). DB2 Universal Database provides many APIs to manipulate the database environment. External tools supported by DB2 Universal Database also use these APIs. These interfaces are described later in the section "Application Programming Interfaces (APIs)."

- *External tools* provide a variety of additional functions. For an example of accessing data through Lotus Approach, see Day 10, "Accessing the Data."

The server version of DB2 contains communication support that enables access from remote workstations in a LAN environment.

What Is a Relational Database?

A *relational database* presents data as a collection of tables. A *table* is a collection of rows and columns. SQL is used to retrieve or update data by specifying columns, tables, and the various relationships between them.

SQL (*Structured Query Language*) is a standardized language for defining and manipulating data in a relational database. SQL statements are executed by a *database manager*, a computer program that manages the data.

The following sections describe these basic elements used by DB2 Universal Database to define and manage databases:

Database objects	Directories
Data integrity	Storage objects
Object relational capabilities	Configuration files
System catalog tables	Recovery objects
Instances	Database objects

Figure 2.2 shows the major relational database objects in a DB2 Universal Database common server. Examples of objects are databases, table spaces, tables, views, indexes, packages, functions, or data types.

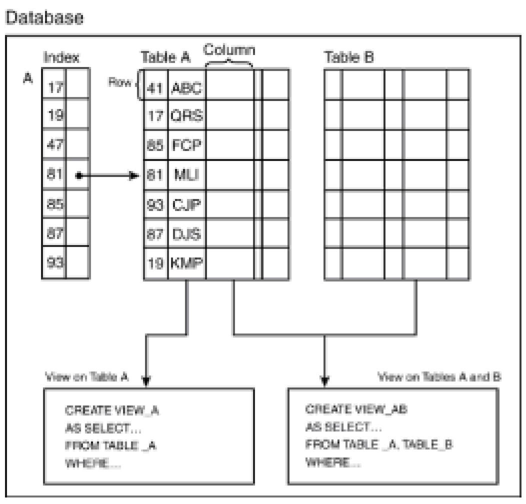


Figure 2.2: Basic elements: Relational database objects.

Each database includes a set of *system catalog tables* to describe the logical and physical structure of the data, a *configuration file* to contain the parameter values allocated for the database, and a *recovery log* to log ongoing transactions and archivable transactions.

Tables, Columns, and Rows

A *table* consists of data logically arranged in columns and rows. Figure 2.3 shows a simple table with several rows and columns. Data in the table is logically related, with relationships defined between different tables in a database. Data is viewed and manipulated based on mathematical principles and operations called *relations*, and table data is accessed via SQL.

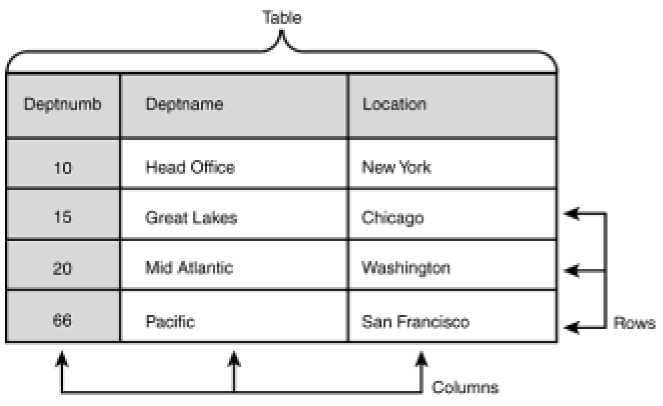


Table
Rows
Columns

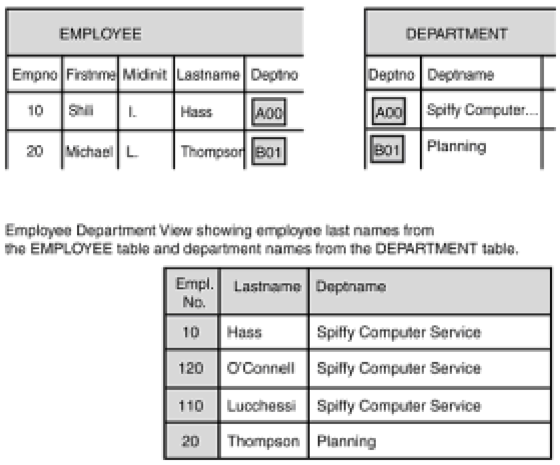
Deptnumb	Deptname	Location
10	Head Office	New York
15	Great Lakes	Chicago
20	Mid Atlantic	Washington
66	Pacific	San Francisco

Figure 2.3: Tables consist of data arranged by columns and rows.

A table can contain up to 500 columns, any number of rows, and up to 64GB of data (not including large objects). A *column* is a set of values of the same data type. A *row* consists of a sequence of values, one for each column. The rows aren't necessarily ordered within the table. To order the result set, you have to explicitly specify ordering in the SQL statement that selects data from the table. At the intersection of every column and row is a specific data item called a *value*.

Views

A *view* is the result of a query of one or more tables. A view looks like a real table, but actually it's just a representation of the data from one or more tables. You can think of a view as a logical or virtual table; it exists only as a result of a query. A view can include all or some of the columns contained in the tables on which it's defined. Figure 2.4 shows the view called `DEPTNAME`, created by combining some columns from the`EMPLOYEE` table and others from the `DEPARTMENT` table.



EMPLOYEE				
Empno	Firstnme	Midinit	Lastname	Deptno
10	Shili	I.	Hass	A00
20	Michael	L.	Thompson	B01

DEPARTMENT	
Deptno	Deptname
A00	Spiffy Computer...
B01	Planning

Employee Department View showing employee last names from the EMPLOYEE table and department names from the DEPARTMENT table.

Empl. No.	Lastname	Deptname
10	Hass	Spiffy Computer Service
120	O'Connell	Spiffy Computer Service
110	Lucchessi	Spiffy Computer Service
20	Thompson	Planning

Figure 2.4: A simple view.

A view efficiently represents data without having to maintain it. A view isn't an actual table and requires no permanent storage; instead, a *virtual table* is created and used.

Views enable multiple users to see different presentations of the same data. For example, several users may be accessing a table of data about some employees but not others, and other users may see some data about all employees but not their salaries. Each user is operating on a view derived from the real table. Each view appears to be a table and has a name of its own.

Tip One advantage to using views is that you can use them to control access to sensitive data. Different people can have access to different columns or rows of the data.

Schemas

A *schema* is a set of objects in the database and is used to logically group the objects in the database. A schema can contain database objects such as tables, views, indexes, packages, distinct types, functions, or triggers.

A schema is used as the first part of a two-part object name. When an object is created, you can assign it to a specific schema. If you don't specify a schema, the object is assigned to the default schema, whose name is usually the username of the person who created the object. The second part of the name is the name of the object. For example, a user named Smith might have a table called `SMITH.PAYROLL`.

Keys

A *key* is a set of columns that identifies a particular row. Without a key, you can't identify or access a row of data. A key composed of more than one column is called a *composite key*.

Unique and Primary Keys

An optional *unique key* is defined to have no two of its values the same. The columns of a unique key can't contain null values. This constraint is enforced by the database manager when data is inserted or updated. A table can have multiple unique keys. Unique keys are defined when the table is created or altered.

A *primary key* is a special kind of unique key that uniquely identifies a row of a table. The columns of a primary key can't contain null values.

A table can have only one primary key, but it can have multiple unique keys. You can define additional unique keys by creating unique indexes (see the later section "Indexes").

> **Tip** Defining a primary key for a new table is recommended because uniquely identifying each row speeds row access. For example, it's faster to access an account by its unique account number than it is to search through all the accounts. It's best to define a primary key at the time you create a table, although you can add a primary key to an existing table.

Primary keys support updates from many ODBC applications. When you choose a column as the primary key, the database checks each new row for a unique value in that column, rejecting any duplicates.

You can choose multiple columns as the primary key. For example, you might want to create a primary key on the first name and last name columns of an employee table to ensure uniqueness. A unique index is created automatically for the columns that make up the primary key.

Foreign Keys

A *foreign key* defines a relationship between two or more tables. The foreign key's value must match the corresponding primary or unique key of the dependent table. For example, the `DEPTNO` column of the `EMPLOYEE` table has a foreign key defined on the `DEPTNO` column of the `DEPARTMENT` table. This means that an employee cannot be assigned to a department unless the department already exists.

A table can have zero or more foreign keys. The value of the composite foreign key is null if any component of the value is null. Foreign keys are optional and are defined when a table is created or altered.

Indexes

When you create a unique key, a primary key, or a foreign key, an index is created. An *index* is a set of keys, each pointing to rows in a table. The `EMPLOYEE` table in Figure 2.5, for example, has an index based on the employee numbers in the table. This key value provides a pointer to the rows in the table; for example, employee number 10 points to employee Haas.

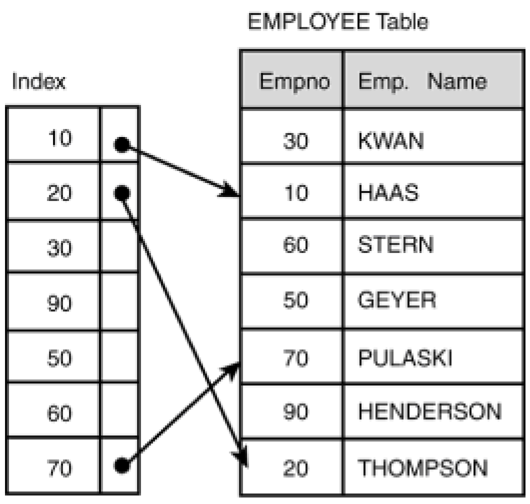


Figure 2.5: Indexes and keys.

An index is a separate object from the table data. When an index is created, the database manager builds this structure and maintains it automatically.

An index gives more efficient access to rows in a table by creating a direct path to the data through these pointers. The optimizer takes indexes into consideration when determining the fastest access path to data.

Unique indexes can be created to ensure uniqueness of the index key. An *index key* is a column or ordered collection of columns on which an index is defined. Using a unique index ensures that each value in the indexed column or columns is unique. For example, in the `EMPLOYEE` table, the column `EMPNO` is a unique key, meaning that each employee in the table must have a different employee number.

Packages

A *package* is an object that includes all the information needed to process specific SQL statements from a single source file. Created during the binding of an application program, a package contains the information needed by the database manager to access data in the most efficient way for a given program. Binding is discussed in more detail in Day 10, "Accessing the Data," in the section "Binding Database Utilities."

Data Types

Data types define acceptable values for constants, columns, host variables, functions, expressions, and special registers.

The data types supported by DB2 include the following:

- A *character string* is a sequence of bytes. The data types in this category include fixed-length character strings, variable-length character strings (`VARCHAR`, `LONG VARCHAR`, and `CLOB`), and `NUL`-terminated character strings.
- All *numbers* have a sign and a precision. *Precision* is the number of bits or digits excluding the sign. The data types in this category include small integer (`SMALLINT`), large integer (`INTEGER`), single-precision floating point (`REAL`), double-precision floating point (`DOUBLE` or `FLOAT`), and decimal (`DECIMAL` or `NUMERIC`).
- The *date/time* data types aren't strings or numbers: date, time, timestamp, string representations of date/time values, date strings, time strings, and timestamp strings.

- The term *large object* refers to any `BLOB`, `CLOB`, or `DBCLOB` data type, including character large object (`CLOB`) strings, double-byte character large object (`DBCLOB`) strings, and binary large objects (`BLOB`).

- A *graphic string* is a sequence of bytes representing double-byte character data. Graphic strings include fixed-length graphic strings, variable-length graphic strings (`VARGRAPHIC`, `LONG VARGRAPHIC`, and `DBCLOB`), and `NUL`-terminated graphic strings.

- A *binary string* is a sequence of bytes. Unlike character strings, which usually contain text data, binary strings are used to hold non-traditional data, such as pictures.

Data types are defined when a table is created.

Distinct Types

Application developers can create distinct types to represent their unique data. A *distinct type* is a user-defined data type that shares its internal representation with an existing type but is considered to be a separate and incompatible type for semantic purposes. For example, distinct types can be created for different currencies. The different currencies can't be compared directly, but a function can be written to convert one to the other before comparing.

Distinct types, like built-in types, can be used for columns of tables as well as parameters of functions. For example, you can define a data type such as `ANGLE` (which varies between 1 and 36) and a set of user-defined functions (UDFs) to act on it, such as `SINE`, `COSINE`, and `TANGENT`.

Large Object (LOB) Support

Large object data types enable multimedia objects such as documents, video, image, and voice to be stored in the database and manipulated like other database objects. Multiple LOB columns, with a maximum size of 2GB each, can be defined per table. The exact number depends on the size defined for each column. LOB data is maintained directly in the database, and SQL is used for storage and retrieval.

Examples of LOB usage include the following:

- Banks storing customer's signatures

- Resumes kept on file with data and photos

- Voice prints for authentication

- Audio clips for a CD collection

Because LOB values can be very large, transferring them from the database server to client application programs can be time-consuming. LOB values are typically processed one piece at a time, however, rather than as a whole. For those cases where an application doesn't need (or want) the entire LOB value to be stored in application memory, it can reference this value by using a large object locator variable. Subsequent statements can then use the locators to perform operations on the data without necessarily retrieving the entire large object.

DB2 provides mechanisms to enable application programs to manage large object data in efficient ways. One of these mechanisms is *locator variables*, used to reduce memory requirements for applications and improve performance by reducing the data flow between the client (where the application resides) and the server (where the database resides). For example, you can use locator variables to retrieve a single chapter from a 35-chapter book stored as a large object. After you retrieve the chapter, you can edit it as required and then replace it. All this is accomplished without the need to retrieve the entire book.

Another mechanism is *file reference variables*, which retrieve a large object directly to a file or update a large object in a table directly from a file. File reference variables are used to reduce the storage requirements for the applications because they don't need to hold the large object data in memory.

The combination of distinct types and LOBs gives you considerable power. You're no longer restricted to using the built-in data types provided by DB2 to model your business data and to capture that data's semantics. You can use distinct types to define large, complex data structures for advanced applications.

Functions

A *function* is a subroutine that performs a distinct set of tasks and usually returns a single set of values. DB2 provides column functions (such as `COUNT`, `MIN`, and `MAX`) and scalar functions (such as `SUBSTR`, `DATE`, and `DAYS`).

Application developers can create their own suite of functions oriented to specific applications or domains, such as scientific and business functions. This expands the availability of common functions, gives greater flexibility to the system, and enables the database

manager to capture more of the data's semantics. These functions can operate over all database types, including LOBs and distinct types.

User-defined functions (*UDFs*) are particularly flexible, as they can operate on data managed within or outside DB2 Universal Database. For example, a function can be enabled to send an electronic message, update a flat file, or perform some other operation. Figure 2.6 shows an example of how UDFs work.

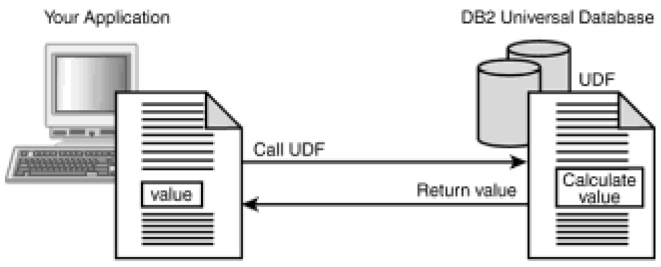


Figure 2.6: Basic elements: User-defined functions.

A retail store could define a price data type for tracking the cost of items it sells. This store might also want to define a `sales_tax` function, which would take a given price as input, compute the applicable sales tax, and return this data to the requesting user or application.

DB2 provides a rich library of UDFs. The library includes a set of mathematical functions such as `SIN`, `COS`, and `TAN`; scientific functions such as `RADIANS`, `LOG10`, and `POWER`; general-purpose functions such as `LEFT`, `DIFFERENCE`, and `UCASE`; and many more.

User-defined functions also support object-oriented programming in your applications. UDFs provide for abstraction, enabling you to define the methods that can be used to control access to the underlying data of an object, protecting it from direct manipulation and possible corruption.

Data Integrity

Data integrity is fundamental to any database management system. Whenever data is shared, there is a need to manage and control operations that ensure the accuracy of the values within database tables. DB2 Universal Database ensures data integrity through the previously mentioned transaction support and concurrency control, as well as through the features described in the following sections.

Roll-Forward Recovery

When you lose online data because of a media failure, such as a hard disk crash, roll-forward recovery enables you to recover it by applying log journal information against

the restored database. Log journals contain the changes made to the database since the last backup. After these journals are applied, the database will be in the same state as it was before the failure.

Constraints

To protect data and to define relationships between your data, you usually define *business rules*. Rules define what data values are valid for a column in a table, or how columns in one or more tables are related to each other.

DB2 provides constraints as a way to enforce those rules with the database system. A *constraint* is a mechanism that ensures that certain conditions relating columns and tables are maintained—for example, an employee number must be unique from person to person and, as such, is under some constraint. By using the database system to enforce business rules, you don't have to write code in your application to enforce them. If a business rule applies to only one application, however, you should code it in the application rather than use a global database constraint.

DB2 provides different kinds of constraints:

- `UNIQUE`
- Referential constraints and relationships such as `NOT NULL` and `FOREIGN KEY`
- `CHECK`

You define constraints when you create or alter a table.

Unique Constraints

Unique constraints ensure that the values in a set of columns are unique and not null for all rows in the table (not-null constraints prevent null values from being entered into a column). For example, a typical unique constraint on a `DEPARTMENT` table may be that the combination of the location number and department number be unique. These constraints are added to the database through the definition of primary keys or unique indexes.

You can optionally define unique constraints when you create or alter a table. The database manager enforces constraints during insert and update operations, ensuring data integrity.

Referential Constraints and Relationships

Another important feature that provides robust data integrity is *referential integrity*, which lets you define required relationships between and within tables. Referential constraints, declared when a table is defined, ensure the consistency of data values between related columns in different tables. DB2 Universal Database maintains these relationships, so you don't need to program this function into applications.

Relationships are expressed as *referential constraints*, which require that all values of a given attribute or column of a table also exist in some other table or column. For example, a typical referential constraint might require that every employee be a member of an existing department. By defining unique constraints and foreign keys, you can define a relationship between the `EMPLOYEE` and `DEPARTMENT` tables and consequently enforce this business rule (see Figure 2.7). With this relationship, you can control updates to the employee table by ensuring that the department number is valid, as defined in the `DEPARTMENT` table. The database can reject any employee record with an invalid department number. This ensures data integrity.

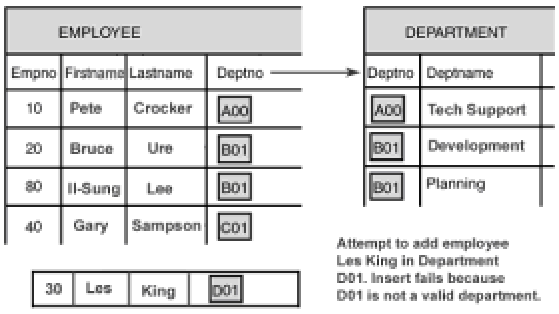


Figure 2.7: Enforcing referential integrity.

To establish a referential constraint, you define the department number in the `EMPLOYEE` table as the foreign key, and the department number in the `DEPARTMENT` table as theprimary key. A foreign key in the `EMPLOYEE` table references the primary key of the`DEPARTMENT` table. This creates a constraint so that any value in the foreign key must match an existing value in the referenced key.

You can also choose the action to be taken when rows of the parent table are deleted. In the preceding example, you can decide that when a department is deleted, all employees in that department are also deleted. Conversely, you could decide that if any employees are still members of the department, the department can't be deleted.

Check Constraints

Check constraints are generally used to enforce business rules not covered by key uniqueness or referential integrity constraints. A check constraint is a rule in the database that specifies the values allowed in one or more columns of every row of a table. This includes specifying a range or domain that valid entries must exist in. For example, you can define constraints on the `EMPLOYEE` table that specify that the job of an employee can be only one of Sales, Manager, or Clerk. Each entry in this column must exist in this domain to be valid.

Check constraints also enable you to have a single constraint defined at the table level involving multiple columns in the table. For example, on a table that contains columns for an employee's salary, bonus, and date of hire, you can define a single constraint to ensure that no employees with more than one year of service can receive a total annual income (including salary and bonus) of less than $35,000.

Table check constraints specify evaluated conditions for each row of a table. You can specify check constraints on individual columns.

User-Defined Types (UDTs)

Every data element in the database is stored in a column of a table, and each column is defined to have a data type. The data type places limits on the types of values you can put in the column and the operations you can perform on them. For example, a column of integers can contain only numbers within a fixed range. DB2 includes a set of built-in data types with defined characteristics and behaviors: character strings, numerics, datetime values, and large objects.

Sometimes, however, the built-in data types might not serve the needs of your applications. DB2 provides user-defined types that enable you to define the distinct data types you need for your applications.

UDTs are based on the built-in data types. When you define a UDT, you also define its valid operations. For example, you might define a money data type based on the decimal data type. For the money data type, however, you might allow only addition and subtraction operations between two money data types, not multiplication and division operations.

User-defined distinct types ensure data integrity through *strong typing*, which guarantees that only functions and operations defined on the distinct type can be applied to the type. For example, the system doesn't enable you to directly compare a picture type with an audio type, even though they share the same underlying type. If you want to do such a comparison, you can *cast* (convert) one or both of the values to a common value by using a UDF. Encapsulation ensures that the behavior of distinct types is restricted by the functions and operators that can be applied to them.

Triggers

A *trigger* defines a set of actions that are executed or triggered by an update, insert, or delete operation on a specified base table. When such a SQL operation is executed, the trigger is said to be *activated*. You can activate the trigger before or after the SQL operation. Triggers can be used to perform validation of input data, automatically generate a value for a newly inserted row, read from other tables for audit-trail purposes, or support alerts through electronic mail.

The use of triggers promotes faster application development because they're stored in the relational database. The actions that triggers perform don't have to be coded in each application. Maintenance of applications and databases is easier because if a business policy changes, only a change in the corresponding trigger is needed instead of changes in each application program. Multiple triggers per table can be set up, and one trigger can cascade and initiate other triggers.

Figure 2.8 gives an example of how triggers work. It shows two tables: one for employees of the sales department and one that lists the total number of employees. When a new employee is added to the sales department, a trigger is activated that updates the information in the table showing the total number of employees. If you didn't have a trigger, you would have to perform a query that goes through the sales department and counts each employee.

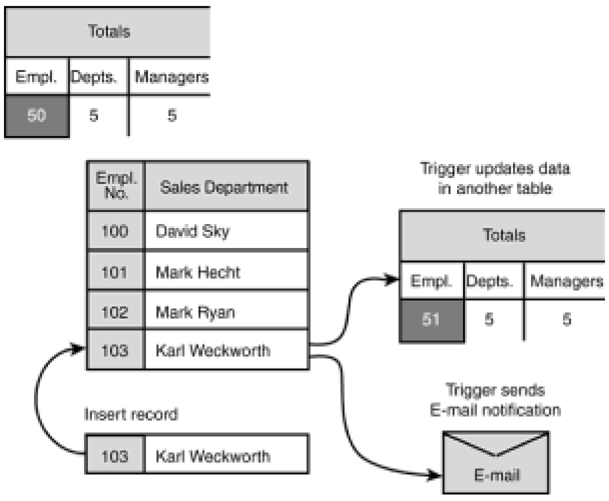


Figure 2.8: Basic elements: Triggers.

Triggers provide integrity checking by doing complex, cross-table validation of data beyond the scope of referential integrity and check constraints.

One powerful use of a trigger is to globally enforce a business rule. For instance, a trigger could be used to ensure that a manager of a department earns more than the minimum salary of his employees.

Triggers create greater flexibility within the system and enable the database manager to capture more of the data semantics. For example, a trigger could be written to verify that orders don't exceed a customer's credit limit. If an order does, the trigger could refuse to place the order.

When combined with UDFs, triggers can be used to implement alerts that signal users or other applications that a given event has occurred within the database manager. For example, if you want to notify a certain member of the personnel department whenever an engineer's salary is changed to exceed $100,000, DB2's triggering mechanism can support this function. Such support provides considerable power and flexibility.

Object Relational Capabilities

DB2 Universal Database provides several features that enable object relational capabilities. These features include large object support, user-defined types, user-defined functions, and triggers. These features offer you many benefits, including the capability to reuse code, create objects that more closely imitate the way you think of things, keep objects and functions for use by multiple applications closer to data, and create more flexible applications.

Large object support enables multimedia data such as documents, video, image, and voice to be stored in the database and manipulated like other database objects. Figure 2.9 shows some of the many large objects that you can store and manipulate in DB2 Universal Database.

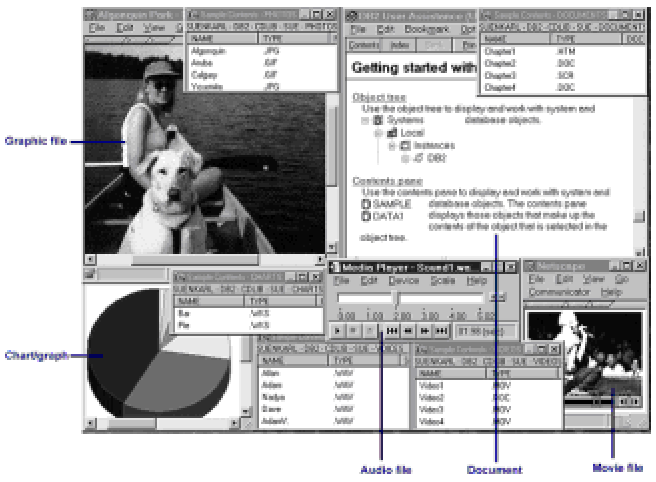


Figure 2.9: Basic elements: Large objects.

The capability to define distinct types gives you freedom beyond the system-provided types. Distinct types are just one building block for object-oriented extensions.

User-defined functions provide the *abstraction* of the object, which defines its behavior and central characteristics. Defining scalar functions to use with distinct types enables the creation of *methods*. The creation of methods is a central concept of object orientation.

With user-defined functions, you also can overload. *Overloading* enables you to use the same name for different actions against data but have the system examine the parameters to determine how to execute the function. For example, you can create a `length` function on `AUDIO`, `TEXT`, and `PICTURE` data types. For the `AUDIO` data type, you want `length` to return the number of seconds; for `TEXT`, the number of words; and for `PICTURE`, the number of bytes.

User-defined functions also provide encapsulation of objects. *Encapsulation* is the means of controlling access to the underlying data of an object. Data is hidden or protected from direct manipulation and, therefore, protected from corruption.

A trigger defines a set of actions that are executed at, or *triggered* by, an update operation on a specified base table. Triggers can be used to enforce complex business rules that may involve objects.

The synergy among these object-orientation extensions gives more power to application developers. Complex data structures can be modeled as user-defined types on large objects, with their methods implemented as user-defined functions and their integrity rules specified as triggers. For example, `MATRIX` could be a user-defined type defined on a binary large object, and user-defined functions could be `MULTIPLY` (which does matrix multiplication) and `ROW` (which retrieves a given row of a matrix).

System Catalog Tables

Each database includes a set of system catalog tables that describe the logical and physical structure of the data. These tables contain information about database objects such as tables, views, packages, referential integrity relationships, functions, distinct types, triggers, and indexes, as well as security information about the authority that users have for these objects. They're created when the database is created and are updated in the course of normal operation. You can't explicitly create or drop them, but you can query and view their contents. Some tables can be modified through views and Visual Explain to update statistics that help you model different data characteristics.

Instances

A DB2 *instance* is a logical database manager environment where you catalog databases and set configuration parameters. Depending on your needs, you can create more than one instance on a single server. This means that you can create several instances of the same product on a physical machine and have them running concurrently. This provides flexibility in setting up environments. You can use multiple DB2 instances to do the following:

- Use one instance for a development environment and another instance for a production environment.
- Separately tune a database instance for a particular environment.
- Protect access to sensitive information. For example, you may want to have your payroll database protected on its own instance so that owners of other instances can't see payroll data.
- Control the assignment of `SYSADM`, `SYSCTRL`, and `SYSMAINT` authority for each instance.
- Optimize the database manager configuration for each database instance.
- Limit the effect of an instance crash. In the unlikely event of an instance crash, only one instance is affected. The other instance may continue to function normally.

However, multiple instances have some minor disadvantages:

- Additional system resources (virtual memory and disk space) are required for each instance.
- More administration is required because you have additional instances to manage.

DB2 program files are physically stored in one location on a particular machine. Each created instance points back to this location so that the program files aren't duplicated for each instance created. Several related databases can be located within a single instance. Figure 2.10 shows how databases, instances, and systems relate to one another.

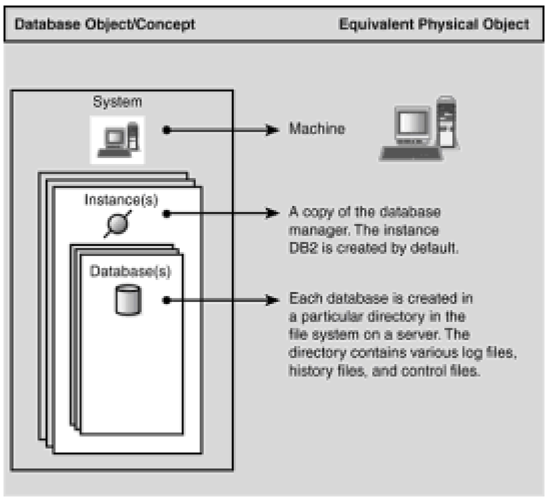

Database Object/Concept
Equivalent Physical Object
System
Machine
Instance(s)
A copy of the database manager. The instance DB2 is created by default.
Database(s)
Each database is created in a particular directory in the file system on a server. The directory contains various log files, history files, and control files.

Figure 2.10: Basic elements: Instances.

You can connect to more than one database at the same time. These databases can be in multiple instances on the same or different machines. Instances are cataloged as local or remote in the node directory. When you attempt to access a database not in your default instance, the node directory is used to determine how to communicate with the database.

You can *attach* to an instance to perform maintenance and utility tasks that can be done only at an instance level, such as create database, force application, database monitor commands, or database manager configuration updates.

Directories

Directories are necessary for accessing local and remote databases. They ensure that access to a database is transparent to users and applications, regardless of where the database physically resides.

DB2 Universal Database uses three types of database directories, which identify the location of databases, and a node directory, which contains network connection information for remote databases. The database directories are as follows:

- The *system database directory* identifies the name, alias, and physical location of each cataloged database. For a user or application to access a database, an entry for that database must be in this directory. All clients and servers must have a system database directory.

- A *local database directory* contains the name of a database and the subdirectory pathname in which the database files are stored. One directory exists in every path that contains a database. An entry is added to it when a database is created.

- The *database connection services* (*DCS*) directory is used only if DB2 Connect is installed on your system. It contains an entry for each distributed relational database that your node can access.

The node directory contains an entry for all nodes to which your database client and protocol can connect. Each entry contains the node's name, communication information, and instance information. A database server's node must be cataloged in this directory before remote database clients can access it or a DB2 Connect gateway.

The system database directory, local database directory, DCS directory, and node directory can be cataloged through the Client Configuration Assistant or the Control Center.

Figure 2.11 shows how the various directories interact. It shows a client connecting to the DB2DB database on a DB2 server.

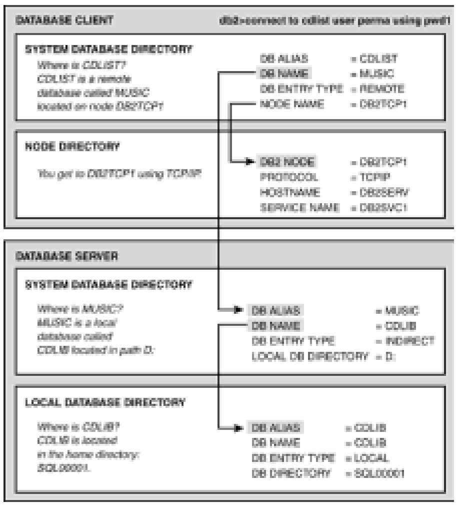


Figure 2.11: Basic elements: An example of directories interacting.

Storage Objects

Table spaces, containers, and buffer pools let you define how you'll store the data on your system and how you can improve performance related to accessing the data. You aren't required to create a table space, container, or buffer pool to be able to add data to tables in a database. You can accept the defaults for each when you create a database and a table. Use the information in this section to understand how you can tune these storage objects for your environment.

Table Spaces

Databases are organized into partitions called *table spaces*. Essentially, a table space is a place to store tables. When creating a table, you can decide to have certain objects, such as indexes and large object data, kept separately from the rest of the table data. Figure 2.12 shows some of the flexibility you have in spreading data over table spaces.

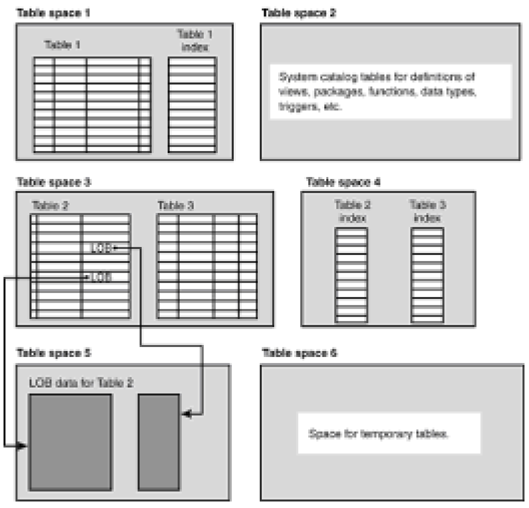


Figure 2.12: Basic elements: Table spaces.

It's easier to manage large databases by partitioning them into table spaces. A table space enables you to assign the location of table and index data directly to media storage devices or containers and enables a database to exceed the size of the file system. A single table space can span several file systems or devices. For example, if you have a

10GB database, you'll need to spread this database over several containers. You could spread the data over five containers, each being 2GB.

You have the flexibility to assign portions of a table (such as data, indexes, and long field data) to different table spaces. You can then assign different storage devices, depending on the contents of each table space. If additional space is required for any of these table spaces, you can increase the size by adding more devices or storage space. This can increase the size without stopping the database, and the data in the table space is automatically rebalanced for maximum performance.

Table spaces can also be backed up or restored as a unit. Backing up only the changed data reduces the time needed to perform a backup, enabling you to back up frequently changed data more often.

A table space can be a *system managed space* (*SMS*) or a *database managed space* (*DMS*). For an SMS table space, each container is a directory in the file space of the operating system, and the operating system's file manager controls the storage space. (Containers are described in the next section.) For a DMS table space, each container is a fixed-size preallocated file or a physical device such as a disk, and the database manager controls the storage space. Figure 2.13 illustrates the differences between SMS and DMS table spaces.

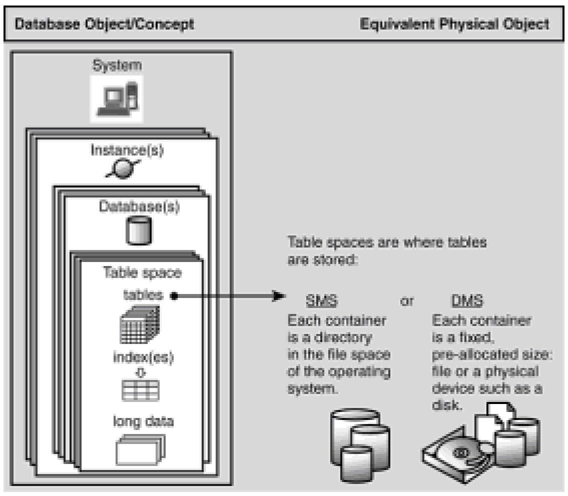

Database Object/Concept
Equivalent Physical Object
System
Instance(s)
Database(s)
Table space
tables
index(es)
long data
Table spaces are where tables are stored:
SMS
or
DMS
Each container is a directory in the file space of the operating system.
Each container is a fixed, pre-allocated size: file or a physical device such as a disk.

Figure 2.13: Basic elements: SMS or DMS table spaces: Figure 2.14.

Tables containing user data exist in the regular table spaces. By default, a table space known as `USERSPACE1` is created. Indexes and system catalog tables are also stored in regular table spaces. The default system catalog table space is known as `SYSCATSPACE`. Tables containing long field data or long object data, such as multimedia objects, exist in the long table spaces. The temporary table space is used during SQL operations for things such as sorting or reorganizing tables, creating indexes, and joining tables. A database should have one temporary table space. You can create more, although for most situations only one is required. By default, a table space called `TEMPSPACE1` is created. Figure 2.14 shows the different types of table spaces.

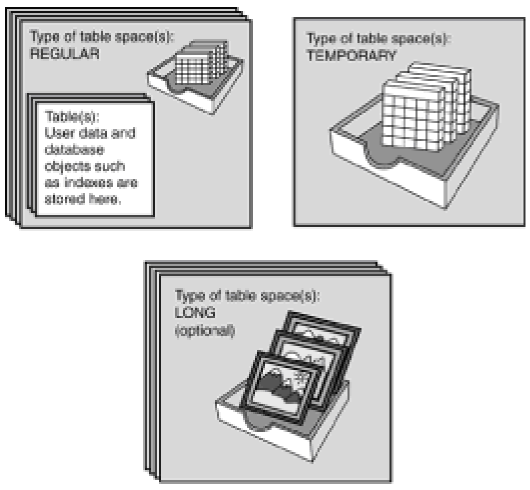


Figure 2.14: Basic elements: Types of table spaces:

Containers

A *container* is a physical storage device that can be identified by a directory name, a device name, or a filename. A container is assigned to a table space (all database and table data are assigned to table spaces). A table space's definitions and attributes are recorded in the database system catalog. When a table space is created, you can then create tables within this table space.

A container isn't explicitly shown in the Control Center, but it's listed when you view table spaces in the Journal (one of the DB2 database administration tools). A single table space can span many containers, but each container can belong to only one table space. Figure 2.15 shows the `PAYROLL` table space spanning five containers.

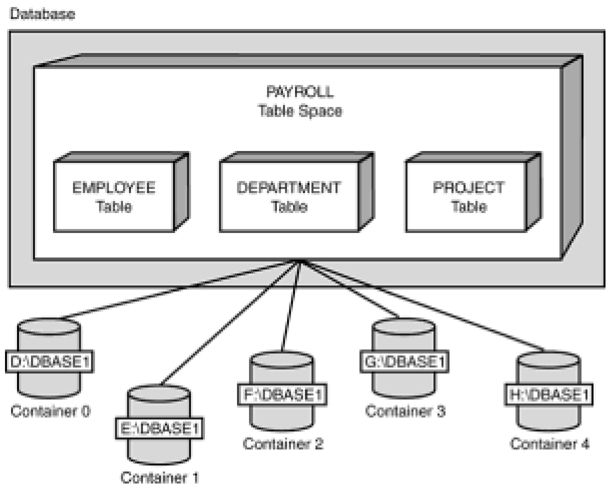


Figure 2.15: Basic elements: Containers.

Data for any table is stored on all containers in a table space in a round-robin fashion. This way, data is balanced across the containers that belong to a given table space. The number of pages that the database manager writes to one container before a different one is used is called the *extent size*.

Buffer Pool

A *buffer pool* is an allocation of main memory allocated to cache table and index data pages as they're being read from disk or being modified. The purpose of the buffer pool is to improve database system performance. Data can be accessed much faster from memory than from disk; therefore, the fewer times the database manager needs to be read from or written to disk, the better the performance. (You can create more than one buffer pool, although for most situations only one is required.) Figure 2.16 shows the relationship between the buffer pool and the other database objects.

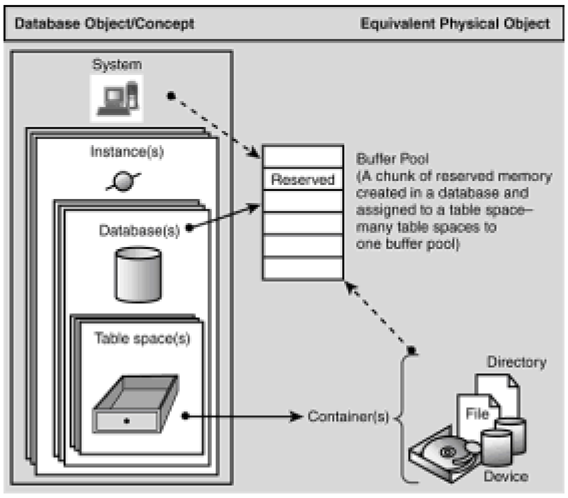

Database Object/Concept
Equivalent Physical Object
System
Instance(s)
Database(s)
Table space(s)
Reserved
Buffer Pool (A chunk of reserved memory created in a database and assigned to a table space– many table spaces to one buffer pool)
Directory
File
Container(s)
Device

Figure 2.16: Basic elements:Buffer pool.

Configuration Files

Configuration files contain parameter values that define the resources allocated to DB2 and to individual databases. The three types of configuration files are the database manager configuration file for DB2 as a whole, the database configuration file for each individual database, and the administration configuration file for the DB2 Administration Server.

The database manager configuration file is created when DB2 is installed or when an instance of DB2 is created. The parameters it contains affect system resources at a global (product or instance) level, independent of any one database stored on the system. Many of these parameters can be changed from the system default values to improve performance or increase capacity, depending on your system's configuration. Figure 2.17 shows the relationship between the various configuration files.

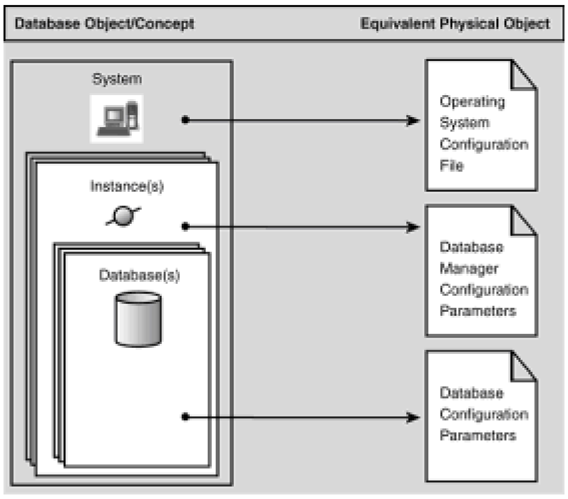

Database Object/Concept
Equivalent Physical Object
System
Instance(s)
Database(s)
Operating System Configuration File
Database Manager Configuration Parameters
Database Configuration Parameters

Figure 2.17: Basic elements: Configuration files.

There's one database manager configuration file for each installation of a client as well. This file contains information about the client enabler for a specific workstation. A subset of the parameters available for a server is applicable for the client.

A database configuration file is created when a database is created and resides where that database physically resides. There's one configuration file per database. Its parameters specify the amount of resources to be allocated to that database; many of the parameters can be changed to improve performance or increase capacity. Different changes may be required, depending on the type of activity in that specific database.

The administration configuration file is created when DB2 Universal Database is installed or when the DB2 Administration Server is created. The parameters it contains affect the remote administration of DB2 servers.

Recovery Objects

Log files and the recovery history file are created automatically when a database is created. You can't modify them; however, they are important should you need to use your database backup to recover lost or damaged data.

Log Files

Each database includes recovery logs, which are used to recover from application or system errors. With the database backups, they're used to recover the consistency of the database right up to the point in time when the error occurred. Figure 2.18 shows the interaction between the active and offline log files.

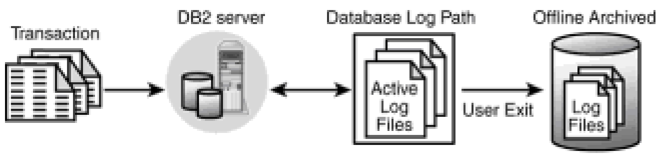


Figure 2.18: Basic elements:Log files.

Should you lose online data because of a media failure, such as a hard disk crash, roll-forward recovery enables you to recover it by applying log journal information against the restored database. Log journals contain the changes made to the database since the last backup. After these journals are applied, the database will be in the same state as it was before the failure.

Circular Logging

Circular logging is the default when a database is created. As the name suggests, circular logging uses a circle (ring) of online logs to provide recovery from transaction failures and system crashes. The logs are used and retained only to the point of ensuring the integrity of current transactions.

Circular logging doesn't enable you to roll forward a database through prior transactions from the last full backup. You recover from media failures and disasters by restoring a full, offline backup. All changes since the last backup are lost. The database must be offline (inaccessible to users) when a full backup is taken. Because this type of restore recovers your data to the specific point in time of the full backup, it's sometimes called *version recovery*.

Circular logging records the changes to your database in a set of logs. When the last log file series is filled, the first log file is reused in a circular fashion. Circular logging supports restore-only recovery.

Archive Logging

Archive logging doesn't overwrite older log files. Instead, it continuously creates additional logs to record every transaction since your last backup. This type of logging supports roll-forward recovery.

This complete recovery option allows for all logs up to the last complete transaction to be replayed. Archive logging is activated when the `LOGRETAIN` parameter is set to `YES`. When archive logging is activated, a full offline backup is required. If `LOGRETAIN` is turned off, logging reverts to circular, and the online logs are automatically deleted. If it's necessary to back out of the changes from an errant application that damaged data, a database can use the logs to roll forward to any point in time between the full, offline backup and the last completed transaction.

With archive logging, it becomes necessary to pay more attention to the handling of the logs and to ensure their safety. The capability to perform roll-forward recovery of your database depends on the integrity of the logs. Consideration should be given to backing up log data and to keeping it on disk arrays or mirrored volumes.

Active Logs

Active logs contain transactions that haven't yet been committed or rolled back, or whose changes haven't yet been written to disk. Active logs are located in the database log path directory.

Online Archived Logs

When all changes in the active log are no longer needed for normal processing, the log is closed and becomes an archived log. An archived log is said to be *online* when it's stored in the database log path directory. Figure 2.19 shows an example of archived active logs.

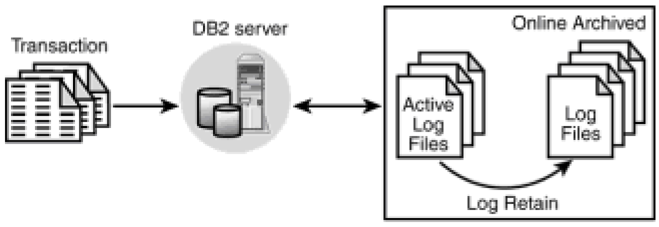


Figure 2.19: Basic elements: Archive logging.

Offline Archived Logs

You also can store archived logs in a location other than the database log path directory by using a user exit program. An archived log is said to be *offline* when it's not stored in the database log path directory.

Recovery History File

The recovery history file contains a summary of the backup information that can be used in case all or part of the database must be recovered to a given point in time. It's used to track recovery-related events such as backups, restores, and loads. A recovery history file is created with each database and is automatically updated when certain actions are performed, such as database or table space backup or restore.

Application Programs

DB2 provides several different ways to access and manipulate data in DB2 databases through application programs. You can use embedded SQL, the DB2 CLI, Java, or ODBC end user tools. To perform administrative functions, you can use the DB2 APIs.

A client application that needs to access a DB2 database sends a request to the server. The request is coded in SQL (the language used to access DB2 databases). The server processes the request and returns the data to the client application. In this context, DB2 programming is the development of client applications that access DB2 databases to retrieve, store, or manipulate data.

DB2 programming also includes developing user-defined functions and stored procedures that run on the server. DB2 also provides many features to build advanced applications: triggers, constraints, stored procedures, user-defined functions, user-defined data types, and support for large objects (such as audio, video, and image) to build multimedia applications.

You can develop client applications by using the following:

- Personal Developer's Kit
- Universal Developer's Kit
- Java Development Kit
- An Open Database Connectivity (ODBC) end user tool such as Lotus Approach

Application programs that can access DB2 databases include those created by an end user or developer in your company, as well as those commercially available. All applications, including query products such as Lotus Approach and fourth-generation languages such as VisualAge, use the same SQL interface described here.

You can choose several methods to access DB2 databases. The way your application accesses DB2 databases depends on the type of application you want to develop. For example, if you want a data-entry application, you might choose to embed static SQL statements in your application. If you want an application that performs queries over the World Wide Web, you would probably choose to develop Java applications. If you just want a simple application to query a database, you might code it by using LotusScript.

Embedded SQL

SQL is the database interface language used to access and manipulate data in DB2 databases. You can embed SQL statements in your applications, enabling them to perform any task supported by SQL, such as retrieving and storing data. With DB2, you can code your embedded SQL applications in the COBOL, FORTRAN, C, C++, and REXX programming languages.

An application in which you embed SQL statements is called a *host program*. A programming language that you compile and in which you embed SQL statements is called a *host language*. The program and language are defined this way because they *host* or accommodate SQL statements.

Static and Dynamic SQL

Embedded SQL can be static or dynamic. The method of compiling a SQL statement and the persistence of its operational form distinguish static SQL from dynamic SQL. Static SQL statements are ones where 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. You can represent those values in the host language variables. You prepare or build static SQL statements before you run your application.

In contrast, dynamic SQL statements are those statements that your application builds and executes at runtime. An interactive application that prompts users for key parts of a SQL statement, such as the names of the tables and columns to be searched, is a good example of dynamic SQL. The application builds the SQL statement while it's running and then submits the statement for processing.

Generally, static SQL statements are well suited for high-performance applications with predefined transactions. A reservation system is a good example of such an application. Generally, dynamic SQL statements are well suited for applications that run against a rapidly changing database where transactions need to be specified at runtime. An interactive query interface is a good example of such an application. You can write applications that have only static SQL statements, only dynamic SQL statements, or a mix of both.

Precompiling and Binding

When you embed SQL statements in your applications, you must *precompile* and *bind* your applications to a database. The precompiler converts the SQL statements in each source file into DB2 runtime API calls to the database manager. The precompiler also produces an access package in the database and an optional bind file, if you specify that you want one created. The access package contains access plans selected by the DB2 optimizer for the static SQL statements in your application. The access plans contain the information required by the database manager to execute the static SQL statements in the most efficient manner, as determined by the optimizer.

For dynamic SQL statements, the optimizer creates access plans when you run your application. The bind file contains the SQL statements and other data required to create an access package. You can use the bind file to rebind your application later without having to precompile it first. Rebinding optimizes access plans for current database conditions. You need to rebind your applications if your application will access a different database from the one against which it was precompiled.

With static SQL applications, you can limit the access users have to data. For example, you can grant a user permission to run an application that updates a table without giving that user general access to the table. You do this by granting authorizations to run an access package when you bind your application. This encapsulates access authority in the package, giving selected users permission only to run a package, not to access the entire table. In dynamic SQL applications, authorizations are validated at runtime on a per-statement basis. Therefore, to run a dynamic SQL statement, users must have explicit access privileges for each database object.

Open Database Connectivity (ODBC)

In some cases, you might need an application to perform a basic task, such as querying a database. You can use ODBC end-user tools such as Lotus Approach, Microsoft Access, and Microsoft Visual Basic to create these applications. ODBC tools provide a simpler alternative to developing applications than using a high-level programming language.

Lotus Approach provides two ways to access DB2 data:

- You can use the graphical interface to perform queries, develop reports, and analyze data.

- You can develop applications by using LotusScript, an object-oriented programming language that comes with a wide array of objects, events, methods, and properties, along with a built-in program editor.

DB2 Call Level Interface (DB2 CLI)

DB2 CLI is a programming interface that your C or C++ applications can use to access DB2 databases. DB2 CLI is based on the Microsoft ODBC specification and the ISO CLI standard. Through this interface, applications use standard function calls at execution time to connect to a database and issue SQL statements to access and update data.

You don't need to precompile or bind DB2 CLI applications because they use common access packages provided with DB2. You simply compile and link your applications. Therefore, applications developed with this interface are independent of any particular database server. This independence means a DB2 CLI application doesn't have to be recompiled to access different database servers but instead selects the appropriate one at runtime.

Before your DB2 CLI or ODBC application can access DB2 databases, however, the DB2 CLI bind files must be bound on each DB2 database that will be accessed. This occurs automatically on the first connection to the database, but it's recommended that the database administrator bind the bind files from one client on each platform that will access a DB2 database. Suppose that you have OS/2, AIX, and Windows 95 clients that each access two DB2 databases. The administrator must bind the bind files from one OS/2 client on each database that will be accessed. Next, the administrator must bind the bind files from one AIX client on each database that will be accessed. Finally, the administrator must do the same on one Windows 95 client.

When you use DB2 CLI, your application passes dynamic SQL statements as function arguments to the database manager for processing. As such, DB2 CLI is an alternative to embedded dynamic SQL, providing another way to access DB2 databases by using dynamic SQL.

Application Programming Interfaces (APIs)

When writing your applications, you might need to perform some database administration tasks, such as creating, activating, backing up, or restoring a database. DB2 provides numerous APIs so that you can perform these tasks from your applications, including embedded SQL and DB2 CLI applications. This way, you can program the same administrative functions into your applications that you can perform by using the DB2 Administration Tools.

You also might need to perform specific tasks that can be performed only by using the DB2 APIs. For example, you might want to retrieve the text of an error message so that your application can display it to users. To retrieve the message, you must use the `Get Error Message` API.

System Management Facilities

DB2 provides many facilities in addition to the Control Center to aid in the management of a large, diverse database system. You can administer database clients from one central location, perform database administration tasks remotely from a client workstation, monitor database activity, spread databases across multiple file systems, force users off the system, and diagnose problems.

Online Administrative Capability

A number of database administration tasks can be performed while the database is still operational (while users or applications are still connected). This provides for greater availability of data to users. Some tasks that can be done online include backing up data, reorganizing data, creating tables or table spaces, and altering tables or table spaces.

Remote Program Execution

DB2 can be installed on a LAN server where multiple clients on the LAN can share the same copy of DB2, but each client can maintain its own configurations at its local machine.

DCE Directory Services

Distributed Computing Environment (DCE) Directory Services simplifies system management of a large network by removing the requirement to catalog information at each individual client. With DCE Directory Services, information about a database has to be recorded only once in a central location, not in every single client workstation. Whenever the database moves or any information about the database changes that a client needs to know, only one update has to be made on that central location.

DCE Security

DB2 supports the Open Software Foundation's (OSF) DCE Security component for use in authentication of database users. Using the DCE Security component provides a more secure authentication mechanism and more central management of users, passwords, and groups.

DB2 Governor

The DB2 Governor lets you set rules that indicate how to handle application behavior. For example, it monitors resource usage by executing applications and, if specified limits are exceeded, reprioritizes query execution or issues the `force` command to terminate them. Also, you can use the governor to generate accounting records for executing applications that can be used for charge-back accounting.

Database and Directory Migration

Support is provided to enable back-level DB2 databases and directories to be converted to a format usable by DB2 Version 5.

Summary

In today's lesson, you learned that a relational database presents data as a collection of tables. The basic elements used by DB2 Universal Database include database objects, data integrity, object relational capabilities, system catalog tables, instances, directories, storage objects, configuration files, and recovery objects.

DB2 provides a rich development environment through the DB2 Personal Developer's Kit and the DB2 Universal Developer's Kit. These kits enable you to access and manipulate DB2 data by using embedded SQL, DB2 CLI, Java, ODBC end-user tools, or DB2 APIs.

DB2 provides several system-management facilities to help you manage very large databases and large networks. These facilities include being able to perform many administration tasks while users are still connected to the database, enabling client code to be shared to simplify configuration, and a governor to help you define how applications behave.

What Comes Next?

On Day 3, "Installing and Configuring DB2 Server," you learn how to install the DB2 Universal Database Workgroup Edition product provided in demonstration mode on the CD-ROM included with this book.

Q&A

Q What different kinds of constraints does DB2 provide to enforce business rules?

A DB2 provides unique constraints to ensure that the values in a set of columns are unique and not null for all rows in the table. Referential constraints let you define relationships between and within tables by requiring that all values of a given column of a table also exist in some other table or column. Check constraints are provided to enforce rules not covered by uniqueness or referential integrity. Check constraints enable you to specify a range of values allowed in one or more columns of every row of a table.

Q What are the two types of table spaces?

A A table space can be system managed space (SMS) or database managed space (DMS). For SMS table spaces, the operating system manages the directories that contain the data files and controls the storage space. For DMS table spaces, the DB2 controls the storage space, and the data is stored in preallocated files or physical devices.

Q How does roll-forward recovery work?

A To use roll-forward recovery, you must make an initial backup of your database and set the `LOGRETAIN` parameter to `YES`. As you make changes to the data in your database, the transactions are stored in log journals. If you have a media failure, you must first restore the database image and then use roll-forward recovery to reapply the transactions stored in the log journals. You can choose to roll data forward to a specific point in time or to the end of the logs.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. What's the advantage of using a view?

2. Give some reasons why you would want to define a primary key for a table.

3. What's an instance? What's the advantage of using an instance?

4. What's the difference between static and dynamic SQL?

Exercise

The sample database used throughout this book is an inventory of a music library. Start thinking about the types of information that should be stored in such a database, what tables will be needed, what the keys should be for each table, and the types of relationships needed between the various tables. The design of the sample database is covered in detail at the end of Week 1.

Day 3: Installing and Configuring DB2 Server

Overview

Although the DB2 documentation covers the installation and configuration of DB2 servers and clients, today's lesson provides more detailed instructions on installing the server. (Day 6, "Installing DB2 Clients," shows how to install the DB2 clients.) I also explain each decision you'll be faced with when installing.

As you know from Day 1's lesson, the DB2 server runs on many different operating systems. Because this book focuses primarily on the Windows NT platform, however, the examples are specific to that interface, as are the installation instructions in this lesson.

In today's lesson, you see how do an interactive Typical, Custom, or Compact install of the DB2 server product. Typical and Compact installs require fewer questions to answer, but to see exactly what the install program adds to your system, you'll also look at the Custom install.

Note If you're migrating from previous versions of DB2, you must complete certain procedures before installing DB2 version 5. The instructions for migrating from a previous version are beyond the scope of this book.

Preparing for the Install

Go through the steps in this section before you begin installing to ensure that you have the required items and information.

Hardware and Software Requirements

Before installing the DB2 product, check to see that you have the proper hardware and software components available on your system:

- **Disk space.** You should have 150 MB of free disk space to install DB2. You also need additional disk space to store the applications and databases that you'll use with DB2 (this amount depends on the amount and type of data you store in your databases).

- **Processor.** You can run DB2 on a 486 computer, but it's recommended that you use a Pentium-based computer.

- **Memory.** To run the entire set of DB2 graphical tools, you need at least 24 MB of memory installed. The more memory you have, the faster the tools will run.

- **Operating systems.** You need to have Windows NT 3.51 or later installed on your computer.

  Note The DB2 Personal Edition product runs on Windows 95. It has the same function as the DB2 server, but it can't accept requests from remote clients.

- **Communications support.** You need communications support only if you want to access remote databases or have remote clients access your databases. Regardless of the operation, you have several communication protocols to choose from: TCP/IP, IPX/SPX, NetBIOS, Named Pipes, and APPC. All these protocols, except APPC, are installable options in Windows NT. You don't need to purchase separate products to provide this support. Ensure that you've installed and configured the protocol support that you intend to use.

Creating a Username for Installing DB2 Products

You need a username to install DB2. The username must belong to the Administrators group and it must be a valid DB2 username or have the Act as Part of the Operating System advanced user right. To create a username on a Windows NT system, choose Start | Programs | Administrative Tools | User Manager.

Note If you're using DB2 Personal Edition on a Windows 95 system, you must still log on with a valid username. To log on, choose Start | Shutdown and select Close All Programs and Log On as Another User. If logging on with a valid username isn't possible, DB2 creates the username `db2admin`, with which you'll need to log on to DB2.

A valid DB2 username is eight characters or less and complies with DB2's naming rules:

- The username must begin with `A` through `Z`, `@`, `#`, or `$`.
- It can contain alphanumeric characters and `@`, `#`, `$`, or `_`.
- You can't use these special words: `USERS`, `ADMINS`, `GUESTS`, `PUBLIC`, or `LOCAL`.
- You can't begin the username with the `IBM`, `SQL`, or `SYS`.

Tip This username will be removed from the system when the installation is complete, unless it is used by the DB2 Administration Server. See the later section, "Performing a Custom Install," and Day 17, "Working with DB2 Instances," for more information about the Administration Server.

Performing a Typical Install

Use the Typical install type if you want to install the DB2 components used most often, and if you want to have the DB2 instance and DB2 Administration Server configured to use the communications protocols detected on your system. The Typical install type sets up the DB2 components used most often, including all required components, ODBC support, documentation, and commonly used DB2 tools such as the Information Center, the Client Configuration Assistant, and the Control Center. The DB2 instance and the Administration server are created and customized to use the protocols detected on your system.

Note You can't selectively uninstall components after the setup program completes the installation. If you don't want to install all the components mentioned in the preceding paragraph, see the sections on the Compact and Custom installs.

Table 3.1 lists the products and selectable components installed when you choose the Typical install type. The check marks in the Enterprise Edition and Workgroup Edition columns indicate components installed for each edition.

Table 3.1: Products and components for DB2 Universal Database for Windows NT.

Components and Subcomponents	Enterprise Edition	Workgroup Edition
Graphical Tools	✓	✓

Client Configuration Assistant	✓	✓
Control Center	✓	✓
Performance Monitor	✓	✓
Visual Explain	✓	✓
DB2 ODBC Driver	✓	✓
DB2 Connect Support	✓	N/A
East Asian Conversion Support	✓	✓
Documentation	✓	✓

Note The appropriate East Asian Conversion Support subcomponent is part of the Typical install for East Asian countries that uses the Double Byte Character Set (DBCS).

To do a Typical DB2 server install on a Windows NT workstation, follow these steps:

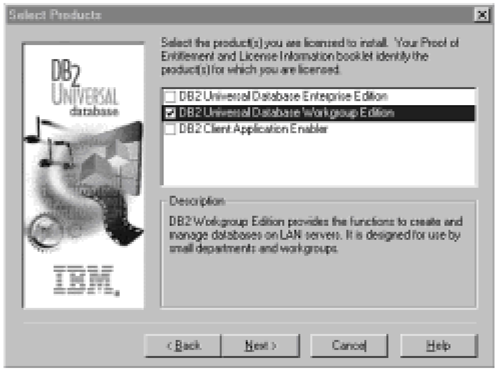
1. Log on as a user that meets the requirements for installing DB2. (For more information, see the previous section "Creating a Username for Installing DB2 Products.")

2. Shut down any other programs so that the setup program can update files as required.

3. Insert the CD-ROM into the drive. The auto-run feature automatically starts the setup program. The setup program determines the system language and launches the setup program for that language.

   Tip To run the setup program in a different language, you need to manually invoke the setup program. Choose Start | Run. In the Open text box, type **`x:\setup /i=LANGUAGE`**, where `x:` represents your CD-ROM drive and *`LANGUAGE`* represents the two-character country code for your language (for example, `EN` for English). Choose OK.

4. In the Welcome dialog box, click the Next button to open the Select Productsdialog box (see Figure 3.1).



Select the product(s) you are licensed to install. Your Proof of Entitlement and License Information booklet identify the product(s) for which you are licensed.

- [ ] DB2 Universal Database Enterprise Edition
- [x] DB2 Universal Database Workgroup Edition
- [ ] DB2 Client Application Enabler

Description

DB2 Workgroup Edition provides the functions to create and manage databases on LAN servers. It is designed for use by small departments and workgroups.

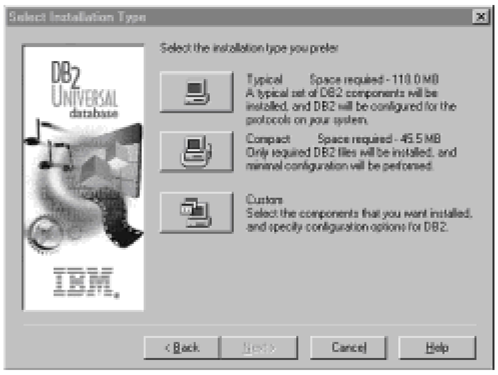
Figure 3.1: The Select Products dialog box.

5. In the Select Products dialog box, choose a DB2 product depending on the type of license you've purchased (the installation instructions are the same whether you're installing the Workgroup or Enterprise Edition):

- Choose DB2 Universal Database Enterprise Edition if you want the DB2 server plus the capability of having your clients access enterprise servers such as DB2 for MVS.
- Choose DB2 Universal Database Workgroup Edition if you want the DB2 server.
- Choose DB2 Client Application Enabler if you want a remote client. The Enabler is built into the DB2 server products, so you need to install this product only on a system that doesn't need a local database.

Tip Although you can install a DB2 client by following the instructions in this section, it's best if you follow Day 6, "Installing DB2 Clients," when installing a client. The instructions vary slightly.

6. Click the Next button. The Select Installation Type dialog box appears (see Figure 3.2).



Select the installation type you prefer

Typical Space required - 110.0 MB
A typical set of DB2 components will be installed, and DB2 will be configured for the protocols on your system.

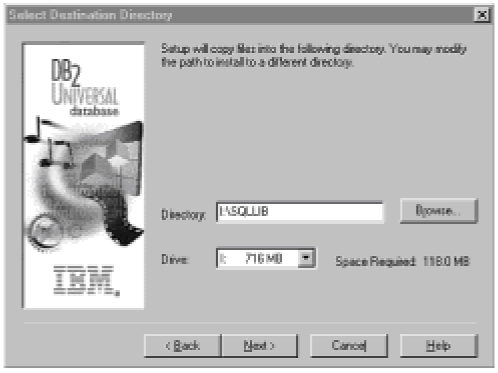
Compact Space required - 45.5 MB
Only required DB2 files will be installed, and minimal configuration will be performed.

Custom
Select the components that you want installed, and specify configuration options for DB2.

Figure 3.2: The Select Installation Type dialog box.

7. Choose the installation type you prefer by clicking the appropriate button—for these steps, the Typical button.

8. In the Select Destination Directory dialog box, select a directory and a drive where DB2 is to be installed (see Figure 3.3). Click the Browse button to help you select a directory with enough available disk space. (The amount of space required for the product also appears onscreen.)


Select Destination Directory

Setup will copy files into the following directory. You may modify the path to install to a different directory.

Directory: I:\SQLLIB

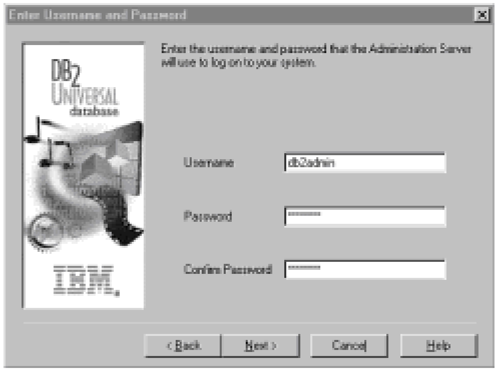
Drive: I: 716 MB

Space Required: 118.0 MB

Figure 3.3: The Select Destination Directory dialog box.

Note If a DB2 version 5 product is already installed on this system, you must install subsequent products and components in the same path. The Directory and Drive boxes are disabled if this is the case.

9. Click Next. In the Enter Username and Password dialog box, the setup program by default fills the Username, Password, and Confirm Password text boxes with `db2admin` (see Figure 3.4). You can accept these default values or provide your own. If you provide your own username, you must ensure that it's eight characters or less and complies with DB2's naming rules, as explained earlier.


Enter Username and Password

Enter the username and password that the Administration Server will use to log on to your system.

Username: db2admin

Password: ********

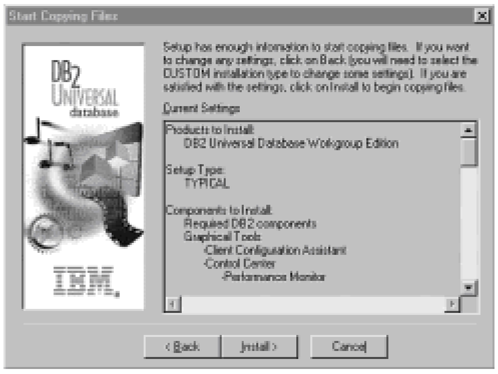
Confirm Password: ********

Figure 3.4: The Enter Username and Password dialog box.

Tip If you use the default username `db2admin`, change the password. You can also change the password after installing the product if you don't changeit now.

The DB2 Administration Server uses the username and password provided here to log on to the system and start itself as a service. (You use the DB2 Administration Server to enable remote administration.) The setup program checks to see whether the username specified for the DB2 Administration Server exists. If it doesn't, you'll be asked if you want it created, provided that the username you're using to install DB2 has the Act as Part of the Operating System advanced user right. If it does exist, the setup program verifies that the username is a member of the Administrators group and verifies that the password is valid. Click Next tocontinue.

10. You've given DB2 all the information required to install the product on your system. In the Start Copying Files dialog box (see Figure 3.5), you're given one last chance to verify the values you've entered. Click Install to have the files copied to your system. You also can click Back to return to the dialog boxes that you've already completed to make any changes.


Start Copying Files

Setup has enough information to start copying files. If you want to change any settings, click on Back (you will need to select the CUSTOM installation type to change some settings). If you are satisfied with the settings, click on Install to begin copying files.

Current Settings:

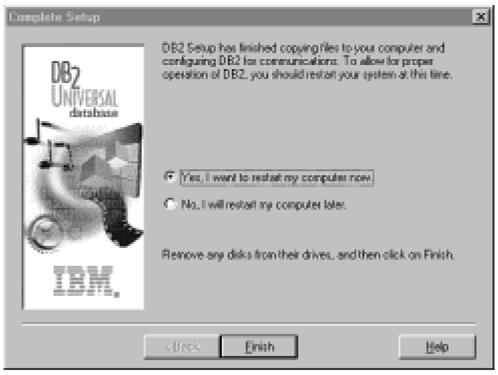
```
Products to Install:
    DB2 Universal Database Workgroup Edition

Setup Type:
    TYPICAL

Components to Install:
    Required DB2 components
    Graphical Tools
        -Client Configuration Assistant
        -Control Center
            -Performance Monitor
```

Figure 3.5: The Start Copying Files dialog box.

11. The installation progress bars appear onscreen while the product is being installed. After the product is installed, you must reboot your system before you can begin to use it. Select a reboot option, as shown in the Complete Setup dialog box in Figure 3.6, and click the Finish button to complete the installation.


Complete Setup

DB2 Setup has finished copying files to your computer and configuring DB2 for communications. To allow for proper operation of DB2, you should restart your system at this time.

[✓ Yes, I want to restart my computer now.]

No, I will restart my computer later.

Remove any disks from their drives, and then click on Finish.

Figure 3.6: The Complete Setup dialog box.

Tip For information on errors encountered during product installation, see the `db2.log` file, which stores general information and error messages resulting from install and uninstall activities. By default, this file is located in the `x:\db2log` directory, where `x:` is the drive on which Windows NT is installed.

The installation program has completed the following:

- Created DB2 program groups and items (or shortcuts).
- Registered a security service.
- Updated the Windows Registry.
- Created a default instance named DB2, added it as a service, and configured it for communications.
- Created the DB2 Administration Server, added it as a service, and configured it so that DB2 tools can administer the server. The service's start type was set to Automatic.
- Activated DB2 First Steps to start automatically following the first boot after installation.

You've completed all the installation steps. Proceed to Day 4, "Getting Started," to verify the installation and to get the system ready for general use.

Performing a Compact Install

The steps to perform a Compact installation are the same as performing a Typical installation, except for the installed product components. During a Compact installation, only the required components are installed, which greatly reduces the amount of disk space required for the product but also greatly reduces the functionality available. For example, the Graphical Tools and online books aren't installed during a Compact installation.

Tip Use Compact installation on systems where you use another application to interface to the database. For example, if you're planning to use Lotus Approach as the interface to DB2, you may not want to use the DB2 tools or online documents.

The following table lists the products and selectable components installed when you choose the Compact install type. The check marks in the Compact Install column indicate the installed components.

Table 3.3. Products and components for DB2 for Windows NT.

Components and Subcomponents	Enterprise Edition	Workgroup Edition
Graphical Tools		
Client Configuration Assistant		
Control Center		
Performance Monitor		
Visual Explain		
DB2 ODBC Driver	✓	✓
DB2 Connect Support		N/A
East Asian Conversion Support		
Documentation		

Tip The appropriate East Asian Conversion Support Subcomponent is part of the Typical install for East Asian countries that uses the Double Byte Character Set (DBCS).

Now that you've successfully installed DB2, see Day 4, "Getting Started," to get started with DB2 databases.

Summary

In today's lesson, you saw that you must use a valid DB2 username when using and installing DB2. The username must be fewer than eight characters long or have the Act as Part of the Operating System advanced user right.

The three different installation types are custom, compact, and typical. Normally you would use the typical install type, but you should understand what the options are to ensure that this install type is right for you.

DB2 requires a username and password to be specified for the Administration Server. If you don't change the default values provided during installation, the username is `db2admin`, as is the password. If you didn't change these values during the install, you should change them after the product is installed to ensure that your system is secure.

What Comes Next?

On Day 4, "Getting Started," you learn about what to do immediately following the installation of the product.

Q&A

Q I have a limited amount of disk space available. Which install type should I use?

A Normally if you have a limited amount of disk space available on your machine, you would select the Compact installation type. This installation type installs the basic components of the product. The DB2 Tools are *not* installed if you use this installation type, so it will be difficult—if not impossible—to complete the rest of the examples in this book. If you have a limited amount of space, try to free up space or find another machine. As a last resort, you could choose the Custom installation type and deselect the DB2 books. This will save a considerable amount of space, but you won't have access to the DB2 library.

Q Do I have to customize the communication protocols during the installation?

A If you choose the Typical or Compact installation type, your system is automatically set up to use the default values for the communication protocols detected on your system. You don't need to do anything additional. If you want to see or modify the default values, you must choose the custom installation type. It's recommended that you don't change the default values, but if you must, ensure that you keep a record of the values you've used.

Q Where can I find information about the success or failure of the installation?

A During installation, DB2 keeps all information, warning, or error messages in the `db2.log` file. This file is located on the drive where Windows NT is installed in the directory `\db2log`. This directory and file remains on your system even if you perform an uninstall.

Workshop

The purpose of the Workshop is to allow you to test your knowledge of the material covered in the lesson. See if you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. What's the one difference concerning the default instance when using the Custom install type rather than the Typical install type?

2. Under what circumstances would you need to manually invoke the installation program?

3. How do you know which subcomponents are installed on your system when using the Custom install type?

4. When would you want the Control Center to automatically start each time you boot your system?

Day 4: Getting Started

Overview

So, now that DB2 Universal Database is installed, what's next? Where do you begin? Today's lesson steps you through everything you need to do immediately following the installation:

- Logging on to the system

- Changing default passwords

- Modifying the Startup folder

- Starting and stopping DB2

- Using DB2 First Steps

- Granting privileges to other users

- Registering the software

You don't need to perform these steps in the order presented, but you should consider each step before proceeding to further lessons in this book. Today's lesson also introduces you to DB2's desktop.

Logging On to the System

When you start your system, log on with a username that belongs to the Administrators group and meets DB2's naming rules so that you can create databases and set up the security for the users on your system.

Note In many cases, the username `db2admin` was created during installation. You might choose to log on with this username because it has the highest level of authority. The password for this username is set to `db2admin` unless you changed the value during installation.

By default, any user belonging to the Administrators group has System Administration (`SYSADM`) authority on the instance created during the installation. To change the group that has `SYSADM` authority on the default instance, see the later section "Changing

Default Privileges for Users." To use DB2, log on with a valid username that has the appropriate authority level for the commands you want to execute. The section "Changing Default Privileges for Users" lists the authority level required to execute certain commands.

Tip Most database administration tasks described in this book require that you have `DBADM` or `SYSADM` authority, so log on with a username that's a member of the Administrators group.

Understanding the Desktop

This section describes each program you see on the desktop following an install. On a Windows NT system, all the DB2-related programs are in Start | Programs | DB2 for Windows NT. The following is what you should see:

- Choose the Administration Tools to launch the tools shown in Table 4.1. Information on how to use each tool is covered throughout this book.

Table 4.1: Available administration tools.

Tool	Description
	The Alert Center monitors your system for early warnings of potential problems or automates actions to correct problems discovered.
	The Control Center manages systems, instances, databases, and database objects, such as tables and views. In the Control Center, you can display all your systems, databases, and database objects and perform administration tasks on them. From the Control Center, you can also open other centers and tools to help you work with DB2 commands, jobs, and scripts; optimize queries; and monitor performance.
	Use the Event Analyzer to monitor your system for early warnings of potential problems or to automate actions to correct problems discovered.
	Use the Journal to view all available information about jobs that are pending execution, are executing, or have completed execution; the recovery history log; the alerts log; and the messages log. The Journal also enables you to review the results of jobs that run unattended.
	Use the Script Center to create mini-applications known as *scripts*, which can be stored and invoked at a later time. These scripts can contain DB2 commands, SQL statements, and operating-system commands. You can schedule scripts to run unattended. These jobs can be run once or set up to run on a repeating schedule. (A repeating schedule is particularly useful for tasks such as backups.)
	Use the Tool Settings to change the settings for the Control Center, Alert Center,and Replication.

- In the Problem Determination folder, select Support Through Internet to launch your Web browser with a file that can link you to the DB2 home page, the DB2 Support and Service page, and the Software Servers home page. Select Trace to create a diagnostic trace file for DB2 Support personnel.
- Choose Certification to launch your Web browser with a file that can link you to the DB2 Professional Certification program information on the Web. This program enables you to take a series of examinations to verify your database skills.
- Choose Client Configuration Assistant to help you manage database connections to remote database servers. With the Client Configuration Assistant, you can define connections to databases, remove cataloged databases, modify the properties of a cataloged database, test connection to local or remote databases, bind applications, and tune client configuration parameters. Each function is described throughout this book.
- Choose Command Center to enter DB2 commands and SQL statements in an interactive window and see the execution result in a result window. You can scroll through the results and save the output to a file. You can also create access plan graphs by using the Command Center. Each function is described in this book.
- Choose Command Line Processor to enter commands and SQL statements at a DB2 command prompt. Choose Command Window to enter commands and SQL statements at a Windows NT command prompt. These tools are useful if you know the commands you want to execute and prefer using the command line rather than the graphical tools.
- Choose First Steps to create the SAMPLE database, view the data in the database, and launch the Information Center. First Steps, discussed more fully later in this lesson, is automatically launched after DB2 is installed, so you might have noticed it already.
- Choose the Information Center to browse the DB2 online library in HTML format and to access task help, samples, and useful external Web links. The Information Center is described later in today's lesson.
- Choose Registration to register your DB2 license. The Registration notebook is launched automatically after DB2 is installed.
- Choose Release Notes to view last-minute news about the DB2 product.
- The HTML search server starts automatically. If you want to avoid the overhead of the search engine, choose Stop HTML Search Server. When you want to use Net.Question program to help you search through the DB2 online library, choose Start HTML Search Server to start again.
- Choose Uninstall to remove the DB2 program and all its associated files and Registry entries. The Uninstall program won't remove any data you might have created.

Choose Start | Programs | License Use Runtime | Client to access the Nodelock Administration tool and its documentation. Use the Nodelock Administration tool to upgrade a try-and-buy license to a permanent license or to track the number of concurrent users connecting to a DB2 Workgroup Edition server.

Changing Default Passwords

If you used the default username `db2admin`, remember to change the default password for this username. You should change this password immediately following the installation (if you didn't change the password during installation) because when the setup program creates the `db2admin` username, it also makes it a member of the Administrators group. Follow these steps to change the password:

1. Choose Start | Programs | Administrative Tools | User Manager.
2. In the User Manager window, select the `db2admin` username.
3. Choose User | Properties from the menu.
4. Type the new password, confirm the new password, and click OK.
5. Click the Close button to exit the User Manager window.

Then follow these steps to change the password for the DB2-DB2DAS00 service to match the new password for the `db2admin` username:

1. Choose Start | Settings | Control Panel | Services.
2. In the Services dialog box, select the DB2-DB2DAS00 service and click Startup.
3. In the Startup dialog box, change the password and confirm the new password. Click OK when done.
4. Click Close when you've finished with the Services dialog box.

Modifying the Startup Folder

During installation, the Control Center is placed in the Startup folder so it can be started each time you boot your system. If you don't want the Control Center to start each time you boot your system, remove it from the Startup folder. Follow these steps to remove the Control Center from the folder:

1. Right-click Start and select Open All Users from the pop-up menu.
2. In the Start Menu folder, double-click the Programs icon.
3. In the Programs folder, double-click the StartUp icon.
4. In the Startup folder, drag the Control Center or Start HTML Search Server icon to a new folder or onto the desktop if you don't want either to start each time you boot your system.

Starting or Stopping DB2

During installation, a default instance named DB2 and the DB2 Administration Server named DB2DAS00 were created, added as services and configured for communications. If you performed a Typical install, these services are set to start automatically each time you boot your system. If you performed a Custom install, you chose whether to have these services started automatically or manually. In either case, to change how the service is started, follow these steps:

1. Choose Start | Settings | Control Panel and double-click the Services icon.

2. In the Services dialog box, select the DB2-DB2 service.

3. To modify whether the services are started automatically or manually, click the Startup icon. Select the radio button that corresponds with Automatic, Manual, or Disable.

4. Repeat steps 2 and 3 for the DB2-DB2DAS00 service.

5. Click Close to exit the Services dialog box.

Tip If the startup is set to manual, click Start to start the service. If startup is set to automatic, click Stop to stop the service.

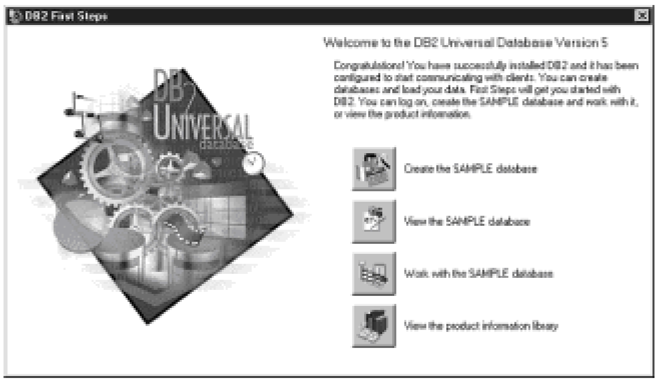
If you select Manual, users wanting to launch DB2 as a service must have certain privileges:

- They must belong to one of the following groups: `SYSADM_GROUP`, `SYSCTRL_GROUP`, or `SYSMAINT_GROUP`. These privileges are specified in the database manager configuration file.

- They must have the correct privileges as defined by Windows NT. User accounts can be members of the Administrators, Server Operators, or Power Users group.

Using DB2 First Steps

When your system restarts the first time after installation, the DB2 First Steps tool executes automatically (see Figure 4.1). To invoke DB2 First Steps at any time, choose Start | Programs | DB2 for Windows NT | First Steps.


DB2 First Steps
DB2 Universal database
Welcome to the DB2 Universal Database Version 5

Congratulations! You have successfully installed DB2 and it has been configured to start communicating with clients. You can create databases and load your data. First Steps will get you started with DB2. You can log on, create the SAMPLE database and work with it, or view the product information.

Create the SAMPLE database
View the SAMPLE database
Work with the SAMPLE database
View the product information library

Figure 4.1: Launching DB2's First Steps program.

Use DB2 First Steps to perform the following (each task is described in detail in the following sections):

- Choose Create the SAMPLE Database to create a sample database (named, of course, SAMPLE) against which you can issue commands to verify that the system is working correctly. The product documentation and sample programs refer to the tables and fields of this database.

- Choose View the SAMPLE Database to start the Command Center with a predefined script that shows you some contents of the SAMPLE database. From this tool, you can also write your own SQL statements against the SAMPLE database.

- Choose Work with the SAMPLE Database to start the Control Center (an administration tool). From the Control Center, you can perform various actions on the SAMPLE database and its objects.

- Choose View the Product Information Library to start the DB2 Information Center. This enables you to view and search the DB2 online library with your Web browser and to access task help, samples, and useful external Web links.

Creating the SAMPLE Database

To help you verify that the system is installed and configured correctly, create the SAMPLE database (click the Create the SAMPLE Database button) and issue a few commands against it. It takes a few minutes for this database to be created. When it's created, you receive the message shown in Figure 4.2.

DB2 First Steps

The sample database has been successfully created.

Figure 4.2: The message indicating that the database is created.

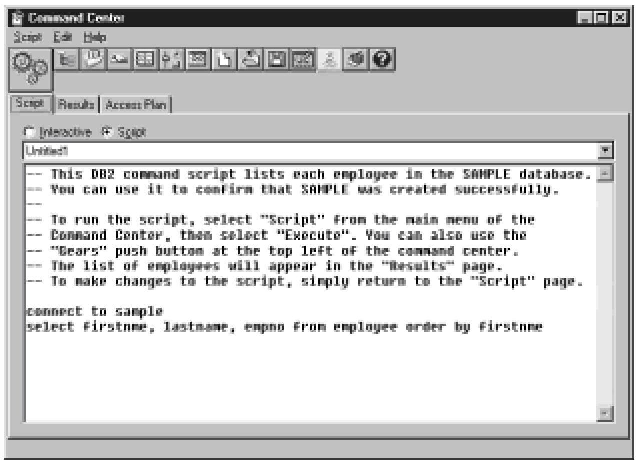
The product documentation and the sample programs use the tables in the SAMPLE database to demonstrate DB2's features. This book uses examples that call these tables as well. Some tables that the SAMPLE database contains include the following:

Table	Columns Included
`EMPLOYEE`	`EMPNO`, `FIRSTNME`, `MIDINIT`, `LASTNAME`, `WORKDEPT`,`PHONENO`, `HIREDATE`, `JOB`, `EDLEVEL`, `SEX`, `BIRTHDATE`, `SALARY`, `BONUS`, and `COMM`
`STAFF`	`ID`, `NAME`, `DEPT`, `JOB`, `YEARS`, `SALARY`, and `COMM`
`DEPARTMENT`	`DEPTNO`, `DEPTNAME`, `MGRNO`, `ADMRDEPT`, and `LOCATION`
`ORG`	`DEPTNUMB`, `DEPTNAME`, `MANAGER`, `DIVISION`, and `LOCATION`
`EMP_PHOTO`	`EMPNO`, `PHOTO_FORMAT`, and `PICTURE`
`EMP_RESUME`	`EMPNO`, `RESUME_FORMAT`, and `RESUME`

The SQL Reference describes each table in detail. Later in today's lesson, you'll see how to access this information in the HTML version of the book.

Viewing the SAMPLE Database

To issue a few commands against the SAMPLE database, click the View the SAMPLE Database button. The Command Center opens and loads in a script that connects to the database and selects several of the columns from the `EMPLOYEE` table (see Figure 4.3).

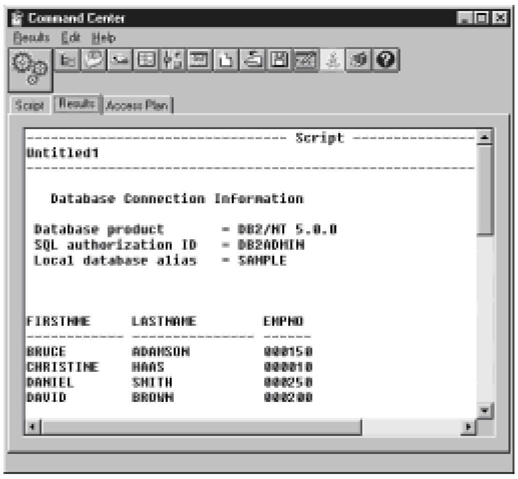


```
-- This DB2 command script lists each employee in the SAMPLE database.
-- You can use it to confirm that SAMPLE was created successfully.
--
-- To run the script, select "Script" from the main menu of the
-- Command Center, then select "Execute". You can also use the
-- "Gears" push button at the top left of the command center.
-- The list of employees will appear in the "Results" page.
-- To make changes to the script, simply return to the "Script" page.

connect to sample
select firstnme, lastname, empno from employee order by firstnme
```

Figure 4.3: The Command Center window.

Click the Gears button to run these commands. If DB2 was installed correctly and the SAMPLE database was created without a problem, the result of this query appears (see Figure 4.4).



```
------------------------------------ Script ------------------
Untitled1
--------------------------------------------------------------

   Database Connection Information

 Database product        = DB2/NT 5.0.0
 SQL authorization ID    = DB2ADMIN
 Local database alias    = SAMPLE


FIRSTNME     LASTNAME        EMPNO
------------ --------------- ------
BRUCE        ADAMSON         000150
CHRISTINE    HAAS            000010
DANIEL       SMITH           000250
DAVID        BROWN           000200
```

Figure 4.4: The results in the Command Center window.

If you know SQL, you can try some of your own queries. Click the Script tab to return to the area where you can enter SQL statements.

If you had a problem creating the SAMPLE database, make sure that you have enough space on the drive where DB2 is installed for the database. You should have a minimum of 10MB available for the database.

If you had a problem with the installation, refer to Day 21, "Diagnosing Problems," for a possible cause and solution.

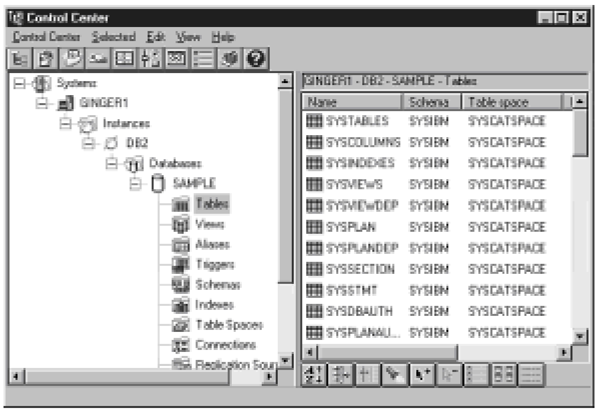
Working with the SAMPLE Database

Now that you're confident that DB2 was installed correctly, you can try out the graphical administration tools. The central tool is the Control Center, a graphical tool used for DB2 administration tasks. Use the Control Center to do the following:

- Manage databases
- Manage tables, views, and indexes
- Configure DB2 systems
- Perform database backup and recovery tasks
- Schedule jobs
- Replicate data

To start the Control Center at this point, click the Work with the SAMPLE Database button in the First Steps window. The Control Center enables you to see the objects that make up the SAMPLE database (see Figure 4.5).

Tip You can also start the Control Center by choosing Start | Programs | DB2 for Windows NT | Administration Tools | Control Center.



GINGER1 - DB2 - SAMPLE - Tables

Name	Schema	Table space
SYSTABLES	SYSIBM	SYSCATSPACE
SYSCOLUMNS	SYSIBM	SYSCATSPACE
SYSINDEXES	SYSIBM	SYSCATSPACE
SYSVIEWS	SYSIBM	SYSCATSPACE
SYSVIEWDEP	SYSIBM	SYSCATSPACE
SYSPLAN	SYSIBM	SYSCATSPACE
SYSPLANDEP	SYSIBM	SYSCATSPACE
SYSSECTION	SYSIBM	SYSCATSPACE
SYSSTMT	SYSIBM	SYSCATSPACE
SYSDBAUTH	SYSIBM	SYSCATSPACE
SYSPLANAU...	SYSIBM	SYSCATSPACE

Figure 4.5: The Control Center.

First, you should know the terminology used to describe the various Control Center elements. The Control Center interface has five elements that help you define and manage systems and databases: a menu bar, a toolbar, an object tree, the contents pane, and the contents pane's toolbar.

The Menu Bar

Use the menu bar to work with objects in the Control Center, open other administration centers and tools, and access online help:

- With the Control Center menu, you can open another Control Center or close all the tools now open.
- With the Selected menu, you can see the actions that you can perform on the currently highlighted object. This menu provides the same options as the right-click pop-up menu.
- The Edit menu helps you find an object or select all the objects within the currently highlighted folder.

- Use the View menu to customize the look of the contents pane or to launch other tools and centers. This menu provides the same functions as the contents pane's toolbar and the Control Center main toolbar.

- The Help menu gives you access to online help or the Information Center.

The Control Center Toolbar

Use the toolbar icons above the object tree to access other functions, such as setting preferences and alerts and viewing job status, logs, and messages. These functions can also be selected from the View menu.

The Object Tree

Use the object tree to display and work with systems and database objects. The top-level objects are Systems, Instances, and Databases. Click the **+** sign in front of an object to reveal the objects it contains. For example, clicking the **+** sign in front of the Databases folder reveals a list of databases that you have access to within the selected instance.

The Contents Pane

Use the contents pane to display and work with systems and database objects. The contents pane displays those objects that make up the contents of the selected object in the object tree. For example, when you click the Databases folder, you see a list of the databases available in the selected instance.

The Contents Pane's Toolbar

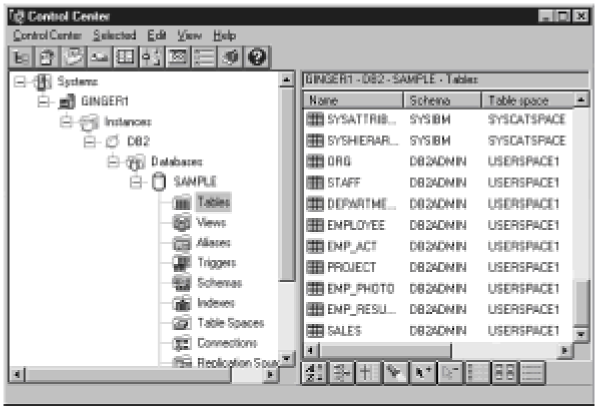
Use the toolbar below the contents pane to tailor the view of objects and information in the contents pane to suit your needs. You can choose to sort, filter, or customize columns; find, select all, or deselect all objects; or view the objects in details, icon, or text formats. (You also can select these functions from the Edit and View menus.)

One thing you might want to do right now is view the contents of a database table, such as `EMPLOYEE`. Follow these steps:

1. Click the + (plus sign) in front of the Systems folder. This shows you the systems, instances, and databases you have on your system.

2. Click the `SAMPLE` folder to see the objects available within the SAMPLE database.

3. Click Tables to see a listing of all the tables in the contents pane. Scroll down until you see the `EMPLOYEE` table as shown in Figure 4.6.



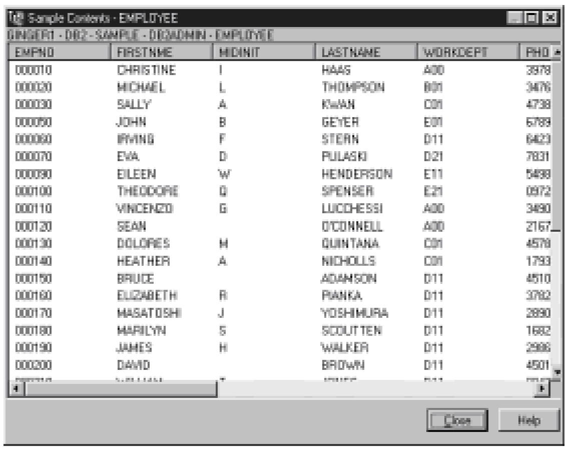
GINGER1 - DB2 - SAMPLE - Tables

Name	Schema	Table space
SYSATTRIB...	SYSIBM	SYSCATSPACE
SYSHIERAR...	SYSIBM	SYSCATSPACE
ORG	DB2ADMIN	USERSPACE1
STAFF	DB2ADMIN	USERSPACE1
DEPARTME...	DB2ADMIN	USERSPACE1
EMPLOYEE	DB2ADMIN	USERSPACE1
EMP_ACT	DB2ADMIN	USERSPACE1
PROJECT	DB2ADMIN	USERSPACE1
EMP_PHOTO	DB2ADMIN	USERSPACE1
EMP_RESU...	DB2ADMIN	USERSPACE1
SALES	DB2ADMIN	USERSPACE1

Figure 4.6: The list of tables inthe contents pane.

Notice the table names that begin with `SYS`. These are system catalog tables. The system maintains the contents of these tables, but you can view them. Also notice the columns shown in the contents pane. You can see the schema and table space that each table belongs to.

4. Right-click the `EMPLOYEE` table and select Sample Contents from the pop-up menu to see the first 20 rows of the `EMPLOYEE` table. The Sample Contents window appears as shown in Figure 4.7.



Sample Contents - EMPLOYEE
GINGER1 - DB2 - SAMPLE - DB2ADMIN - EMPLOYEE
Close
Help

EMPNO	FIRSTNME	MIDINIT	LASTNAME	WORKDEPT	PHO
000010	CHRISTINE	I	HAAS	A00	3978
000020	MICHAEL	L	THOMPSON	B01	3476
000030	SALLY	A	KWAN	C01	4738
000050	JOHN	B	GEYER	E01	6789
000060	IRVING	F	STERN	D11	6423
000070	EVA	D	PULASKI	D21	7831
000090	EILEEN	W	HENDERSON	E11	5498
000100	THEODORE	Q	SPENSER	E21	0972
000110	VINCENZO	G	LUCCHESSI	A00	3490
000120	SEAN		O'CONNELL	A00	2167
000130	DOLORES	M	QUINTANA	C01	4578
000140	HEATHER	A	NICHOLLS	C01	1793
000150	BRUCE		ADAMSON	D11	4510
000160	ELIZABETH	R	PIANKA	D11	3782
000170	MASATOSHI	J	YOSHIMURA	D11	2890
000180	MARILYN	S	SCOUTTEN	D11	1682
000190	JAMES	H	WALKER	D11	2986
000200	DAVID		BROWN	D11	4501

Figure 4.7: The Sample Contents window.

Further details about the tasks you can perform with the Control Center are covered throughout this book.

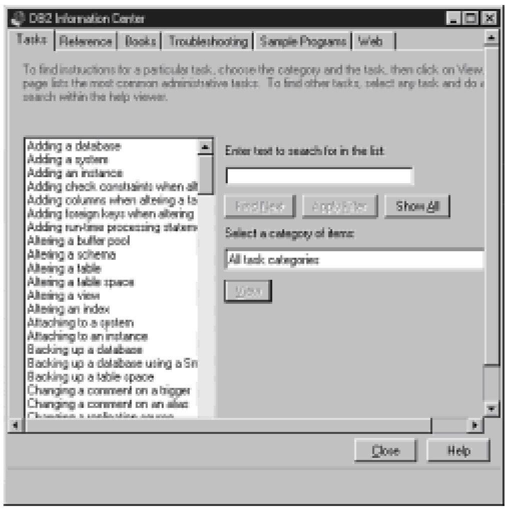
Tip It's recommended that you keep the SAMPLE database (if disk space allows), because it provides a good place for you to test ideas without damaging data in your own databases. If, after you finish using and testing the SAMPLE database, you want to remove it to free up disk space, right-click the `SAMPLE` folder and select Drop to remove the database.

Viewing the Product Information Library

You can view the DB2 product library through the Information Center. The Information Center launches the Web browser you normally use and displays the entire set of DB2 information, as well as lets you access task help, samples, and external Web links.

To see how to use the Information Center, you can locate information that describes the tables of the SAMPLE database by following these steps:

1. Click the View the Product Information Library button in the First Steps window. The Information Center appears (see Figure 4.8).



To find instructions for a particular task, choose the category and the task, then click on View. page lists the most common administrative tasks. To find other tasks, select any task and do a search within the help viewer.

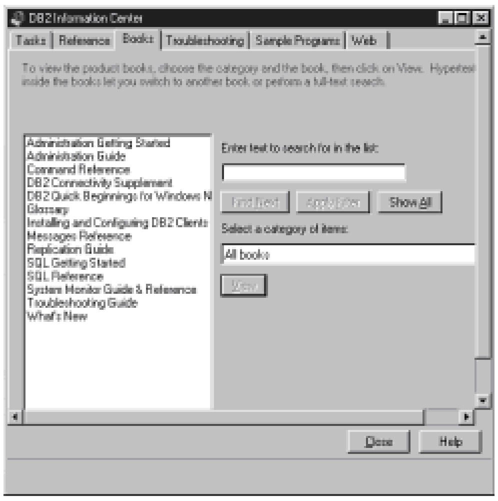
Adding a database
Adding a system
Adding an instance
Adding check constraints when alt
Adding columns when altering a ta
Adding foreign keys when altering
Adding run-time processing statem
Altering a buffer pool
Altering a schema
Altering a table
Altering a table space
Altering a view
Altering an index
Attaching to a system
Attaching to an instance
Backing up a database
Backing up a database using a Sn
Backing up a table space
Changing a comment on a trigger
Changing a comment on an alias

Enter text to search for in the list:

Select a category of items: All task categories

Figure 4.8: The Information Center.

2. Because you want to view the information that you know is in the SQL Reference, click the Books tab. All the books in the DB2 library are listed (see Figure 4.9).



To view the product books, choose the category and the book, then click on View. Hypertext inside the books let you switch to another book or perform a full-text search.

Administration Getting Started
Administration Guide
Command Reference
DB2 Connectivity Supplement
DB2 Quick Beginnings for Windows N
Glossary
Installing and Configuring DB2 Clients
Messages Reference
Replication Guide
SQL Getting Started
SQL Reference
System Monitor Guide & Reference
Troubleshooting Guide
What's New

Enter text to search for in the list:

Select a category of items: All books

Figure 4.9: The Books page of the Information Center.

3. Double-click the SQL Reference item. Your system browser, such as Netscape, is launched, and you see the title page for the SQL Reference.

   Tip To customize the launched browser, click the Choose Browser button on the Web tabbed page of the Information Center.

4. Click Search the DB2 Books to begin searching the entire DB2 library for information about the sample tables.

5. In the Type the Word or Phrase You Are Searching For text box, enter the search words **`sample tables`** and click Search. A list of the documents that contain this information appears.

6. Double-click an entry to read the online information about the sample tables.

Organizing and Viewing Objects by Schema

The objects in a relational database are organized into *schemas*, which provide a logical classification of objects in the database. The schema object is identified in the first part of a two-part object name. When an object such as a table, view, alias, distinct type, function, index, package, or trigger is created, it's assigned to a schema. This assignment is done explicitly or implicitly.

Use the following instructions to explicitly create a schema:

1. Start the Control Center.

2. Expand the objects until you see the Schemas folder.

3. Right-click the Schemas folder and select Create from the pop-up menu.

4. Type a schema name. The name must be one to eight characters long and begin with any letter from A through Z. (A schema name can't begin with `SYS`.)

5. From the Authorization Name drop-down list of users, select the owner of the schema.

6. Click OK to have the schema created.

As you create objects with the Control Center, you can select from a list of existing schema names. To implicitly create a schema, simply type a valid name in the Schema Name text box, overwriting the default schema name. For example, when using the SmartGuide to create a table, the first field enables you to choose or enter a schema name. The default name is the username you're signed in as.

Before creating your own objects, you need to consider whether you want to create them in your default schema (identified by your username) or by using a separate schema (such as `PAYROLL` for database objects relating to the payroll function) that logically groups the objects. If you create objects that will be shared, using a different schema name can be very beneficial.

Connecting to a Database

You need to connect to the database before you can use SQL statements to query or update it. The `CONNECT` statement associates a database connection with a username.

A database is created in an instance through one of three different authentication types: `CLIENT`, `SERVER`, or `DCS`. If an authentication type isn't specified when the instance is created, the default `SERVER` authentication is used.

Use the following instructions to connect to a database with the Control Center:

1. Start the Control Center.

2. Expand the objects until you see the database you want.

3. Right-click the database and select Connect from the pop-up menu.

4. Enter a username and password. If the authentication type is `CLIENT`, you must enter a valid username and password on the client system. If the authentication type is `SERVER`, you must enter a valid username and password on the server system. `DCS` authentication acts the same as `SERVER` authentication.

To connect to a local database called SAMPLE by using the Command Center or the command-line processor, type the following command:

Input `connect to sample`

Granting Privileges to Other Users

To give privileges to the SAMPLE database to other users on the system, follow these steps:

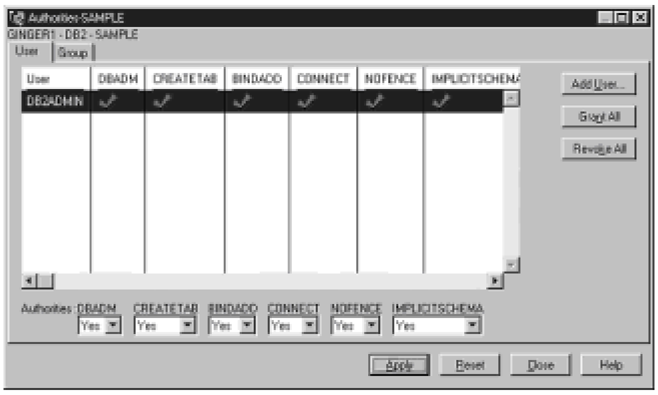
1. Start the Control Center, if it isn't already started.

2. Expand the objects until you see the SAMPLE database.

3. Right-click the `SAMPLE` database folder and select Authorities from the pop-up menu.

4. In the Authorities SAMPLE window, click Add User, select one or more users from the list, and then click the Add button. Click Close when you finish selecting users.

5. A window showing the privileges now assigned to each user opens (see Figure 4.10). A check mark in a column means that the user has the privilege; an X means that the user doesn't have the privilege. You can grant access to `DBADM`, `CREATETAB`, `BINDADD`, `CONNECT`, `NOFENCE`, and `IMPLICITSCHEMA`. Table 4.2 lists the definitions for each privilege.



Authorities-SAMPLE

GINGER1 - DB2 - SAMPLE

User	DBADM	CREATETAB	BINDADD	CONNECT	NOFENCE	IMPLICITSCHEMA
DB2ADMIN	✓	✓	✓	✓	✓	✓

Authorities: DBADM Yes, CREATETAB Yes, BINDADD Yes, CONNECT Yes, NOFENCE Yes, IMPLICITSCHEMA Yes

Figure 4.10: Privileges assigned to each user.

Table 4.2: Available user privileges.

Privilege	Definition
`SYSADM`	Grants the highest level of authority within the database manager; controls all database objects. `SYSADM` authority is given to all users in the Administrators group; all other authorities are implicit. This authority level can't be assigned through DB2; it's assigned through Windows NT's security system. DB2 simply assigns `SYSADM`

	authority to all users who are members of the Administrator group. See the next section, "Changing Default Privileges for Users," for instructions on changing the members of this group.
`DBADM`	Grants all privileges against all objects in the database and might grant these privileges to other users or groups. `CREATETAB`, `CONNECT`, `BINDADD`, and `NOFENCE` authorities are automatically granted to an authorization name granted `DBADM` authority.
`CREATETAB`	Grants the authority to create base tables within the database. The creator of a base table automatically has the `CONTROL` privilege on that table. The creator retains this privilege even if `CREATETAB` is subsequently revoked.
`BINDADD`	Grants the authority to create packages in the database. The creator of a package automatically has the `CONTROL` privilege on that package and retains this privilege even if the `BINDADD` authority is subsequently revoked.
`CONNECT`	Grants the authority to access the database.
`NOFENCE`	Grants the authority to register functions that execute in the database manager's process. After users register a function as not-fenced, the function continues to run in the database manager's process even if `NOFENCE` authority is revoked.
`IMPLICITSCHEMA`	Grants the authority to create a schema for an object by specifying the schema's name in the definition of the object.

Tip To create databases and assign authorities to others, you must have `SYSADM` authority.

6. To grant privileges to an entire group, use the Windows NT User Manager to assign users to a group. Click the Group tab in the Authorities window to grant access to the group.

Tip You might want to assign privileges to users according to their jobs. For example, a query user will need `CONNECT` on one or more databases and `SELECT`, `INSERT`, and `DELETE` on some tables and views.

Changing Default Privileges for Users

By default, `SYSADM` privileges are granted to any valid DB2 username that belongs to the Administrators group on Windows NT. You can change the users who have administrator privileges for each DB2 instance by changing the `SYSADM_GROUP` parameter; before you do, however, ensure that the group exists. To check to see whether this group exists, use the Windows NT User Manager Administrative Tool (click Start | Programs | Administrative Tools | User Manager). If the group exists, it's listed in the lower section of the User Manager window.

To use another group as the System Administration group (`SYSADM_GROUP`), update the database manager configuration file. To change the System Administration group

(`SYSADM_GROUP`) on the server instance, follow these steps:

1. Start the Control Center.
2. Click the + sign beside the Systems folder to list all the systems known to your workstation.
3. Click the + sign for the system that contains the instance you want to update.
4. Right-click the instance that you want to change—for example, DB2—and select Configure from the pop-up menu.
5. In the Configuration Instance notebook, select the Administration tab to see the configuration parameters associated with administration.
6. Select the System Administration Authority Group parameter and type the name of an existing group that you want to assign this privilege to in the Value box.
7. Click OK when you've finished.
8. Stop all applications that are using DB2, including the Control Center. When restarted, the new value for `SYSADM_GROUP` is used.

Registering the Software

The first time you boot your system following the installation of DB2, you're presented with the IBM Software Registration Tool. Use this tool to tell IBM a bit about yourself so that you can receive information on upgrades, new products, or new releases.

Summary

In today's lesson, you saw that when using DB2, you must log on to Windows NT with a username that complies with DB2's naming rules. This username must have `DBADM` or `SYSADM` authority to perform most of the tasks described in this book. You might want to use the `db2admin` username created during installation.

You can reach most of the DB2-related programs through the Control Center. The Control Center may have been automatically started when you booted your system; if not, you can start it by choosing Start | Programs | DB2 for Windows NT | Administration Tools | Control Center.

Use the DB2 First Steps program to create the SAMPLE database, view the contents of the database, work with the database, or access the product library. Each function is available separately as well.

If you plan to have other users on your system who are required to have fewer privileges, you can create groups of usernames and use the Control Center to grant authorities.

What Comes Next?

On Day 5, "Configuring Server Communications with the Control Center," you learn how to view or modify the communication parameters set up on your machine during installation.

Q&A

Q What username should I use to log on to Windows NT if I want to use DB2?

A To use DB2, you must log on to Windows NT with a username that complies with DB2's naming rules. The most important rule is that the username must have eight characters or fewer. The username `db2admin` is usually created during installation, so you may want to log on with this username.

Another factor you should consider is the authority the username has. Many things can be performed only by a username with `SYSADM` authority. By default, if your username is a member of the Administrators group, it has `SYSADM` authority. The `db2admin` username is created in the Administrators group and therefore has `SYSADM` authority.

Q What DB2 programs start automatically following a reboot?

A Following the first reboot after installation, several DB2 programs are automatically started. First, DB2 First Steps opens to allow you to create the SAMPLE database, view the contents of the database, work with the database, and access the DB2 product library. The Software Registration program also starts, enabling you to register your software.

Depending on the answers you provided during installation, you might also see that the Control Center has started. It automatically starts if you did a Typical install. If you did a Custom install, you had a choice whether the Control Center automatically starts.

Another choice you had during Custom install was whether you wanted the default instance (DB2) and the DB2 Administration Server (DB2DAS00) to start automatically. If you used the typical install type, both instances start automatically.

The last program automatically started is the HTML Search Server. This program enables you to search the entire DB2 product library available in HTML format.

Q How can I view data in the SAMPLE database?

A DB2 uses SQL to query tables in a database and returns the results to you in the form of a table. You can use SQL in many ways to view the information in a table. First, you can use the Control Center and select to sample the contents of a selected table. By using the Command Center, you can type in SQL statements to `CONNECT` to a database and `SELECT` data from a table. If you registered for ODBC, you can use ODBC-enabled applications such as Lotus Approach or Microsoft Access to view the data. You can view the data in other ways as well.

Workshop

The purpose of the Workshop is to allow you to test your knowledge of the material covered in the lesson. See if you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. Why would you want to create and use a schema?

2. What authority do you need to create databases?

3. What's the name of the tool that you use to view the DB2 product library?

4. What are the two ways you can start administration tools such as the Script Center?

Exercise

The Control Center introduced in this lesson is the main administration tool that you use with DB2. Much of the rest of this book discusses what you can do with the Control Center and the various tools associated with it. At this time, explore various options available with the Control Center to get a feel for how the tool works.

Day 5: Configuring Server Communications with the Control Center

Overview

The DB2 for Windows NT product enables you to use TCP/IP, NetBIOS, IPX/SPX, and APPC protocols to communicate with client workstations. During the installation of DB2 servers, the protocols detected on your system are configured automatically so that DB2 can use them. Usually, you don't need to do anything further to enable your server to accept requests from clients.

In today's lesson, you learn how to view or change protocol information for the TCP/IP, NetBIOS, and IPX/SPX protocols. (Updating the information for the APPC protocol is beyond the scope of this book.) You also see how to set up the communications on your server to allow remote clients to access DB2 databases and how to set up the communications for the DB2 Administration Server.

You also use the Control Center's Setup Communications function to configure communications on the server. The Control Center enables you to display the protocols and configuration parameters that a server instance is configured to use. With the Control Center, you can also maintain the configured protocols by modifying the parameter values of a configured protocol and by adding or deleting a protocol.

To add an undetected protocol, you must supply all parameter values before you proceed. When you add support for a new protocol to the server system, the Setup Communications function detects and generates parameter values for the new protocol; you can accept or modify them before use. When you remove support for an existing protocol from the server system, the Setup Communications function detects that the protocol has been removed and disables its use by the instance. You can also use the Setup Communications function to maintain communications of both local and remote server instances.

Modifying the DB2 Communication Configuration of Server Instances

To view or modify the configuration of the communication protocols for a DB2 instance, follow these steps:

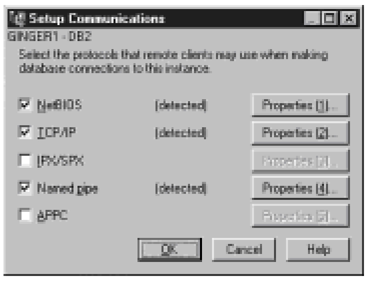
1. To start the Control Center on a Windows NT system, choose Start | Programs | DB2 for Windows NT | Administration Tools | Control Center.

2. Click the + sign beside the Systems icon in the Control Center for a list of systems.

3. Click the plus (+) sign beside the system name for a list of that system's database instances.

4. Right-click the instance you want to configure, and select Setup Communications from the pop-up menu.

5. To enable a protocol that wasn't selected during install in the Setup Communications window (see Figure 5.1), click the check box for the protocol, and click its Properties button.


Setup Communications

GINGER1 - DB2

Select the protocols that remote clients may use when making database connections to this instance.

☑ NetBIOS	[detected]	Properties (1)...
☑ TCP/IP	[detected]	Properties (2)...
☐ IPX/SPX		Properties (3)...
☑ Named pipe	[detected]	Properties (4)...
☐ APPC		Properties (5)...

Figure 5.1: The Setup Communications window.

6. In the Configure window, click Default to use DB2's recommended values, or type appropriate values for your system, as explained in the following sections. Click OK to return to the Setup Communications window.

7. Click OK when you've completed all the changes.

8. Stop and start the instance for these changes to take effect:

   - To stop the database manager instance, right-click it and select Stop from the pop-up menu.

   - To start the database manager instance, right-click it and select Start from the pop-up menu.

Modifying an instance's communications settings might require you to update the database connection configuration on the client. Use the Client Configuration Assistant to update the connection information on the client, as discussed on Day 6, "Installing DB2 Clients."

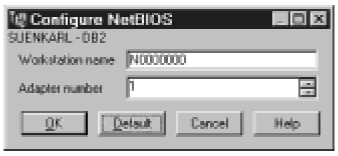
Note If clients are having trouble connecting to the server, ensure that the `DB2COMM` Registry value is defined to use the protocol that you intend to use for the client and server communications. To check the `DB2COMM` value on the server, type **`db2set db2comm`** at a command prompt.

Also ensure that the database manager at the server is started. A client can't access the server unless a `db2start` has been issued. The database manager is usually set up to start each time you boot your system.

Configuring NetBIOS

When you click Properties next to NetBIOS, you can enter values for workstation name and adapter number in the Configure NetBIOS window. The Workstation Name is the name of this system and must be unique on the network. The Adapter Number is the LAN adapter number used by this machine for NetBIOS connections.

Click Default to have DB2 assign values for these parameters (see Figure 5.2). Click OK to have these values saved and to return to the Setup Communications window.



SUENKARL - DB2

Workstation name: N0000000

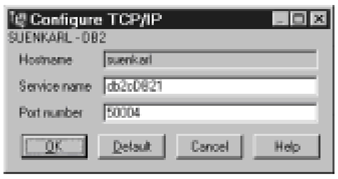
Adapter number: 1

Figure 5.2: The Configure NetBIOS window.

Configuring TCP/IP

Click Properties next to TCP/IP to enter values for service name and port number in the Configure TCP/IP window. The Hostname can't be changed through DB2 because it's a systemwide value. The Service Name is used in the services file to map to the port number used for this server instance. The Port Number is the TCP/IP port number used by this server instance to listen for connection requests from remote clients.

Click Default to have DB2 assign values for these parameters (see Figure 5.3). Click OK to save these values and to return to the Setup Communications window.



SUENKARL - DB2

Hostname: suenkarl

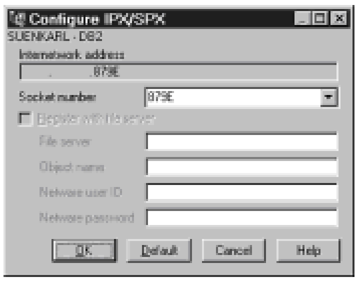
Service name: db2cDB21

Port number: 50004

Figure 5.3: The Configure TCP/IP window.

Configuring IPX/SPX

Click Properties next to IPX/SPX to open the Configure IPX/SPX window, shown in Figure 5.4 (because IPX/SPX isn't on this system, the options are disabled). For IPX/SPX, you can enter values for socket number, file server, object name, NetWare user ID, and NetWare password.



SUENKARL - DB2

Internetwork address: . .879E

Socket number: 879E

Figure 5.4: The Configure IPX/SPX window.

The Internetwork Address field displays the address a client system must use to connect to this server—for example, `09212700.400011527745.879E`. The first two values are unique to your system; you won't see them if IPX/SPX isn't installed on your system.

The hexadecimal Socket Number specifies a well-known static socket number and represents the connection endpoint in a DB2 server's internetwork address. The `Socket Number` defaults to `0x879E`; change this value to a supported value in the range of `0x879E`–`0x87A1` if you're running more than one DB2 server instance. DB2 has registered well-known sockets with Novell in the range of `0x879E`–`0x87A1` for the customer's use.

For File Server, specify in uppercase the name of the NetWare file server where the internetwork address of the database manager will be registered. The internetwork address of the database manager is stored in the bindery at the NetWare file server.

Note The socket number must be unique for every DB2 server instance on a machine. It must also be unique among all IPX/SPX applications running on the DB2 server machine, to ensure that the DB2 server can listen to incoming IPX/SPX connections with this socket number.

For Object Name, enter in uppercase a value that represents a particular database manager in the network. The object name must be unique for each DB2 server instance registered at a NetWare file server.

Enter a NetWare user ID for registering the DB2 object name at a NetWare file server. This ID must have supervisory or equivalent authority.

Click Default to have DB2 assign values for these parameters.

Tip You can use direct addressing or file server addressing to configure IPX/SPX:

- For direct addressing, you need only the socket number. Select the socket number used for this server instance in the Socket Number drop-down box.
- For file server addressing, you need the file server, object name, socket number, and NetWare user ID and password. You must also select the Register With File Server check box.

Configuring Named Pipes

Click Properties next to Named Pipes. The Configure Named Pipes window shows simply the computer name of your system (see Figure 5.5); because it's a systemwide value, you can't change the computer name in DB2. Click OK to return to the Setup Communications window.

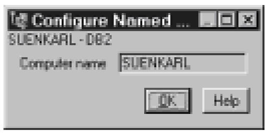
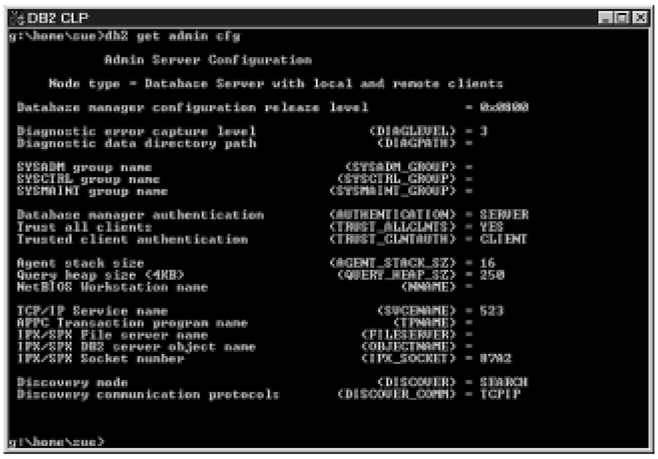

Configure Named ...
SUENKARL - DB2
Computer name SUENKARL

Figure 5.5: The Configure Named Pipes window.

Viewing and Modifying Configuration Files

You can use the DB2 Command window to view and modify the values in the DB2 Database Administration Server configuration file (see Figure 5.6). To start the DB2 Command window, choose Start | Programs | DB2 for Windows NT | Command Window. To view the file, enter the following command at the command prompt:

Input `db2 get admin cfg`



```
g:\home\sue>db2 get admin cfg

            Admin Server Configuration

     Node type = Database Server with local and remote clients

 Database manager configuration release level            = 0x0800

 Diagnostic error capture level              (DIAGLEVEL) = 3
 Diagnostic data directory path               (DIAGPATH) =

 SYSADM group name                        (SYSADM_GROUP) =
 SYSCTRL group name                      (SYSCTRL_GROUP) =
 SYSMAINT group name                    (SYSMAINT_GROUP) =

 Database manager authentication        (AUTHENTICATION) = SERVER
 Trust all clients                      (TRUST_ALLCLNTS) = YES
 Trusted client authentication          (TRUST_CLNTAUTH) = CLIENT

 Agent stack size                       (AGENT_STACK_SZ) = 16
 Query heap size (4KB)                   (QUERY_HEAP_SZ) = 250
 NetBIOS Workstation name                        (NNAME) =

 TCP/IP Service name                          (SVCENAME) = 523
 APPC Transaction program name                  (TPNAME) =
 IPX/SPX File server name                   (FILESERVER) =
 IPX/SPX DB2 server object name             (OBJECTNAME) =
 IPX/SPX Socket number                      (IPX_SOCKET) = 8702

 Discovery mode                               (DISCOVER) = SEARCH
 Discovery communication protocols       (DISCOVER_COMM) = TCPIP


g:\home\sue>
```

Figure 5.6: The DB2 Administration Server configuration file.

You can enter commands in the DB2 Command window to change a value in the DB2 Database Administration Server configuration file. You more than likely don't need to change any values in this configuration file, but if you want to change the `DISCOVER` parameter from `SEARCH` to `KNOWN`, for example, use the following command:

Input `db2 update admin cfg using discover known`

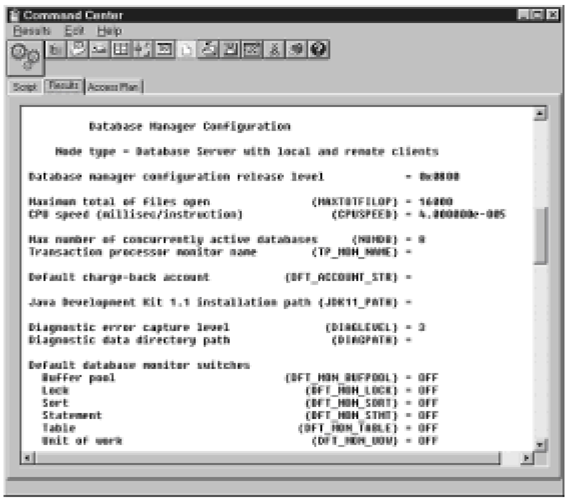
For this command to take effect, you must refresh the cache by issuing the following command:

Input `db2 terminate`

You can use the Command Center to view and update the values in the database manager configuration file (see Figure 5.7). Choose Start | Programs | DB2 for Windows NT | Command Center, and enter the following command:

Input `get dbm cfg`

This file is quite long; you'll need to scroll down to see all the values.



```
          Database Manager Configuration

     Node type = Database Server with local and remote clients

 Database manager configuration release level            = 0x0800

 Maximum total of files open             (MAXTOTFILOP) = 16000
 CPU speed (millisec/instruction)           (CPUSPEED) = 4.000000e-005

 Max number of concurrently active databases     (NUMDB) = 8
 Transaction processor monitor name        (TP_MON_NAME) =

 Default charge-back account           (DFT_ACCOUNT_STR) =

 Java Development Kit 1.1 installation path (JDK11_PATH) =

 Diagnostic error capture level              (DIAGLEVEL) = 3
 Diagnostic data directory path               (DIAGPATH) =

 Default database monitor switches
   Buffer pool                         (DFT_MON_BUFPOOL) = OFF
   Lock                                   (DFT_MON_LOCK) = OFF
   Sort                                   (DFT_MON_SORT) = OFF
   Statement                              (DFT_MON_STMT) = OFF
   Table                                 (DFT_MON_TABLE) = OFF
   Unit of work                            (DFT_MON_UOW) = OFF
```

Figure 5.7: The database manager configuration file.

To change the values in the database manager file, enter an **`update dbm cfg`** command. For example, to change the `DISCOVER` value from `SEARCH` to `KNOWN`, use the following command:

Input `update dbm cfg using discover known`

Again, for this to take effect, you must refresh the cache by using the `terminate`command.

Setting Up Communications forthe Administration Server

If you've installed the Control Center, you can use the Setup Communications function to configure the instance for communications. It can't be used to set up communications for the Administration Server, however.

If you add support for a new protocol after DB2 is installed, you must set up the Administration Server manually. For example, if you had TCP/IP and Named Pipes set up on your system when you installed DB2, DB2 is automatically configured to use both protocols. If you decide that you'd also like to support NetBIOS and add it to your system after DB2 is installed, you need to use the Control Center to configure DB2 to use NetBIOS, and use DB2 commands to configure the Administration Server to use NetBIOS.

Tip It's simplest to have the protocols you intend to use installed and configured on your system before DB2 is installed. This enables you to configure DB2 to use these protocols during the installation, leaving no additional steps to perform.

The following sections explain how to configure the Administration Server manually for each protocol.

Setting Up for TCP/IP

For the DB2 Administration Server to use TCP/IP, you must update the `svcename` parameter in the administration configuration file to use port 523. Use the following command to update this file:

Input `update admin cfg using svcename 523`

For these changes to take effect, you must stop and restart the database manager by using the `db2stop` and `db2start` commands.

Setting Up for NetBIOS

For the DB2 Administration Server to use NetBIOS, you must set the `DB2NBADAPTERS` Registry value if you want to use a value other than the default logical adapter number `0`. To change the adapter number, use the command

Input `db2set db2nbadapters =` *`adapter_number`* `-I DB2DAS00`

where *`adapter_number`* is the logical adapter number you want to use for the DB2 Administration Server.

After you set the adapter number, you must update the administration configuration file as follows:

Input `update admin cfg using nname` *`server1`* where *server1* is the server's workstation name.

For these changes to take effect, you must again stop and restart the database manager by using the `db2stop` and `db2start` commands.

You must also create a dependency for the Administration Server on NetBIOS. To create a dependency, you must enter the following command from the `sqllib\misc` directory on the drive where DB2 was installed:

Input `db2depnb DB2DAS00`

This records a dependency on the startup order, which causes NetBIOS to start before DB2.

Setting Up for IPX/SPX

For the DB2 Administration Server to use IPX/SPX, you must update the administration configuration file as follows:

Input `update admin cfg using ipx_socket 87A2`

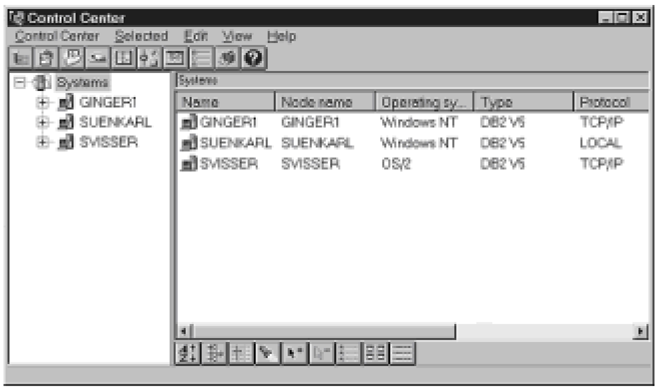
For these changes to take effect, you must stop and restart the database manager by using the `db2stop` and `db2start` commands.

Adding Systems Through the Control Center

You can use the Control Center to administer databases on remote servers as well as the databases on your local system. To see the databases and instances on a remote computer, you must first add the system to the Control Center object tree.

Each system must have an entry in the admin node directory that gives DB2 the information it needs to communicate with the remote system. Use the Add System window to specify information about the system (or computer) you want to connect to.

On a server, the Control Center displays a local system under the `Systems` line in the object tree. You can work with objects from the local system, such as instances and databases, but you can't change or remove the local system because it represents your local computer. Figure 5.8 shows the Control Center with three systems. `SUENKARL` is the local system; `GINGER1` and `SVISSER` are remote systems.



Name	Node name	Operating sy...	Type	Protocol
GINGER1	GINGER1	Windows NT	DB2 V5	TCP/IP
SUENKARL	SUENKARL	Windows NT	DB2 V5	LOCAL
SVISSER	SVISSER	OS/2	DB2 V5	TCP/IP

Figure 5.8: The Control Center with three systems.

For DRDA and Version 2 systems, the Control Center displays a system name that corresponds to the name cataloged for that machine. You can perform a subset of the regular functions on these objects.

If a version 5 system (`DB2V5`) appears as a version 2 system (`DB2V2`), use the Change System window to update the admin node directory so that the system can be recognized correctly. To access the Change System window, right-click the system name, and select Change from the pop-up menu. Figure 5.9 shows the Change System window for the remote system `GINGER1`.

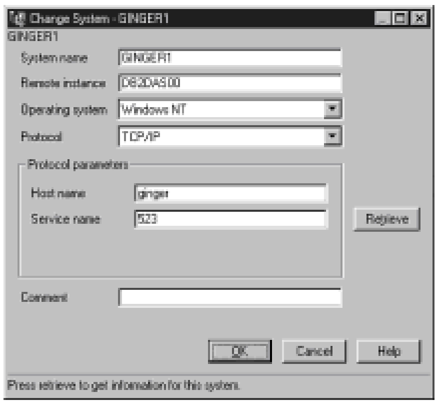

Change System - GINGER1
GINGER1
System name: GINGER1
Remote instance: DB2DAS00
Operating system: Windows NT
Protocol: TCP/IP
Protocol parameters
Host name: ginger
Service name: 523
Comment:
Press retrieve to get information for this system.

Figure 5.9: The Change System window for system `GINGER1`.

To add a remote system, right-click the `SYSTEM` folder, and select Add System from the pop-up menu. In this window, you must enter the hostname of the remote system you want to access, and click Retrieve. The rest of the fields are filled in as soon as the information is retrieved from the remote system (see Figure 5.10). Click Apply to add the remote system to the object tree in the Control Center.

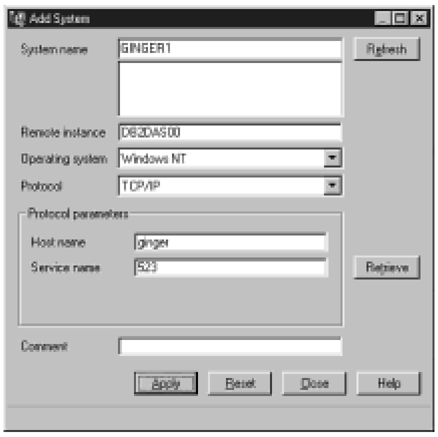

Add System
System name: GINGER1
Remote instance: DB2DAS00
Operating system: Windows NT
Protocol: TCP/IP
Protocol parameters
Host name: ginger
Service name: 523
Comment:

Figure 5.10: The Add System window for system `GINGER1`.

Now that you have a remote system listed in the Control Center, you must add the instances and databases you intend to use on the remote system. To add an instance, expand the objects within the remote system. Right-click the Instances folder, and then select Add from the pop-up menu. In the Add Instance window (see Figure 5.11), click Refresh to have information about the remote instances returned to your system. Select an instance you want to add, and click Apply to add it to the object tree in the Control Center.

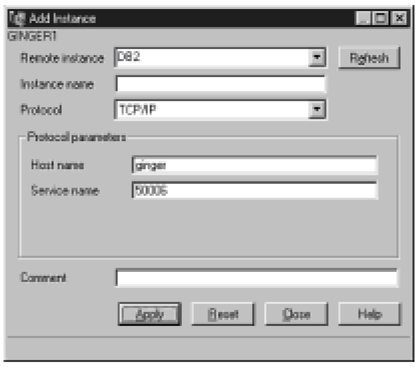

Add Instance
GINGER1
Remote instance: DB2
Instance name:
Protocol: TCP/IP
Protocol parameters
Host name: ginger
Service name: 50006
Comment:

Figure 5.11: The Add Instance window for system `GINGER1`.

The last step is to add the databases from the remote system. The databases will stay on the remote system, but you can view the data and perform administration tasks for this database. To add a database, expand the objects below the instance you have just added. Right-click the Databases folder, and select Add from the pop-up menu. In the Add Database window (see Figure 5.12), click Refresh to list the remote databases. Click Apply to add the database to the object tree in the Control Center. You can now administer the remote database.

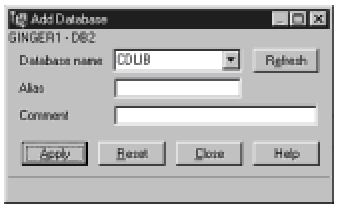

Add Database
GINGER1 - DB2
Database name: CDLIB
Alias:
Comment:

Figure 5.12: The Add Database window for system `GINGER1`.

Summary

In today's lesson, you saw that the Control Center enables you to view and modify the communications on the server. The server is configured automatically to use the protocols that are detected on your system when DB2 is installed.

You can use one or more of the following protocols to listen for requests from clients if your server is on Windows NT: APPC, IPX/SPX, Named Pipes, NetBIOS, or TCP/IP.

Usually, you should accept the default values used to configure the supported communications. If you want to change any values, you should proceed with caution.

What Comes Next?

On Day 6, "Installing DB2 Clients," you learn how to install the DB2 Client Application Enabler code on a Windows NT or Windows 95 workstation.

Q&A

Q When should I install communications support on my system?

A If you're installing a new system from scratch, it's best to install and fully configure any communication protocols you intend to use with DB2 before you install DB2. Then, when you install DB2, the installation program will detect that these protocols are available and will proceed to configure DB2 to use these protocols. If you install protocols after installing DB2, you'll need to manually configure the protocols so that DB2 can use them.

Q How many protocols can I use at the same time?

A You can use all five supported protocols at the same time. This means that if you have five different clients, each set up to use a different protocol, you can configure the DB2 server to accept requests from each client concurrently. The only thing that you have to ensure is that all five protocols are installed and configured on the server.

If, however, you intend to use only one protocol, you should configure your system to use only this protocol. It takes additional resources to configure and start each protocol every time DB2 is used.

Q How do I configure the DB2 Administration Server?

A DB2 automatically configures the DB2 Administration Server to use TCP/IP or NetBIOS if these protocols are installed on your system when DB2 is installed. If you install DB2 before these protocols are installed and configured, you need to manually configure the DB2 Administration Server to use one or more of these protocols. You can't configure the DB2 Administration Server by using the Control Center. Instead, you need to enter a series of commands at a command prompt. As a result, it's highly recommended that you have the protocols you intend to use available on your system before you install DB2.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. What properties can you change if you're modifying the TCP/IP protocol?

2. Do you configure communications for an entire instance or for each individual database?

Exercise

Use the Control Center to view the protocol values set during installation. Don't change any values—just look at them. Then, use the `get admin cfg` and `get dbm cfg` commands to view the database configuration file. Verify that the values are consistent in both locations.

Day 6: Installing DB2 Clients

Overview

Today's lesson shows how to perform an interactive installation of a DB2 client on a Windows NT or Windows 95 system.

LAN clients that will connect to a DB2 Universal Database must have the DB2 Client Application Enabler component installed.

The DB2 Client Application Enabler component is installed with every DB2 version 5 product. If a DB2 version 5 product now exists on this workstation, the DB2 Client

Application Enabler component has already been installed, and there's no need to re-install it. To configure your client to access remote servers, see the later section "Configuring Client-to-Server Communications with the CCA."

DB2 Client Application Enablers and DB2 SDKs are available for AIX, DOS, HP-UX, Macintosh, OS/2, SCO OpenServer, Silicon Graphics IRIX, SINIX, Solaris, Windows 3.1x, Windows 95, and Windows NT. This book covers how to install on Windows 95 and Windows NT systems.

Tip You can also run unattended installations of DB2 products to remote target machines running OS/2, Windows 3.x, Windows 95, or Windows NT. Unattended installations use response files that you set in place beforehand to install and configure DB2 products without any intervention from you. A response file contains the answers to the installation and configuration prompts you need to answer if you perform an interactive installation. Instructions for the unattended installation methods are beyond the scope of this book.

The DB2 Software Developer's Kits are on the DB2 SDK CD-ROM that comes with the DB2 Developer's Edition products. The DB2 Client Application Enablers (except the DOS client) are on the Client Pack CD-ROM or on IBM's Web site (`http://www.software.ibm.com`). Search for the IBM Database 2 Client Application Enabler Pack and follow the instructions provided.

You can install DB2 Client Application Enablers on any number of workstations; no licensing restrictions for installation exist. DB2 Enterprise Edition enables an unlimited number of users to connect to the server. DB2 Workgroup Edition has use-based entitlement restrictions for the number of concurrent users that can connect to the server.

Preparing for the Install

Today's lesson shows how to install the DB2 Client Application Enabler on Windows 95 and Windows NT workstations by using the Typical, Custom, and Compact install types. Before you begin the install, however, read and complete the following sections to ensure that you have the required items and information you need.

Hardware and Software Requirements

Before installing the DB2 product, check to see that you have the proper hardware and software components available on the workstation:

- **Disk space.** You need 95MB of free disk space to install DB2. You also need additional disk space to store the applications that you use with DB2; this amount depends on the size of your applications.

- **Processor.** You can run DB2 Client Application Enabler on a 486 computer, although it's recommended that you use a Pentium-based computer.

- **Memory.** To run the entire set of DB2 graphical tools, you need at least 24MB of memory. The more memory you have, the faster the tools run.

- **Operating systems.** You need to have Windows NT 3.51 or later or Windows 95 version 4.00.950 or later installed.

- **Communications.** You need to use communications to access remote databases. You have a wide selection of communication protocols to choose from: TCP/IP, IPX/SPX, NetBIOS, Named Pipes, and APPC. All these protocols, except APPC, are installable options in the Windows NT and Windows 95 operating systems. You don't need to purchase separate products to provide this support. Ensure that you installed and configured the protocol support that you intend to use.

Creating a Username for Installing DB2 Products

You need to have a valid DB2 username to install DB2. A valid DB2 username is eight characters or less and complies with DB2's naming rules. To comply with DB2's naming rules, the username must begin with letter `A` through `Z`, `@`, `#`, or `$` and contain characters `A` through `Z`, `1` through `9`, `@`, `#`, `$`, or `_`. Don't use the special words `USERS`, `ADMINS`, `GUESTS`, `PUBLIC`, or `LOCAL`, and don't begin the username with `IBM`, `SQL`, or `SYS`.

For Windows NT Workstation, the username must belong to the Administrators group. It must also be a valid DB2 username or have the Act as Part of the Operating System advanced user right. You can create a username on a Windows NT system in the User Manager dialog box (choose Start | Programs | Administrative Tools | User Manager).

If the username doesn't comply with DB2's naming rules but has the Act as Part of the Operating System advanced user right, the setup program creates the username `db2admin` to perform the installation.

Performing a Typical Install

Use the Typical install type if you want to install the DB2 components used most often and to have the DB2 instance created. A Typical install installs all required components, ODBC support, documentation, and commonly used DB2 tools such as the Client Configuration Assistant and the Information Center.

Note You can't selectively uninstall components after the setup program completes the installation.

Table 6.1 lists the products and selectable components installed when you choose the Typical install type. The check marks in the Typical Install column indicate the installed components.

Table 6.1: Products and components for DB2 Client Application Enabler forWindows NT.

Components and Subcomponents	Typical Install
Graphical Tools	✓
Client Configuration Assistant	✓
Control Center	
Performance Monitor	
Visual Explain	
DB2 ODBC Driver	✓
Documentation	✓

Note The Control Center, Performance Monitor, and Visual Explain are installed if you select the Install Components Required to Administer Remote Servers check box.

To install DB2 Client Application Enabler on a Windows NT or Windows 95 workstation by doing a Typical install, follow these steps:

1. Log on as a user that meets the requirements for installing DB2.

2. Shut down any other programs so that the setup program can update files as required.

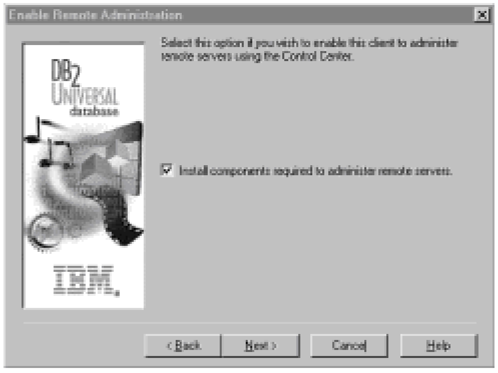
3. Insert the DB2 Client Pack CD-ROM or the DB2 Universal Database for Windows NT CD-ROM into the drive. The auto-run feature automatically starts the setup program.

   Note These steps assume that you're using the DB2 Universal Database for Windows NT CD-ROM.

   Tip The setup program determines the system language and launches itself for that language. If you want to run the setup program in a different language, you need to invoke the program manually. Choose Start | Run. In the Open text box, type **`x:\setup /i=LANGUAGE`**, where **`x:`** represents your CD-ROM drive and ***`LANGUAGE`*** represents the two-character country code for your language (for example, **`EN`** for English). Click OK.

4. In the Welcome dialog box, click the Next button to continue.

5. In the Enable Remote Administration dialog box (see Figure 6.1), select the Install Components Required to Administer Remote Servers check box if you want to administer remote servers from this client and install the Control Center andrelated documentation. Click Next to continue.



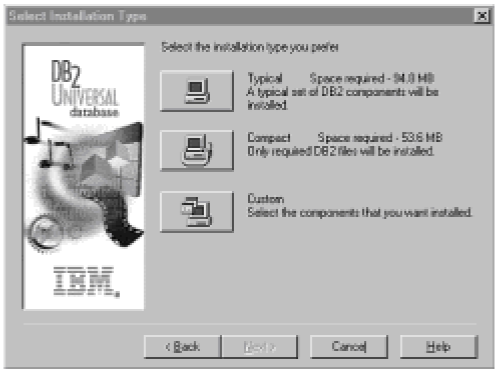
Enable Remote Administration

Select this option if you wish to enable this client to administer remote servers using the Control Center.

[✓] Install components required to administer remote servers.

Figure 6.1: The Enable Remote Administration dialog box.

6. In the Select Installation Type dialog box (see Figure 6.2), select the installation type you prefer by clicking the appropriate button. For these steps, choose Typical.


Select Installation Type

Select the installation type you prefer

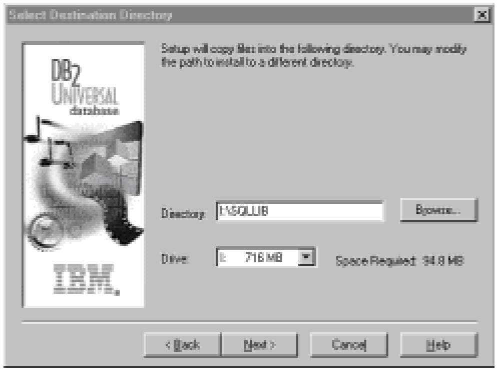
Typical   Space required - 94.8 MB
A typical set of DB2 components will be installed.

Compact   Space required - 53.6 MB
Only required DB2 files will be installed.

Custom
Select the components that you want installed.

Figure 6.2: The Select Installation Type dialog box.

7. In the Select Destination Directory dialog box, select a directory and a drive where DB2 is to be installed (see Figure 6.3). Click Browse to help you select a directory with enough available disk space. (The amount of space required for the product also appears onscreen.)


Select Destination Directory

Setup will copy files into the following directory. You may modify the path to install to a different directory.

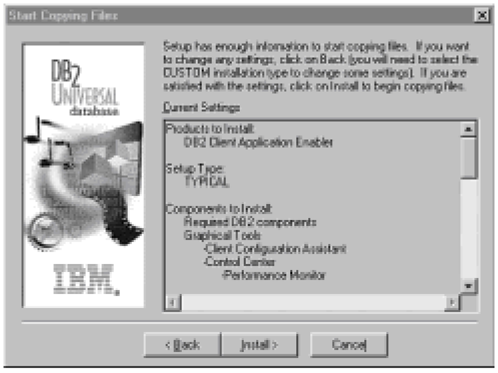
Directory: I:\SQLLIB

Drive: I: 716 MB   Space Required: 94.8 MB

Figure 6.3: The Select Destination Directory dialog box.

Tip If a DB2 version 5 product is already installed on this system, you must install subsequent products and components in the same path. The Directory and Drive text boxes appear grayed out if this is the case.

8. Click Next. DB2 has all the information required to install the product on your system. In the Start Copying Files dialog box (see Figure 6.4), you have one last chance to verify the values you entered. Click Install to have the files copied to your system. You also can click Back to return to the installation screens that you already completed. Make any changes and complete the installation.


Start Copying Files

Setup has enough information to start copying files. If you want to change any settings, click on Back (you will need to select the CUSTOM installation type to change some settings). If you are satisfied with the settings, click on Install to begin copying files.

Current Settings

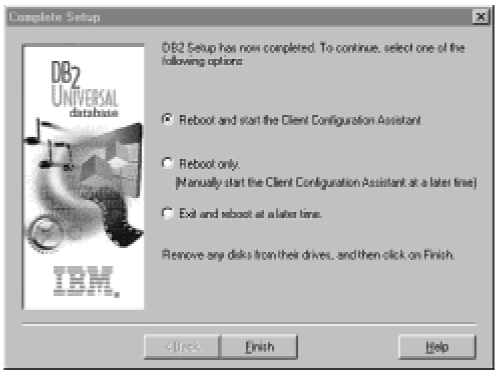
```
Products to Install:
    DB2 Client Application Enabler

Setup Type:
    TYPICAL

Components to Install:
    Required DB2 components
    Graphical Tools
        -Client Configuration Assistant
        -Control Center
            -Performance Monitor
```

Figure 6.4: Start Copying Files window.

9. The installation progress bars appear onscreen while the product is being installed. After the product is installed, you must reboot the workstation before you can begin to use it. Select a reboot option (see Figure 6.5) and click the Finish button to complete the installation.



Complete Setup

DB2 Setup has now completed. To continue, select one of the following options:

[✓ Reboot and start the Client Configuration Assistant]

Reboot only.
(Manually start the Client Configuration Assistant at a later time)

Exit and reboot at a later time.

Remove any disks from their drives, and then click on Finish.

Figure 6.5: The Complete Setup dialog box.

Tip For information on errors encountered during product installation, see the `db2.log` file, which stores general information and error messages resulting from installation and uninstall activities. By default, the file is in the `x:\db2log` directory, where `x:` is the drive on which Windows is installed.

The installation program has completed the following:

- Created DB2 program groups and items (or shortcuts)
- Updated the Windows Registry
- Registered a security server
- Created a default instance named DB2

You've completed all the installation steps. See the later section "Verifying the Connection" to verify the installation and get the workstation ready for general use.

Performing a Custom Install

Use the Custom install type if you want to choose the installed DB2 components. Through a Custom install, only those components that you select are installed and the DB2 instance is created. The default for a Custom install is to install those components and subcomponents you get in a Typical install.

Note If you want to modify the protocols that DB2 uses or to customize the communications parameters, you must choose the Custom install.

To do a Custom install for DB2 Client Application Enabler on a Windows NT or a Windows 95 workstation, follow these steps:

1. Log on as a user that meets the requirements for installing DB2 (see the earlier section "Creating a Username for Installing DB2 Products").

2. Shut down any other programs so that the setup program can update files as required.

3. Insert the DB2 Client Pack CD-ROM or the DB2 Universal Database for Windows NT CD-ROM into the drive. The auto-run feature automatically starts the setup program.

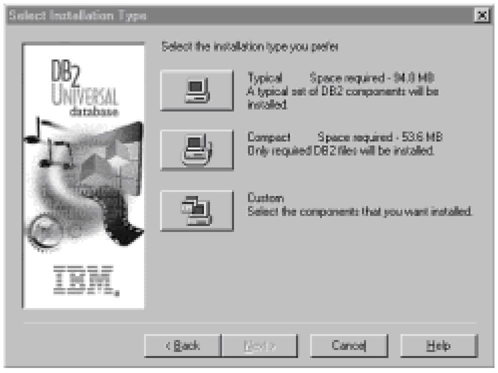
   Note These steps assume that you're using the DB2 Client Pack.

   Tip The setup program determines the system language and launches itself for that language. If you want to run the install in a different language, you need to invoke the setup program manually. Choose Start | Run. In the Open text box, type ***`x`*** **`:\setup /i=`*****`LANGUAGE`***, where ***`x`*** **`:`** represents your CD-ROM drive and ***`LANGUAGE`*** represents the two-character country code for your language (for example, **`EN`** for English). Click OK.

4. In the Welcome dialog box, click Next to continue.

5. In the Enable Remote Administration dialog box, select the Install Components Required to Administer Remote Servers check box if you want to administer remote servers from this client. When this option is selected, the Control Center and related documentation are installed. Then click Next to continue.

6. In the Select Installation Type dialog box (see Figure 6.6), select the installation type you prefer by clicking the appropriate button. For these steps, choose Custom.



Select the installation type you prefer

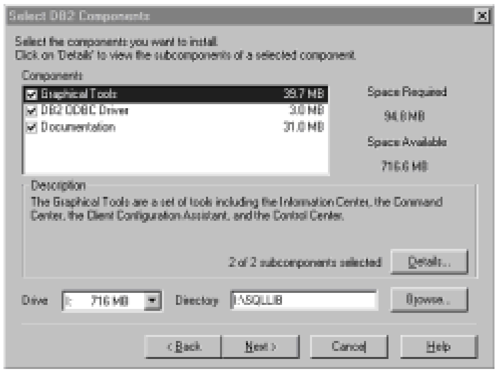
Typical   Space required - 94.0 MB
A typical set of DB2 components will be installed.

Compact   Space required - 53.6 MB
Only required DB2 files will be installed.

Custom
Select the components that you want installed.

Figure 6.6: The Select Installation Type dialog box.

7. In the Select DB2 Components dialog box, decide which components you want to install (see Figure 6.7). The components installed with the Typical install option are checked by default; if you don't want an option, remove the check mark. To see a description for the component, select it and click the Help button.



Select the components you want to install.
Click on 'Details' to view the subcomponents of a selected component.

Components

✓ Graphical Tools	39.7 MB
✓ DB2 ODBC Driver	3.0 MB
✓ Documentation	31.0 MB

Space Required
94.0 MB
Space Available
716.6 MB

Description
The Graphical Tools are a set of tools including the Information Center, the Command Center, the Client Configuration Assistant, and the Control Center.

2 of 2 subcomponents selected

Drive: I: 716 MB   Directory: I:\SQLLIB

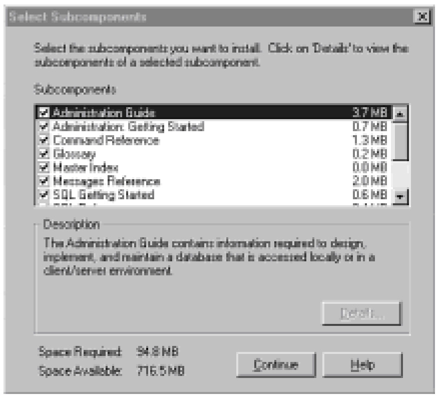
Figure 6.7: The Select DB2 Components dialog box.

Tip Be sure to install the Graphical Tools. This book primarily discusses how to use these tools to maintain your databases. If you don't install the tools, you must use the command-line processor to perform these tasks.

You can install the following products and selectable components on a DB2 workstation:

Graphical Tools	Visual Explain
Client Configuration Assistant	DB2 ODBC Driver
Control Center	Documentation
Performance Monitor	

Some components have subcomponents, the most obvious being Documentation. If you click the Details button, you see each DB2 book that you can choose to install or not install (see Figure 6.8). Select the appropriate subcomponent(s) and then click Continue to return to the Select DB2 Components dialog box. From the Select DB2 Components dialog box, click Next to proceed to the next installation dialog box.


Select Subcomponents

Select the subcomponents you want to install. Click on 'Details' to view the subcomponents of a selected subcomponent.

Subcomponents

✓ Administration Guide	3.7 MB
✓ Administration: Getting Started	0.7 MB
✓ Command Reference	1.3 MB
✓ Glossary	0.2 MB
✓ Master Index	0.0 MB
✓ Messages Reference	2.0 MB
✓ SQL Getting Started	0.6 MB

Description

The Administration Guide contains information required to design, implement, and maintain a database that is accessed locally or in a client/server environment.

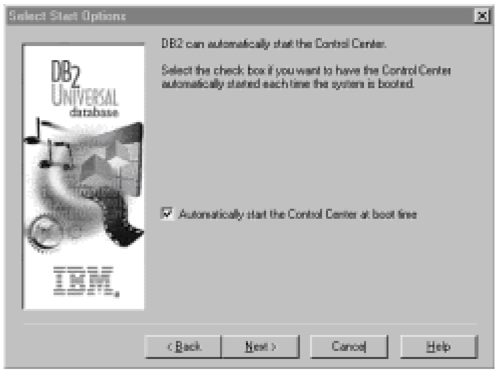
Space Required: 94.8 MB
Space Available: 716.5 MB

Figure 6.8: The Select Subcomponents dialog box.

8. Select a directory on which to install the product. Use Browse to help you select a directory with enough available disk space. (The amount of space required for the product also appears onscreen.)

9. Click Next. In the Select Start Options dialog box (see Figure 6.9), you can choose to have the Control Center automatically started each time the system is booted. Having the Control Center started automatically reduces the number of steps you need to perform each time you boot your system.

Tip If a DB2 version 5 product is already installed on this system, you must install subsequent products and components in the same path. The Directory and Drive text boxes are disabled if this is the case.


Select Start Options

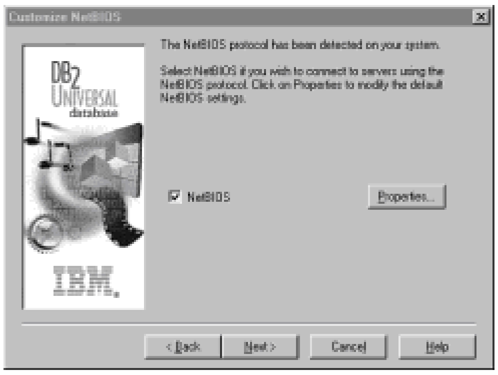
DB2 can automatically start the Control Center.

Select the check box if you want to have the Control Center automatically started each time the system is booted.

☑ Automatically start the Control Center at boot time

Figure 6.9: The Select Start Options dialog box.

10. Click Next. In the Customize Protocol dialog box (see Figure 6.10), select the protocols you want to use when connecting to remote DB2 servers, and then click Properties to modify the default values for the protocol (see Figure 6.11).

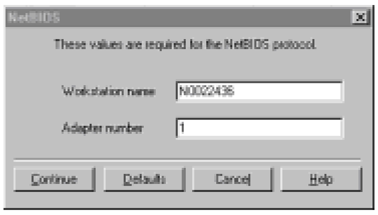

Customize NetBIOS

The NetBIOS protocol has been detected on your system.

Select NetBIOS if you wish to connect to servers using the NetBIOS protocol. Click on Properties to modify the default NetBIOS settings.

☑ NetBIOS

Figure 6.10: The Customize Protocol dialog box.


NetBIOS

These values are required for the NetBIOS protocol.

Workstation name: N0022436

Adapter number: 1

Figure 6.11: The properties dialog box.

These parameters are required for the TCP/IP protocol:

- **Service Name.** The service name of the client. This value is arbitrary.
- **Port Number.** The port number the client will use when making remote connections. By default, this value is set to 0.

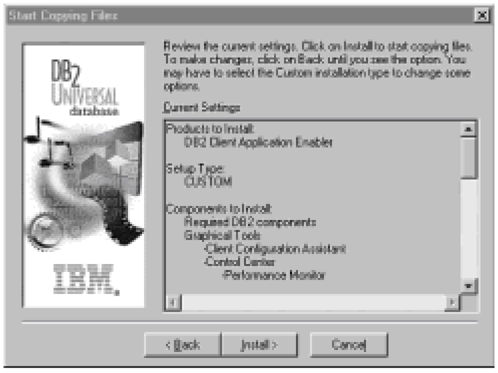
These parameters are required for the NetBIOS protocol:

- **Adapter Number.** The adapter number is the logical network adapter used for the NetBIOS connection. The client uses adapter 0 by default.

- **Workstation Name.** The workstation name of the local machine. One is assigned by DB2 if you don't currently have one. You can accept the value that DB2 assigns or change the value to another name.

  The Transaction Program parameter is required for the APPC protocol. This value is used during remote connections.

11. DB2 has all the information that it needs to install the product on you system. In the Start Copying Files dialog box, you have one last chance to verify the values you've entered (see Figure 6.12). Click Install to have the files copied to your system, or click Back to return to the installation screens that you've already completed to make any changes.



Start Copying Files

Review the current settings. Click on Install to start copying files. To make changes, click on Back until you see the option. You may have to select the Custom installation type to change some options.

Current Settings

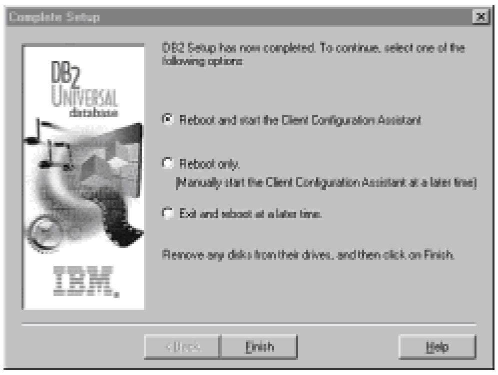
```
Products to Install:
    DB2 Client Application Enabler

Setup Type:
    CUSTOM

Components to Install:
    Required DB2 components
    Graphical Tools
        -Client Configuration Assistant
        -Control Center
            -Performance Monitor
```

Figure 6.12: The Start Copying Files dialog box.

12. The installation progress bars appear onscreen while the product is being installed. After the product is installed, you must reboot the workstation before you can begin to use it. Select a reboot option (see Figure 6.13) and click the Finish button to complete the installation.



Complete Setup

DB2 Setup has now completed. To continue, select one of the following options:

[✓ Reboot and start the Client Configuration Assistant]

Reboot only.
(Manually start the Client Configuration Assistant at a later time)

Exit and reboot at a later time.

Remove any disks from their drives, and then click on Finish.

Figure 6.13: The Complete Setup dialog box.

Tip For information on errors encountered during product installation, see the `db2.log` file, which stores general information and error messages resulting from installation and uninstall activities. By default, the file is located in the *x*:`\db2log` directory, where *x*: is the drive on which Windows is installed.

The installation program has done the following:

- Created DB2 program groups and items (or shortcuts)

- Registered a security service
- Updated the Windows Registry
- Created a default instance named DB2

You've completed all the installation steps. See the later section "Verifying the Connection" to verify the installation and to get the workstation ready for general use.

Performing a Compact Install

The steps to perform a Compact installation are the same as performing a Typical installation, except for the installed product components. During a Compact installation, only the required components are installed. This greatly reduces the amount of disk space required for the product, but it also greatly reduces the available functionality. For example, the graphical tools and the online books aren't installed during a Compact installation, unless you select the Install Components Required to Administer Remote Servers check box.

Compact installation should be used on systems where you use another application to interface to the database. For example, if you plan to use Lotus Approach as the interface to DB2, you may not want to use the DB2 tools or online documents.

The only component installed during a Compact install is the DB2 ODBC Driver.

Configuring Client-to-Server Communications with the CCA

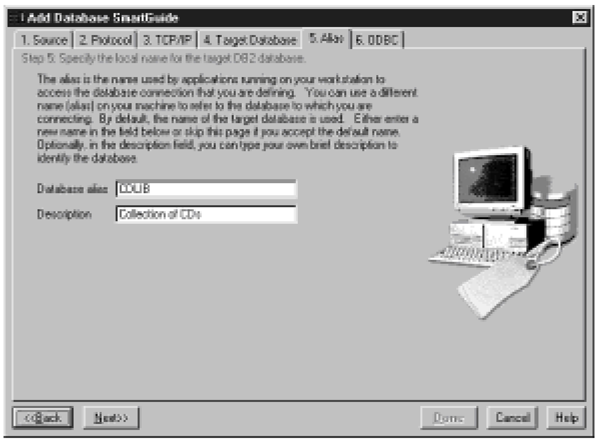
Use the Client Configuration Assistant (CCA) to configure your client workstation to access remote DB2 Universal Database servers. The Client Configuration Assistant provides three configuration methods: one uses a server's access profile, one searches the network for databases, and the other requires that you enter the database name and the communication protocol values of the DB2 server.

If your administrator provided you with a file containing database access information, follow these steps to configure your workstation to access remote servers by using an access profile:

1. Start the CCA by choosing Start | Programs | DB2 for Windows NT | Client Configuration Assistant.

   Note The Welcome panel opens each time you start the CCA, until you add at least one database to your client.

2. In the Add Database SmartGuide, click the Add Database or Add button to configure connections.
3. On the Source page, select Use an Access Profile and click the Next button.
4. On the Target Database page, click the Browse button to select an access profile, or enter the path and name of the file in the File text box.
5. A list of systems, instances, and databases appears. Select the database that you want to use and click Next.
6. On the Alias page, specify the database alias name or add a description of the database (see Figure 6.14). If you don't specify a database alias name, the default will be the same as the database name. This step is optional.


Add Database SmartGuide

1. Source | 2. Protocol | 3. TCP/IP | 4. Target Database | 5. Alias | 6. ODBC

Step 5: Specify the local name for the target DB2 database.

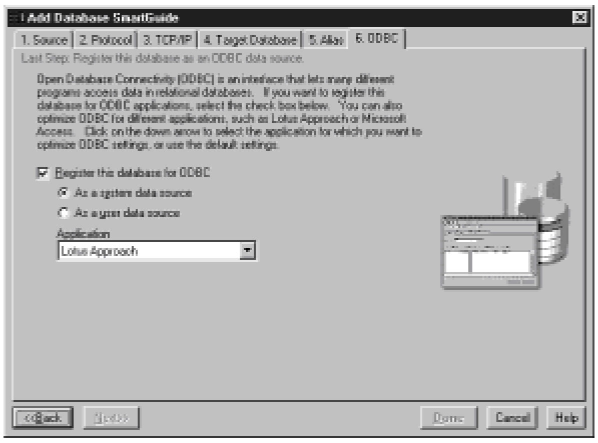
The alias is the name used by applications running on your workstation to access the database connection that you are defining. You can use a different name (alias) on your machine to refer to the database to which you are connecting. By default, the name of the target database is used. Either enter a new name in the field below or skip this page if you accept the default name. Optionally, in the description field, you can type your own brief description to identify the database.

Database alias: CDLIB

Description: Collection of CDs

Figure 6.14: The Alias page ofthe Add Database SmartGuide.

7. If you plan to use applications such as Lotus Approach, advance to the ODBC page to enable ODBC applications (see Figure 6.15). Click the Enable check box on the ODBC page if you plan to use ODBC applications.


Add Database SmartGuide

1. Source | 2. Protocol | 3. TCP/IP | 4. Target Database | 5. Alias | 6. ODBC

Last Step: Register this database as an ODBC data source.

Open Database Connectivity (ODBC) is an interface that lets many different programs access data in relational databases. If you want to register this database for ODBC applications, select the check box below. You can also optimize ODBC for different applications, such as Lotus Approach or Microsoft Access. Click on the down arrow to select the application for which you want to optimize ODBC settings, or use the default settings.

☑ Register this database for ODBC

● As a system data source

○ As a user data source

Application: Lotus Approach

Figure 6.15: The ODBC page ofthe Add Database SmartGuide.

If you don't plan to use ODBC applications, click the Done button to finish using the Add Database SmartGuide and proceed to step 9.

8. If you're using ODBC applications, select the Register This Database for ODBC check box, select the radio button that describes the type of data source you want to register this database as, and select from the Application drop-down box the application that you want to use (for example, Lotus Approach). Click the Done button to finish using the Add Database SmartGuide.

9. In the Confirmation dialog box, click the Test Connection button to test the connection to the specified database. In the Connect to DB2 Database dialog box, enter your username and password to access the database and click OK. If the connection is successful, a message confirming the connection appears. If the connection fails, click the Help button for more information. For example, if the username or password that you entered isn't known on the server, the connection fails. The help message that appears asks you to enter a valid username and password.

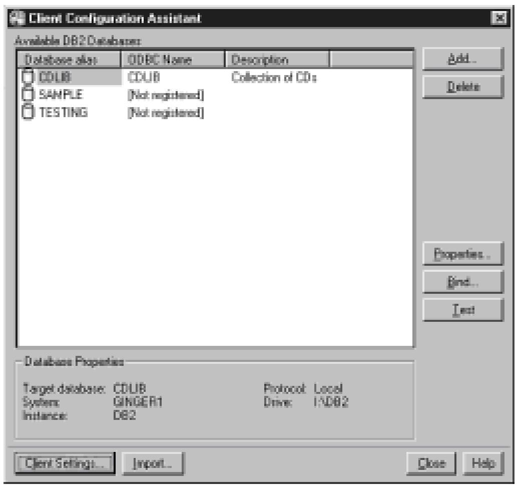
10. You now can use the database. If you want to access another database, select the Add Another button. To finish using the Client Configuration Assistant, click the Close button.

If you want to search the network for databases, follow these steps:

1. Start the CCA by choosing Start | Programs | DB2 for Windows NT | Client Configuration Assistant.

   Note The Welcome panel opens each time you start the CCA, until you've added at least one database to your client.

2. The Client Configuration Assistant dialog box appears (see Figure 6.16). Click Add to move to the Add Database SmartGuide (see Figure 6.17).



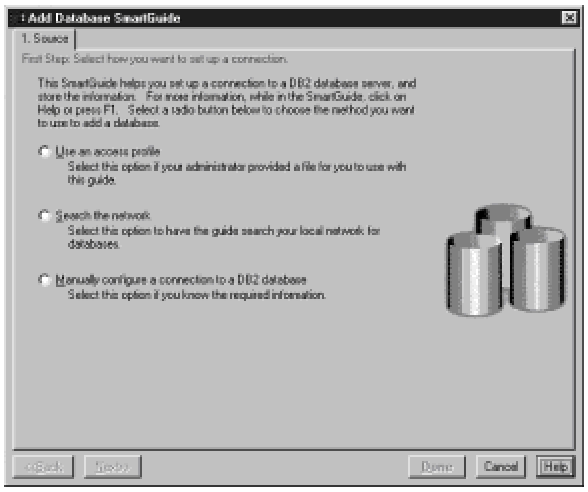
Available DB2 Databases

Database alias	ODBC Name	Description
CDLIB	CDLIB	Collection of CDs
SAMPLE	[Not registered]	
TESTING	[Not registered]	

Database Properties

Target database: CDLIB
System: GINGER1
Instance: DB2
Protocol: Local
Drive: I:\DB2

Figure 6.16: The Client Configuration Assistant.



First Step: Select how you want to set up a connection.

This SmartGuide helps you set up a connection to a DB2 database server, and store the information. For more information, while in the SmartGuide, click on Help or press F1. Select a radio button below to choose the method you want to use to add a database.

( ) Use an access profile
Select this option if your administrator provided a file for you to use with this guide.

( ) Search the network
Select this option to have the guide search your local network for databases.

( ) Manually configure a connection to a DB2 database
Select this option if you know the required information.

Figure 6.17: The Add Database SmartGuide.

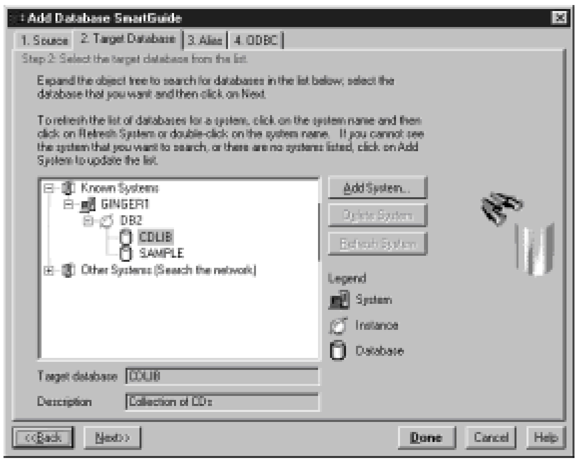
3. On the Source page, select the Search the Network radio button and click Next.

4. On the Target Database page, click the Browse button to select an access profile, or enter the path and name of the file in the File text box.

5. A list of systems, instances, and databases appears. Select the database that you want to use and skip to step 10.

Tip The Other Systems (Search the Network) icon appears only if the client's `DISCOVER` parameter is set to `SEARCH`.

6. Click the + sign beside the Known Systems icon to list all the systems known to your client (see Figure 6.18).



Add Database SmartGuide

1. Source | 2. Target Database | 3. Alias | 4. ODBC

Step 2: Select the target database from the list.

Expand the object tree to search for databases in the list below; select the database that you want and then click on Next.

To refresh the list of databases for a system, click on the system name and then click on Refresh System or double-click on the system name. If you cannot see the system that you want to search, or there are no systems listed, click on Add System to update the list.

```
Known Systems
  GINGER1
    DB2
      CDLIB
      SAMPLE
Other Systems (Search the network)
```

Legend: System, Instance, Database

Target database: CDLIB

Description: Collection of CDs

Figure 6.18: The known systemsin the Client Configuration Assistant.

7. Click the plus sign (+) beside the system to get a list of the instances and databases on it. Select the database that you want to add and skip to step 10.

8. If the system that contains the database that you want isn't listed, click the + sign beside the Other Systems (Search the Network) icon to search the network for additional systems. Click the + sign beside the system to get a list of the instances and databases on it. Select the database that you want to add and skip to step 10.

9. If the system you want is still not listed, you can add it to the list of systems by clicking the Add Systems button. Enter the required communication protocol parameters and click OK. Select the database that you want to add.

10. On the Alias page, specify the database alias name or add a description of the database (refer to Figure 6.14). If you don't specify a database alias name, the default will be the same as the database name. This step is optional.

11. If you plan to use applications such as Lotus Approach, advance to the ODBC page to enable ODBC applications (refer to Figure 6.15). Click the Enable check box on the ODBC page if you plan to use ODBC applications.

    If you don't plan to use ODBC applications, click the Done button to finish using the Add Database SmartGuide and skip to step 13.

12. If you're using ODBC applications, select the Register This Database for ODBC check box, select the radio button that describes the type of data source that you want to register this database as, and select from the Application drop-down box the application that you want to use (for example, Lotus Approach). Click the Done button to finish using the Add Database SmartGuide.

13. In the Confirmation dialog box, click the Test Connection button to test the connection to the specified database. On the Connect to DB2 Database dialog box, enter your username and password to access the database and click OK. If the connection is successful, a message confirming the connection appears. If the connection fails, click the Help button for more information. For example, if the username or password that you entered isn't known on the server, the connection will fail. The help message

that appears will ask you to enter a valid username and password.

14. You now can use the database. If you want to access another database, select the Add Another button. To finish using the Client Configuration Assistant, click the Close button.

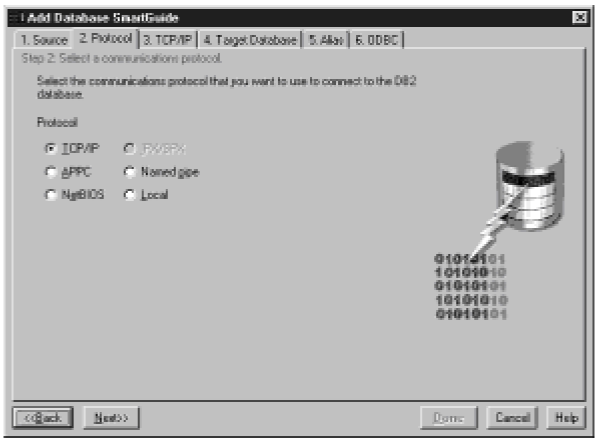
If you don't have an access profile and don't want to search the network, follow these steps to enter the protocol information for the database that you want to connect to:

1. Start the CCA by choosing Start | Programs | DB2 for Windows NT | Client Configuration Assistant. The Welcome panel opens each time you start the CCA, until you've added at least one database to your client.

2. In the Add Database SmartGuide, click the Add Database or Add button to configure connections.

3. On the Source page, select Manually Configure a Connection to a DB2 Database and click Next.

4. Select the radio button that corresponds to the protocol you want to use from the Protocol list (see Figure 6.19). Click Next.


Add Database SmartGuide

Step 2: Select a communications protocol.

Select the communications protocol that you want to use to connect to the DB2 database.

Protocol

[✓ TCP/IP] IPX/SPX
APPC Named pipe
NetBIOS Local

Figure 6.19: The Protocol list.

5. Enter the required communication protocol parameters and click Next. These parameters are required for the TCP/IP protocol:

   - **Hostname.** The hostname of the server where the target database resides.

   - **Service Name.** The service name of the server that matches the entry in the server's TCP/IP services file.

   - **Port Number.** The port number associated with the DB2 instance on the server that contains the target database.

   These parameters are required for the NetBIOS protocol:

   - **Adapter Number.** The adapter number is the logical network adapter used for the NetBIOS connection. The client uses adapter 0 by default.

   - **Server Workstation Name.** The workstation name of the server machine where the target database resides.

- **Local Workstation Name.** The workstation name of the local machine. One is assigned by DB2 if you don't currently have one. You can accept the value that DB2 assigns or change it to another name.

  The Internetwork Address parameter is required for the IPX/SPX protocol. The internetwork address of the DB2 instance on the server contains the target database.

  These parameters are required for the Named Pipes protocol:

  - **Computer Name.** The computer name of the server where the target database resides.
  - **Instance.** The DB2 instance where the target database resides.

  The Symbolic destination name parameter is required for the APPC protocol. This name is found in the CPIC side information in the SNA subsystem.

6. Specify the name of the database that you want to connect to in the Target Database text box and click Next.
7. On the Alias page, specify the database alias name or add a description of the database (refer to Figure 6.14). If you don't specify a database alias name, the default will be the same as the database name. This step is optional.
8. If you plan to use applications such as Lotus Approach, advance to the ODBC page to enable ODBC applications (refer to Figure 6.15). Click the Enable check box if you plan to use ODBC applications.

   If you don't plan to use ODBC applications, click the Done button to finish using the Add Database SmartGuide and skip to step 10.
9. If you use ODBC applications, select the Register This Database for ODBC check box, select the radio button that describes the type of data source that you want to register this database as, and select from the Application drop-down box the application that you want to use (for example, Lotus Approach). Click the Done button to finish using the Add Database SmartGuide.
10. In the Confirmation dialog box, click the Test Connection button to test the connection to the specified database. On the Connect to DB2 Database dialog box, enter your username and password to access the database and click OK. If the connection is successful, a message confirming the connection appears. If the connection fails, click the Help button for more information.
11. You now can use the database. If you want to access another database, select the Add Another button. To finish using the Client Configuration Assistant, click the Close button.

Verifying the Connection

To test the connection, you need to connect to a remote database. If you don't have a database on the server, create the SAMPLE database on the server to test the connection. See Day 4, "Getting Started," for more information.

When the configuration of the server and client is complete, issue the `db2start` command on the server to start the database manager (if it wasn't automatically started at boot time). Then issue the following command in the client's Command Center orcommand-line processor to connect the client to the remote database:

Input
```
connect to database_alias user username using password
```

The database alias is an arbitrary local nickname for the remote database on the client. If you don't provide one, the default is the same as the database name. You use this name when connecting to a database from a client.

The values for *`username`* and *`password`* must be valid for the system on which they're authenticated. By default, authentication takes place on the server. If the database manager is configured for client authentication, the *`username`* and *`password`* must be valid on the client.

If the connection is successful, you get a message showing the name of the database you're connected to:

Output
```
Database product        = DB2/NT 5.0.0
SQL authorization ID    = DB2ADMIN
Local database alias    = SAMPLE
```

Tip If you connect to a database from a Windows NT client by using client authentication and providing a username and password on the `CONNECT` command, you must start the DB2 for Windows NT Security Service on the system. The Security Service is installed by DB2 and set up as a Windows NT service; however, it's not started automatically. To start the DB2 Security Service, double-click the Services icon in Control Panel, select the DB2 Security Server option, and click the Start button.

If you receive an error message, make sure that the SAMPLE database exists on the server and the database manager was started on the server.

You can now retrieve data from the database. For example, to retrieve a list of all the table names listed in the system catalog table, enter the following SQL command in the Command Center:

Input
```
select tabname from syscat.tables
```

To select data from the `EMPLOYEE` table, you must enter the appropriate schema name. For example, the schema name for my tables is `DB2ADMIN`. I use the following command to see the data from the `EMPLOYEE` table on my computer:

Input
```
select * from DB2ADMIN.EMPLOYEE
```

When you finish using the database connection, issue the **`connect reset`** command to end the database connection.

You're ready to start using the DB2 client to communicate with a DB2 server. See Day 10, "Accessing the Data," for details.

Summary

In today's lesson, you saw that the DB2 client function is provided by software known as DB2 Client Application Enabler. We can't call this a product because this software is "free." It's legally known as a distributive option, meaning that you can copy the software to any workstation that supports it. Licensing is taken care of at the server level.

You can acquire the DB2 Client Application Enabler software in three ways:

- By using the DB2 Client Pack CD-ROM included with DB2's Workgroup and Enterprise Editions

- By downloading the software from IBM's Web site
- By installing the software from the DB2 Universal Database version 5 CD-ROM

If you want your server to also act as a client, you don't need to install the DB2 Client Application Enabler software separately on the server. The DB2 Client Application Enabler software is already built into each DB2 product. You need to install this software only on a remote system that's to act as a client.

What Comes Next?

On Day 7, "Designing a CDLIB Database," you learn the procedure for designing a relational database and are introduced to the sample database used throughout this book.

Q&A

Q When are the graphical tools installed with the client?

A You can install the DB2 Tools if you're using DB2 Client Application Enabler for OS/2, Windows NT, or Windows 95. The tools aren't provided with the clients for the other platforms.

If you perform a Custom install, you can choose to have the tools installed. If you perform a Typical or Compact install, you're asked whether you intend to administer remote servers from this client. If you respond yes, the tools are installed.

For the purposes of this book, you need to have the tools. If you intend to perform the exercises from the server, you don't need the tools on the client.

Q Where can I install the product?

A If you already have a DB2 Version 5 product installed on your system, you must install any subsequent product on the same drive in the same directory. If this is your first DB2 version 5 product, you can install on any drive that has the required amount of space.

Q What about communications?

A The protocols that you need on a client to communicate with a remote server aren't configured or selected during installation. You use the Client Configuration Assistant after the client software is installed to configure the protocols you need for communications. Five protocols are supported for the Windows NT client: APPC, IPX/SPX, Named Pipes, NetBIOS, and TCP/IP. For the Windows 95 client, all but the APPC protocol are supported. You must use a protocol that matches one of the protocols set up on the server you intend to access.

Workshop

The purpose of the Workshop is to allow you to test your knowledge of the material covered in the lesson. See if you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. What three configuration methods are available with the Client Configuration Assistant?

2. When should the client's `DISCOVER` parameter be set to `SEARCH`?

3. Why would you want to register a database for ODBC?

4. If you can't connect to the server, what's the most likely problem?

Exercise

Because this is the first time that the Client Configuration Assistant was introduced, try out the different panels and options to see what's available with this tool.

Day 7: Designing a CDLIB Database

Overview

Today's lesson takes you through the steps of designing a database, tables, and referential constraints to hold the data for a CD collection. This lesson focuses on planning and designing a database structure; the actual creation of the database and its tables is covered on Day 8, "Creating Databases." Planning a database structure is as important as actually creating the database—in fact, good planning helps prevent hard-to-fix problems down the road.

This lesson takes you through the usual steps for designing a database to show you the structure of the database for the CD collection. These steps include deciding what data to store in the database, choosing appropriate columns and data types, defining relationships between the tables, identifying the primary key of the table, identifying constraints and foreign keys, and normalizing tables.

As you learned on Day 2, " Exploring the Capabilities of DB2 Universal Server," a table is a collection of rows and columns. Each row contains information about one object, such as the information for one CD. Each column contains the same type of information, such as the CD's title. A *table* is a group of information regarding one subject, such as a collection of songs. One or more tables are related to each other through the addition of foreign keys that link information, such as a song title to a particular album.

Deciding What Data to Store in the Database

For a CD collection, you may want to record information such as the name of the artist or group, the title of the CD, the names of each song on the CD, the length of each song, the length of the CD, the year the CD was released, members of the group, type of music, record label, and perhaps an audio clip or photo of the artist. Before you design your tables, however, you must understand each entity and its relationship to other entities. Artist, Album, and Song are examples of entities. The artist records one or more albums, each of which contains one or more songs. The relationship between artists, songs, and albums applies to each row of the tables.

The information contained within a table depends on the relationships to be expressed, the amount of flexibility needed, and the data retrieval speed desired. The following sections describe each table and the attributes for each column within the table.

For each column, you need to provide the following information:

- **Column name.** You can have up to 500 columns in a table. Each column name can contain up to 18 characters and must be different than all other columns in the table. Choose a column name that describes the data you intend to put in the column. For example, you can choose the column name `ALBUM_TITLE` for the column that will contain the title of each album in the table.

- **Type of data and length.** You must analyze the data that you intend to store in the column. This quite often is difficult to do because you don't always know in advance what data needs to be put into the table. Data usually falls into the categories of character, numeric, or large object. There are many data types for each category.

  The following data types can be used for the columns in the table. There are some size restrictions inherent in the data types, so it's important to choose the data type that gives you the flexibility you need.

Data Type	Description
`INTEGER`	For a large integer.
`SMALLINT`	For a small integer.
`DOUBLE`	For a floating-point number.
`DECIMAL`	For a decimal number.
`CHARACTER`	For a fixed-length character string.
`VARCHAR`	For a varying-length character string with a maximum length of 4000.
`LONG VARCHAR`	For a varying-length character string with a maximum length of 32700.
`BLOB`	For a binary large object string.
`CLOB`	For a character large object string.
`DBCLOB`	For a double-byte character large object string.
`GRAPHIC`	For a fixed-length graphic string.
`VARGRAPHIC`	For a varying-length graphic string with a maximum length of 2000.
`LONG VARGRAPHIC`	”For a varying-length graphic string with a maximum length of 16350.
`DATE`	For a date.
`TIME`	For a time.
`TIMESTAMP`	For a timestamp.
`Distinct type`	For a type you want to create, based on your specific needs. Distinct types are listed in the form of `schema.type`.

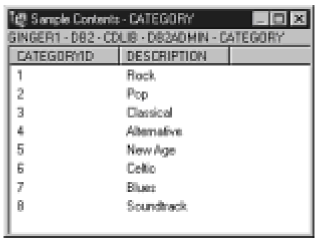
- **Whether the column has a default value or is null.** If a column in the table will likely contain a certain value, you can define the value as the default for the column. For example, if you have a table containing addresses and most of the people in your table are from the same city or state, you can define this as the default value. That way, if you don't enter a value for the column, the default value is automatically used.

  Some columns may never contain a value. In this case, you should define the column as being null. A *null value* indicates that the column value is unknown or not applicable. For example, if you have a column for email addresses in your address table, you should make the column null to accommodate those people in your table who don't currently have an email address.

The CATEGORY Table

The CATEGORY table contains the musical categories for the CD collection (see Figure 7.1). This table includes the following attributes:

Column Name	*Data Type, Size, and Whether Nullable*
CATEGORYID	SMALLINT; not nullable
DESCRIPTION	VARCHAR, 50; not nullable


Sample Contents - CATEGORY
GINGER1 - DB2 - CDLIB - DB2ADMIN - CATEGORY

CATEGORYID	DESCRIPTION
1	Rock
2	Pop
3	Classical
4	Alternative
5	New Age
6	Celtic
7	Blues
8	Soundtrack

Figure 7.1: A sample CATEGORY table.

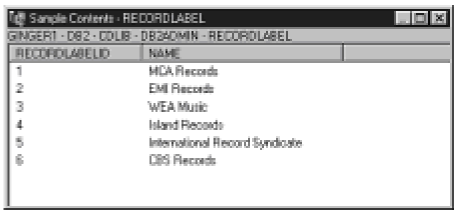
The primary key of this table, CATEGORYID, contains a unique ID number for each category type. Category types include rock, classical, country, and soundtrack.

The RECORDLABEL Table

The RECORDLABEL table contains the names of each record label for the CD collection (see Figure 7.2). Examples of record labels include MCA Records, EMI/Capitol Records, and WEA Music. The table includes the following attributes:

Column Name	***Data Type, Size, and Whether Nullable***
`RECORDLABELID`	`SMALLINT`; not nullable
`NAME`	`VARCHAR`, 50; not nullable

The primary key of this table, `RECORDLABELID`, contains a unique ID number for each record company.



GINGER1 - DB2 - CDLIB - DB2ADMIN - RECORDLABEL

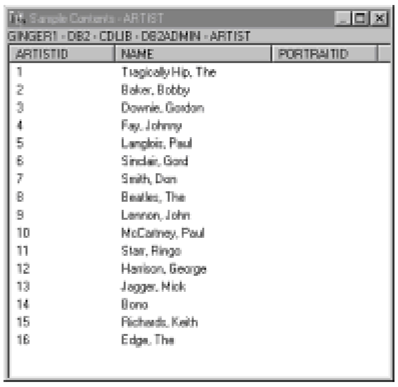
RECORDLABELID	NAME
1	MCA Records
2	EMI Records
3	WEA Music
4	Island Records
5	International Record Syndicate
6	CBS Records

Figure 7.2: A sample `RECORDLABEL` table.

The `ARTIST` Table

The `ARTIST` table contains information about each person who is a performer on an album (see Figure 7.3). Artist names could include John Lennon, Bono of U2, and David Bowie. This table includes the following attributes:

Column Name	***Data Type, Size, and Whether Nullable***
`ARTISTID`	`SMALLINT`; not nullable
`NAME`	`VARCHAR`, 50; not nullable
`PORTRAITID`	`SMALLINT`; nullable



Sample Contents - ARTIST

GINGER1 - DB2 - CDLIB - DB2ADMIN - ARTIST

ARTISTID	NAME	PORTRAITID
1	Tragically Hip, The	
2	Baker, Bobby	
3	Downie, Gordon	
4	Fay, Johnny	
5	Langlois, Paul	
6	Sinclair, Gord	
7	Smith, Don	
8	Beatles, The	
9	Lennon, John	
10	McCartney, Paul	
11	Starr, Ringo	
12	Harrison, George	
13	Jagger, Mick	
14	Bono	
15	Richards, Keith	
16	Edge, The	

Figure 7.3: A sample ARTIST table.

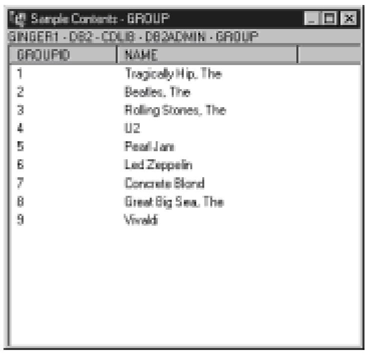
The primary key of this table, ARTISTID, contains a unique ID number for each artist. PORTRAITID is a foreign key to the PORTRAIT table, where a photo of the artist can be stored.

The GROUP Table

The GROUP table, shown in Figure 7.4, contains information regarding a musical group that performs on an album. Group names could include the Beatles, U2, and the Rolling Stones, as well as solo artists such as John Lennon or Loreena McKennitt or composers such as Vivaldi. This table includes the following attributes:

Column Name	***Data Type, Size, and Whether Nullable***
GROUPID	SMALLINT; not nullable
NAME	VARCHAR, 50; not nullable

The primary key of this table, GROUPID, contains a unique ID number for each group.



Sample Contents - GROUP

GINGER1 - DB2 - CDLIB - DB2ADMIN - GROUP

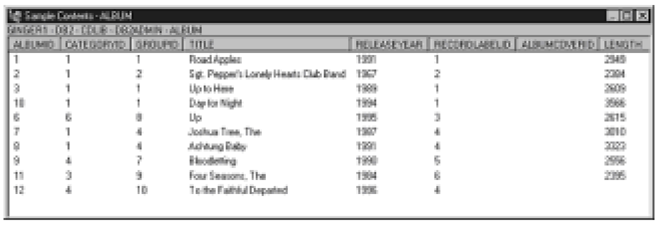
GROUPID	NAME
1	Tragically Hip, The
2	Beatles, The
3	Rolling Stones, The
4	U2
5	Pearl Jam
6	Led Zeppelin
7	Concrete Blond
8	Great Big Sea, The
9	Vivaldi

Figure 7.4: A sample GROUP table.

The **ALBUM** Table

The ALBUM table contains information about a musical recording on CD (see Figure 7.5). Album titles could include *Pop*, *Abbey Road*, and *Four Seasons*. This table includes the following attributes:

Column Name	***Data Type, Size, and Whether Nullable***
ALBUMID	SMALLINT; not nullable
CATEGORYID	SMALLINT; not nullable
GROUPID	SMALLINT; not nullable
TITLE	VARCHAR, 50; not nullable
RELEASEYEAR	SMALLINT; not nullable
RECORDLABELID	SMALLINT; not nullable
ALBUMCOVERID	SMALLINT; nullable
LENGTH	SMALLINT; nullable


Sample Contents - ALBUM

GINGER1 - DB2 - CDLIB - DB2ADMIN - ALBUM

ALBUMID	CATEGORYID	GROUPID	TITLE	RELEASEYEAR	RECORDLABELID	ALBUMCOVERID	LENGTH
1	1	1	Road Apples	1991	1		2949
2	1	2	Sgt. Pepper's Lonely Hearts Club Band	1967	2		2384
3	1	1	Up to Here	1989	1		2609
10	1	1	Day for Night	1994	1		3566
6	6	8	Up	1995	3		2815
7	1	4	Joshua Tree, The	1987	4		3010
8	1	4	Achtung Baby	1991	4		3323
9	4	7	Bloodletting	1990	5		2556
11	3	9	Four Seasons, The	1984	6		2395
12	4	10	To the Faithful Departed	1996	4		

Figure 7.5: A sample ALBUM table.

The primary key of this table, ALBUMID, contains a unique ID number for each CD in the collection.

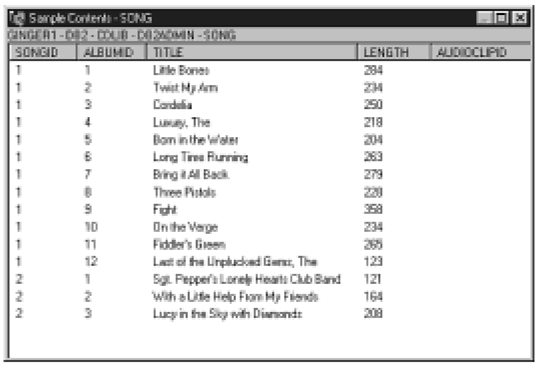
For each album, the year it was released and the length of the album are included.A check constraint is defined on the RELEASEYEAR to ensure that the date is enteredaccurately.

This table has several foreign keys: CATEGORYID is a foreign key to the CATEGORY table, GROUPID is a foreign key to the GROUP table, RECORDLABELID is a foreign key to the RECORDLABEL table, and ALBUMCOVERID is a foreign key to the ALBUMCOVER table.

The **SONG** Table

The `SONG` table contains information regarding each song on an album (see Figure 7.6). This table includes the following attributes:

Column Name	***Data Type, Size, and Whether Nullable***
`SONGID`	`SMALLINT`; not nullable
`ALBUMID`	`SMALLINT`; not nullable
`TITLE`	`VARCHAR`, 50; not nullable
`LENGTH`	`SMALLINT`; not nullable
`AUDIOCLIPID`	`SMALLINT`; nullable



SONGID	ALBUMID	TITLE	LENGTH	AUDIOCLIPID
1	1	Little Bones	284	
1	2	Twist My Arm	234	
1	3	Cordelia	250	
1	4	Luxury, The	218	
1	5	Born in the Water	204	
1	6	Long Time Running	263	
1	7	Bring it All Back	279	
1	8	Three Pistols	228	
1	9	Fight	358	
1	10	On the Verge	234	
1	11	Fiddler's Green	265	
1	12	Last of the Unplucked Gems, The	123	
2	1	Sgt. Pepper's Lonely Hearts Club Band	121	
2	2	With a Little Help From My Friends	164	
2	3	Lucy in the Sky with Diamonds	208	

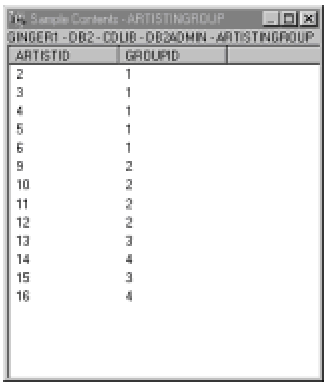
Figure 7.6: A sample `SONG` table.

The primary key of this table is a combination of `SONGID` and `ALBUMID`. The same song can be recorded by many artists, but typically a CD contains only a single version of the song. (There are exceptions, but this lesson ignores these cases for simplicity.)

The `TITLE` column contains the title of each song, such as *Salvation*, *Paint It Black*, and *Courage*. The length of each song is also included. `AUDIOCLIPID` is a foreign key to the `AUDIOCLIP` table, where a sample of the song may be stored.

The `ARTISTINGROUP` Table

The `ARTISTINGROUP` table associates each artist with a group (see Figure 7.7). This table takes into account that a group is made of several members, and these members might actually be parts of different groups in the course of their careers. For example, Paul McCartney was a member of the Beatles and Wings.



Sample Contents - ARTISTINGROUP

GINGER1 - DB2 - CDLIB - DB2ADMIN - ARTISTINGROUP

ARTISTID	GROUPID
2	1
3	1
4	1
5	1
6	1
9	2
10	2
11	2
12	2
13	3
14	4
15	3
16	4

Figure 7.7: A sampleARTISTINGROUP table.

The primary key of this table is a combination of ARTISTID and GROUPID. ARTISTID is a foreign key to the ARTIST table, and GROUPID is a foreign key to the GROUP table:

Colunn Name	***Data Type, Size, and Whether Nullable***
ARTISTID	SMALLINT; not nullable
GROUPID	SMALLINT; not nullable

The PORTRAIT Table

The optional PORTRAIT table contains GIF images of some artists. This table includes the following attributes:

Column Name	***Data Type, Size, and Whether Nullable***
PORTRAITID	SMALLINT; not nullable
PORTRAIT	BLOB, 1MB; not nullable

The primary key is the PORTRAITID.

The AUDIOCLIP Table

The optional AUDIOCLIP table contains WAV samples of songs. This table includes the following attributes:

Column Name	***Data Type, Size, and Whether Nullable***
AUDIOCLIPID	SMALLINT; not nullable
AUDIOCLIP	BLOB, 1MB; not nullable

The primary key is AUDIOCLIPID.

The ALBUMCOVER Table

The optional ALBUMCOVER table contains GIF samples of CD covers. This table includes the following attributes:

Column Name	***Data Type, Size, and Whether Nullable***
ALBUMCOVERID	SMALLINT; not nullable
ALBUMCOVER	BLOB, 1MB; not nullable

The primary key is the ALBUMCOVERID.

Defining Tables for Each Type of Relationship

In a database, you can express several types of relationships. The type of relationship can vary, depending on the specific environment. You'll want to define separate tables for different types of relationships:

- One-to-one relationships are single valued in both directions. For example, each song has one title. This type of relationship can be assigned to the album table or the song table. This sample database associates song titles with the SONG table.

- One-to-many relationships are multivalued in one direction. For example, the relationship between categories and albums is one-to-many because each category has many albums and each album has one category defined. (In reality, an album can belong to many categories, but for simplicity I've associated a single category per album.) The *many* side in the album/category relationship is *album*, so this example defines a table for albums.

- Many-to-one relationships are also multivalued in one direction. For example, an album has many songs. The *many* side of the songs-album relationship is *songs*, so this example defines a `SONG` table.

- Many-to-many relationships are multivalued in both directions. For example, it usually takes many performers to record one song, and each performer usually records many songs. This type of relationship can be expressed in a table with a column for each entity.

Figure 7.8 shows the many types of relationships in the CDLIB database.

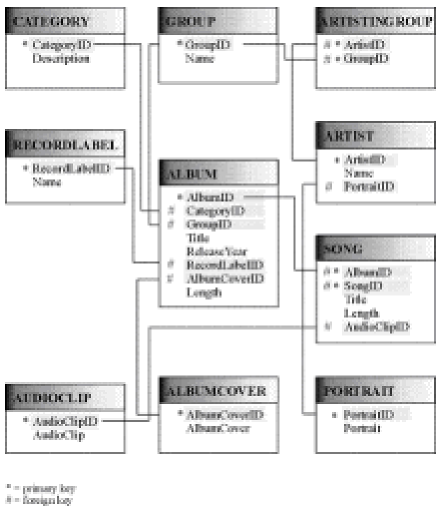


Figure 7.8: Structure of the CDLIB database.

Identifying the Primary Key

You should determine whether you want to define a primary key for your table. As you learned on Day 2, "Exploring the Capabilities of DB2 Universal Server," a *primary key* is one of the unique keys defined on a table but is selected as the column of first importance on the table. The table can have only one primary key, but the primary key can be made up of one or more columns. An asterisk (*) next to the column name in Figure 7.8 indicates that it's a primary key.

The key columns to consider should have the following qualities:

- **Persistence.** A value must always be present in the row. The columns of a primary key can't contain null values.

- **Uniqueness.** Each key value is and always will be different for each row. A primary key is a special kind of unique key that uniquely identifies a row of a table. You can choose multiple columns as the primary key. For example, you might want to create a primary key on the song ID and album ID columns of a song table to ensure uniqueness.

- **Stability.** The key value is one that should never change to another value. For example, you don't want to define a person's name as the primary key because the person may change his or her name—for example, John Mellencamp originally recorded under the name of John Cougar and then as John Cougar Mellencamp. It's common to use an identifier column such as employee number as the primary key because this value is unlikely to change.

If you have more than one table describing properties of the same entity, make sure that equal values represent the same entity. For example, if you add an `ADDRESS` table to contain the Web site and email addresses for each group, you would have a column for the

GROUPID that would match the GROUPID column in the GROUP table. The connecting columns can have different names or the same name. For example, you can call the column GROUPID in the GROUP table and GRPID in the ADDRESS table.

Identifying Constraints and Foreign Keys

You can define several types of constraints in your table, including foreign key constraints, table check constraints, and triggers.

Foreign key constraints let you define relationship tables and require that all values of a given column of a table also exist in some other table. For example, the DEPTNO column of the EMPLOYEE table has a foreign key defined on the DEPTNO column of the DEPARTMENT table. This means that an employee can't be assigned to a department unless the department already exists.

A number sign (#) next to the column name in Figure 7.8 indicates that it's a foreign key. Foreign keys are defined on the dependent table and rely on the data in the parent table.

Table check constraints enforce business rules that you define in your database design. These constraints specify conditions that are enforced for each row of the table and are automatically activated when data is updated or inserted in the table.

You can use a table check constraint for validation. For example, you can define a constraint to check the validation of the RELEASEYEAR of an album. You can make sure that the entered value is greater than 1900 and less than the current year.

You can also define triggers to help support business rules. A trigger is activated when data is inserted, updated, or deleted from the table. For example, you can create a trigger that sends you an email message whenever a business rule such as a credit limit is exceeded.

Normalizing Your Table

Normalizing helps you avoid redundancies and inconsistencies in your data. The main idea behind normalizing is to reduce tables to a set of columns in which all the non-key columns depend on the entire primary key of the table.

You might think that normalizing your tables will hurt performance because querying data from normalized tables requires joining one or more of the tables. The actualperformance difference between accessing data in normalized tables as opposed to non-normalized tables is minimal. In fact, the benefits of avoiding redundancies and inconsistencies far outweighs the minimal differences in performance.

The process of normalizing your tables involves analyzing them to see whether they violate a series of normalization rules. These rules are called *first normal form*, *second normal form*, *third normal form*, and so on. The following sections briefly describe the rules for first, second, and third normal forms and provide examples of tables that violate the rules and how they can be normalized. The fourth and fifth normal forms are beyond the scope of this book.

First Normal Form

In first normal form, one value exists in each row and column position, never a set of values. For example, the following table violates first normal form because the ARTIST column contains several values for each occurrence of GROUP.

Group ID (Primary Key)	*Group Name*	*Artist*
1	Rolling Stones	Mick Jagger, Keith Richards, Charlie Watts, Ronnie Woods, Bill Wyman,Brian Jones
2	The Beatles	John Lennon, Paul McCartney, George Harrison, Ringo Starr

The following example shows the table in first normal form.

Group ID (Primary Key)	*Group Name*	*Artist ID (Primary Key)*	*Artist Name*
1	Rolling Stones	1	Mick Jagger
1	Rolling Stones	2	Keith Richards
1	Rolling Stones	3	Ronnie Woods
1	Rolling Stones	4	Charlie Watts
1	Rolling Stones	5	Bill Wyman
1	Rolling Stones	6	Brian Jones
2	The Beatles	7	John Lennon
2	The Beatles	8	Paul McCartney
2	The Beatles	9	George Harrison
2	The Beatles	10	Ringo Starr

Second Normal Form

In second normal form, each column not in the key depends on all columns in the key. This reduces repetition among database tables. Second normal form is violated when a non-key column depends on a subset of a composite key, as in the following example:

Song ID (Primary Key)	Album ID (Primary Key)	Album Title	Song Title
1	1	Sgt. Pepper's Lonely Hearts Club Band	Fixing a Hole
2	2	Pop	Please
3	1	Sgt. Pepper's Lonely Hearts Club Band	Being for the Benefit of Mr. Kite!

The key consists of the `ALBUMID` and `SONGID` columns together. Because the album title depends only on the value of the `ALBUMID` column (a subset of the composite key), the table violates the rule for second normal form.

The problems with this design are as follows:

- The album name is repeated for each song. An album like *Sgt. Pepper's Lonely Hearts Club Band* has 13 songs, so the album title will be repeated 13 times.
- The album title won't change, but when you're inserting data into the table, you must ensure that you spell the album title correctly each time you enter it.

To satisfy second normal form, the information in the preceding table should be split into the following two tables, with the non-key columns depending on the entire key in each table:

Song ID(Primary Key)	Album ID(Primary Key)	Song Title
1	1	Fixing a Hole
2	2	Please
3	1	Being for the Benefit of Mr. Kite!
4	1	Getting Better

Album ID (Primary Key)	Album Title

1	Sgt. Pepper's Lonely Hearts Club Band
2	Pop

Third Normal Form

In third normal form, each non-key column is independent of other non-key columns and depends only on the key. The following table violates the third normal form:

Album ID(Primary Key)	**Album Name**	**Record Label**	**Phone Number**
1	Abbey Road	Apple	555-1234
2	Pop	MCA	555-5678
3	Let It Be	Apple	555-1234

Third normal form is violated in this table because a non-key column such as `PHONE-NUMBER` depends on another non-key column such as `RECORDLABEL`. Changing the phone number for the record label of a single album won't change the phone number for all other rows that refer to the same record label. The inconsistency after an update is as follows:

Album ID (Primary Key)	***Album Name***	***Record Label***	***Phone Number***
1	Abbey Road	Apple	555-9999
2	Pop	MCA	555-5678
3	Let It Be	Apple	555-1234

You can normalize this table by providing a new table with columns for record labels and phone numbers. In this case, updating a phone number is much easier because the update has to be made only to the new table. A SQL query that shows the phone number with the name of an album is more complex to write because it requires joining the two tables. The following tables are defined as a result of normalizing the tables:

Album ID (Primary Key)	Album Name	Record Label ID
1	Abbey Road	1
2	Pop	2
3	Let It Be	1

Record Label ID (Primary Key)	Name	Phone Number
1	Apple	555-9999
2	MCA	555-5678

Summary

In today's lesson, you saw the several requirements behind designing a database. You must decide what you want to store in the database, define tables for each relationship, identify the primary key for each table, then identify constraints and foreign keys, and finally normalize your tables.

Today's lesson also introduced the sample database used in this book. It's introduced while describing the steps that must take place when designing a database. The database is called CDLIB and contains the information for a collection of CDs.

No matter how carefully you plan for your database, you can't anticipate some situations. For this reason, you should go through a scenario with as much different data that you can think of to see whether your design works. Remember, the more flexible you try to make your database, the more difficult it can become.

Normalizing tables is a common topic for relational databases. This book covers up to third normal form.

What Comes Next?

On Day 8, "Creating Databases," you learn how to use the Control Center to create the CDLIB database and the tables introduced in this lesson.

Q&A

Q How do I choose a primary key?

A You must look at all the columns in your table and choose the column that's always present, always unique, and rarely changes. Most often you'll decide to introduce a new column to your table that holds these properties. For example, in a table containing information about employees, an employee number is assigned to each employee. The number is unique for each person, and the employee uses the number for the entire time her or she is employed at the company. Often when an employee leaves, the number isn't reassigned, so if the employee returns, he or she is assigned the same number as before.

Q How many columns can I have in a table?

A DB2 enables you to have 500 columns in a single table.

Q How do I choose a data type?

A You must analyze the data that you intend to store in the column. This is quite often difficult to do because you don't always know in advance what data will need to be put into the table. Data usually falls into the categories of character, numeric, or large object. There are many data types for each category.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson. The answers appear in Appendix A, "Answers to Quiz Questions."

Quiz

1. What different types of relationships can you represent in a table?

2. Where are foreign keys defined?

3. Why should you normalize your tables?

4. Is it a good idea to use a phone number as a primary key? Why or why not?

Exercise

Go through the lesson and define a database that you have an idea for—for example, a database that contains an inventory of videos or books.

Week 1

In Review

Congratulations—you've survived Week 1! On Day 1, you learned about the various products that make up the DB2 family. Day 2 showed you many of the concepts that relate to a relational database in general and how these related to DB2 Universal Database. On Day 3, you installed and configured DB2 on your workstation to prepare for future lessons. Day 4 introduced you to the tasks you need to complete once the product is installed. Day 5 taught you how to customize the communication protocols that were set during installation. On Day 6, you installed and configured a database client. On Day 7, you were introduced to the CDLIB database that you will use throughout the rest of the book.

Week 2: At a Glance

Overview

During Week 2, you'll see how to create databases, tables, and table spaces; how to access data in your tables; how to look at messages or jobs; how to back up and recover data; and how to export, import, load, and replicate data. You will perform these tasks in your day-to-day job as a database administrator. Here's a day-by-day summary of Week 2:

- On Day 8, "Creating Databases," you'll use the Control Center to create a database called CDLIB. When the database is created, you'll use SmartGuides to create several tables in the database.
- Day 9, "Creating Table Spaces," teaches you the concepts of table spaces and how to use them. You'll also learn how to create table spaces to store your tables.
- Day 10, "Accessing the Data," goes through the many choices you have to access the data stored in DB2 databases.
- On Day 11, "Using System Administration Tools," tools such as the Script Center and Journal are introduced.
- Day 12, "Backing Up and Recovering Data," explains the very important topic of producing a backup and recovery plan for your data.
- On Day 13, "Moving Data," you'll learn how to export data from a DB2 table, and import data and load data into a DB2 table.
- Day 14, "Replicating Data," covers the topic of replicating data from one table to another.

Day 8: Creating Databases

Overview

The easiest way to create a database is to use the Create Database SmartGuide. You can create a very simple database that uses the default values for table spaces or a very sophisticated database that has separate table spaces defined for different data types and spans different containers.

For today's lesson, you create a few databases to see the available choices. You'll create the sample CDLIB database (defined on Day 7, "Designing a CDLIB Database") to store information about a collection of CDs. The example will continue when you create tables and set up the referential constraints.

Note SmartGuides are similar to Experts in Microsoft products; they make a difficult task easier to accomplish.

A Simple Database

First, you create a simple database where you accept most of the available defaults. To create a database with the Create Database SmartGuide, follow these steps:

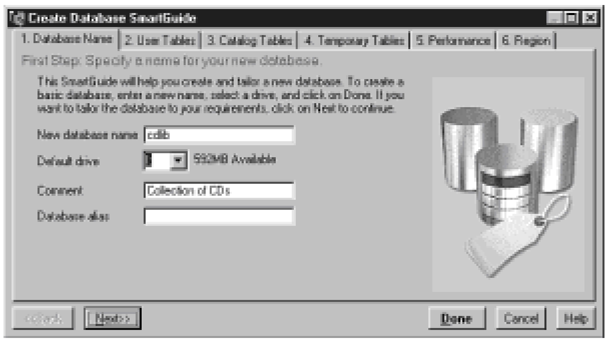
Tip You must have `SYSADM` authority to create a database. Your username must belong to the Administrators group to have `SYSADM` authority.

1. Start the Control Center, if not already started (choose Start | Programs | DB2 for

Windows NT | Administration Tools | Control Center).

2. Expand the object folders until you see the Databases folder. Right-click the folder and select Create from the pop-up menu.

3. You're given the choice to create a new database or open one from a backup. Choose New. The Create Database SmartGuide appears, open at the first page (see Figure 8.1).



Create Database SmartGuide

First Step: Specify a name for your new database.

This SmartGuide will help you create and tailor a new database. To create a basic database, enter a new name, select a drive, and click on Done. If you want to tailor the database to your requirements, click on Next to continue.

New database name: cdlib

Default drive: [illegible] 592MB Available

Comment: Collection of CDs

Database alias:

Figure 8.1: The Create Database SmartGuide.

4. Fill in the first page as follows:

   • **New database name.** Enter a name to identify your database. This name must be different from any other database names cataloged on this drive. The name must contain no more than eight characters—using characters `A–Z`, `0–9`, `@`, `#`, or `$`—and can't begin with a number. For this example, enter **`CDLIB`**.

   Tip If you're planning to use the database in a communications environment or outside North America, avoid using the `@`, `#`, and `$` symbols in the name.

   • **Default drive.** Select the drive that you want to use. Ensure that you'll have enough space on this drive to contain all the data you'll eventually have in your database. All associated database files will be stored in this drive. (Later, you'll see how to create and spread data over table spaces.)

   Tip To estimate the size of a database, you must include the size of the system catalog tables (initially 1,600KB, but they grow as objects and privileges are added to the database), the size of each user table in the database, the index space, log file space, and temporary work space.

   • **Comment.** In this text box, enter an optional description for your database. The comment can contain up to 30 characters. Enter a comment such as **`Collection of CDs`**.

   • **Database alias.** An alias is a nickname given to the database. If you leave this text box blank, it will default to the name of the database. Leave this blank so that it will default to `CDLIB`.

5. Click the Done button to have the database created. (Pages 2–6 are more advanced topics and are covered in the next section.)

After a few minutes, you'll receive a message stating that the database is created. Check the main panel of the Control Center to see a folder added for the CDLIB database.

At this point, the database is empty. To make it useful, you need to create tables, add data, and set up referential constraints.

Creating Tables

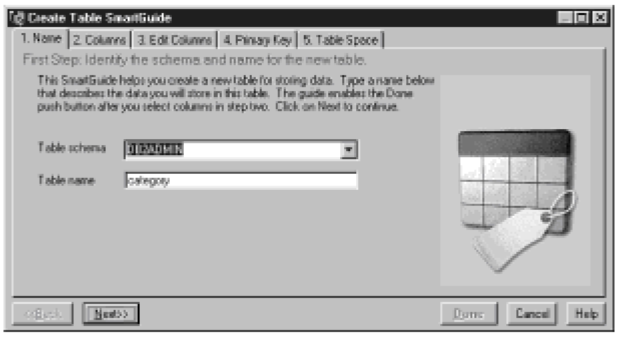
You can create tables with the Control Center in two ways: through the Create Table SmartGuide and the Create Table notebook. If you want to create referential constraints while creating the table, use the Create Table notebook. If you're using the Create Table SmartGuide, you can add foreign keys and constraints by using the Alter option.

For this example, you'll use the Create Table SmartGuide and leave the creation of the constraints for later. To create the `CATEGORY` table, which will contain the music categories of the CDs in this collection, follow these steps:

1. Start the Control Center, if it's not already started.

2. Expand the object folders under the `CDLIB` database folder until you see the Tables folder. Right-click that folder.

3. Select Create | Table Using SmartGuide from the pop-up menu to start the SmartGuide (see Figure 8.2).



First Step: Identify the schema and name for the new table.

This SmartGuide helps you create a new table for storing data. Type a name below that describes the data you will store in this table. The guide enables the Done push button after you select columns in step two. Click on Next to continue.

Table schema: DB2ADMIN

Table name: category

Figure 8.2: The Create Table SmartGuide's first page.

4. On the first page, enter the following information:

   - **Table schema.** Specify the schema for the table you're creating. The initial value is the username that you're using. My username is `SUE`, so tables I create with my username are named `SUE.`*`TABLE_NAME`*. This is fine if you're the only user of the database. You might want to have a more universal schema name, such as a department or function name. For example, the Payroll department might want to use the schema `PAYROLL` instead of someone's username.

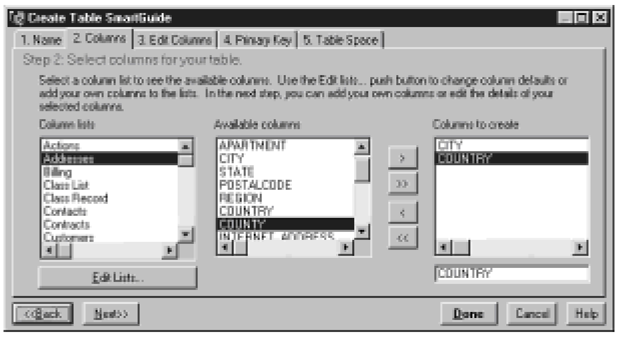
     Select an existing schema or type the name of a new schema. (You must have `IMPLICIT_SCHEMA` authority to be able to create new schemas.) Rather than use my username, I use `DB2ADMIN`.

   - **Table name.** The name of your table must be unique within the schema and can contain up to 18 characters. For this example, name the table `CATEGORY`.

   Click Next to continue.

5. On the second page (see Figure 8.3), you can select from a list of predefined columns to use for your table. For example, if you're keeping a list of addresses, click the Addresses item in the Column lists to see all the predefined columns for an address list. Each column is defined with preset column sizes and characteristics.

The selected column moves to the Columns to create list.



Step 2: Select columns for your table.

Select a column list to see the available columns. Use the Edit lists... push button to change column defaults or add your own columns to the lists. In the next step, you can add your own columns or edit the details of your selected columns.

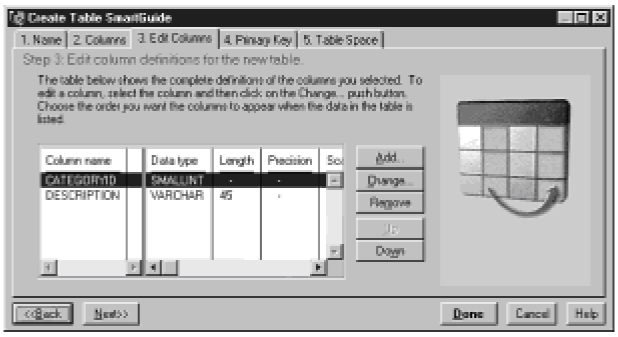
Column lists	Available columns	Columns to create
Actions	APARTMENT	CITY
Addresses	CITY	COUNTRY
Billing	STATE	
Class List	POSTALCODE	
Class Record	REGION	
Contacts	COUNTRY	
Contracts	COUNTY	
Customers	INTERNET ADDRESS	

COUNTRY

Figure 8.3: The Create Table SmartGuide's second page.

For this example, you want to create two columns that aren't in any of these predefined lists. To create these, you need to advance to the next page. Click Next to continue.

6. On the third page (see Figure 8.4), you can edit column definitions for the new table. If you selected predefined columns on the second page, this page gives you an opportunity to customize the column characteristics.



Step 3: Edit column definitions for the new table.

The table below shows the complete definitions of the columns you selected. To edit a column, select the column and then click on the Change... push button. Choose the order you want the columns to appear when the data in the table is listed.

Column name	Data type	Length	Precision	Sc
CATEGORYID	SMALLINT	-	-	
DESCRIPTION	VARCHAR	45	-	

Figure 8.4: The Create Table SmartGuide's third page.

Because you didn't select columns, you must click the Add button to open the Add Column window. Fill in the details as follows:

Tip Plan carefully when setting the characteristics for each column. They can't be altered after you finish creating the table.

- **Column name.** Enter a unique name for the column. The name can contain up to 18 characters. For this example, enter **`CATEGORYID`**.
- **Data type.** Many data types are available in DB2, and you also can make your own unique data types. Examples of some predefined data types include `INTEGER`, `SMALLINT`, `DOUBLE`, `DECIMAL`, `CHARACTER`, and `VARCHAR`. For this example, select `SMALLINT` because you want `CATEGORYID` to be a small integer.
- **Data type characteristics.** The data type characteristics that you need to define depend on the data type that you selected. You selected the `SMALLINT` data type, so you have no additional characteristics to define.
- **Default.** You can enter an optional value that you want used as a default whenever a value isn't entered. For example, if for a field named `COUNTRY` the most common

entry is UNITED STATES, you should set the default to this value so that when you don't enter a value, DB2 will fill in this value. For this example, leave it blank.

- **Nullable.** Select the Nullable check box if you want to allow this field to be blank. Nullable is the default. For this example, you want CATEGORYID to be the primary key, so clear the Nullable check box.

Tip If the column will be a primary key column, it can't be nullable because the primary key must exist in every row in the table. This guarantees that the row can be found.

Click the Add button when finished. You'll remain in the Add Column window, but the column has been added. For this example, you want one more column, so add this information:

- **Column name**. Enter the name **DESCRIPTION** for this column.

- **Data type.** For this example, select VARCHAR because you want the description to contain characters and the length to be variable.

Tip Using the VARCHAR data type is more efficient than using the CHARACTER data type to store character values. Because VARCHAR is variable, it uses only the necessary space to store the value, whereas the CHARACTER data type reserves a block of space for the value, regardless of its actual size.

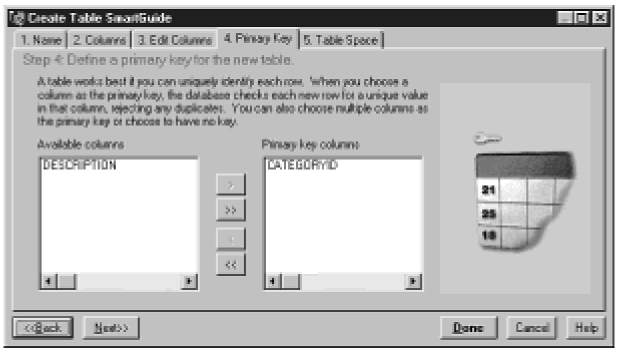
- **Data type characteristics.** An upper value must still be specified forVARCHAR. For a category description, 30 characters should be enough.

- **Default.** Don't define a default for this field.

- **Nullable.** Make the description field required, even though it won't be the primary key.

Click Close to exit the Add Column window and click Next to continue to the next page of the Create Table SmartGuide.

7. On the fourth page, you can define a primary key for the new table from the list of columns that you have in the table. In this example, you want to have CATEGORYID as the primary key. Move CATEGORYID into the Primary key columns list (see Figure 8.5) and choose Next to continue.


Create Table SmartGuide

1. Name | 2. Columns | 3. Edit Columns | 4. Primary Key | 5. Table Space

Step 4: Define a primary key for the new table.

A table works best if you can uniquely identify each row. When you choose a column as the primary key, the database checks each new row for a unique value in that column, rejecting any duplicates. You can also choose multiple columns as the primary key or choose to have no key.

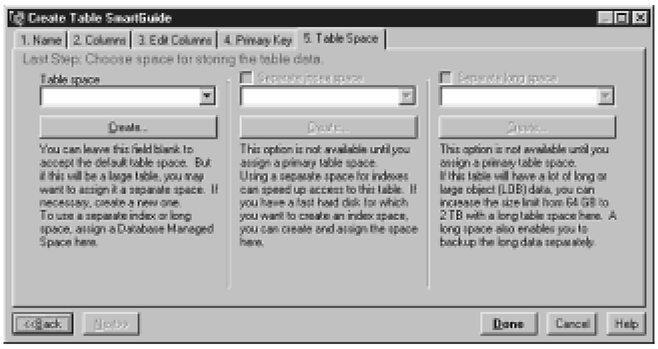
Available columns: DESCRIPTION

Primary key columns: CATEGORYID

Figure 8.5: The Create Table SmartGuide's fourth page.

Tip If the column selected to be the primary key was defined as nullable in the previous page, you'll receive an error message saying that you can't use this column as part of the primary key. If you receive this message, select Back to return to the previous page, select the column that you need to modify, click Change, and make the column not nullable.

8. On the last page (see Figure 8.6), you can create table spaces for storing the table data. This topic is very important for more sophisticated databases, but because you are creating a simple and small database, leave this topic for later. Click Done to have the columns created.



Last Step: Choose space for storing the table data.

You can leave this field blank to accept the default table space. But if this will be a large table, you may want to assign it a separate space. If necessary, create a new one.
To use a separate index or long space, assign a Database Managed Space here.

This option is not available until you assign a primary table space.
Using a separate space for indexes can speed up access to this table. If you have a fast hard disk for which you want to create an index space, you can create and assign the space here.

This option is not available until you assign a primary table space.
If this table will have a lot of long or large object (LOB) data, you can increase the size limit from 64 GB to 2 TB with a long table space here. A long space also enables you to backup the long data separately.

Figure 8.6: The Create Table SmartGuide's last page.

If you want to look at your empty table, use the Control Center. Click the Tables folder in the main Control Center panel. In the right panel, you'll see all the tables now in the database. Most of these are DB2 system tables that you can view but can't change. Scroll down until you see the `CATEGORY` table just created. Right-click `CATEGORY` and choose Sample Contents from the pop-up menu. You'll see the two table columns that you added, with no data. Next, you'll add data to this table. Now close this window by clicking Close.

Adding Data to Tables

You can add data to tables in several ways in DB2. Most likely, you'll have an application from which you can enter data into DB2 databases. As mentioned in earlier lessons, you can create your own applications by using many popular programming languages such as C++, C, FORTRAN, Java, or higher level systems such as Net.Data or Lotus Approach to help you enter data.

If you don't have an application, you can enter data by using the Command Center and simple SQL statements. Follow these steps:

1. Start the Command Center, if it's not already open (choose Start | Programs | DB2 for Windows NT | Command Center).

2. In the main panel, enter the following commands:

Input
```
connect to CDLIB;
insert into CATEGORY (CATEGORYID, DESCRIPTION)
values (1, 'Rock'), (2, 'Classical'), (3, 'Country');
```

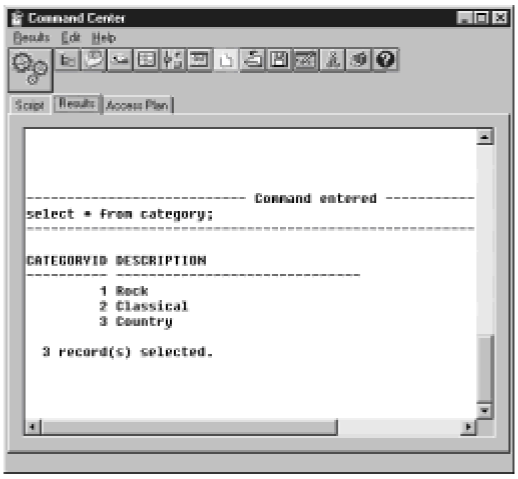
3. Click the gears button to process the command. The Results screen opens, showing that the command completed successfully.

4. Continue to add data in this way. Return to the Script screen, click the down arrow to recall the command, change the values as appropriate to add another row, and click the gears button again. (You need to connect to the database only once.)

5. To see the data that you've just entered, you can issue a `SELECT` statement from the Script screen:

   Input `select * from CATEGORY;`

6. Click the gears button to see the data that you've entered (see Figure 8.7).



```
--------------------------- Command entered -----------
select * from category;
--------------------------------------------------------

CATEGORYID DESCRIPTION
---------- ------------------------------
         1 Rock
         2 Classical
         3 Country

  3 record(s) selected.
```

Figure 8.7: Sampling data in the Command Center.

Setting Up Referential Integrity

A database usually isn't a simple table with a list of items, as you've created so far. It's usually made up of many tables, each containing normalized data.

As discussed on Day 7, "Designing a CDLIB Database," normalizing helps you avoid redundancies and inconsistencies in your data. The main idea behind normalizing is to reduce tables to a set of columns where all the non-key columns depend on the entire primary key of the table.

Each table is linked to ensure that certain business rules are followed. These rules are normally called *referential integrity*. For example, in the `CATEGORY` table, you've created a list of music categories. Now create a table that lists the record company associated with an album and another table that lists the albums available on CD for each artist. You can even take this further and add a table that contains graphic files of the band or album cover, audio samples of the CD itself, and even video clips. Each table that you create will contain a certain type of information and will be connected to one or more of the tables with foreign keys.

To see how this is done, create `ALBUM`, a table that contains the name of the album and its year of release and length, along with relationships to the `CATEGORY`, `RECORDLABEL`, `ALBUMCOVER`, and `GROUP` tables.

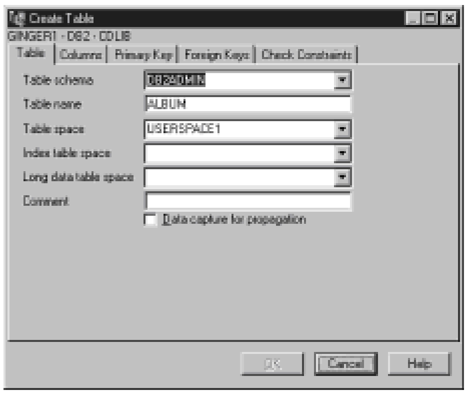
Because you want to add foreign keys to the table as you define it, use the Create Table notebook:

1. Start the Control Center, if it's not already started.

2. Expand the object folders under the `CDLIB` database folder until you see the Tables folder. Right-click that folder.

3. Select Create | Table from the pop-up menu to start the Create Table notebook.

Figure 8.8 shows the first page displayed.



Create Table

GINGER1 - DB2 - CDLIB

Table schema	DB2ADMIN
Table name	ALBUM
Table space	USERSPACE1
Index table space	
Long data table space	
Comment	

☐ Data capture for propagation

Figure 8.8: The Create Table notebook's Table page.

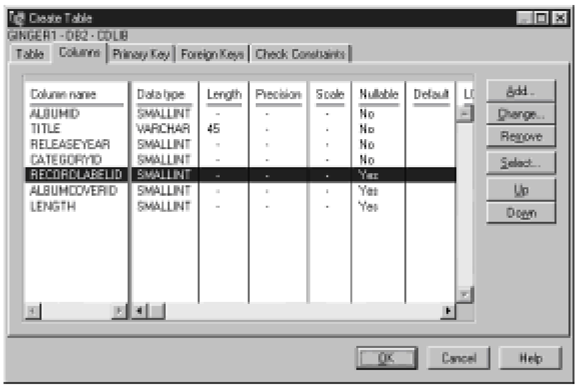
4. On the Table page, enter the following information:

   • **Table schema.** Select DB2ADMIN from the list of existing schemas.

   • **Table name.** For this example, name the table ALBUM.

   The rest of the information on the page refers to creating table spaces. These topics are covered later in Day 9, "Creating Table Spaces"; leave the default values for these fields for now. Click Next to continue.

5. On the Columns page of the notebook (see Figure 8.9), select from a list of predefined columns to use for your table. For this example, you want to create columns for ALBUMID, TITLE, RELEASEYEAR, CATEGORYID, RECORDLABELID, ALBUMCOVERID, and LENGTH.



Create Table

GINGER1 - DB2 - CDLIB

Column name	Data type	Length	Precision	Scale	Nullable	Default
ALBUMID	SMALLINT	-	-	-	No	
TITLE	VARCHAR	45	-	-	No	
RELEASEYEAR	SMALLINT	-	-	-	No	
CATEGORYID	SMALLINT	-	-	-	No	
RECORDLABELID	SMALLINT	-	-	-	Yes	
ALBUMCOVERID	SMALLINT	-	-	-	Yes	
LENGTH	SMALLINT	-	-	-	Yes	

Figure 8.9: The Create Table notebook's Columns page.

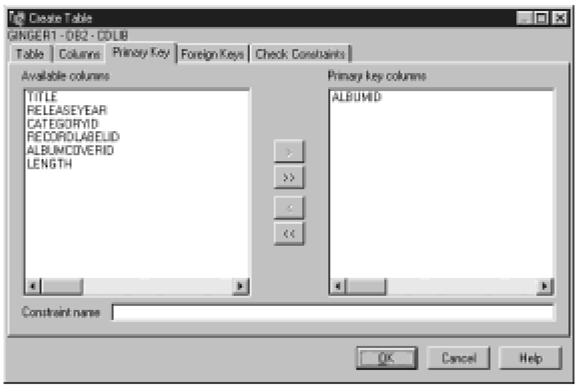
6. Click the Add button to open the Add Column window and fill in the details as follows:

Column Name	Data Type	Nullable

`ALBUMID`	`SMALLINT`	No
`TITLE`	`VARCHAR`, 45	No
`RELEASEYEAR`	`SMALLINT`	No
`CATEGORYID`	`SMALLINT`	No
`RECORDLABELID`	`SMALLINT`	No
`ALBUMCOVERID`	`SMALLINT`	Yes
`LENGTH`	`SMALLINT`	Yes

Click Close to exit the Add Column window and Next to continue to the next page.

7. On the Primary Key page (see Figure 8.10), define a primary key for the new table. From the list of columns in the table, select the columns you want to have as a primary key. In this example, you want to have `ALBUMID` as the primary key, so move `ALBUMID` into the Primary key columns list and choose Next to continue.


Create Table
GINGER1 - DB2 - CDLIB
Tabs: Table | Columns | Primary Key | Foreign Keys | Check Constraints

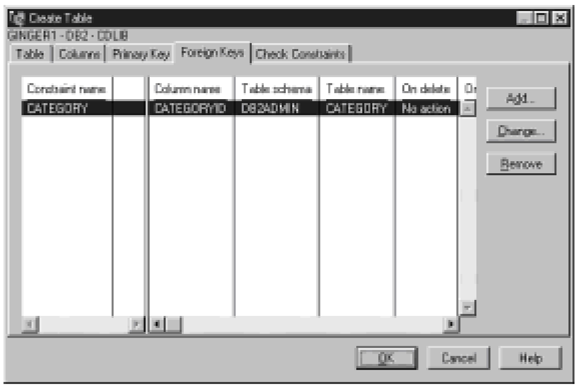
Available columns:
TITLE
RELEASEYEAR
CATEGORYID
RECORDLABELID
ALBUMCOVERID
LENGTH

Primary key columns:
ALBUMID

Constraint name:

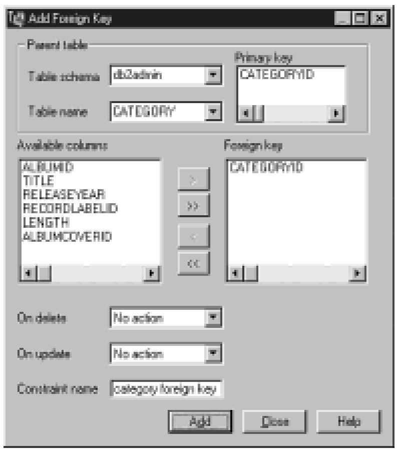
Figure 8.10: The Create Table notebook's Primary Key page.

8. On the Foreign Keys page (see Figure 8.11), you create table spaces for storing the table data. Click the Add button to open the Add Foreign Key window (see Figure 8.12).


Create Table
GINGER1 - DB2 - CDLIB
Tabs: Table | Columns | Primary Key | Foreign Keys | Check Constraints

Constraint name		Column name	Table schema	Table name	On delete	O
CATEGORY		CATEGORYID	DB2ADMIN	CATEGORY	No action	

Figure 8.11.: The Create Table notebook's Foreign Keys page.



Parent table
Table schema: db2admin
Table name: CATEGORY
Primary key: CATEGORYID
Available columns: ALBUMID, TITLE, RELEASEYEAR, RECORDLABELID, LENGTH, ALBUMCOVERID
Foreign key: CATEGORYID
On delete: No action
On update: No action
Constraint name: category foreign key

Figure 8.12: The Add ForeignKey window.

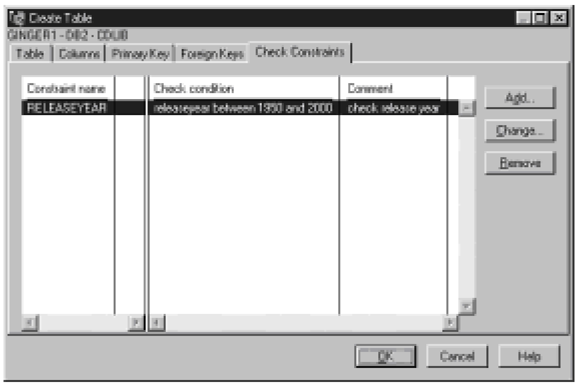
Fill in these fields:

- **Table schema.** Specify the name of the schema of the parent key. In this example, the schema for all the tables is `DB2ADMIN`.
- **Table name.** Specify the name of the parent table. In this example, you want `CATEGORY` as the parent table. Choose `CATEGORY`; its primary key value will appear in the Primary key box.
- **Primary key.** This box is filled in automatically when you select the parent table. The primary key of the `CATEGORY` table is `CATEGORYID`, so this appears in the Primary key box.
- **Foreign key.** In the Available columns box, select columns that you want to define as foreign keys. Each column you select must match a primary key column for the parent table in meaning and data type. In this example, select `CATEGORYID` to be the foreign key.
- **On delete.** From this optional list, specify the action to take place on the dependent table when a row is deleted from the parent table. In this example, the `ALBUM` table is dependent on the parent table `CATEGORY`. Suppose that you have Rock listed in the `CATEGORY` table and decide to delete it. Because you also have a row in the `ALBUM` table that uses Rock as a category and these rows are linked, you now have to decide what to do with all the rows in `ALBUM`. Your choices are No action (the default), Restrict, Cascade, or NULL. If you select No action or Restrict, DB2 will give an error message when you attempt to delete a row in the parent table if there are dependent rows. No rows will be deleted. If you choose Cascade, the row in the parent table will be deleted, along with any dependent rows in dependent tables. If you choose NULL, when the parent row is deleted, any dependent rows will replace all columns with a `NULL` value if it's allowable. For this example, leave this at the default value: No action.
- **On update.** From this optional list, specify the action to take place on the dependent table when a row of the parent table is updated. Your choices are No action and Restrict. If you choose No action or Restrict, an error will occur if you attempt to update a row in the parent table. For this example, use the default value: No action.
- **Constraint name.** In this optional text box, type a name for the constraint that

you're defining. If a name isn't entered, one is generated for you. It's useful to have a meaningful constraint name. If an error occurs, the constraint name is used to describe the error condition. A meaningful name helps correct the error.

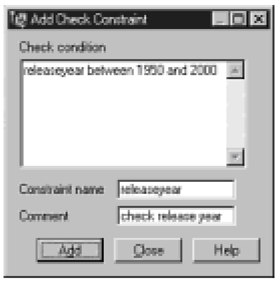
Click Add to add the foreign key.

9. On the Check Constraints page of the Create Table notebook, you can add rules to help ensure that only valid data is inserted into the table (see Figure 8.13). For example, the column `RELEASEYEAR` is intended to contain the year the album or CD was released. This date is printed on the back of each CD. Depending on your collection, the release years can range from 1950 to present day. You can add a check constraint to ensure that only dates in a specific range are entered. Click the Add button to open the Add Check Constraint window (see Figure 8.14).



Constraint name	Check condition	Comment
RELEASEYEAR	releaseyear between 1950 and 2000	check release year

Figure 8.13: The Create Table notebook's Check Constraints page.



Check condition: releaseyear between 1950 and 2000

Constraint name: releaseyear

Comment: check release year

Figure 8.14: The Add Check Constraint window.

You can define a check constraint that references a single column by specifying that column in the Check condition box. For example, you can add a check constraint to validate the year entered for the release date of a CD. A check condition of `releaseyear between 1950 and 2000` would be suitable. (On Day 11, "Using System Administration Tools," you'll add a trigger to ensure that a year beyond the current year isn't entered.)

Note Check constraints aren't checked for inconsistencies, duplicate conditions, or equivalent conditions. Therefore, contradictory or redundant check constraints can be defined, resulting in possible errors at execution time.

Although the Constraint name and Comment text boxes are optional, they're very useful. If you don't enter values for these text boxes, the system will generate a unique but incomprehensible name for the constraint. If you violate a check constraint condition, having a useful name appear in the message makes it easier to correct the error.

You can add multiple check constraints before closing this window. Click Add in the Add Check Constraint dialog to add the constraint, and click Close when you've finished adding constraints.

10. Click OK to create the table.

Altering Tables

After you create a table, the attributes that you can change are limited. You can change the following:

- Change the comment and the data capture for propagation option.
- Add new columns and change, remove, or rearrange the new columns.
- Add foreign keys for the table.
- Add or drop columns from the primary keys for the table or change the constraint name.
- Add, change, or remove new check constraints.

Tip If you're altering a table by adding a foreign key, you need to turn off constraint checking before you alter the table. After you add the foreign key, you need to turn constraint checking back on.

Note If you turn off constraint checking for a parent table, the foreign key constraints of all its dependent and descendent tables are put in check-pending state.

Right-click the table you want to alter and select Alter from the pop-up menu. The Alter Table notebook opens. If you want to change an attribute in the table but it isn't in this list, you'll need to create a new table with the new attributes and import the data from the existing table.

Summary

In today's lesson, you saw how to use the Create Database SmartGuide, which you launch through the Control Center, to create simple or sophisticated DB2 databases. The SmartGuide prompts you for the required information and creates the database you need.

You use the Create Table SmartGuide or Create Table notebook to create tables in an existing database. Again, you're prompted for information about the columns of the table, and the table is created for you.

Referential integrity is added to your database when you're using the Create Database notebook or when you alter the table after it's created.

You can add data to tables in many ways. One way is to use SQL statements to insert the data into the table. Of course, an easier way to add data is by using an ODBC application.

What Comes Next?

On Day 9, "Creating Table Spaces," you learn how to create more sophisticated databases that use table spaces to organize the data into partitions.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. Can you define a primary key column to be nullable? Why or why not?
2. What are some differences between the data types `VARCHAR` and `CHARACTER`?
3. What are the rules for naming a database?
4. What are the differences between the Create Table SmartGuide and the Create Table notebook?

Exercise

Create the video database that you designed on Day 7 and the tables that must be created within the database. When you're using the Create Table SmartGuide, explore the available options in the list of predefined columns.

Day 9: Creating Table Spaces

Overview

For databases that are very large, contain large objects like photos, or require high performance, you need to use advanced methods to store your data. DB2 provides table spaces, containers, and buffer pools for you to define how data is stored on your system.

Databases are organized into partitions called *table spaces*. A *container* is a physical storage device assigned to a table space. A single table space can span many containers. A *buffer pool* is an allocation in memory used to cache tables and to index data pages being read from disk or being modified.

You aren't required to create a table space, container, or buffer pool to add data to tables in a database. You can accept the defaults for each when you create a database and a table. When you created a database earlier with default values for table spaces, you ended up using system-managed space and three standard table spaces:

- `SYSCATSPACE`. Regular table space used to store the system catalog tables.
- `TEMPSPACE1`. Temporary table space used to sort or reorganize tables, create indexes, and join tables.
- `USERSPACE1`. Regular table space used to store the table's data andindexes.

Using table spaces to store your data gives you the flexibility to assign portions of a table—such as data, indexes, and long field data—to different table spaces. This gives you the opportunity to assign different storage devices, depending on the content of each table space. Table spaces can also be backed up and restored as a unit. If you separate your data into table spaces according to how often it's updated, you can choose more often to back up the table space containing the more frequently updated data.

The easiest option for choosing space for storing table data is to accept the default table space in the Create Table SmartGuide. A common starting point is to store all your tables in one table space. However, if this will be a large table, you might want to assign it a separate table space. It's recommended that you create one table space for tables containing user data and their indexes. Create one table space each for the temporary tables and the system catalog tables.

In today's lesson, you learn how to create and tune these storage objects for your environment. You'll use the Create Table Space SmartGuide and the Create Table Space notebook to create table spaces in your database that can be used to store your data. You'll also learn how to separate different types of data to improve performance.

Using the Create Table Space SmartGuide

Use the Create Table Space SmartGuide to create table spaces that contain your data tables. The SmartGuide enables you to define containers as well as the type of management. After you create a table space, you can assign data tables to it. Assign a table to a table space when you create the table.

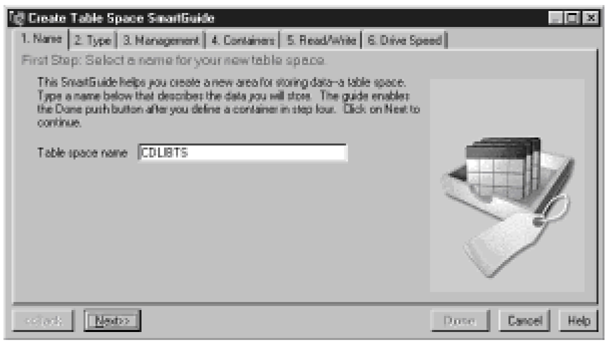
To create table spaces for your database, follow these steps:

1. Start the Control Center.

2. Expand the folders until you see the objects in the CDLIB database.

3. Right-click the Table Spaces folder. Select Create | Table Space Using SmartGuide from the pop-up menu. The Create Table Space SmartGuide appears.

4. On the first page (see Figure 9.1), you must type a name for the table space. The name must be different from any other table spaces on your system and can contain up to 18 characters. For the example, enter **CDLIBTS** as the name of the table space and click Next to continue.



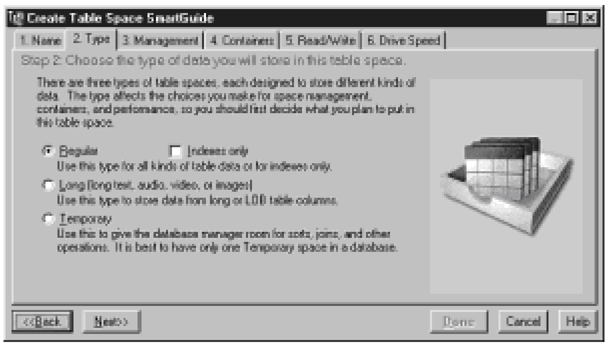
First Step: Select a name for your new table space.

This SmartGuide helps you create a new area for storing data--a table space. Type a name below that describes the data you will store. The guide enables the Done push button after you define a container in step four. Click on Next to continue.

Table space name: CDLIBTS

Figure 9.1: The Create Table Space SmartGuide's first page.

5. On the second page (see Figure 9.2), decide what type of table space you want to create. Before you select a type, you should know what data you plan to put in this table space. The type of table space affects the type of space management, containers, and performance options available.



Step 2: Choose the type of data you will store in this table space.

There are three types of table spaces, each designed to store different kinds of data. The type affects the choices you make for space management, containers, and performance, so you should first decide what you plan to put in this table space.

[✓ Regular] Indexes only
Use this type for all kinds of table data or for indexes only.

Long (long text, audio, video, or images)
Use this type to store data from long or LOB table columns.

Temporary
Use this to give the database manager room for sorts, joins, and other operations. It is best to have only one Temporary space in a database.

Figure 9.2: The Create Table Space SmartGuide's second page.

You can choose from among these types:

- **Regular.** Use this type for all kinds of table data except temporary tables. Select the Indexes Only check box if the table space will contain only indexes. When this option is selected, DB2 uses the high-performance management option for this table space (more about this in a little bit).

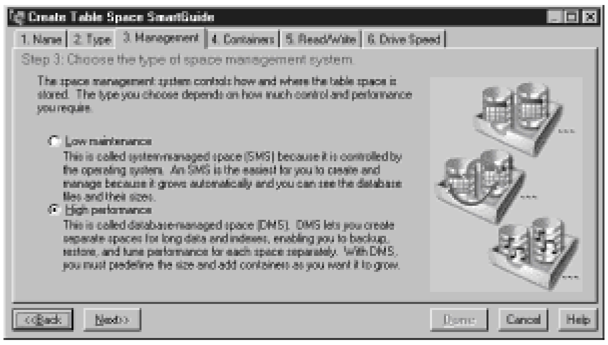
- **Long.** Use this type to create a table space that stores large object data such as images, audio, video, or long text fields. When this type is selected, DB2 uses the high-performance management option for this table space.

- **Temporary.** Use this type to create a table space to store temporary tables. The temporary space is used during sorts, joins, and other operations. You need to have only a single temporary table space in a database because DB2 uses only one at a time.

For the example, choose Regular and click Next to continue.

6. On the third page (see Figure 9.3), choose whether you want the table space to be Low Maintenance or High Performance:

- **Low Maintenance.** This type of space management is also known as *system-managed space* (*SMS*). An SMS table space is the easiest type for you to create and manage because it grows automatically and you can see the database files and their sizes. SMS table space is well suited for temporary data and database catalogs because the space is consumed only when it's needed. (If you selected Indexes Only or Long on the second page, you can't select Low Maintenance.)



Step 3: Choose the type of space management system.

The space management system controls how and where the table space is stored. The type you choose depends on how much control and performance you require.

Low maintenance
This is called system-managed space (SMS) because it is controlled by the operating system. An SMS is the easiest for you to create and manage because it grows automatically and you can see the database files and their sizes.

[✓ High performance]
This is called database-managed space (DMS). DMS lets you create separate spaces for long data and indexes, enabling you to backup, restore, and tune performance for each space separately. With DMS, you must predefine the size and add containers as you want it to grow.

Figure 9.3: The Create Table Space SmartGuide's third page.

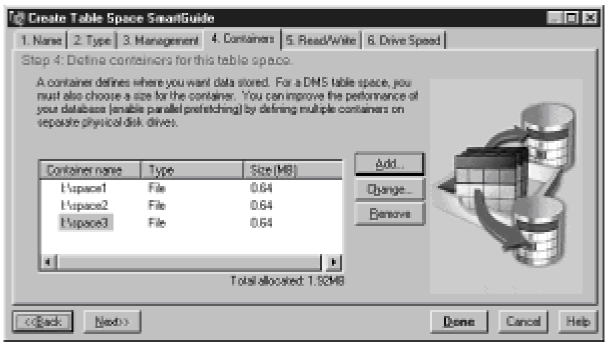
- **High Performance.** This type of space management provides you with many tuning options, including predefining the size of each table space and adding containers as your data grows. This management type is also known as*database-managed space* (*DMS*). One advantage of using DMS over SMS is that you can separate indexes and long data from the rest of your regular data. You can back up, restore, and tune the performance for each table space separately.

Tip You might want to use a separate table space for each very large table and to group all small tables in a single table space. This separation also enables you to select an appropriate extent size based on the table space used. This is available only if using DMS table spaces.

Tip If you plan to store many small tables in a table space, consider using SMS for that table space. The DMS advantages with I/O and space management efficiency aren't as important with small tables. The SMS advantage of allocating space one page at a time, and only when needed, is more attractive with smaller tables. If one of your tables is larger or you need faster access to the data in the tables, you should consider a DMS table space with a small extent size.

Because the goal is to have three separate table spaces—one each for regular data, indexes, and long data—you must choose High Performance. (Data can be split across several table spaces only by using DMS.) Click Next to continue.

7. On the fourth page, specify where the table space will be stored (see Figure 9.4). A container defines where you want data stored. If you've defined a DMS table space, you must also define a size for the container. By defining multiple containers on separate physical drives, you can improve the performance of your database.



Step 4: Define containers for this table space.

A container defines where you want data stored. For a DMS table space, you must also choose a size for the container. You can improve the performance of your database (enable parallel prefetching) by defining multiple containers on separate physical disk drives.

Container name	Type	Size (MB)
I:\space1	File	0.64
I:\space2	File	0.64
I:\space3	File	0.64

Total allocated: 1.92MB

Figure 9.4: The Create Table Space SmartGuide's fourth page.

If you don't see any containers in the list, click the Add button to open the Add Container window, which varies, depending on the space management selected. For SMS, you need to define only the filename, drive, and directory for the container. DB2 takes care of the storage for you, increasing storage size when necessary. For DMS, you must define the file size and specify whether the container will be stored in a file or on a raw device:

- **File Size.** For DMS, you must define the size of the container. You can specify the size in units of megabytes (MB) or 4KB pages. (There are 256 4KB pages in 1MB.)

- **File.** Choose between a file or raw device for the container. Files are easier to use than raw devices because DB2 works through the operating system's file system to create and access the data in files. However, files don't perform as well.

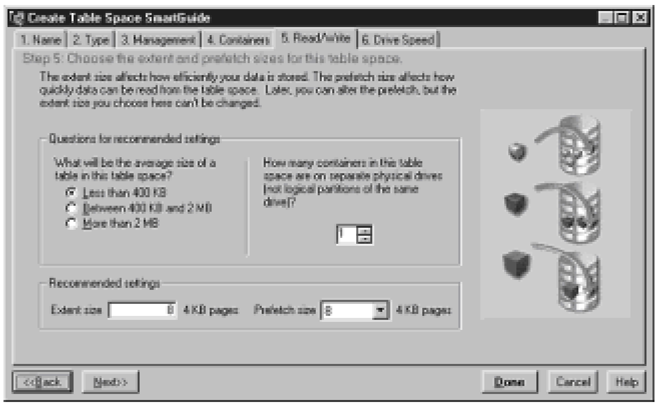
Note The database manager manages the data files for containers by creating the first table data file (`SQL00001.DAT`) in the first container specified for the table space. This file cangrow to the extent size that you specify. After it reaches this size, the database manager writes the data to `SQL00001.DAT` in the next container. This continues until all the containers contain `SQL00001.DAT` files, at which time the database manager returns to the first container. This process is known as striping.

- **Raw.** A *raw device* is a large contiguous chunk of disk space. You can use an unformatted partition or an unpartitioned disk as a raw device. It performs better than a file because DB2 accesses the disk directly.
- **File Name, Drive, and Directory.** Choose a name for the container, as well as a drive and directory where the container is to be located. Use a directory that won't be used for anything else. You should specify a filename that doesn't exist, because it will be automatically created.

Because you chose High Performance in step 6, the table space is using DMS. Add several containers on a drive where you want to store your data. For example, add containers such as in `i:\tspace\space1`, `i:\tspace\space2`, and `i:\tspace\space3`.

Now you can click Done to create the table space and container. For this example, however, click Next to proceed to the Read/Write and Drive Speed pages.

8. On the fifth page (see Figure 9.5), the SmartGuide can assist you with the extent and prefetch size that you define.


Create Table Space SmartGuide

Step 5: Choose the extent and prefetch sizes for this table space.

The extent size affects how efficiently your data is stored. The prefetch size affects how quickly data can be read from the table space. Later, you can alter the prefetch, but the extent size you choose here can't be changed.

Questions for recommended settings

What will be the average size of a table in this table space?
[✓ Less than 400 KB] Between 400 KB and 2 MB More than 2 MB

How many containers in this table space are on separate physical drives (not logical partitions of the same drive)? 1

Recommended settings

Extent size: 8 4 KB pages   Prefetch size: 8 4 KB pages

Figure 9.5: The Create Table Space SmartGuide's fifth page.

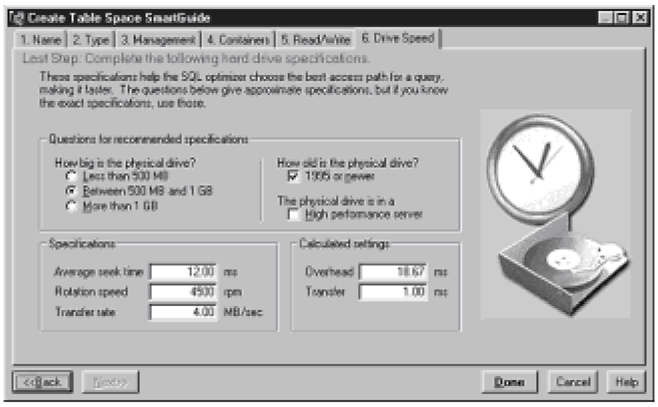
Data for any table is stored in a round-robin fashion on all containers in a table space. The data is balanced across the containers for a given table space. The number of pages that DB2 writes to one container before a different one is used is called the *extent size*. The extent size affects how efficiently your data is stored.

Prefetch size indicates how much data is retrieved from disk in anticipation of its use. To determine the prefetch size, you must know how many containers in the table space are on separate physical drives (as opposed to separate logical partitions on the same drive). Type the number of containers that are on separate drives. This number is multiplied by the extent size to give the recommended prefetch size. The prefetch size recommendation is based on whether you have containers on separate drives; DB2 can prefetch from all those drives at the same time. The prefetch size affects how quickly data can be read from the table space.

To determine the extent size, you must estimate the average size of a table in this table space. You can calculate the size of a table by multiplying the number of bytes in a single row by the number of rows in a table. A rough estimate is all you need. The extent size recommendations are based on smaller tables being more efficiently stored in smaller extent blocks.

You can alter the prefetch size later, but you can't alter the extent size. After you determine the extent and prefetch size for this table space, click Next to continue.

9. On the last page, you provide information to help the SQL optimizer choose the best path for a query (see Figure 9.6). Enter actual specifications for your hard drive or answer the questions to come up with approximate specifications.



Last Step: Complete the following hard drive specifications.

These specifications help the SQL optimizer choose the best access path for a query, making it faster. The questions below give approximate specifications, but if you know the exact specifications, use those.

Questions for recommended specifications

How big is the physical drive?
- Less than 500 MB
- [✓ Between 500 MB and 1 GB]
- More than 1 GB

How old is the physical drive?
- ✓ 1995 or newer

The physical drive is in a
- High performance server

Specifications

Average seek time	12.00	ms
Rotation speed	4500	rpm
Transfer rate	4.00	MB/sec

Calculated settings

Overhead	18.67	ms
Transfer	1.00	ms

Figure 9.6: The Create Table Space SmartGuide's last page.

10. Click Done when you've completed all the pages. You might want to review your choices before you do so, however, to make sure that everything is as you want.

To see the table space that you created, click the Table Space icon. You'll see the `CDLIBTS` table space listed with the default table spaces already created. To view or change any values for your table space, right-click the table space and select Alter from the pop-up menu.

Creating a Buffer Pool

A *buffer pool* is an area of main memory into which database pages are temporarily read and changed. The purpose of the buffer pool is to improve database system performance. Data can be accessed much faster from memory than from disk; therefore, the fewer times DB2 needs to read from or write to a disk, the better the performance.

You can create more than one buffer pool, although for most situations, one will suffice. Having more than one buffer pool allows you to tune the performance for different applications.

To create a buffer pool, follow these steps:

1. Right-click the Buffer Pools folder and select Create from the pop-up menu. The Create Buffer Pool dialog box opens.

2. Enter a name for the buffer pool. The name can contain up to 18 characters. For this example, use the name `BUFFER1`.

3. In the Size In 4 KB Pages text box, type the size of the buffer pool.

4. Ignore the Use Extended Storage check box because this option is for UNIX systems only.

5. Click OK to create the buffer pool.

When you create new table spaces, you can select to have this new buffer pool used instead of the default buffer pool. You can also alter existing table spaces and have them use this new buffer pool.

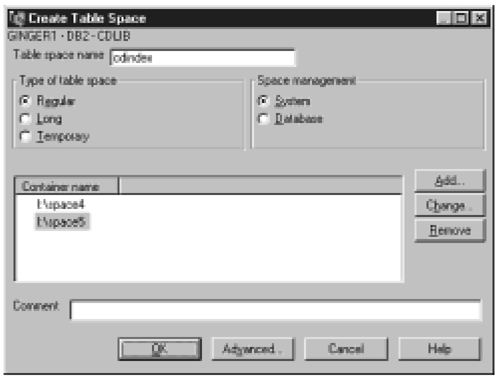
Creating More Table Spaces

Now that you've learned many of the concepts behind creating a table space, use the Create Table Space notebook to create two more table spaces—one to contain the indexes and one to contain the long data. Follow these steps:

1. Start the Control Center.

2. Expand the folders until you see the objects in the CDLIB database.

3. Right-click the Table Space folder. Select Create | Table Space from the pop-up menu. The Create Table Space notebook opens (see Figure 9.7).



Create Table Space

GINGER1 - DB2 - CDLIB

Table space name: cdindex

Type of table space: [✓ Regular] Long Temporary

Space management: [✓ System] Database

Container name:
i:\space4
i:\space5

Comment:

Figure 9.7: The Create Table Space notebook.

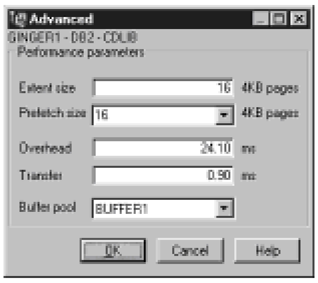
4. Type a name for the table space. The name must be different from any other table spaces on your system and can contain up to 18 characters. For this example, use `CDINDEX` as the name of the table space.

5. From the Type Of Table Space options, indicate the type of table space you want to create. When you used the SmartGuide, you learned that indexes are stored in regular table spaces. Select the Regular radio button.

6. From the Space Management options, indicate whether to use system or database management. When you used the SmartGuide, you learned that indexes are stored in the high-performance or DMS-type space management. Select the Database radio button.

7. Click the Add button to open the Add Container window to specify where the table space will be stored. For this example, set the file size to 2MB, click the File radio button, and type names for the containers (such as `i:\space4` and `i:\space5`). Click OK to add the containers, and close the Add Containers window.

8. Click the Advanced button to specify performance parameters such as extent size, prefetch size, overhead, transfer, and buffer pool (see Figure 9.8).



GINGER1 - DB2 - CDLIB
Performance parameters

Extent size	16	4KB pages
Prefetch size	16	4KB pages
Overhead	24.10	ms
Transfer	0.90	ms
Buffer pool	BUFFER1	

Figure 9.8: The Advanced window of the Create Table Space notebook.

The extent size is the number of pages that DB2 writes to one container before a different one is used. For the example, use 16, meaning that 16 4KB pages are written to one container before the next one is used.

To determine the prefetch size, you must multiply the number of containers (for example, 1) that are on separate drives by the extent size (for example, 16) to give the recommended prefetch size (for example, 16).

For overhead and transfer, use the default values. For buffer pool, select the buffer pool you created earlier (`BUFFER1`). Click OK to return to the Create Table Space notebook.

9. Click OK after you enter all the information to create the table space you have defined. You might want to review your choices before you do so, to make sure that everything is as you want it.

Repeat these steps to create a table space to store long data. Use these values:

Option	Value
Table Space Name	`CDLONG`
Type of Table Space	Long
Space Management	Database
File Size	10MB File
Container	`i:\long`

In your list of table spaces, you should see `CDLIBTS`, `CDINDEX`, and `CDLONG`. Next you'll create a table that uses these table spaces.

Creating Tables in Table Spaces

Earlier, you created the `CATEGORY` table by using the Create Table SmartGuide. Now look at how to create tables by using the Create Table notebook. You'll also assign the tables to the table spaces you created. The Create Table notebook also lets you create foreign keys and constraints while creating the table.

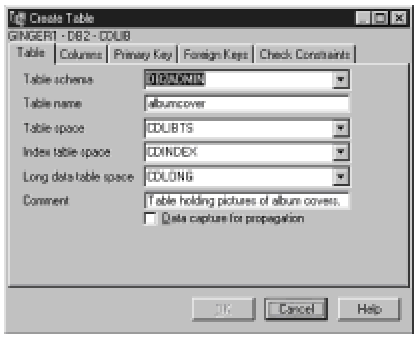
To use the notebook to create a table, follow these steps:

1. Start the Control Center, if it's not already started.

2. Expand the object folders until you see the Tables folder. Right-click that folder.

3. Select Create | Table from the pop-up menu to start the notebook.

4. Use the Table page of the notebook to define the table properties (see Figure 9.9). In the example, you'll create the table `ALBUMCOVER`, which will be used to store graphics of each album cover.



GINGER1 - DB2 - CDLIB

Table schema	DB2ADMIN
Table name	albumcover
Table space	CDLIBTS
Index table space	CDINDEX
Long data table space	CDLONG
Comment	Table holding pictures of album covers.

☐ Data capture for propagation

Figure 9.9: The Create Table notebook.

On the Table page, enter the following information:

- **Table Schema.** Specify the schema for the table you're creating. Click the down arrow to display a list of the existing schemas. For this example, use the `DB2ADMIN` schema.

- **Table Name.** The table name must be unique within the schema and can contain up to 18 characters. For this example, name the table `ALBUMCOVER`.

- **Table Space.** Use the name of the table space you created for regular data: `CDLIBTS`.

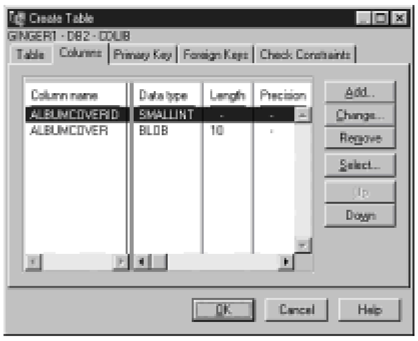
- **Index Table Space.** Use the name of the table space you created for index data: `CDINDEX`.

- **Long Data Table Space.** Use the name of the table space you created for long data: `CDLONG`.

- **Comment.** Optionally enter a comment to describe the contents of the table.

- **Data Capture for Propagation.** Select this check box if the table will be replicated. For now, leave it deselected.

5. Click the Columns tab to create columns in the table (see Figure 9.10). For the table, you want to create two columns: one for the photo and one for the ID.


Create Table

GINGER1 - DB2 - CDLIB

Column name	Data type	Length	Precision
ALBUMCOVERID	SMALLINT	-	-
ALBUMCOVER	BLOB	10	-

Figure 9.10: The Create Table notebook's Columns page.

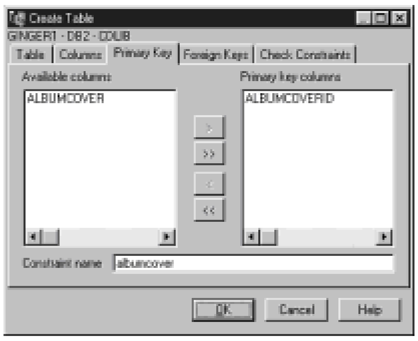
To create a column for the ID, click Select to see a list of predefined columns. Click ID Numbers and select the column `ID_NUMBER` from the list. Click the right arrow to move the column into the Column List. Click OK.

For the photo column, click Select again. Click Media and select the column `PHOTO` from the list. Click the right arrow to move the column into the Column List. Click OK.

You'll see that both columns have been added. Select the `PHOTO` column and click Change. You'll see all the predefined characteristics of this column. In the Name text box, change the name to `ALBUMCOVER`. In the Length text box, the size is set to `10MB`, but change it to `1MB`. Click OK. Select the `ID_NUMBER` column and click Change. In the Name text box, change the name to `ALBUMCOVERID`. Click OK.

Move the `ALBUMCOVERID` column ahead of the `ALBUMCOVER` column by selecting `ALBUMCOVERID` and clicking Up. The order of the columns doesn't matter, but it's standard practice to have the ID of a column in the first position.

6. Click the Primary Key tab to define a primary key for the new table (see Figure 9.11). From the list of columns that you have in the table, select the column you want to have as a primary key. For this example, select `ALBUMCOVERID`.


Create Table

GINGER1 - DB2 - CDLIB

Available columns: ALBUMCOVER

Primary key columns: ALBUMCOVERID

Constraint name: albumcover

Figure 9.11: The Create Table notebook's Primary Key page.

7. Click the Foreign Keys tab to define a foreign key for the table (see Figure 9.12). Click the Add button to open the Add Foreign Key window. The table `ALBUMCOVER` is related to the `ALBUM` table, but `ALBUMCOVER` is the parent table and `ALBUM` is the dependent table. Because you must define the foreign keys on the dependent table, leave this window blank. Later, you'll have to alter the `ALBUM` table to specify this foreign key.

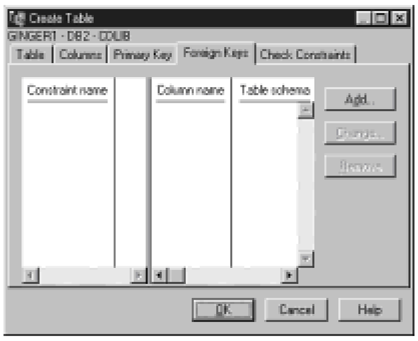


Figure 9.12: The Create Table notebook's Foreign Keys page.

8. Click the Check Constraints tab to define restrictions on data added to the table (see Figure 9.13). Check constraints are enforced when rows in the table are inserted or updated. Because you don't want to create a check constraint on the `ALBUMCOVER` table, proceed to the next step.

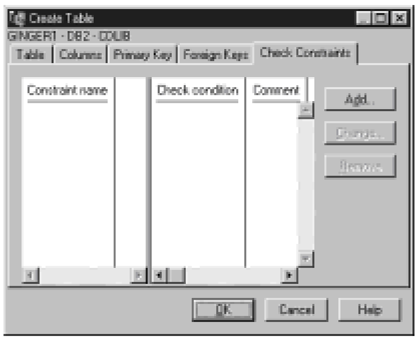


Figure 9.13: The Create Table notebook's Check Constraints page.

9. Click OK to create the table and close the Create Table notebook.

Summary

In today's lesson, you saw how to use table spaces, containers, and buffer pools to define how data is stored on your system.

You can create table spaces by using the Create Table Space SmartGuide (launched through the Create Database SmartGuide or the Control Center) or the Create Table Space notebook. When creating a table space, you have the option of using low-maintenance or high-performance space management.

A *container* defines where the data will be stored. Table spaces can span more than one container. The database manager attempts to balance the load of the data across the containers.

What Comes Next?

On Day 10, "Accessing the Data," you see several ways to access data in DB2 databases, including ODBC applications such as Lotus Approach, the Control Center, the Command Center, Net.Data applications, and JDBC applications.

Q&A

Q What three table spaces must a database contain?

A A database must contain a table space for the system catalog tables. This is known as the *catalog table space* and is named `SYSCATSPACE`. This space is created when the database is created, and it can't be dropped.

The database must contain one or more user table spaces that contain all user-defined tables. By default, one table space is created to contain user tables and is called `USERSPACE1`. When you create a table, you specify the table space that you want to use for the data.

The database also contains one or more temporary table spaces, which contain temporary tables. Temporary tables are necessary for performing tasks such as sorting. By default, one temporary table space is created that's called `TEMPSPACE1`. Although you can create many temporary table spaces, as few as possible is preferable.

Q When should I use DMS table spaces instead of SMS table spaces?

A SMS table spaces are lower maintenance than DMS table spaces because the size automatically grows as space is needed. SMS table spaces are best suited for temporary data and database catalogs. SMS is also best to use if your database contains many small tables.

DMS table spaces, on the other hand, provide you with more options for managing your data. With DMS table spaces, you can separate each type of data into a different table space. By separating data, you can back up, restore, and tune the performance for each table space separately. If you're storing long data, such as audio or photos, you must use DMS table spaces. This is also true if you want to store your indexes on a separate table space. DMS table spaces are more difficult to maintain because you must predefine the size of each table space and add containers when necessary.

Q How do containers relate to a file?

A When you're creating SMS table spaces, you must specify the number of containers that you want to use for your table space. It's very important to identify all the containers that you want to use, because you can't add or delete containers after an SMS table space is created. Each container used for an SMS table space identifies an absolute or relative directory name. Each directory can be located on a different file system or physical disk.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. What's the difference between the Create Table Space SmartGuide and the Create Table Space notebook?

2. How many buffer pools should you define for each database?

3. At what point in the process do you specify the tables to be stored in a table space?

4. How can you view the defined table spaces?

Exercise

One of the most difficult concepts covered in this lesson is containers. As you go through the lesson and create table spaces and containers, check your file system to see the relationship between your file system and containers.

Day 10: Accessing the Data

Overview

You can view the data in DB2 databases in several ways:

- By using the Control Center
- By using the Command Center
- By using ODBC applications, such as Lotus Approach
- By using Internet applications, such as Java or Net.Data
- By using applications written in C, C++, or other programming languages

Each access method uses SQL to access the data.

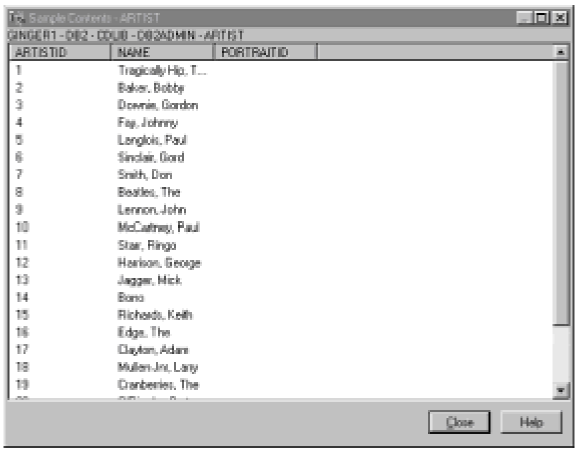
Today's lesson shows how to use each method to view the data in the CDLIB database. Before you can actually access data in any database, you must first have the authority to access the database and the privileges to perform the chosen actions. At a minimum, you must have `CONNECT` authority on the database. Day 4, "Getting Started," provides details on how to set authorities.

Accessing Data Through the Control Center

The Control Center enables you to view the first 200 rows of the selected table. The names of the table's columns are displayed at the top of each column in the Sample Contents window. Each column's contents are displayed below the corresponding column name. You might need to scroll right to see all the columns and scroll down to see all the rows.

To view the contents of the CDLIB database, follow these steps:

1. Open the Control Center, if it's not already started.
2. Expand the objects within the CDLIB database until you see the Tables folder.
3. Click the folder in the main Control Center panel. In the right panel, you'll see all the tables now in the database.
4. Scroll down until you see the `ARTIST` table. Right-click it and choose Sample Contents from the pop-up menu. You'll see all the columns and 200 rows of the `ARTIST` table (see Figure 10.1).



ARTISTID	NAME	PORTRAITID
1	Tragically Hip, T...	
2	Baker, Bobby	
3	Downie, Gordon	
4	Fay, Johnny	
5	Langlois, Paul	
6	Sinclair, Gord	
7	Smith, Don	
8	Beatles, The	
9	Lennon, John	
10	McCartney, Paul	
11	Starr, Ringo	
12	Harrison, George	
13	Jagger, Mick	
14	Bono	
15	Richards, Keith	
16	Edge, The	
17	Clayton, Adam	
18	Mullen Jnr, Larry	
19	Cranberries, The	

Figure 10.1: The Sample Contents window in the Control Center.

You can only view the data through the Control Center. If you want to modify the data or create more complex queries, you must use one of the other methods described in this lesson.

Accessing Data Throughthe Command Center

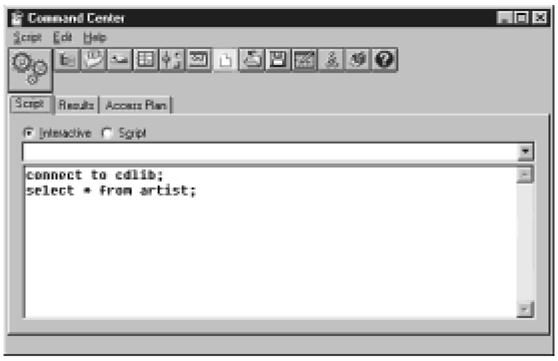
The Command Center is installed automatically with all DB2 Universal Database products on the OS/2, Windows 95, and Windows NT operating environments. You can use the Command Center to access and manipulate databases from an interactive window. From it, you can

- Issue SQL statements, DB2 commands, and operating system commands.
- See the execution result of one or many SQL statements and DB2 commands in a result window. You can scroll through the results and save the output to a file.
- Save a sequence of SQL statements and DB2 commands to a script file. You can then schedule the script to run as a job. When a saved script is modified, all jobs dependent on the saved script inherit the new modified behavior.
- Recall and run a script.
- See the execution plan and statistics associated with a SQL statement before execution. You do this by invoking Visual Explain in the interactive window.
- Get quick access to database administrative tools such as the Control Center and the Journal from the main toolbar.
- Display all the command scripts known to the system through the Script Center, with summary information listed for each.
- Access local and remote databases.
- Request command syntax help for DB2 commands and message help for DB2 messages.

The Command Center enables you to enter simple SQL statements to view and manipulate the data in your databases. Follow these steps to view all the rows and columns of the `EMPLOYEE` table:

1. Start the Command Center by choosing Start | Programs | DB2 for Windows NT | Command Center. If you already have the Control Center open, simply click the Command Center icon from the toolbar. (The hover help displays when your mouse pointer is on the toolbar icons.)

2. Type your commands in the large input area (see Figure 10.2). If you want to run multiple commands, you must end each command with a semicolon and then press Enter to start the next command on a new line.



```
connect to cdlib;
select * from artist;
```

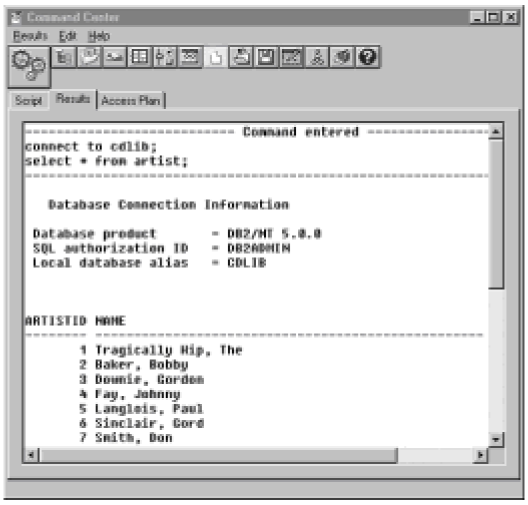
Figure 10.2: The input page in the Command Center.

To connect to the CDLIB database and issue a `SELECT` statement, use the following commands:

Input
```
connect to cdlib;
select * from artist;
```

Note The semicolon is optional at the end of a single quick command and at the end of the final command in a series.

3. Click the gears button to process the commands. The Command Center displays the Results window, in which you'll see all the rows and columns in the `ARTIST` table (see Figure 10.3).



```
--------------------------- Command entered ----------------
connect to cdlib;
select * from artist;
-----------------------------------------------------------

   Database Connection Information

 Database product       = DB2/NT 5.0.0
 SQL authorization ID   = DB2ADMIN
 Local database alias   = CDLIB


ARTISTID NAME
-------- --------------------------------------------------
       1 Tragically Hip, The
       2 Baker, Bobby
       3 Downie, Gordon
       4 Fay, Johnny
       5 Langlois, Paul
       6 Sinclair, Gord
       7 Smith, Don
```

Figure 10.3: The Command Center's Results page.

To recall commands that you've typed in interactive mode, return to the Command Center's Script page and click the arrow beside the narrow input window.

To save commands entered in interactive mode as scripts, choose Script | Save As from the menu.

Tip To schedule commonly used SQL statements or DB2 commands, use the Script Center (click the Script icon on the main toolbar).

Using the Command-Line Processor

The command-line processor is installed automatically with all DB2 Universal Database products. Use the DB2 command-line processor to access and manipulate databases from the system command prompt or command files. From it, you can

- Issue SQL statements, DB2 commands, and Windows NT commands.
- Maintain a history file of all requests.
- Redirect the output of requests.
- Access local and remote databases.
- Request command syntax help for DB2 utilities and message help for DB2 messages.

Entering Commands

To invoke the command-line processor to enter DB2 commands or SQL statements and view their output, choose Start | Programs | DB2 for Windows NT | Command Line Processor.

You can also invoke the command-line processor by entering **`db2cmd`** at a command prompt and then entering **`db2`**.

The prompt for the command-line processor looks like this:

```
db2 =>
```

This prompt indicates that you don't type DB2 commands with a `db2` prefix; instead, you just type the DB2 command—for example,

```
list node directory
```

To run Windows NT commands in the command-line processor, precede the command with an exclamation mark (`!`):

```
!dir db2*.log
```

If you need to enter a long command that doesn't fit on one line, use the \ line-continuation character :

```
db2 => select group.name, artist.name from \
db2 (cont.) => group, artist, artistingroup \
db2 (cont.) => where group.groupid = artistingroup.groupid \
db2 (cont.) => and artist.artistid = artistingroup.artistid \
db2(cont.) => order by group.name
```

To exit the command-line processor, type **`terminate`**.

Entering Commands in a Command Window

To use the command-line processor like a Windows NT command window, where you can enter DB2 commands or SQL statements and view their output, use one of the following methods:

- Choose Start | Programs | DB2 for Windows NT | Command Window.

- Enter **`db2cmd`** at a command prompt.

After you invoke the DB2 command environment, you can enter DB2 commands at the command prompt. You must include the `db2` prefix—for example,

```
db2 list node directory
```

Enter Windows NT commands as usual, without a prefix:

```
dir db2*.log
```

To exit the command-line processor, type **`exit`**.

Accessing Data Through Lotus Approach

Lotus Approach, provided with the DB2 package, is a powerful database interface that enables you to use the mouse to access data and search your databases. Use the instructions provided with Lotus Approach to install it. You can install it on the same system where your databases are located or on a remote client.

To access DB2 databases through Lotus Approach, you must first register the DB2 database that you want to access as an ODBC data source. Follow these steps:

1. Choose Start | Programs | DB2 for Windows NT | Client Configuration Assistant. The Client Configuration Assistant opens.

2. Select the database you want to access in the ODBC application (the CDLIB database for this example) and click Properties. The Database Properties – CDLIB window opens.

3. Select the Register This Database For ODBC check box and ensure that the As A System Data Source radio button is selected. This makes the database source available to all users on the system.

4. In the Connection window, click Properties to set the protocol information for this connection. Day 6, "Installing DB2 Clients," covers the settings you can use for the properties.

5. Click Done when the information has been added.

6. Click Settings. You're asked whether you want to connect to database; click Yes. Enter a valid username and password and click OK.

7. Click Advanced to open the CLI/ODBC – Advanced Settings notebook. This notebook is divided into tabbed pages that enable you to select a group of configuration parameters to set or modify. Each tabbed page has a list of parameters with the corresponding `db2cli.ini` file keyword parameter beside it. When you select a

parameter, you can change its setting in the Value section of the window. Hint text is available for each parameter.

> **Tip** This book doesn't cover all configuration parameters that you can set, but you should consider setting two parameters: `SCHEMALIST` and `SYSSCHEMA`. As you learned on Day 2, "Exploring the Capabilities of DB2 Universal Server," schemas are used to group database objects. If you have many schemas on your database server, performance can degrade. The `SCHEMALIST` and`SYSSCHEMA` configuration parameters improve performance in this case by limiting the number of schemas searched during a query.

Click the Default button to return the selected parameter's value to the DB2 CLI/ODBC default. Use the Reset button to return the selected parameter's value to the last saved value.

Click OK to close the notebook.

8. Click Optimize. Select Lotus Approach and click OK. This will update the ODBC settings so that they're optimized for using Lotus Approach.

9. Exit from the Client Configuration Assistant and make sure that DB2 has started. (Issue a `db2start` command at the command-line processor.)

Now that DB2 is registered as an ODBC data source and DB2 is started, you're ready to view the tables in the CDLIB database with Lotus Approach. For example, to view the `ALBUM` table, follow these steps:

1. Ensure that DB2 is started.

2. Choose Start | Programs | Lotus SmartSuite | Lotus Approach 97 to start Lotus Approach.

3. Click Cancel to close the Welcome To Lotus Approach window.

4. Click File | Open. In the Open dialog box, select `IBM DB2 (*)` from the Files Of Type drop-down list. If you aren't connected to a DB2 database, the Connect To DB2 dialog box opens. Select a database alias from the drop-down list, enter a valid username and password, and click OK.

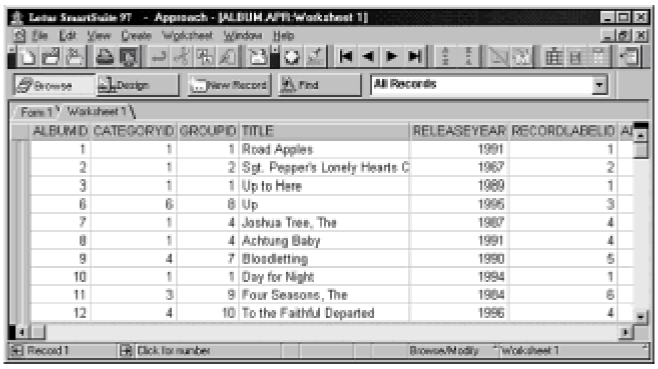
5. The Open dialog box is updated with a list of schema names (represented as folders). Double-click the `db2admin@DB2` folder, double-click the `DB2admin` folder, and then double-click the `ALBUM` folder.

> **Tip** If the table you're opening doesn't have a unique index, a message appears, stating that the table will be opened in read-only format. You'll need to create unique indexes for the table before you can edit any of the data.

6. A default form called `Form 1` appears with the field names of the `ALBUM` table. Click the Record # arrow at the bottom of the screen to scroll through the records in the table.

7. Click the Worksheet 1 tab to see all the data from the `ALBUM` table presented as a spreadsheet (see Figure 10.4).

8. Use the instructions in the Lotus Approach books and help to perform additional tasks with the DB2 data.



ALBUMID	CATEGORYID	GROUPID	TITLE	RELEASEYEAR	RECORDLABELID	A
1	1	1	Road Apples	1991	1	
2	1	2	Sgt. Pepper's Lonely Hearts C	1967	2	
3	1	1	Up to Here	1989	1	
6	6	8	Up	1995	3	
7	1	4	Joshua Tree, The	1987	4	
8	1	4	Achtung Baby	1991	4	
9	4	7	Bloodletting	1990	5	
10	1	1	Day for Night	1994	1	
11	3	9	Four Seasons, The	1984	6	
12	4	10	To the Faithful Departed	1996	4	

Figure 10.4: The Worksheet 1 page in Lotus Approach.

Creating Unique Indexes

To update data in a DB2 database, you must have unique indexes. To create a unique index for the CDLIB database, follow these steps:

1. Start the Control Center.

2. Click the + (plus sign) in front of the `CDLIB` folder.

3. Right-click the `Indexes` folder and select Create.

4. In the Create Index window, complete the information as follows:

   - **Index schema.** Click the down arrow to see a list of schemas to choose from. The default schema is the username that you used to log in. Choose a schema appropriate for your system. In this example, `DB2ADMIN` is the schema.

   - **Index name.** Choose a name for the index you're creating. The name can be up to 18 characters long. This name must be unique within the indexes' schema—no other object in the schema can have the same name. The name used in the sample database is `CDLIBINX`.

   - **Table schema.** Click the down arrow to see a list of the schemas to choose from. Select the schema that contains the `CDLIB` table.

   - **Table name.** Click the down arrow to see a list of the tables in this schema. Select the `CDLIB` table from the list of tables that appears.

   - **Selected columns.** Select the column or columns that you want to define as part of the index key. Click the arrow to move the column from the Available columns list to the Selected columns list.

   - **Unique.** Check the Unique check box to create a unique index. A unique index prevents the table from containing two or more rows with the same value of the index key.

5. Click OK to have the index created.

Accessing Data Through Java Applications

In some cases, you might need an application that can access DB2 databases across the Internet. By using the Java programming language, you can develop applications and applets that access and manipulate data in DB2 databases.

DB2 provides support for the Sun Microsystem's Java Database Connectivity (JDBC) API through a JDBC driver that comes with DB2. The JDBC API, similar to the ODBC APIs, provides a standard way to access databases from Java code. Your Java code passes SQL statements as function arguments to the DB2 JDBC driver, which then calls the CLI/ODBC driver.

DB2 includes support for the JDBC API as distributed with Java Development Kit (JDK) 1.1. The JDK provides the tools and environment you need to develop Java applications and applets. To access DB2 databases, you need to use the DB2 JDBC driver.

You can use DB2 JDBC support to run the following types of Java programs:

- Java applications that rely on the DB2 Client Application Enabler to connectto DB2
- Java applets that don't require any other DB2 component code on the client

You can also use the Java programming language to develop user-defined functions and stored procedures that run on the server. This process is similar to creating UDFs and stored procedures in any other programming language. After you create and register your Java UDFs and stored procedures, you can call them from applications coded in any supported programming language.

Installing and Configuring the JDBC Environment

To create and use Java applications or applets with DB2 data, follow these steps:

1. Install and configure the Java Development Kit that you can download from `http://www.javasoft.com`.

2. After you install the JDK, choose Start | Settings | Control Panel | System, and click the Environment tab to set the following environment variables:

   - `CLASSPATH`. Make sure that this environment variable includes `.` and the file `sqllib\java\db2java.zip`. This file contains the DB2 Java libraries. A path to this file is necessary to run Java programs.

   - `PATH`. Make sure that this environment variable includes the directory`sqllib\bin`. This is a path to the DB2 files.

   Close the System notebook.

3. To modify the behavior of DB2 JDBC data sources, use the same `db2cli.ini` file as you set for the ODBC driver.

You can download the following source code samples from `http://www.software.ibm.com/data/db2/java/`:

File	Description
`DB2Appl.java`	JDBC application driver sample

`DB2Applt.java`	JDBC applet driver sample
`DB2Applt.html`	HTML file that runs `DB2Applt.java` sample
`DB2Stp.java`	Java stored procedures and JDBC shell
`DB2Udf.java`	Java UDF and JDBC shell

View the file and choose File | Save As in your Web browser to store these files on your local drive. The following sections discuss how to compile and run the Java application and Java applet samples.

JDBC Applications

JDBC applications rely on the DB2 Client Application Enabler to connect to DB2. You start your application from the desktop or command line, like any other application. The DB2 JDBC driver handles the JDBC API calls from your application and uses the DB2 Client Application Enabler to communicate the requests to the server and to receive the results.

If you've downloaded the JDBC application sample, compile and run it as follows:

1. To produce a `DB2Appl.class` file, type the following command in a Windows NT command window:

   Input **`javac DB2Appl.java`**

   Note These filenames are case-sensitive.

2. Make sure that DB2 is started. You can issue the `db2start` command in the command window to make sure.

3. Type the following command to run the application sample:

   Input **`java DB2Appl`**

   You'll receive a message that one row was updated.

JDBC Applets

Because JDBC applets are delivered over the Web, you treat them a bit differently than JDBC applications. On the server, you must install DB2, a Web server, and the Java Development Kit (JDK) 1.1 from Sun Microsystems. On the client, you need a Java-enabled Web browser.

JDBC applets don't require the DB2 Client Application Enabler on the client. You imbed the applet in an HTML page.

When you load your HTML page, the applet tag downloads the Java applet to your machine, which then downloads the Java class files, including the `COM.ibm.db2.java.sql` and `COM.ibm.db2.jdbc.net` classes and DB2's JDBC driver. When your applet calls the JDBC API to connect to DB2, the JDBC driver establishes separate communications with the DB2 database through the JDBC applet server residing on the DB2 server.

Compile and run your applets as follows:

1. To produce a `DB2Applt.class` file, type the following command in a Windows NT command window:

   Input **`javac DB2Applt.java`**

   Note These filenames are case-sensitive.

2. Zip the class file into an archive file named `db2java.zip` on your local drive by using the JAVA zip utility (JAR).

3. Modify the `DB2Applt.htm` file according to the instructions in the file.

4. Start the DB2 JDBC applet server on your Web server by entering

   Input **`db2jstrt`** ***`portno`***

   where *`portno`* is the number of the unused TCP/IP port that you specified in the `DB2Applt.java` file.

5. Start DB2 on the server.

6. On your client system, start your Web browser and load the `DB2Applt.htm` file that imbeds the applet. Alternatively, you can use the `appletviewer` in the JDK to run the applet:

   ```
   appletviewer DB2Applt
   ```

   You'll receive a message that one row was updated.

Accessing Data ThroughNet.Data Applications

With Net.Data, you can create dynamic Web pages with data from various sources. These interactive pages use HTML and SQL to access live data from local or remote databases. With HTML alone, you can create only static Web pages, meaning that they don't change unless you edit them.

Net.Data simplifies writing interactive Web applications by using macros to add logic, variables, program calls, and reports to HTML. These macro files define the following:

- The layout of the HTML form where the query is specified
- The information required to connect to DB2
- The SQL statement sent to DB2
- The format of the HTML and DB2 data returned

To use Net.Data, you must have a Web server installed in your environment. DB2 provides Lotus Domino Go WebServer, which you can install by following the instructions provided with Domino Go. You should install Domino Go before you install Net.Data. You could use another Web server, but these instructions assume that you're using Domino

Go.

Net.Data is provided in the DB2 package. Use the instructions provided with Net.Data to

install it.

After Domino Go and Net.Data are installed on your system, follow these steps to use Net.Data:

1. Set a number of variables in the Net.Data initialization file. This file, `db2www.ini`, is located in the Web server's HTML subdirectory. In this file, set the following path statements:

   - Set `MACRO_PATH` to the subdirectories that are searched when a Net.Data macro is called.

   - Set `EXEC_PATH` to the subdirectories that are searched when a macro attempts to execute an external program.

   - Set `INCLUDE_PATH` to the subdirectories that are searched when a macro specifies additional macros by using the `%INCLUDE` statement.

   It's reasonable to point these paths to the same subdirectory.

   Tip If you install Net.Data after Domino Go, most of these path statements are set to the correct locations.

2. Make sure that the following statements point to the appropriate subdirectories in the Web server directory:

   - `BIND_FILE` should be set to the `CGI-BIN` subdirectory.

   - `HTML_PATH` should be set to the `HTML` subdirectory.

3. Set the following two statements concerning DB2:

   - Set `DB2INSTANCE` to the instance name used in DB2.

   - Set `DB2PATH` to the `SQLLIB` subdirectory of your DB2 server.

Now you can begin creating macros. Various sample Net.Data macro files are provided to start you offwith common tasks. Follow these steps to retrieve and run the sample macros:

1. Download and unzip the `gosamp.zip` file in the subdirectory where your Net.Data macros are stored. (Point your browser to `http://www.software.ibm.com/data/db2/db2lotus/gosamp.htm` to download this file.)

2. Move the `goexamp.htm` file to the subdirectory where your HTML files are stored and accessed by Domino Go WebServer.

3. Make sure that both DB2 and Domino Go servers are running. If you're using two separate systems, make sure that the Domino Go system can connect to the DB2 server.

4. Make sure that the DB2 sample database is created. Use DB2 First Steps to create it now, if necessary.

5. Load the Web page at `goexamp.htm` to access the sample macros.

Use the sample macros and the information provided in the Net.Data Programming Guide to create your own macros.

Accessing Data ThroughYour Own Applications

This lesson has discussed the various types of applications that can access DB2 databases, including ODBC applications, JDBC applications and applets, and Net.Data macros. You can also develop applications by using the DB2 Software Developer's Kit, which includes the following:

- Embedded SQL
- APIs
- Stored procedures
- User-defined functions
- Calls to the DB2 CLI

As mentioned in earlier lessons, DB2 Software Developer's Kits are shipped with the DB2 Personal Developer's Edition and the DB2 Universal Developer's Edition.

An application on a DB2 client can access a remote database without knowing its physical location. The DB2 client determines the location of the database, manages the transmission of the requests to the database server, and returns the results. In general, to run a database client application, follow these steps:

1. Make sure that the database manager is started on the database server to which the application program is connecting. If it's not, you must issue the `db2start` command at the server before starting the application.

2. Ensure that you can connect to the database that the application uses.

3. Bind the utilities and the applications to the database before you run the application program.

Binding Database Utilities

You must bind the database utilities (import, export, reorg, the command-line processor, and DB2 CLI) to each database before they can be used with that database. In a network environment, if you're using multiple clients that run on different operating systems or are at different versions of DB2, you must bind the utilities once for each operating system/DB2-version combination.

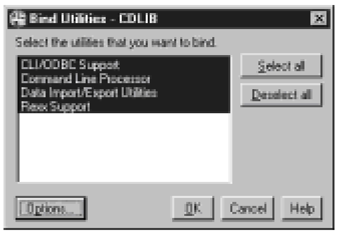
When a database is created, the database manager attempts to bind the database utilities in `db2ubind.lst` to the database. This file is stored in the `\sqllib\bnd` subdirectory.

Binding a utility creates a *package*, an object that includes all the information needed to process specific SQL statements from a single source file. The bind files are grouped together in different `.lst` files in the `bnd` directory under the installation directory (typically, `sqllib`). Each file is specific to a server.

Follow these steps to bind the database utilities to a database by using the Client Configuration Assistant (CCA):

1. Start the CCA by choosing Start | Programs | DB2 for Windows NT | Client Configuration Assistant.
2. Select the database to which you want to bind the utilities.
3. Click the Bind button.
4. Select the Bind DB2 Utilities radio button.
5. Click the Continue button.
6. Enter a username and password to connect to the database. The username must have the authority to bind new packages against this database.
7. Select the utilities you want to bind (in Figure 10.5, all the utilities are selected to be bound to the database) and click OK.

Binding can take a few minutes to complete.


Bind Utilities - CDLIB

Select the utilities that you want to bind.

CLI/ODBC Support
Command Line Processor
Data Import/Export Utilities
Rexx Support

Figure 10.5: Binding utilities with the Client Configuration Assistant.

The `db2ubind.list` file contains the list of bind (`.bnd`) files required to create the packages for the database utilities. The `db2cli.lst` file contains the list of bind (`.bnd`) files required to create packages for the DB2 CLI and the DB2 ODBC drivers.

Summary

In today's lesson, you saw several ways that you can access the data in DB2 databases. Some ways include the Control Center; SQL statements entered in the Command Center; ODBC applications such as Lotus Approach and Microsoft Access; Internet applications created with Java or Net.Data; or applications written in C, C++, or other programming languages.

When accessing data through the Control Center, you can sample the first 200 rows of the selected table. When accessing data through the Command Center, you can enter valid SQL statements.

Before you can use an ODBC application, you must register the database to be ODBC-enabled. When this is done, using an ODBC application makes accessing data in a DB2 database quite simple.

What Comes Next?

On Day 11, "Using System Administration Tools," you learn how to use tools such as the Script Center to create scripts that can be saved and scheduled.

Q&A

Q What's the difference between the Command Center, the command-line processor, and the Command window?

A The main difference between these tools is that the Command Center provides a graphical interface to the command line. When you use the Command Center, you can move between the entry and results pages and save queries or results to files.

The command-line processor and Command window are basically Windows NT command-line sessions. The command-line processor provides several different modes of operation, the most commonly used being interactive mode. When you're in interactive mode, you'll see a DB2 prompt and can enter commands and statements the same as you would in the Command Center (without prefixing the commands with `db2`).

The Command window is a Windows NT command window prepared to accept DB2 commands. All DB2 commands and SQL statements must be prefixed with `db2` for the commands to be valid.

Overall, it makes very little difference which method you use for commands and statements. Try all three methods, and use the one you become most comfortable with. One or two commands can't be entered in the Command Center. If you receive an error message that the command isn't known or accepted, try entering the command in the command-line processor or the Command window to see whether you can successfully run the command.

Q How can I register a database to be ODBC-enabled?

A The easiest way to enable a database for ODBC applications is through the Client Configuration Assistant, which also enables you to optimize the ODBC settings for the application you intend to use.

Q What do I need to create Java applications or applets?

A Java Database Connectivity (JDBC) support is built into each DB2 product. When you have DB2 running, you'll need to download the Java Development Kit (JDK) before you can create Java applications and applets.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. What's a benefit of using Net.Data to create Web applications?

2. What's the main difference between a JDBC applet and a JDBC application?

3. If you receive a message that the table you're opening in an ODBC application is read-only, what's the most likely problem?

4. When entering commands in the Command Center, do you need to end each line with a `;` (semi-colon)?

Exercise

Lotus Approach is one example of an ODBC-enabled application. If you have access to another ODBC-enabled application, such as Microsoft Access, try to access DB2 data from this application, following the instructions from today's lesson.

Using the Script Center

You use the Script Center to view summary information about all command scripts, to back up scripts, to reorganize table scripts known to the system, and to work with these scripts. A mini-application known as a *script* is a set of one or more commands created through the Command Center or the Script Center.

Scripts don't contain scheduling information. If you schedule or run a saved script, an entry called a *job* is logged on the Jobs page of the Journal. If you modify a saved script, all jobs that depend on this saved script inherit the modified behavior.

Creating a Command Script

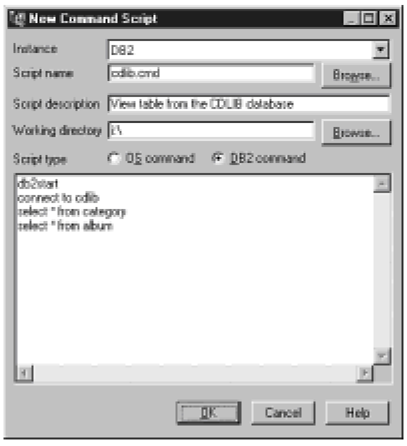
Use the Script Center to create command scripts that can be stored and invoked at a later time. Scripts can contain DB2 commands, SQL statements, and operating-system commands. Follow these steps to create a command script:

1. Start the Script Center. Click the Script Center icon on the toolbar of the Control Center, or choose Start | Programs | DB2 for Windows NT | Administration Tools | Script Center.

2. From the System Name box, select a system on which to store and run the command scripts. The default is your local system.

3. Choose Script | New from the menu.

4. In the New Command Script window, type the commands that will make up your script (see Figure 11.1). Put each command on a separate line; don't separate the lines with semicolons.



Instance: DB2
Script name: cdlib.cmd
Script description: View table from the CDLIB database
Working directory: i:\
Script type: ( ) OS command [✓ DB2 command]

```
db2start
connect to cdlib
select * from category
select * from album
```

Figure 11.1: The Script Center's New Command Script window.

5. Select an instance name from the Instance drop-down box to associate with the

command script.

6. In the Script Name text box, type a descriptive name for your command script, specifying the full path of the file. If you don't specify the path, the default value is the administration server instance directory. You can use the Browse button to browse for the path of your command file. The name of the script file must contain an appropriate filename extension, such as `.cmd` or `.bat`.

7. Type a brief description of your command script in the Script Description text box.

8. In the Working Directory text box, type the full path of the directory from which the script will run. If you don't specify the full path, the default path is the Windows NT operating-system directory.

9. Select a radio button that indicates the script type. Select OS Command if the script contains Windows NT commands; select DB2 Command if the script contains DB2 commands.

10. Type the commands that will make up your script. For example, to start the database manager, connect to the `CDLIB` database, and view the contents of the `CATEGORY` and `ALBUM` tables; enter the following commands:

Input
```
db2start
connect to cdlib
select * from category
select * from album
```

11. Click OK when you've finished entering the commands that will make up your script.

To create a new command script from an existing file, choose Script | Import from the menu. In the File Browser window, choose the file you want to import and click OK. The New Command Script window opens and shows the contents of the imported file. Enter a new script name, script description, working directory, and whether the new script will contain DB2 or OS commands. Modify the script and click OK to save it.

Scripts are also created when you create a backup plan by using the Backup SmartGuide. These scripts can't be modified through the Script Center, but you can run them from the Script Center.

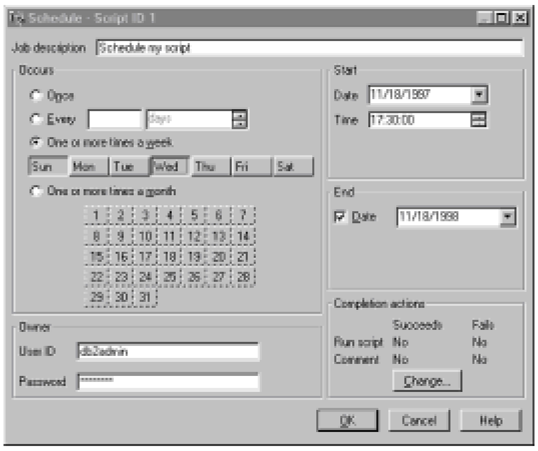
Running Scripts

You can run a saved script or schedule to run it at a later time or date. To immediately run a saved command script, right-click it and select Run Now from the pop-up menu. In the Run Now window, type a valid username and password. The results of the script will appear in the Journal Jobs page (see the next section "Using the Journal" for details).

To schedule a saved script to run every Sunday and Wednesday at 5:30 p.m. for the next year, for example, follow these steps:

1. Right-click the saved script that you want to schedule to run, and select Schedule from the pop-up menu.

2. In the Schedule window, enter a job description, start date, and start time (see Figure 11.2).


Schedule - Script ID 1

Job description: Schedule my script

Occurs: One or more times a week — Sun, Wed

Start: Date 11/18/1997, Time 17:30:00

End: [✓] Date 11/18/1998

Owner: User ID db2admin, Password ********

Completion actions	Succeeds	Fails
Run script	No	No
Comment	No	No

Figure 11.2: The Script Center's Schedule window.

3. In the Occurs section, select one of the following radio buttons to indicate how often the job is to run:

   - Choose Once to have the script run as a single job.
   - Choose Every to schedule the job to occur every few hours, weeks, or months. Specify the frequency in the adjacent fields.
   - Choose One Or More Times A Week to select the days on which your job is to run within a week.
   - Choose One Or More Times A Month to select the dates on which your job is to run within a month.

   For this example, choose to have the script run every Sunday and Wednesday at 5:30 p.m.

4. In the Owner section, type a valid username and password.

5. In the Completion Actions section of the Schedule window, you click Change to define completion actions, which should be taken when a job succeeds or fails. You can choose to run another command script or to have a comment appear in the results. For this example, leave this section blank.

6. Click OK to have the job scheduled. A job ID is assigned to the job.

Look at the Journal in the Jobs page to see the job you've just scheduled.

Using the Journal

You use the Journal to view all available information about jobs that are pending execution, are executing, or have completed execution. You also can view the recovery history log, view the alerts log, view the messages log, and review the results of jobs that run unattended.

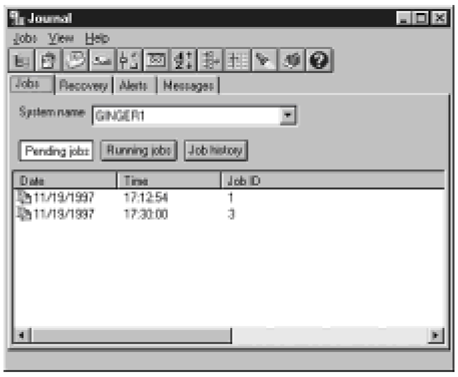
Viewing the Results of a Job

From the Jobs page in the Journal, you can view pending, running, and executed jobs. Follow these steps:

1. Open the Journal by clicking the Journal icon on the Control Center's toolbar or by choosing Start | Programs | DB2 for Windows NT | Administration Tools | Journal.

2. From the System Name drop-down box, select the system on which to store and run the command scripts. The default is your local system.

3. To see all jobs scheduled to run, click the Pending Jobs button (see Figure 11.3). To view all the jobs now running, click the Running Jobs button. To view the results of a job, click the Job History button.



System name: GINGER1

Date	Time	Job ID
11/19/1997	17:12:54	1
11/19/1997	17:30:00	3

Figure 11.3: The Journal's Pending Jobs view.

On the Journal's Pending Jobs view, you can remove jobs, reschedule jobs, run jobs immediately, and disable jobs:

- To remove a job, right-click it and select Remove from the pop-up menu.

- To reschedule a job, right-click it and select Reschedule from the pop-up menu. In the Reschedule window, indicate when and how often you want the job to run, and click OK.

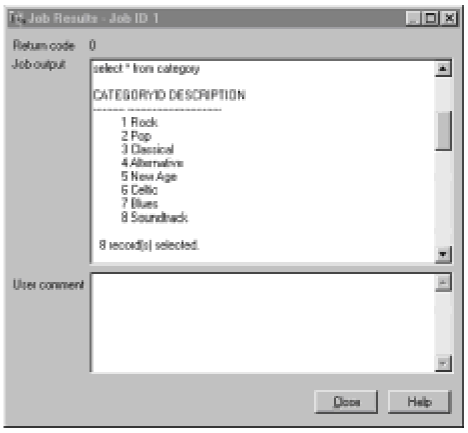
- To run a job now, right-click it and select Run Now from the pop-up menu. In the Run Now window, type a valid username and password.

- To disable a job, right-click it and select Disable from the pop-up menu. The job definition remains, but the job isn't run. To enable a disabled job, right-click it and select Enable from the pop-up menu.

On the Journal's Job History view, you can remove a job entry, show the results of the script, or show the script:

- To remove a job entry, right-click it and select Remove from the pop-up menu.

- To view the results of a script that has completed, right-click it and select Show Results from the pop-up menu. Figure 11.4 shows the results of the script run earlier, including a statement saying that the database manager was started and that the connection to the CDLIB database worked successfully, and showing the rows in the `CATEGORY` and `ALBUM` tables.



```
select * from category

CATEGORYID DESCRIPTION
---------- -----------------
         1 Rock
         2 Pop
         3 Classical
         4 Alternative
         5 New Age
         6 Celtic
         7 Blues
         8 Soundtrack

  8 record(s) selected.
```

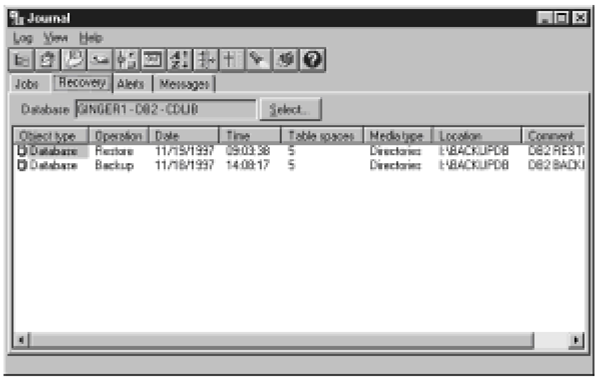
Figure 11.4: Showing script results in the Journal.

- To view the contents of a saved script, right-click the job whose script you want to view, and select Show Script from the pop-up menu. The Show Command Script window opens.

  Note You can't modify the contents of a script in the Show Command Script window; return to the Script Center to do so.

Viewing the Recovery History Log

Use the Journal's Recovery page to restore databases, table spaces, or the recovery history of a selected database. You can also view table space information from this page. Figure 11.5 shows an example of the Journal's Recovery page.



Database GINGER1 - DB2 - CDLIB

Object type	Operation	Date	Time	Table spaces	Media type	Location	Comment
Database	Restore	11/15/1997	09:03:38	5	Directories	E:\BACKUPDB	DB2 REST
Database	Backup	11/18/1997	14:08:17	5	Directories	E:\BACKUPDB	DB2 BACK

Figure 11.5: The Journal's Recovery page.

Tip No entries for a database means that you haven't yet backed up the data in your database. Entries are posted in this log as soon as a backup is performed.

On the Recovery page, click the Select button. Choose an instance and database that you want to recover. Click OK to return to the Recovery page to see entries that relate to the backups performed on this database. With the entries displayed, you can do the following:

- To restore a selected database from a backup image that has been made from this database, right-click it and select Restore from the pop-up menu. The Restore Database notebook opens. See Day 12, "Backing Up and Recovering Data," for information on using this notebook.

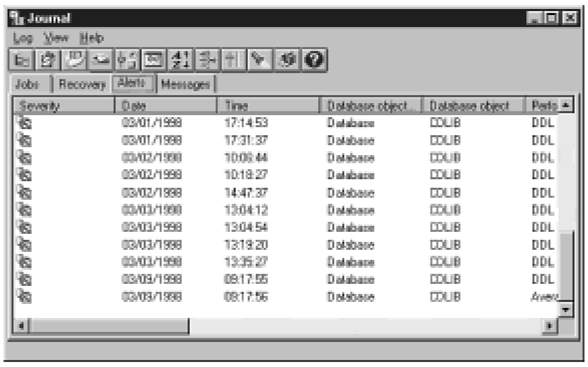
- To restore a database image to a different database than the one from which it was made, right-click it and select Restore To New from the pop-up menu. The Restore Database To New notebook opens. By restoring to a new database, you can clone a database. You can restore database images from other systems or platforms as well as databases created on this system.

- To restore the recovery history of a database, right-click it and select Restore Recovery Histories from the pop-up menu. The Restore Database Recovery History notebook opens. If a backup image for the database isn't found, the window opens in manual mode. When in manual mode, you must enter the name of another system, instance, and database to recover.

- To see a list of table spaces associated with this database, right-click it and select View Table Spaces from the pop-up menu. The View Table Spaces window opens.

Viewing the Alerts Log

Use the Alerts page to view and prune the alerts log (see Figure 11.6). An alert message is added to the log when a threshold that you defined through the Snapshot Monitor is reached. An alert message can tell you that a performance objective hasn't been met. See Day 18, "Performance Aids," for information on defining monitors and alerts.



Severity	Date	Time	Database object ...	Database object	Perfo
	03/01/1998	17:14:53	Database	CDLIB	DDL
	03/01/1998	17:31:37	Database	CDLIB	DDL
	03/02/1998	10:06:44	Database	CDLIB	DDL
	03/02/1998	10:18:27	Database	CDLIB	DDL
	03/02/1998	14:47:37	Database	CDLIB	DDL
	03/03/1998	13:04:12	Database	CDLIB	DDL
	03/03/1998	13:04:54	Database	CDLIB	DDL
	03/03/1998	13:19:20	Database	CDLIB	DDL
	03/03/1998	13:35:27	Database	CDLIB	DDL
	03/09/1998	09:17:55	Database	CDLIB	DDL
	03/09/1998	09:17:56	Database	CDLIB	Avera

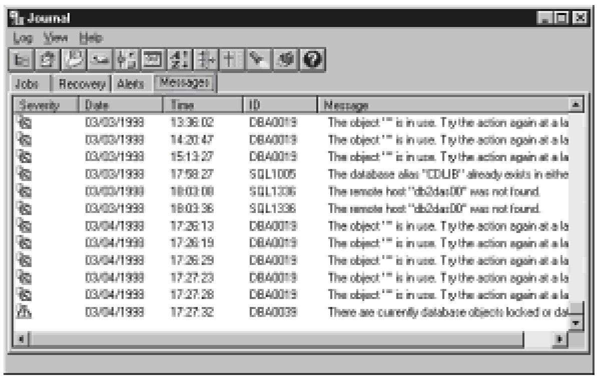
Figure 11.6: The Journal's Alerts page.

The information on the Alerts page indicates the severity of the alert (warning or alarm); the date and time when the alert occurred; the object name and type of the object being monitored; and the name, identifier, value, category, description, defined thresholds, and history of the performance variable that caused the alert.

To prune all messages from the log, choose Log | Remove All from the menu. To delete individual messages, right-click the message you want to delete, and choose Remove from the pop-up menu.

Viewing the Messages Log

Use the Messages page to view the messages log (see Figure 11.7). Each error, warning, or informational message that occurs while operating DB2 is logged here. The log lists the severity of the message, the date and time the message occurred, the message identifier, and the message text.



Severity	Date	Time	ID	Message
	03/03/1998	13:36:02	DBA0019	The object "" is in use. Try the action again at a la
	03/03/1998	14:20:47	DBA0019	The object "" is in use. Try the action again at a la
	03/03/1998	15:13:27	DBA0019	The object "" is in use. Try the action again at a la
	03/03/1998	17:58:27	SQL1005	The database alias "CDLIB" already exists in eithe
	03/03/1998	18:03:08	SQL1336	The remote host "db2das00" was not found.
	03/03/1998	18:03:36	SQL1336	The remote host "db2das00" was not found.
	03/04/1998	17:26:13	DBA0019	The object "" is in use. Try the action again at a la
	03/04/1998	17:26:19	DBA0019	The object "" is in use. Try the action again at a la
	03/04/1998	17:26:29	DBA0019	The object "" is in use. Try the action again at a la
	03/04/1998	17:27:23	DBA0019	The object "" is in use. Try the action again at a la
	03/04/1998	17:27:28	DBA0019	The object "" is in use. Try the action again at a la
	03/04/1998	17:27:32	DBA0039	There are currently database objects locked or da

Figure 11.7: The Journal's Messages page.

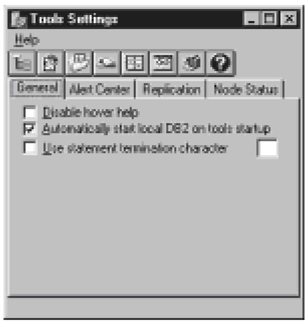
Prune this log when it becomes full. Right-click one or more entries and select Remove to remove only the highlighted messages, or select Remove All to remove all entries in the log.

Customizing Tools Settings

You use the Tools Settings notebook to customize settings and set properties for the administration tools and for replication tasks. To open the Tools Settings notebook, click the Tools Settings icon on the Control Center's toolbar, or choose Start | Programs | DB2 for Windows NT | Administration Tools | Tools Settings.

On the notebook's General page (see Figure 11.8), you can disable hover help, automatically start the local instance, and define a statement termination character:

- Select the Disable Hover Help check box to stop displaying the help that appears when you put your mouse pointer over an item in the administration tools.
- Select the Automatically Start Local DB2 On Tools Startup check box to have the local DB2 instance begin each time an administration tool is started.
- Select the Use Statement Termination Character check box to define a character to use at the end of statements in command scripts in the Command Center and Script Center. If you select this check box, the semi-colon (`;`) is used by default to end statements. Optionally, you can define a different character by typing it in the text box.

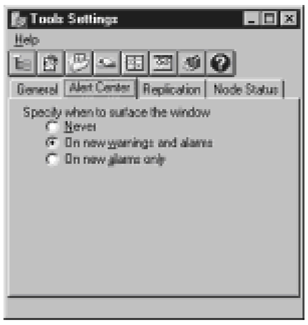


- [ ] Disable hover help
- [x] Automatically start local DB2 on tools startup
- [ ] Use statement termination character

Figure 11.8: The Tools Settings notebook's General page.

> **Note** You can't specify the backslash (\) character to continue statements in command scripts.

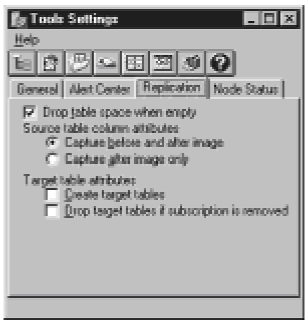
On the notebook's Alert Center page, you can specify when the Alert Center window is displayed (see Figure 11.9). See Day 18, "Performance Aids," for information on snapshot monitors.



Specify when to surface the window
( ) Never
(●) On new warnings and alarms
( ) On new alarms only

Figure 11.9: The Tools Settings notebook's Alert Center page.

On the notebook's Replication page, you can set global properties for replication tasks (see Figure 11.10). You can set properties for certain attributes of source table columns and target tables. See Day 14, "Replicating Data," for information on replication tasks.



[✓] Drop table space when empty
Source table column attributes
(●) Capture before and after image
( ) Capture after image only
Target table attributes
[ ] Create target tables
[ ] Drop target tables if subscription is removed

Figure 11.10: The Tools Settings notebook's Replication page.

The Node Status page is for partitioned databases only. This topic is beyond the scope of this book.

Making Your Database More Useful

You can make your databases more useful in several ways:

- You can create an index on your table that can improve the performance of selecting data from the table.
- You can create distinct data types to better reflect the type of data you intend to store in a column.
- You can use the functions provided with DB2 to perform complex mathematical operations on your data. You can also create your own functions if the provided functions don't exactly suit your needs, although creating user-defined functions is beyond the scope of this book.
- Creating triggers can introduce additional logic in your database. For example, you can create a trigger that sends an email message when a particular event occurs in the database.

The following sections go into more detail.

Creating Indexes

An *index* is the key value of a table. It provides pointers to the rows in the table, which enables more efficient access by creating a direct path to the data. Also, you can define index keys as unique, thereby preventing duplicate rows from occurring within the table.

An index is created when you create a unique key, a primary key, or a foreign key. You can create an index at the time the table is defined or at any time afterward.

To view the indexes defined on each table in the CDLIB database, follow these steps:

1. Start the Control Center.

2. Expand the objects until you see the Indexes folder. Click that folder to see a list of the indexes in the contents pane. Each index has a name, schema, table schema, and table name associated with it.

3. Right-click an index and select Alter from the pop-up menu. In the Alter Index window, you can see the columns used to make up the index, but you can't modify anything except the comment. If you need to make changes, you must drop the existing index and re-create it.

The effectiveness of defining an index depends largely on the type of table access required. An index is most effective if it matches the way you access your data. Although the creation of an index usually helps improve performance on large or frequently updated databases, each index must be kept up to date. Therefore, each insert, update, and delete request against a table will require an update for the index, if any table columns affected are included in the index definition. Such updates might possibly degrade performance. Indexes should therefore be created carefully, with an understanding of how the data is stored and used. The simplest recommendation is that you define a key on the table columns you use the most.

To create a new index, follow these steps:

1. Start the Control Center.

2. Expand the objects until you see the Indexes folder.

3. Right-click the Indexes folder, and select Create from the pop-up menu.

4. In the Create Index window, complete the following information:

   - **Index Schema.** From this drop-down list, select a schema. The default schema is the username that you used to log in; use a schema appropriate for your system.

   - **Index Name.** Choose a name for the index you're creating. The name can be up to 18 characters long. This name must be unique within the indexes' schema; no other object in the schema can have the same name.

   - **Table Schema.** From this drop-down list, select the schema that contains the table for which you're creating an index.

   - **Table Name.** From this drop-down list, select the table in this schema for which you're creating an index.

   - **Selected Columns.** Select the column or columns that you want to define as part of the index key. Click the arrow to move the column from the Available Columns list box to the Selected Columns list box.

- **Unique.** Select the Unique check box to create a unique index, which prevents the table from containing two or more rows with the same value of the index key.

5. Click OK to have the index created.

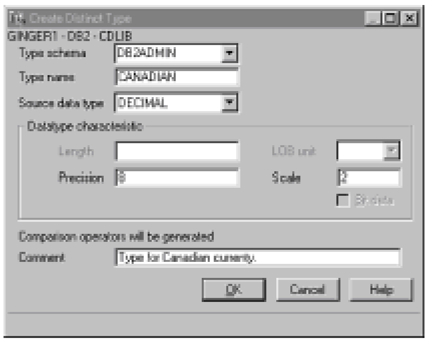
Creating Distinct Types

You'll want to create a distinct type to more closely represent your unique data. A distinct type shares its internal representation with an existing type but is considered a separate and incompatible type for semantic purposes. For example, to create a Canadian dollar type based on the decimal data type, follow these steps:

1. Start the Control Center.

2. Expand the objects until you see the User Defined Types folder. Right-click the folder and select Create from the pop-up menu.

3. In the Create Distinct Type window, select a value for the schema (see Figure 11.11).



Create Distinct Type

GINGER1 - DB2 - CDLIB

Type schema: DB2ADMIN

Type name: CANADIAN

Source data type: DECIMAL

Datatype characteristic

Length:

LOB unit:

Precision: 8

Scale: 2

☐ Bit data

Comparison operators will be generated

Comment: Type for Canadian currency.

Figure 11.11: The Create Distinct Type window.

4. Enter a name for the new type. For example, use `CANADIAN` for the Canadian currency.

5. Choose an existing data type for the source. In this case, you want the type to look and act like the `DECIMAL` data type.

6. For the `DECIMAL` data type, you must enter a precision and scale. *Precision* is the total length of the number, including decimal places; *scale* is just the decimal places. For example, to hold a maximum of $999,999.99, you should use 8 as the precision and 2 as the scale.

7. Type a comment to describe your new type, and click OK. You'll see the type `CANADIAN` in the contents pane.

To use the type, select `CANADIAN` from the list of data types when you're defining the columns for a table.

Listing User-Defined Functions

You might want to create your own suite of functions that are specific to an application or domain. Creating a function is beyond the scope of this book.

To see the list of functions available on your system, follow these steps:

1. Start the Control Center.
2. Expand the objects until you see the User Defined Functions folder.
3. Click the folder to see a list of the functions in the contents pane.

Some functions that DB2 provides include `DAYNAME`, `DEGREES`, and `DIFFERENCE`. Each function has a schema, a name, a specific name, a result, and input parameters associated with it.

Creating Triggers

A *trigger* is an object that initiates an action when an `UPDATE`, `DELETE`, or `INSERT` operation is run against a table. You can use triggers that run before such operations in several ways:

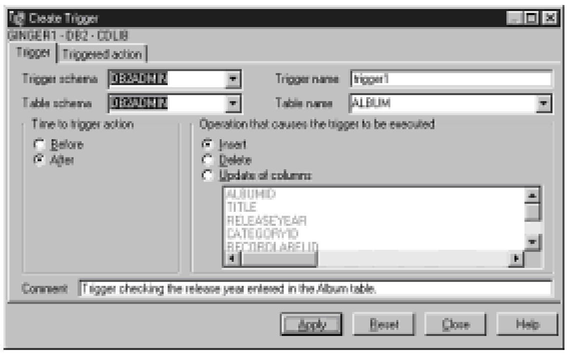
- To check for certain conditions before performing a triggering operation.
- To change the input values before they're stored in the table.
- To check or modify values that run before an `UPDATE`, `INSERT`, or `DELETE` in the database. This is useful if you need to transform data from the way the user sees it to some internal database format.
- To run other nondatabase operations coded in user-defined functions.

Similarly, you can use triggers that run after an update or insert in several ways:

- To update data in other tables. This is useful for maintaining relationships between data or in keeping audit trail information.
- To check against other data in the table or in other tables. This is useful to ensure data integrity when referential integrity constraints aren't appropriate or when table check constraints limit checking to the current table only.
- To run nondatabase operations coded in user-defined functions. This is useful when issuing alerts or to update information outside the database.

To see how triggers work, create a trigger for the CDLIB sample database to verify that the release year of an album is valid. Follow these steps:

1. Start the Control Center.
2. Expand the objects until you see the Triggers folder within the CDLIB database.
3. Right-click the folder, and select Create from the pop-up menu. The Create Trigger notebook opens (see Figure 11.12).


Create Trigger
GINGER1 - DB2 - CDLIB
Trigger | Triggered action

Trigger schema: DB2ADMIN
Table schema: DB2ADMIN
Trigger name: trigger1
Table name: ALBUM

Time to trigger action: Before [✓ After]

Operation that causes the trigger to be executed: [✓ Insert] Delete Update of columns

ALBUMID
TITLE
RELEASEYEAR
CATEGORYID
RECORDLABELID

Comment: Trigger checking the release year entered in the Album table.

Figure 11.12: The Create Trigger notebook.

4. On the Trigger page, select a schema for the trigger and a schema for the table the trigger will be acting on.

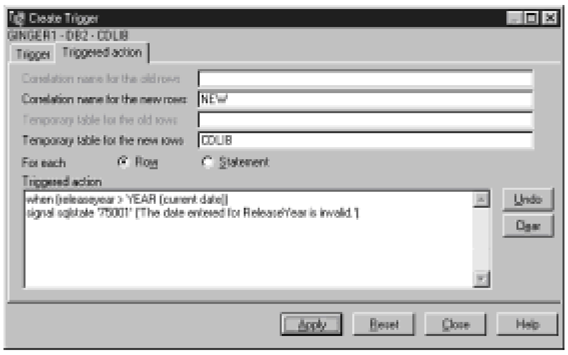
5. Enter a name for the trigger, fewer than 18 characters and unique within the schema—for example, **trigger1**.

6. Select a table for the trigger to act on. For this example, you want the trigger to act on the `ALBUM` table.

7. Select a time to trigger the action. Select Before to activate the trigger before the `UPDATE`, `INSERT`, or `DELETE` action; select After to activate the trigger after the `UPDATE`, `INSERT`, or `DELETE` action. For this example, select After to verify the value entered for `RELEASEYEAR`.

8. Select an operation that causes the trigger to be executed. Your choices are Insert, Delete, or Update Of Columns. If you choose Update Of Columns, you can select a specific column to trigger the action. For this example, however, select Insert.

9. On the Triggered Action page, you can specify correlation names for old and new rows and temporary tables for old and new rows (see Figure 11.13). Correlation names and temporary tables are necessary when you need to refer to the rows before or after they're updated. For this example, use `NEW` for the correlation name and `CDLIB` for the temporary tables.


Create Trigger
GINGER1 - DB2 - CDLIB
Trigger | Triggered action

Correlation name for the old rows:
Correlation name for the new rows: NEW
Temporary table for the old rows:
Temporary table for the new rows: CDLIB
For each [✓ Row] Statement

Triggered action

```
when (releaseyear > YEAR (current date))
signal sqlstate '75001' ('The date entered for ReleaseYear is invalid.')
```

Figure 11.13: The Create Trigger notebook's Triggered Action page.

10. Specify whether the trigger is to occur each time the operation acts on a row or just one time after the operation acts on all rows. For this example, to have the trigger occur each time a row is inserted, select the Row radio button.

11. In the Triggered Action list box, enter a SQL statement to define the trigger. The

following statement compares the value of the RELEASEYEAR column with the current year, which results in an error if the entered date is greater than the current date:

Input
```
when (releaseyear  YEAR(current date))
    signal sqlstate '75001' ('The date entered for
    ReleaseYear is invalid.')
```

12. Click Apply to have the trigger created.

13. To test the trigger, insert a row in the ALBUM table, providing a RELEASEYEAR that's greater than the current year.

Checking Space Available in a Table Space

To check the amount of space available in a DMS table space, follow these steps:

1. Start the Control Center.

2. Expand the objects until you see the Table Spaces folder within the database you want to work with.

3. Click the Table Spaces folder to see a list of table spaces in the contents pane.

4. Scroll to the columns titled Allocated Size, Size Used, and Percentage Used to see details related to the amount of space available in a table space. Space is shown in pages, one page being 4KB. (A *page* is a fixed-length block of memory used in virtual memory management.)

To check the amount of space available in an SMS table space, use the tools provided by your operating system, such as Explorer, to monitor space used and to ensure that space is available for the table space.

Allocating Additional Space

Capacity for a DMS table space is the total size of containers allocated to the table space. When a DMS table space reaches capacity (depending on table space used, 90% is a possible threshold), you should add more space to it. The database manager will then automatically rebalance the tables in the table space across all available containers. During rebalancing, data in the table space remains accessible.

The strategy for adding space to a table space is different for DMS and SMS table spaces. For a DMS table space that has reached capacity, you can add another container by following these steps:

1. Start the Control Center.

2. Expand the objects until you see the Table Spaces folder within the database you want to work with.

3. Click the Table Spaces folder to see a list of table spaces in the contents pane.

4. Right-click the table space in the contents pane and select Alter from the pop-up menu. The Alter Table Space window opens.

5. Click Add to open the Add Container window.

6. If you select the File radio button, enter a size, a drive, and a directory for the new container—for example, `i:\container1`.

   If you select the Raw Device radio button, enter the size of the new container, and select whether the device is an unformatted partition or an unformatted drive.

7. Click OK to return to the Alter Table Space window, and click OK again to have the new container added to the table space.

In general, you can't extend the size of an SMS table space very easily because SMS capacity depends on the space available in the file system and the maximum size of the file supported by the operating system. You can use the operating-system commands to move the files to a larger file system, or you can use the DB2 redirected restore to back up the table spaces and restore them in a larger number of containers. Day 12, "Backing Up and Recovering Data," covers redirected restores.

Summary

In today's lesson, you saw that the Script Center enables you to work with saved scripts. A *script* is a set of one or more commands and is created through the Command Center or the Script Center.

The Journal enables you to view information about any scheduled, running, or completed jobs. You also can view the various logs in which DB2 stores information.

You can create many objects to make your database more useful. These objects include indexes, distinct types, user-defined functions, and triggers.

Check the space available for a table space to determine whether you have enough space for the data you need to add.

What Comes Next?

On Day 12, "Backing Up and Recovering Data," you learn how to develop a backup and recovery plan to protect the data in your database.

Q&A

Q Can I create a script that contains operating system and DB2 commands?

A You can create scripts that contain only operating system commands and scripts that contain only DB2 commands. You can't combine both types of commands in a single script. When you're creating a script, be sure to select which type of commands your script contains.

Q Can a script contain scheduling information?

A A script can contain only commands. A script can, however, be scheduled to run at a specified date and time. If you do schedule to run a script, the scheduling information is part of the Journal and not the script itself. When you create a script, you can use the Journal to run the script now, stop the script from running, schedule the script to run, and view the results of the script. If you decide that you need to make changes to the saved script, you must return to the Script Center to make these changes.

Q What's the purpose of the recovery history log?

A The recovery history log contains information about when a backup or recovery was performed for a particular database. If you don't have information in the log, it means that you haven't yet performed a backup or recovery on the data in the database. If you've performed a backup, this information is contained in the log. You can select to restore a backup image while viewing the recovery history log information. This is also the place where you can view the table spaces (if any) associated with the database.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. How do you create an index? Why do you want to create an index?
2. If you don't see hover help while you're using the DB2 tools, how can you turn this feature on?
3. If you create several distinct types based on the same existing type, can you directly compare these values?
4. Can you modify scripts that are created as a result of using the Backup Database SmartGuide?

Exercise

Today's lesson introduced the Script Center and Journal. Together, these tools help simplify many of the tasks you'll need to perform as a database administrator. Try out the many options available in these tools to gain an understanding of how you create scripts and what you can do after a script is saved.

Day 12: Backing Up and Recovering Data

Overview

When you have data in your database, you must guard against losing this data. DB2 is equipped with tools to help you back up and recover your data.

In this lesson, you'll learn about the different logging methods you can use to protect your data. You'll also see how to set up a recurring schedule for your backups, verify that your backups are valid, restore your data in case of a database failure, and roll forward the transactions logs you've been keeping if you're using the log archiving.

Recovering Data

No matter how careful you are, disasters can strike your data. You might have to deal with media and storage problems, power interruptions, and application failures. The Restore Database SmartGuide helps you handle the basic database recovery. If you find that you have a more complex problem, use the information on the Restore Database notebook covered later.

DB2 automatically recovers from system crashes. If your database system crashes because of a software problem or power failure, DB2 automatically restores your database to the state just after the last committed transaction by using a set of logs that record every transaction not saved to the hard drive. All committed units of work not written to disk will be redone when the system comes back and the first application connects to a database, or when a database is restarted.

Using the Restore Database SmartGuide

The Restore Database SmartGuide helps you restore a database to the point in time of the last database backup or the last completed transaction.

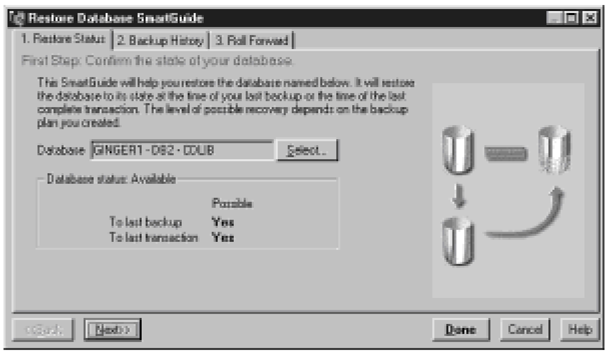
Occasionally, you might need to undo something that happened to your database: For example, if an errant application program damaged your data, you would want to put the database back to the point just before that application was run against it. To do this, use the Restore Database SmartGuide to perform a full database restore, but in the last step, specify a date and time to which you want to roll forward your database by using the logs.

To use the SmartGuide, follow these steps:

1. Start the Control Center.

2. Right-click the database you want to restore, and select Restore | Database Using SmartGuide from the pop-up menu.

3. On the Restore Status page (see Figure 12.11), you see the status of your database and whether it's possible to restore it to the last backup or to the last transaction.



First Step: Confirm the state of your database.

This SmartGuide will help you restore the database named below. It will restore the database to its state at the time of your last backup or the time of the last complete transaction. The level of possible recovery depends on the backup plan you created.

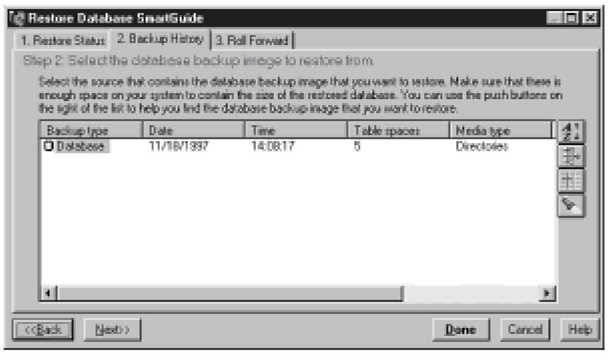
Database: GINGER1 - DB2 - CDLIB

Database status: Available

	Possible
To last backup	Yes
To last transaction	Yes

Figure 12.11: The Restore Database SmartGuide's Restore Status page.

4. If the hard drive that stores your database has crashed or the data has been accidentally erased, you need to restore your database from a backup image. The Backup History page contains a list of database backups (see Figure 12.12).



Step 2: Select the database backup image to restore from.

Select the source that contains the database backup image that you want to restore. Make sure that there is enough space on your system to contain the size of the restored database. You can use the push buttons on the right of the list to help you find the database backup image that you want to restore.

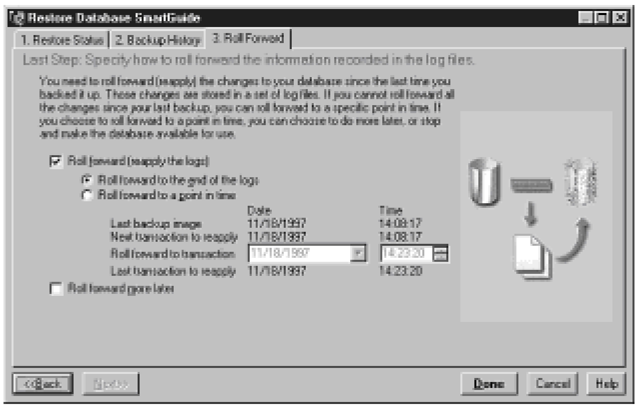
Backup type	Date	Time	Table spaces	Media type
Database	11/18/1997	14:08:17	5	Directories

Figure 12.12: The Restore Database SmartGuide's Backup History page.

Select the image you want to restore, and then click Done to begin the restore procedure (or click Next to move to the Roll Forward page, and go to the next step if you enabled log archiving). You can use the buttons on the right of the list to help you find the database backup image that you want to restore.

5. If you enabled log archiving, you can also roll forward or reapply the changes stored in the logs. (Log archiving is enabled if the `LOGARCHIVE` database configuration parameter is set to `YES`.) Reapplying the changes in the logs means that you can return the database to its state just before the last complete transaction, even if you're dealing with a hard drive crash. Use the Roll Forward page to set the point where the logs should stop being replayed (see Figure 12.13).

Note The Roll Forward page won't appear if you're using circular logging.


Restore Database SmartGuide

1. Restore Status | 2. Backup History | 3. Roll Forward

Last Step: Specify how to roll forward the information recorded in the log files.

You need to roll forward (reapply) the changes to your database since the last time you backed it up. Those changes are stored in a set of log files. If you cannot roll forward all the changes since your last backup, you can roll forward to a specific point in time. If you choose to roll forward to a point in time, you can choose to do more later, or stop and make the database available for use.

[✓] Roll forward (reapply the logs)

[●] Roll forward to the end of the logs

[ ] Roll forward to a point in time

	Date	Time
Last backup image	11/18/1997	14:08:17
Next transaction to reapply	11/18/1997	14:08:17
Roll forward to transaction	11/18/1997	14:23:20
Last transaction to reapply	11/18/1997	14:23:20

[ ] Roll forward more later

Figure 12.13: The Restore Database SmartGuide's Roll Forward page.

Roll-forward recovery builds on a restored database and enables you to rebuild a database to a specified point in time. For example, if a disk failure occurs on the disk containing your database, you can restore and then roll the database forward to recover the updates as near the point of failure as possible.

If an application problem corrupted your database or accidentally deleted data, you can recover it by restoring the backup image and then performing roll forward to the point in time just before the application destroyed your data. Although you lose all changes made after that point in time, you don't lose the earlier data.

Caution If you choose to stop rolling forward and make the database ready for use, be careful! When you click Done, you can't continue rolling forward. Because a new log file is created that can overwrite an existing one, you should copy your log files and your database backup image before you complete your roll forward.

Using the Restore Database Notebook

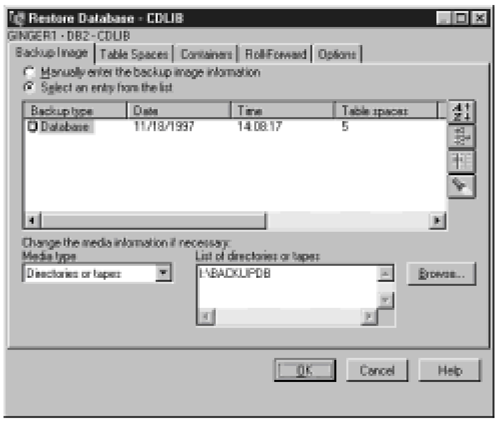
You can choose to use the Restore Database notebook to restore a selected database from a backup image that has been made of your database. The Restore Database notebook varies from the SmartGuide in that it enables you to restore selected table spaces and change the containers assigned to the table spaces.

To open the Restore Database notebook, start the Control Center, right-click the database you want to restore, and then select Restore | Database from the pop-up menu. The notebook has five tabbed pages: Backup Image, Table Spaces, Containers, Roll Forward, and Options.

The following sections explain how and when to use all the Restore Database notebook pages. Click OK when you've set all the appropriate options.

Selecting the Backup Image

On the Backup Image page (see Figure 12.14), you select the backup image you want to use to restore the database. If the database image you want to restore isn't listed, click Manually Enter The Backup Image Information, and use the fields to specify the backup image that you want.


Restore Database - CDLIB

GINGER1 - DB2 - CDLIB

Backup Image | Table Spaces | Containers | RollForward | Options

Manually enter the backup image information [✓ Select an entry from the list]

Backup type	Date	Time	Table spaces
Database	11/18/1997	14.08.17	5

Change the media information if necessary:

Media type: Directories or tapes

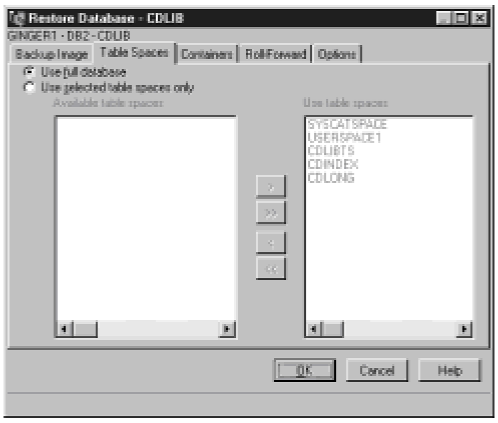
List of directories or tapes: I:\BACKUPDB

Figure 12.14: The Restore Database notebook's Backup Image page.

Also, verify that the media type displayed is correct. If it isn't, select the correct type from the Media Type drop-down list.

Specifying Table Spaces

On the Table Spaces page (see Figure 12.15), you specify whether you want to restore a subset of the table spaces in the databases. Select Use Full Database (the default) to restore all the table spaces in the database to the state they were in at the time that the backup image was taken. Select Use Selected Table Spaces Only to restore only those table spaces that you selected in the Available Table Spaces box.


Restore Database - CDLIB

GINGER1 - DB2 - CDLIB

Backup Image | Table Spaces | Containers | RollForward | Options

[✓ Use full database] Use selected table spaces only

Available table spaces:

Use table spaces:
SYSCATSPACE
USERSPACE1
CDLIBTS
CDINDEX
CDLONG

Figure 12.15: The Restore Database notebook's Table Spaces page.

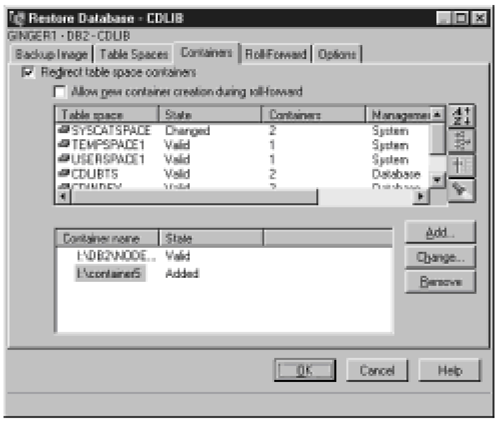
Note Restoring all table spaces isn't the same as performing a full restore of the database. When you restore a full database, the following also occurs:

- The recovery history file for the database is restored.
- The database configuration parameter values are restored to their state at the time of the backup.

You can restore table spaces while the database is active. Performing a full restore requires exclusive access to the database.

Modifying Containers

On the Containers page (see Figure 12.16), you can specify that you want to modify containers during the restore operation. To do so, click the Redirect Table Space Containers check box, and then click Add to create a new container, click Change to change the path of a selected container, or click Remove to remove a selected container. This operation of modifying containers during a restore is known as a *redirected restore*.



[✓] Redirect table space containers

[ ] Allow new container creation during roll-forward

Table space	State	Containers	Management
SYSCATSPACE	Changed	2	System
TEMPSPACE1	Valid	1	System
USERSPACE1	Valid	1	System
CDLIBTS	Valid	2	Database

Container name	State
I:\DB2\NODE...	Valid
I:\container5	Added

Figure 12.16: The Restore Database notebook's Containers page.

During a backup of a database or a table space, a record is kept of all table space containers in use by the backed-up table spaces. During a restore, all containers listed in the backup are checked to see whether they now exist and are accessible. If one or more of the containers is inaccessible because of a media failure (or for any other reason), the restore will fail. To allow a restore in such a case, the redirecting of table space containers is supported during the restore. This support includes adding, changing, or removing table space containers.

You might want to restore a database on a system that wasn't used to create the backup. Any directory or file containers are automatically created on the new system if they don't exist. No redirection is necessary.

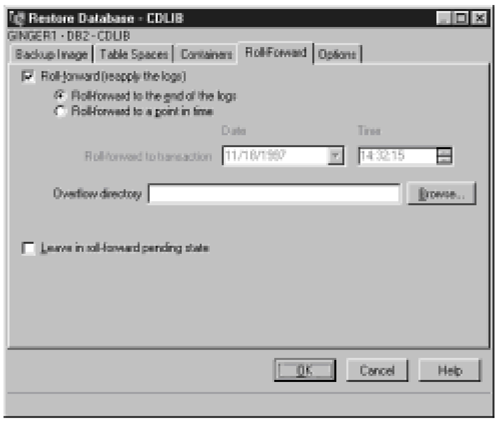
You can't add containers to an SMS database, but if you use a redirected restore, you can effectively accomplish this. You can also increase the size of DMS table space containers the same way.

You can't perform a restore operation if a container is in the `not valid` state (a container might be invalid because of a media failure). In this situation, you must modify the container. You can change the container's path or remove the container. If you want, you can perform an add operation to replace the deleted container.

To display the containers not in valid state (if you can't access data stored in a table space container, it's invalid), select the table space whose state isn't valid. That table space's containers are displayed in the Container Name box along with their states. Select the container you want to modify, and then specify the container modification.

Specifying a Roll-Forward Recovery

On the Roll Forward page (see Figure 12.17), you can specify that you want to perform roll-forward recovery on the database as part of the restore operation. The roll-forward operation checks for logs in the location specified by the `LOGPATH` database configuration parameter. If you've moved any logs from the specified location, use the Overflow Directory text box to specify the logs' new location. To display the list of available directories, click Browse.


Restore Database - CDLIB
GINGER1 - DB2 - CDLIB

[✓] Roll-forward (reapply the logs)
(●) Rollforward to the end of the logs
( ) Rollforward to a point in time
Rollforward to transaction — Date: 11/18/1997 — Time: 14:32:15
Overflow directory:
[ ] Leave in roll-forward pending state

Figure 12.17: The Restore Database notebook's Roll Forward page.

Note To perform roll-forward recovery, you must have the database logs, and the backup image that you use to restore the database must have been made when the database was enabled for forward recovery. (`LOGRETAIN` must be set to `ON` when the backup image is created.)

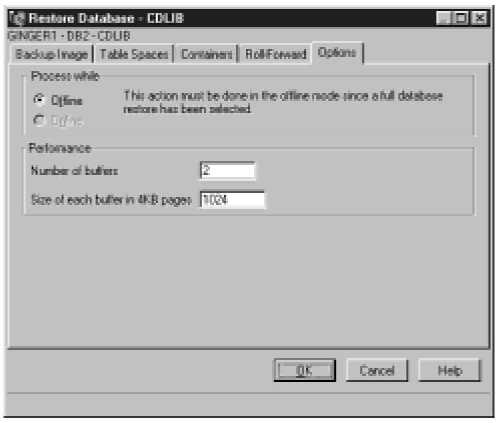
To indicate that you want to perform roll-forward recovery, select the Roll Forward check box. To have all the changes recorded in the log reapplied to the database, select the Roll Forward To The End Of The Logs check box. To have the changes in the logs up to a specified date and time reapplied to the database, select the Roll Forward To A Point In Time check box.

To make the database usable immediately after the restore operation, clear the Leave In Roll-Forward Pending State check box. (By default, this check box is selected.)

More information on rolling data forward appears later in the section "Rolling Data Forward."

Specifying Table Spaces

On the Options page (see Figure 12.18), you can set options used for the restore operation. Select Offline to have the restore database operation occur when no other application or user is connected to the database. Select Online to have the restore databaseoperation occur while other applications and users are connected to the database.



Process while

[✓ Offline] Online

This action must be done in the offline mode since a full database restore has been selected.

Performance

Number of buffers: 2

Size of each buffer in 4KB pages: 1024

Figure 12.18: The Restore Database notebook's Options page.

In the Number Of Buffers text box, type the number of buffers used for the restore operation. The default is 2; the valid range is from 1 to 65535. The more buffers you have, the faster the restore and the more memory used.

In the Size Of Each Buffer In 4KB Pages text box, type the size of each buffer in 4KB pages. The default is 1024.

Restoring to a New Database

If you want to restore a backup image of the database to a new database, right-click the database you want to restore, and select Restore From New from the pop-up menu. Follow the same instructions as described in the preceding sections, except enter a new name for the database on the Description page.

Rolling Data Forward

One aspect unique to database backups is the necessity to consider the database logs. If a database needs to be restored to a point beyond the last full, offline backup, logs are required to roll the data forward to the point of failure.

Rolling data forward refers to the log journal information, which is applied against a restored database. Log journals contain the changes made to the database since the last backup. After these journals are applied, the database will be in the same state it was in before the failure. In the case of a machine or site disaster in which the logs are destroyed, the database can be restored only to the point of the last full backup and only if the backup media was stored at a location unaffected by the disaster.

When you first create a database, only circular logging is enabled for it. This means that logs are reused in a circular fashion and aren't saved or archived. With circular logging, roll-forward recovery isn't possible; only crash recovery or a restore of the database to the time of the last backup is enabled.

When log archiving is performed, roll-forward recovery is possible because logs record changes to the database after the time that the database backup was taken. You perform log archiving by activating the `LOGRETAIN` database configuration parameters.

Roll-forward recovery reapplies the completed units of work recorded in the logs to the restored database or table space(s). You can specify that the roll-forward recovery is to the end of the logs or to a particular point in time.

Roll-forward recovery builds on a restored database and enables you to restore a database to a particular time after the database backup was taken.

With roll-forward recovery enabled, you can take advantage of online backup. For full database roll-forward recovery, you can choose to recover to the end of the logs or to a specified point in time. For instance, you could start with a restored copy of the database and roll forward changes until just before that application started. All units of work in the logs after the specified time won't be reapplied.

Before you can do forward recovery, the database must already be in *roll-forward pending state*, and you must have the logs. A database is in roll-forward pending state following the completion of a full database restore. When these conditions are satisfied, you can perform the roll-forward operation by using one of three methods: the Roll Forward page of the Restore Database SmartGuide (discussed earlier in the section "Using the Restore Database SmartGuide"), the Roll Forward page of the Restore Database notebook (discussed earlier in the section "Specifying a Roll-Forward Recovery"), or the Roll-Forward Database notebook. These methods give you the flexibility to perform the roll-forward operation when it makes sense for you; each method accomplishes the same task.

To use the Roll-Forward Database notebook, follow these steps:

1. Start the Control Center.

2. Expand the objects until you see the database you want to work with—for example, CDLIB.

3. Right-click the CDLIB database, and select Roll Forward from the pop-up menu.

4. On the Roll Forward page of the Roll Forward Database notebook, select to roll forward to the end of the logs or to a specified point in time:

   - Use Roll Forward To The End Of The Logs (the default) to have all the changes recorded in the logs applied to the database.

   - Select Roll Forward To A Point In Time to have changes rolled forward to a specific date and time. The Roll Forward To Transaction date and time text boxes correspond to the last recorded transaction to reapply. If you don't specify a time and date, the default is the current date and time.

5. The roll-forward operation checks for logs in the location specified in the `LOGPATH` database configuration parameter. If you've moved any logs from this location, use the Overflow Directory text box to indicate the new location for the logs.

6. Select the Leave In Roll-Forward Pending State check box (the default) to leave the database in a state in which you can continue to roll forward transactions. To turn off roll-forward pending, perform additional roll-forward recovery, or select the Stop Roll Forward from the pop-up menu of the database.

7. Click OK to have the logs rolled forward.

Recovery History File

A recovery history file is created with each database and is automatically updated whenever there is any of the following:

- A database or table space backup

- A database or table space restore

- A database or table space roll forward
- A table space quiesce
- A load of a table

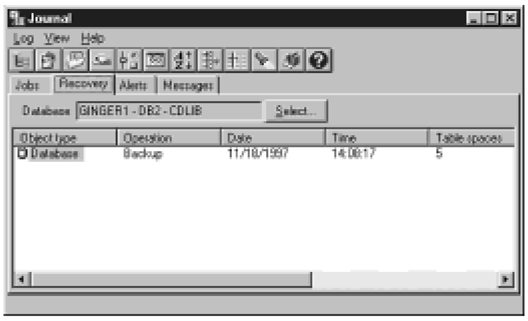
The file contains a summary of the backup information to be used in case all or part of the database must be recovered to a given point in time. The information in the file includes

- The copied part of the database and how it was copied
- The time the copy was made
- The location of the copy
- The last time a restore was done

Every backup operation includes a copy of the recovery history file, which is linkedto the database. Dropping a database deletes the recovery history file. Restoring data doesn't overwrite an existing history recovery file.

If the current database is unusable or unavailable and the associated history file is damaged or deleted, you can follow these steps to restore the file:

1. Open the Journal by clicking its icon on the Control Center's tool bar or by selecting Start | Programs | DB2 for Windows NT | Administration Tools | Journal.
2. Click the Recovery tab to see the recovery history files (see Figure 12.19).



Database GINGER1 - DB2 - CDLIB

Object type	Operation	Date	Time	Table spaces
Database	Backup	11/18/1997	14:08:17	5

Figure 12.19: The Journal's Recovery page.

3. Click the Select button, and choose an instance and database from the list. Click OK to return to the Recovery page to see entries related to the backups performed on this database.
4. Right-click an entry, and select Restore Recovery Histories from the pop-up menu.
5. In the Restore Database Recovery History notebook, select an entry, and click OK to have the database recovered by using the backup image you selected.

The recovery history file helps you locate information when you've performed a backup. For example, it can help you determine the location of a backup or in which backup a DB2 object can be found. If a backup is moved to a different medium—say, from disk to tape—this file can be updated to keep track of the new location of the backup.

Every DB2 backup contains a copy of this file, and it can be restored from any backup. If you choose to restore it, use caution to avoid overwriting the database's existing his-tory file.

Note Although the recovery history file is an ASCII file, manually editing the file should be done only at your own risk and isn't recommended.

To view the recovery history file, follow these steps:

1. Open the Journal.

2. Click the Recovery tab to see the recovery history files.

3. Click the Select button, and choose an instance and database from the list. Click OK to return to the Recovery page to see entries related to the backups performed on this database.

4. If your database hasn't been backed up, no recovery history file will be associated with it. If your database has been backed up, you'll see an entry in the table. Scroll to the right to see the information captured for this database.

Log files and the recovery history file are created automatically when a database is created. You can't modify a log file or the recover history file. However, they are important, should you need to use your database backup to recover lost or damaged data.

Summary

In today's lesson, you saw that creating a backup and recovery plan is an important way that you can protect the data in your database. DB2 enables you to save all transaction information into logs, which means that if you do need to recover your data, you can recover right to the point of failure.

Your system is set up for circular logging by default. With this type of logging, you can recover data only to the last full backup you performed. If you need more complete recovery, you must change the default from circular logging to archive logging, which enables you to fully recover your data to the point of failure.

If you're faced with a media failure and have kept backups of your data and logs of the transactions, you can use the Restore Database SmartGuide or the Restore Database notebook to fully recover your data.

Q&A

Q What is *roll-forward pending state?*

A If your database is enabled for archive logging and you use a backup image to restore the data, your database is left in roll-forward pending state. This means that the data has been restored to the level of the backup, but recorded transactions in the log should be applied if you want to recover to the most recently saved transaction. To remove the database from roll-forward pending state, you must use the Roll Forward tool (launched through the Roll-Forward Database notebook, the Roll Forward page of the Restore Database SmartGuide, and the Roll Forward page of the Restore Database notebook) to restore the data in the logs to a point in time or to the end of the logs.

Q What two things must I do to enable roll-forward recovery?

A To enable roll-forward recovery, you must change the default logging from circular to archive. You can use the Backup Database SmartGuide to make this change, or you can use the Control Center to set the `LOGARCHIVE` database configuration parameter to `YES`. After you enable archive logging, you must make a full offline backup of your data.

Q Can I change the plan created when using the Backup Database SmartGuide?

A When you create a plan by using the Backup Database SmartGuide, you can change when the backup is scheduled to occur but nothing else. If you want to revise your plan in ways other than the schedule, you must delete the current plan from the Script Center and rerun the SmartGuide to create a new plan.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. How can you ensure that you have exclusive access to a database if you want to do a full offline backup of your data?

2. In what cases are you required to use the Backup Database notebook instead of the Backup Database SmartGuide?

3. How does using redirected restore give you additional flexibility when using SMS table spaces?

4. How can you use roll-forward recovery to undo an unexpected error?

Exercise

Backing up your data is one of the most important things you can do. Of course, at this point you might not have much data to be concerned about. Take this opportunity to explore the many backup and restore options that you have, and make an extra effort to understand the power of the database logs. Use any databases on your system (CDLIB, SAMPLE, or your own) for experimentation.

Day 13: Moving Data

Overview

Today, you see how to move data in and out of a DB2 database by importing, loading, or exporting:

- If you want to create a new database, replace data in an existing database, or append to the data in an existing database with data from another source, you use the import utility.

- If you have a substantial amount of data that you need to move into your database, you should consider using the load utility instead of the import utility.

- If you need to move data from a DB2 database to another source, you use the export utility.

The import and load utilities provide roughly the same function, but the load utility performs better when very large amounts of data are involved. All three utilities support several different file formats, giving you the flexibility to choose the format that best suits your environment.

To see how the export, import, and load utilities work, you're going to export the data from the `ORG` table in the SAMPLE database in two formats: IXF and DEL. When you have these files, you'll import the data in the IXF file into a new table. You'll first modify the data in the DEL file and then load it into a table.

Exporting Data

You can export data from DB2 and use it to import into other programs. To export data from DB2, you use the Export notebook, which is launched through the Control Center. The discussion in this section covers how to use the Export notebook to export data from DB2 into one of these supported formats:

- Nondelimited ASCII format (ASC)

- Delimited ASCII format (DEL)

- Worksheet format (WSF)

- Integrated exchange format (IXF)

Nondelimited ASCII format (*ASC*) is a stream of ASCII characters consisting of data values organized by row and column. Rows are separated on new lines, and each column is defined by a beginning and ending position. The data type of a given column in the ASC file is determined by the type of the target column in the table. This file format can be used for data exchange with applications, including word processors that create flat text files with aligned column data.

A *delimited ASCII format* (*DEL*) file is a sequential ASCII file, with each row appearing on a separate line. The columns are separated by a delimiter, such as a comma. Each character string is contained within double quotes (`"`). This file format is commonly used to exchange data with various products that might be using different column delimiters.

Worksheet format (*WSF*) is generated or supported by products such as Lotus 1-2-3 and Symphony. Each WSF file represents one worksheet. DB2 uses the following conventions to interpret worksheets:

- Cell values under any column heading are values for that particular column or field.
- Cells in the first row are reserved for descriptive information about the worksheet. This information is optional and is ignored during import.
- Cells in the second row are used for column labels.
- The remaining rows are data rows.

Integrated exchange format (*IXF*) data interchange architecture is a host file format designed to enable the exchange of relational database structure and data. On the workstation, this file format is PC/IXF and is an adaptation of the host IXF format. Use this format when exchanging data between the various versions of DB2.

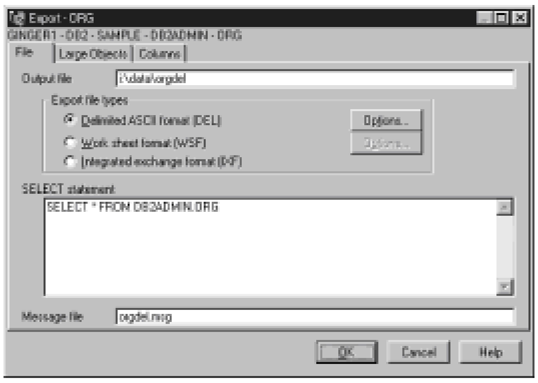
When you choose an output file for the exported data, make careful notes of what options you select.

> **Note** You can also export data from other sources to load into DB2. These sources include applications such as Lotus Approach, Lotus 1-2-3, Microsoft Access, flat files on your host system, or DB2 databases on the host. To export data from a source such as Lotus Approach or 1-2-3, you use the export function available with these tools and choose a format that DB2 can import. Later, the section "Importing and Loading Data" explains how to use the data you exported from another program.

Performing a Simple Export

To export data from DB2, follow these steps:

1. Start the Control Center.
2. Expand the objects until you see the Tables folder within the SAMPLE database. Click this folder to see a list of all the tables in the right pane.
3. Right-click the `ORG` table, and select Export from the pop-up menu. The Export notebook opens to the File page, in which you type the path and name of the file that will receive the data (see Figure 13.1). Export the data to a file—for example, `i:\data\orgdel`.



GINGER1 - DB2 - SAMPLE - DB2ADMIN - ORG

Output file: i:\data\orgdel

Export file types
- [✓ Delimited ASCII format (DEL)]
- Work sheet format (WSF)
- Integrated exchange format (IXF)

SELECT statement:

```
SELECT * FROM DB2ADMIN.ORG
```

Message file: orgdel.msg

Figure 13.1: The Export notebook's File page.

If you don't specify a path, the current working directory is used. If you specify a path, the directory must exist. If the file already exists, its contents are overwritten with the exported data.

4. Choose a format to use for the export: Delimited ASCII Format (DEL), Worksheet Format (WSF), or Integrated Exchange Format (IXF). For this example, use DEL.

5. If you choose DEL or WSF, you have additional options that you might want to set:

   - For DEL files, select the delimiter characters for columns, character strings, and decimal points.

   - For WSF files, enter the version and release of the Lotus product to which you're exporting the data.

   For this example, leave the options at their default values, and export all the data.

6. Near the bottom of the File page, you'll see the `SELECT` statement that will be used to extract data from the table. By default, all data in the table is extracted. You can add `WHERE` clauses if you want only a subset of the data. For example, if you want to export only the rows that have `Eastern` as a division, enter a SQL statement as follows:

   ```
   select * from org where division = 'Eastern'
   ```

   Because three rows in the `ORG` table have `Eastern` as a division, three rows are exported.

7. The last piece of information you need to add is the location of a message file. As soon as you type a path and filename for the message file, the OK button is activated. (You can't continue until a filename is entered in this field.) For this example, choose a name such as `ORGDEL.MSG` for the message file.

8. Click OK to export the data.

Repeat these steps to export the same data in IXF format. Store the exported data in a file—for example, `i:\data\orgixf`; you'll import this data later in this lesson.

Exporting Large Objects (LOBs)

If you want to export tables that include large objects, such as audio or video clips, use the Export notebook's Large Objects page to arrange to have the large objects stored in separate files (see Figure 13.2). You need to click the Store Large Objects (LOBs) In Separate Files (LOBSINFILE) check box before the options on this page becomeavailable.

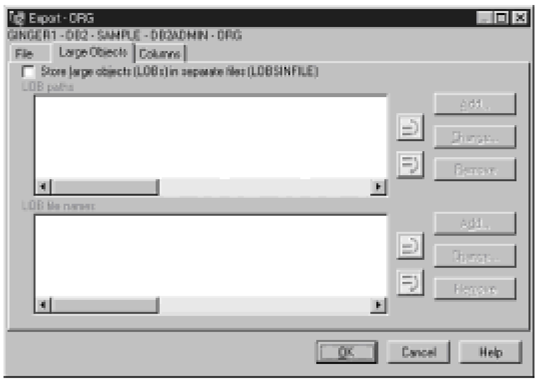
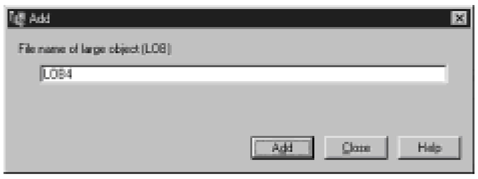


Figure 13.2: The Export notebook's Large Objects page.

Click the Add button next to the LOB Paths box. Type the path to the LOB file, click Add to insert the path into the LOB Paths box, and then click Close to return to the Export notebook. Click Add next to the LOB File Names box. Type the name of the file that will store the LOBs; don't use a file extension for the filename because a three-digit sequence number is appended to the file (see Figure 13.3). Click the Add button to add the filename to the list. Repeat this operation for each file you plan to use to store LOBs. Click Close to return to the Large Objects page in the Export notebook.



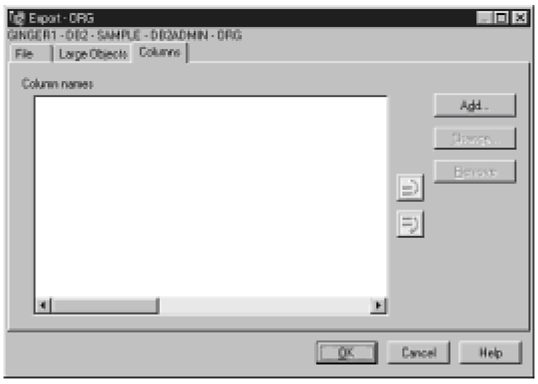
File name of large object (LOB)

LOB4

Figure 13.3: Inserting LOB filenames.

Specifying Column Names

If you want to specify the column names used in the export file, use the notebook's Columns page (see Figure 13.4). Click the Add button to open the Add Column Name dialog. In the Column Name box, type the name of the column you want to add. Click the Add button. The column name appears in the Column Names list, and the Add window remains open. Repeat this step to add as many column names as you want, and then click Close to return to the Columns page in the Export notebook.



GINGER1 - DB2 - SAMPLE - DB2ADMIN - ORG

Column names

Figure 13.4: The Export notebook's Columns page.

Note This option doesn't reorder the columns or let you export a subset of data. It only enables you to change the column name that will be used when importing or loading the data. For example, you might want to change the column names of a table from `COL1`, `COL2`, and `COL3` to `EMPID`, `FIRST`, and `LAST` to better match the data stored in the column.

Note You can specify column names only for the WSF and IXF formats. If you don't add column names to the list box on the Columns page, the column names in the existing table are used.

Importing and Loading Data

DB2 provides import and load utilities to help you move data into a table from existing sources. Both utilities enable you to load data into tables, but there are some differences:

- To create a new table with the data you are inserting, you must use the import utility. The load utility doesn't allow you to create a new table but does enable you to append to or replace the data in an existing table.
- To replace or add data to an existing table, you must use the import utility.
- The load utility is usually faster than the import utility and is well suited for inserting very large amounts into the table. Quite often, tables of data from host database systems are loaded into DB2 databases with the load utility.
- The WSF format is supported with the import utility but not with the load utility.

Importing Data from Files

DB2's import utility takes data from an input file and inserts it into a table or a view. An input file contains data extracted from an existing source of data, such as a Lotus 1-2-3 or ASCII file.

DB2 imports file formats generated from the same sources as supported by the Export notebook: nondelimited ASCII format (ASC), delimited ASCII format (DEL), worksheet format (WSF), and integrated exchange format (IXF). The procedure for generating each file varies with the original source.

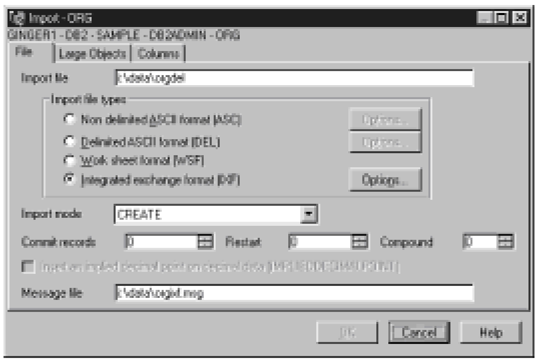
Tip For DEL, WSF, and ASC data file formats, define the table, its columns, and data types before importing the file. The import utility accepts data with minor incompatibility problems, including character data imported with possible padding or truncations, and numeric data imported into different types of numeric fields.

Note You can create a new database only if IXF format is used. Other formats must be imported into existing tables.

When you have an input file in a supported format, use the Import notebook to insert data from the file into an existing table. If the table already contains data, you can replace it or append to the existing data with the data in the file. You can also use the Import notebook to create and populate a new table from an input file or delete existing rows in the selected table and repopulate it by using data from the input file.

Now it's time to actually import some data. You'll use the data that you exported in IXF format in the preceding discussion and create a new table called `NEWORG`. First, you need to open the Import notebook:

1. Start the Control Center.
2. Expand the objects until you see the Tables folder within the SAMPLE database. Select that folder to see a list of tables in the right pane.
3. Right-click any table, and select Import from the pop-up menu. The Import notebook opens (see Figure 13.5).


Import - ORG
GINGER1 - DB2 - SAMPLE - DB2ADMIN - ORG
File | Large Objects | Columns

Import file: i:\data\orgdel

Import file types:
- ( ) Non delimited ASCII format (ASC)
- ( ) Delimited ASCII format (DEL)
- ( ) Work sheet format (WSF)
- (●) Integrated exchange format (IXF)

Import mode: CREATE
Commit records: 0 | Restart: 0 | Compound: 0
☐ Insert an implied decimal point on decimal data (IMPLIEDDECIMAL)
Message file: i:\data\orgixf.msg

Figure 13.5: The Import notebook.

To continue the import process, follow along through the Import notebook page by page, option by option.

Specifying the Path, Import Format, and Other Settings

On the File page, type the path and name of the file that contains the data you want to import—for example, `i:\data\orgixf`. If you don't specify the full path, the current working directory is used.

Next, select a format to use for the import (ASC, DEL, WSF, or IXF). Your choice depends on the format used to export the data, as discussed earlier. To import the IXF data that you exported in the preceding section, choose IXF.

ASC, DEL, and IXF have additional options that you might want to set. Click the Options button to set these options, and then click OK to return to the Import notebook. Because you selected IXF as the format for this example, the following IXF options are available:

- Select the Accept Code Page Mismatches And Suppress Translation check box to accept data from the input file, even if code page mismatches occur, and to suppress translation between code pages.

- Select the Drop Existing Indexes And Create New Ones check box to drop all indexes defined in the table that you selected and to create new ones from the index definitions in the input file. You can use this option only when a table's contents are being replaced—not with a view nor when columns are inserted.

- Select the Load Each Row Without Checking Target Length check box to specify that target fields aren't to be checked and that an attempt be made to load each row. (By default, target fields for fixed-length fields in IXF files are checked to verify that they're large enough for the data.)

- Select the Index Schema check box to specify the schema of the table you're creating. By default, the schema used is the same as the username you used to log in.

The ASC options are as follows:

- Ignore the Do Not Recognize x'1A' As EOF Character option because it's for OS/2 only.

- Select the Truncate Trailing Blank Spaces When Loading Into Variable-Length Database Field check box to drop trailing blanks after the last nonblank character when loading the data into a variable-length table column.

- Select the Pad With 0x00 Characters When Loading Into Variable-Length Database Field check box to add the characters `0x00` after the last nonblank character when loading the data into a variable-length table column.

- Select the Record Length In Characters check box to specify the number of characters to be read for each row.

The DEL options are as follows:

- Select the delimiter characters to use for columns, character strings, and decimal points. All these choices are optional.

- Ignore the Do Not Recognize x'1A' As EOF Character option because it's for OS/2 only.

- Choose the import mode that you want to use. For this example, select `CREATE` (available only for IXF input files) to create the table definition and the row contents. If the data was exported from a database manager, indexes are also created. The other modes available are as follows:

  - `INSERT` adds the imported data to the table without changing the existing table data.

  - `INSERT_UPDATE` adds the rows of the imported data to the target table or updates existing rows of the target table with matching primary keys.

  - `REPLACE` can be used only if the table exists. Select it to delete all existing data in the table and insert the imported data. The table and index definitions aren't changed.

  - `REPLACE_CREATE`, available only for IXF input files, deletes all existing data in the table and inserts the imported data without changing the table or index definitions. If the table doesn't exist, this mode creates the table definition and row contents. If the data was exported from a database manager, indexes are also created.

In the Commit Records box, specify how often the changes are committed. If you specify `1`, the changes to the target table will be committed each time a record is inserted. This reduces the number of lost records if a failure occurs during the import. For this example, specify `1` to have each record committed as it's inserted. This field has no maximum, but if the number is too large, you run the risk of losing data if any kind of failure occurs during the import.

In the Restart box, specify the number of records in the file to skip before the import begins. If an error occurs during an import, you can specify this information to restart the import operation immediately following the last row that was successfully imported and committed.

In the Compound box, specify whether compound SQL will be used. Compound SQL improves import performance by grouping SQL statements into a block; this option can reduce network overhead and improve response time. You can specify any number between 1 and 100. The specified number indicates how many statements are grouped each time. Because you're importing the data into a local database, choose `1`, which means that one statement will be processed at a time.

Select the Insert An Implied Decimal Point On The Decimal Data check box if you want the column definition to determine the location of an implied decimal point. For example, the value `54321` is imported into a `DECIMAL (6,2)` column as `543.21`, not `54321.00`. This option is available only if you're using DEL or ASC file formats.

In the Message File text box, type the path and name of the file that will contain the warning and error messages that occur during import. If you don't specify a path, the current working directory is used—for example, `i:\data\orgixf.msg`.

Note You can't start the import until a message file is specified.

Click OK to begin the import operation. You should check the message file to see whether there were any problems. Use an editor of your choice to view the file, `c:\data\orgixf.msg`, which is specified on the File page of the Import notebook.

Importing Tables with Large Objects

If you're importing tables that include large objects such as audio or video clips, use the Large Objects page (see Figure 13.6). Select the Retrieve Large Objects (LOBs) In Separate Files (LOBSINFILE) check box to retrieve large objects stored in separate files and to activate the rest of the options on this page. Click Add; in the dialog box, type the path of the LOB files. Click the Add button to add the path to the list. Repeat this for as many entries as needed. The paths in this list are searched in the order in which they appear in the LOB paths list box. Click Close to return to the Large Objects page of the Import notebook.

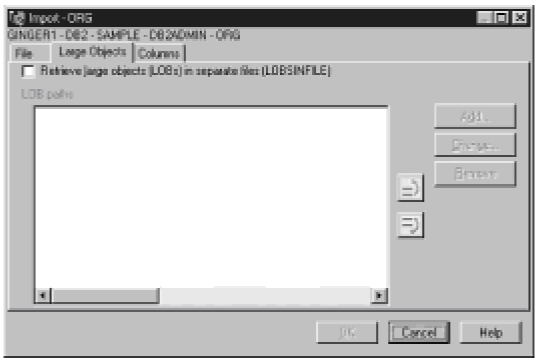

Import - ORG
GINGER1 - DB2 - SAMPLE - DB2ADMIN - ORG
☐ Retrieve large objects (LOBs) in separate files (LOBSINFILE)

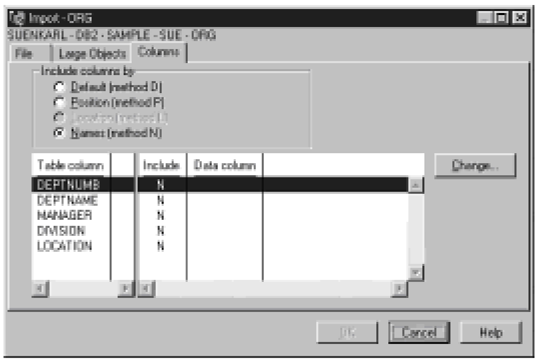
Figure 13.6: The Import notebook's Large Objects page.

Importing Table Subsets

If you're importing a table subset, click the Columns tab to specify column options. In the Include Columns By section, select the appropriate option:

- *Default (Method D)* imports columns from the input file to the target table, using a one-to-one mapping. (The first column in the input file is imported into the first column of the target table, and so on.) If the input file contains more columns than the target table, the additional columns at the end of the input file are dropped.
- Select *Position (Method P)* to import columns from the input file to the target table, using the column's position as the mapping method. Each data column is specified by the number of its position in the input file (for example, `1` for the first column). This option is supported only for IXF or DEL file format.
- Select *Location (Method L)* to import columns from nondelimited ASCII format (ASC) input files to the target table. The starting and ending locations of the data column in the input file are used as the mapping method.
- Select *Names (Method N)* to import columns from the input file to the target table, using column names as the mapping method. This option is supported only for the IXF file format.

If you choose the Position, Location, or Names method, select a column in the Table Column list (see Figure 13.7), and click the Change button. In the Change dialog box, you can include or exclude the column during the import.



Include columns by
- ( ) Default (method D)
- ( ) Position (method P)
- ( ) Location (method L)
- (•) Names (method N)

Table column	Include	Data column
DEPTNUMB	N	
DEPTNAME	N	
MANAGER	N	
DIVISION	N	
LOCATION	N	

Figure 13.7: The Import notebook's Columns page, showing selected columns.

Loading Data from Files, Tapes, or Named Pipes

The Load notebook loads data directly into a DB2 table from one or more files, tapes, or named pipes. (For example, you might need to move in data from databases other than DB2.) During load processing, indexes can be built, primary keys validated, and statistics generated, all without requiring secondary passes through the data.

The Load notebook is faster than the Import notebook but has some functional differences. The Load notebook is intended for bulk loading of new tables or appending large amounts of data to existing tables. It's common to use the load utility when moving large amounts of data from DB2 on the mainframe to DB2 systems on the LAN. Load is restartable and recoverable. If a failure occurs while you're loading data, you have the option of continuing the load without starting from the beginning.

If you're using the Load notebook's `INSERT` mode, for an initial large load of data, create the unique key after loading the data. This avoids the overhead of maintaining the index while the table is being loaded. It also results in the index using the least amount of storage.

If you're using the Load notebook's `REPLACE` mode, create the unique key before loading the data. In this case, creating the index during the load is more efficient than after the load.

When loading a small amount of data to an existing table, you can improve performance if you use the Import notebook instead of the `LOAD INSERT` mode.

To see how the Load notebook works, you'll modify the `ORGDEL` file you created when exporting the `ORG` table in DEL format and load it into the table you created during the import.

Tip You must have an existing table in which to load the data. You can't use the Load notebook to create a new table.

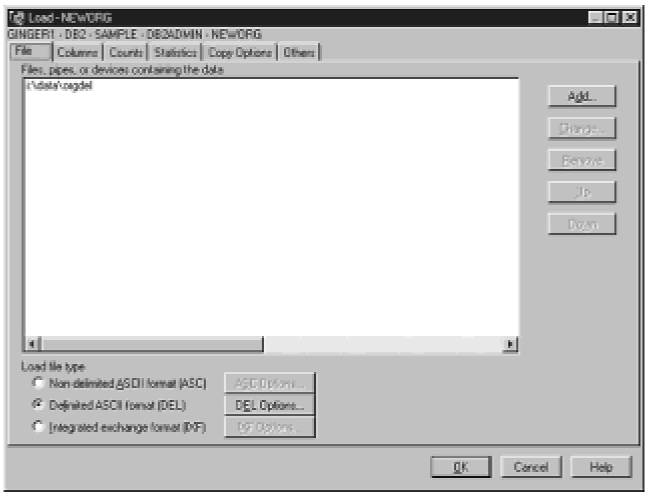
Change some of the data in the `ORGDEL` file by using Notepad. For example, change the cities listed to other cities, such as `New York` to `Toronto`.

To load data from the modified `ORGDEL` file into the `NEWORG` table (created during the import step), follow these steps:

1. Start the Control Center.

2. Expand the objects until you see the Tables folder within the SAMPLE database. Click that folder to see a list of all the tables in the right pane.

3. Right-click the table you want to load data into. (For this example, choose the `NEWORG` table that you created during the Import steps.) Select Load from the pop-up menu. The Load notebook opens, as shown in Figure 13.8.



Files, pipes, or devices containing the data

c:\data\orgdel

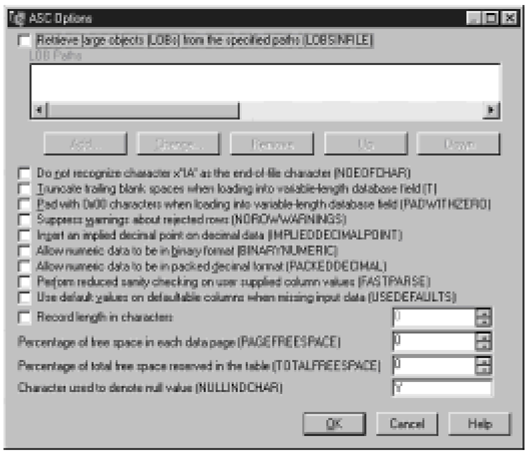
Load file type

( ) Non-delimited ASCII format (ASC)
(●) Delimited ASCII format (DEL)
( ) Integrated exchange format (IXF)

Figure 13.8: The Load notebook.

4. On the File page, click the Add button.

5. In the Add window, specify locations such as files, pipes, devices, LOB paths, or directories (see Figure 13.9). You should fully qualify the filename—for example, if you want to load data from the file `c:\data\orgdel`, type the entire name in the text box.



☐ Retrieve large objects (LOBs) from the specified paths (LOBSINFILE)
☐ Do not recognize character x'1A' as the end-of-file character (NOEOFCHAR)
☐ Truncate trailing blank spaces when loading into variable-length database field (T)
☐ Pad with 0x00 characters when loading into variable-length database field (PADWITHZERO)
☐ Suppress warnings about rejected rows (NOROWWARNINGS)
☐ Insert an implied decimal point on decimal data (IMPLIEDDECIMALPOINT)
☐ Allow numeric data to be in binary format (BINARYNUMERIC)
☐ Allow numeric data to be in packed decimal format (PACKEDDECIMAL)
☐ Perform reduced sanity checking on user supplied column values (FASTPARSE)
☐ Use default values on defaultable columns when missing input data (USEDEFAULTS)
☐ Record length in characters 0

Percentage of free space in each data page (PAGEFREESPACE) 0
Percentage of total free space reserved in the table (TOTALFREESPACE) 0
Character used to denote null value (NULLINDCHAR) Y

Figure 13.9: The Load notebook's Add window.

6. Choose a radio button that corresponds to the file type of the load input file. Your choices are ASC, DEL, and IXF. (For this example, you exported the data in DEL, so use this format.) See the earlier section "Importing Data from Files" for definitions of each type.

7. To load the data by using the default values, click OK.

The rest of the steps are optional and are covered in the following sections. After you complete all the pages of the Load notebook, click OK to begin the load operation. You don't have to set each option on the pages. As soon as the required information is entered, the OK button becomes active.

Setting Load Options

At the bottom of the Load notebook's File page, you can click the Options button for the load file type you've chosen and select the options you need. Then click OK to return to the File page.

The following options are available for all three file types:

- Retrieve Large Objects (LOBs) From The Specified Paths
- Suppress Warning About Rejected Rows
- Use Default Values On Defaultable Columns When Missing Input Data
- Percentage Of Free Space In Each Data Page
- Percentage Of Total Free Space Reserved In The Table

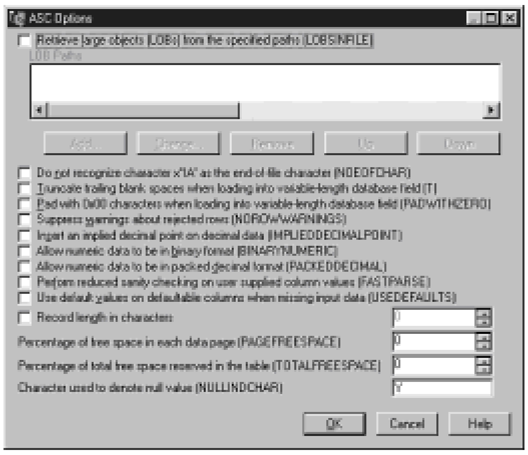
If you choose the ASC format, you can also set the following options (see Figure 13.10):

- Do Not Recognize Character x'1A' As The End-Of-File Character (ignore this option because it's for OS/2 only)
- Truncate Trailing Blank Spaces When Loading Into Variable-Length Database Fields
- Pad With 0x00 Characters When Loading Into Variable-Length Database Field
- Insert An Implied Decimal Point On Decimal Data
- Allow Numeric Data To Be In Binary Format
- Allow Numeric Data To Be Packed Decimal Format
- Perform Reduced Sanity Checking On User Supplied Column Values
- Record Length In Characters
- Character Used To Denote Null Value

The unique options for the DEL file type are as follows (see Figure 13.11):

- Do Not Recognize Character x'1A' As The End-Of-File Character (ignore this option because it's for OS/2 only)
- Insert An Implied Decimal Point On Decimal Data
- Perform Reduced Sanity Checking On User Supplied Column Values

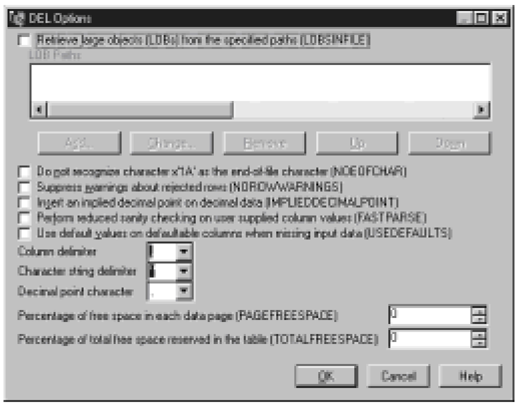
- Column Delimiter
- Character String Delimiter



☐ Retrieve large objects (LOBs) from the specified paths (LOBSINFILE)
LOB Paths

☐ Do not recognize character x'1A' as the end-of-file character (NOEOFCHAR)
☐ Truncate trailing blank spaces when loading into variable-length database field (T)
☐ Pad with 0x00 characters when loading into variable-length database field (PADWITHZERO)
☐ Suppress warnings about rejected rows (NOROWWARNINGS)
☐ Insert an implied decimal point on decimal data (IMPLIEDDECIMALPOINT)
☐ Allow numeric data to be in binary format (BINARYNUMERIC)
☐ Allow numeric data to be in packed decimal format (PACKEDDECIMAL)
☐ Perform reduced sanity checking on user supplied column values (FASTPARSE)
☐ Use default values on defaultable columns when missing input data (USEDEFAULTS)
☐ Record length in characters: 0
Percentage of free space in each data page (PAGEFREESPACE): 0
Percentage of total free space reserved in the table (TOTALFREESPACE): 0
Character used to denote null value (NULLINDCHAR): Y

Figure 13.10: The Load notebook's ASC options.



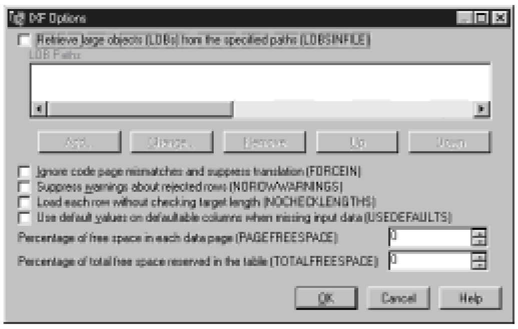
☐ Retrieve large objects (LOBs) from the specified paths (LOBSINFILE)
LOB Paths

☐ Do not recognize character x'1A' as the end-of-file character (NOEOFCHAR)
☐ Suppress warnings about rejected rows (NOROWWARNINGS)
☐ Insert an implied decimal point on decimal data (IMPLIEDDECIMALPOINT)
☐ Perform reduced sanity checking on user supplied column values (FASTPARSE)
☐ Use default values on defaultable columns when missing input data (USEDEFAULTS)
Column delimiter
Character string delimiter
Decimal point character: .
Percentage of free space in each data page (PAGEFREESPACE): 0
Percentage of total free space reserved in the table (TOTALFREESPACE): 0

Figure 13.11: The Load notebook's DEL options.

- Decimal Point Character

The unique options for the IXF file type are as follows (see Figure 13.12):



☐ Retrieve large objects (LOBs) from the specified paths (LOBSINFILE)
LOB Paths

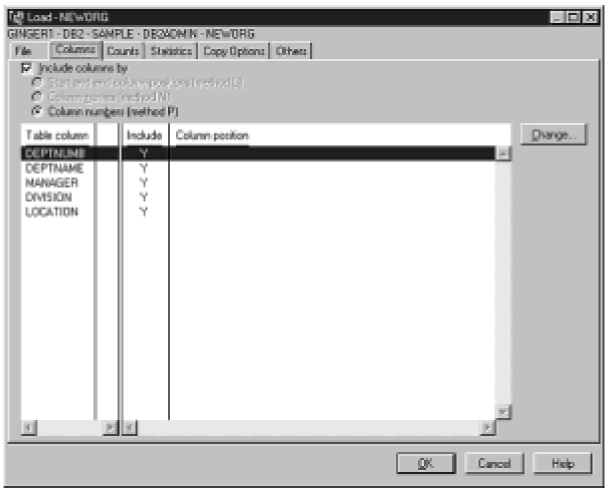
☐ Ignore code page mismatches and suppress translation (FORCEIN)
☐ Suppress warnings about rejected rows (NOROWWARNINGS)
☐ Load each row without checking target length (NOCHECKLENGTHS)
☐ Use default values on defaultable columns when missing input data (USEDEFAULTS)
Percentage of free space in each data page (PAGEFREESPACE): 0
Percentage of total free space reserved in the table (TOTALFREESPACE): 0

Figure 13.12: The Load notebook's IXF options.

- Ignore Code Page Mismatches And Suppress Translation
- Load Each Row Without Checking Target Length

Ordering Columns While Loading

Click the Columns tab (see Figure 13.13) to choose how you want the columns in the data files loaded into the table columns. Click the Include Columns By check box to activate the rest of the page. (For the example, load all the columns in the default order, so leave the Include Columns By check box deselected.)



[✓] Include columns by
( ) Start and end column positions (method L)
( ) Column names (method N)
[✓ Column numbers (method P)]

Table column		Include	Column position
DEPTNUMB		Y	
DEPTNAME		Y	
MANAGER		Y	
DIVISION		Y	
LOCATION		Y	

Figure 13.13: The Load notebook's Columns page.

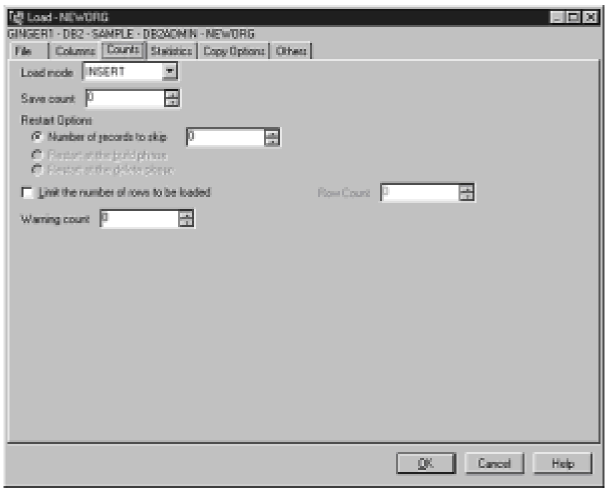
The Columns page contains different information, depending on the load file type you chose on the File page:

- **ASC.** Select a table column you want to load into your table, and click the Change button. Indicate which columns to include in the load, the column start and end positions, and a null indicator. The default for the null indicator is `0`, which means that there must always be a value for the column. Repeat this operation for each column you want to load.
- **DEL or IXF.** Column Numbers (Method P) is the only file-loading method available for the delimited ASCII format. Select a table column you want to load into your table, and click the Change button. Indicate whether you want the column included in the load, and designate the start position of the column. Repeat this operation for each column you want to load.
- **IXF.** Choose a file-loading method. Select the Column Names (Method N) radio button to import columns from the input file to the target table, using column names as the mapping method. Use Column Numbers (Method P) to import columns from the input file to the target table, using the column position as the mapping method. Each data column is specified by the number of its position in the input file (for example, `1` for the first column).

Select a table column you want to load into your table, and click the Change button. Indicate whether you want the column included in the load, as well as the column start position or column name. Repeat this operation for each column you want to load.

Setting a Load Mode, Save Count, and Other Options

Click the Counts tab to select a load mode, to set the save count, to choose restart options, and to limit the number of rows loaded (see Figure 13.14).



Load mode: INSERT

Save count: 0

Restart Options

(●) Number of records to skip: 0

( ) Restart at the build phase

( ) Restart at the delete phase

☐ Limit the number of rows to be loaded — Row Count: 0

Warning count: 0

Figure 13.14: The Load notebook's Counts page.

Use the Load Mode drop-down list to select a load mode:

- The `INSERT` mode is the default. Use it to add the imported data to the table after any existing table data.

- Select `REPLACE` to delete all existing data in the table and to insert the imported data. The table and index definitions aren't changed.

- Select `RESTART` to restart the load if the load is interrupted (the same load mode used before the interrupt is used when the load resumes).

  Note A `TERMINATE` option is also available but isn't recommended for general use; it should be selected only if an unrecoverable error occurred. This mode stops an interrupted load and moves the table spaces in which the table resides from load pending state to recovery pending state. The table spaces can't be used until a backup is restored and the table spaces are rolled forward.

In the Save Count text box, enter an interval (in number of rows) between consistency points. During the load process, after every *N* rows are loaded, a consistency point is established. The default value is `0`, meaning that no consistency point is established. If the value you select isn't high enough, the synchronization of activities performed at each consistency point affects performance.

Note Because a message is issued at each consistency point, you should specify a Save count only if the load is being monitored.

You can also specify where to restart the load process in case of an interruption:

- Select Number Of Records To Skip for the load operation to skip a specified number of records in the data file and then start the load at the next record. You can use this option with the load mode set to `INSERT`, `REPLACE`, or `RESTART`. If you select `RESTART`, you must use the number of rows at the last successful consistency point. The default is `0`, meaning the load will start at the first record.

- Select Restart At The Build Phase for the load to restart at this phase. The load process restarts, no additional rows are loaded, and the indexes are built for the rows already loaded. This option is available only when the load mode is set to `RESTART`.

- Select Restart At The Delete Phase for the load to restart at this phase. Rows that cause key violations are deleted from the table. Use this option if the message file states that the build phase completed and all temporary files are unmodified. This option is available only when the load mode is set to `RESTART`.

Select the Limit The Number Of Rows To Be Loaded check box to limit the number of physical records to be loaded to the value specified in the Row Count box. The default is `0`, meaning that the load will continue until all records are loaded or an unrecoverable error occurs.

Use the Warning Count box to indicate the number of warnings after which the load is stopped. The default is `0`, meaning that the load will continue, regardless of the number of warnings encountered during the load operation.

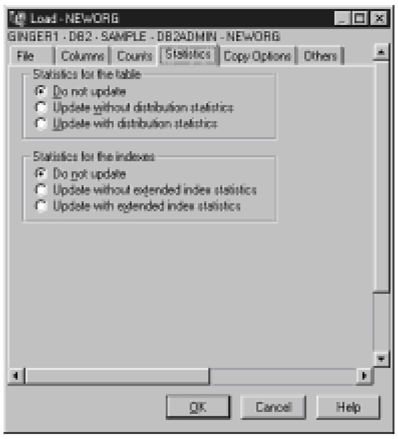
Specifying Collection Statistics

Click the Statistics tab to specify how to collect statistics during the load process (see Figure 13.15). The following table options are available:

- Use the default Do Not Update option if you don't want statistics for the table to be updated during the load.

- Select Update Without Distribution Statistics to have basic-level statistics on the table data updated in the system catalog tables during the load. An example of a basic-level statistic is the number of pages being used by the table.

- Select Update With Distribution Statistics to update basic-level statistics on the table data and to collect statistics of the distribution of data values in the table columns. The query optimizer uses these statistics to more accurately estimate the number of rows in a column that satisfy given equalities or ranges. An example of a distribution statistic is the frequency with which a particular data value occurs in a column.



Statistics for the table
[✓ Do not update]
Update without distribution statistics
Update with distribution statistics

Statistics for the indexes
[✓ Do not update]
Update without extended index statistics
Update with extended index statistics

Figure 13.15: The Load notebook's Statistics page.

Note You can gather statistics during the load process only if you selected `REPLACE` as the load mode on the Counts page.

Here are the choices for index statistics:

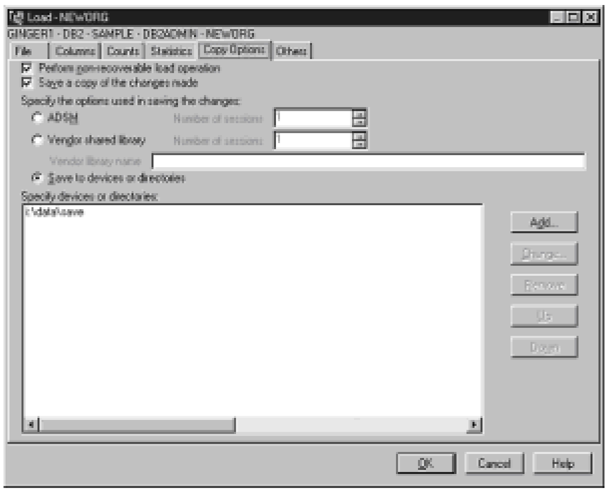
- Use the default Do Not Update option if you don't want statistics for the index to be updated during the load.

- Select Update Without Extended Index Statistics to collect basic-level statistics on the indexes during the load. An example of a basic-level statistic is the number of index leaf pages.

- Select Update With Extended Index Statistics to collect basic-level and detailed statistics on the indexes. The detailed statistics can help the optimizer better estimate the input/output cost of an index scan. An example of a detailed statistic is the number of input/output actions needed to read the data pages into buffer pools of various sizes.

Setting Save Options

On the Copy Options page, you specify how to save a copy of the changes made during the load process (see Figure 13.16). If you don't need to save a copy of the changes, select the Perform Non-Recoverable Load Operation check box to specify that the load is nonrecoverable.



☑ Perform non-recoverable load operation
☑ Save a copy of the changes made

Specify the options used in saving the changes:

○ ADSM — Number of sessions: 1
○ Vendor shared library — Number of sessions: 1
Vendor library name:
● Save to devices or directories

Specify devices or directories:

c:\data\save

Figure 13.16: The Load notebook's Copy Options page.

Note By default, if forward recovery is enabled for the database in which this table resides and you load data into the table, the table spaces for that table are placed in backup pending state. In this case, the data in the table isn't accessible until a table space backup or a full database backup is made.

If you don't want the table spaces to be placed in backup pending state, you can perform a nonrecoverable load or create an image copy of the load. Select the Perform Non-Recoverable Load Operation check box if you want to perform the load without jeopardizing the recoverability of all the other tables in the database and without the overhead of building a copy of all the changes made.

Select the Save A Copy Of The Changes Made check box to specify that a copy of the changes made during the load process be saved.

Tip Forward recovery must be enabled for the database before you can save a copy of the changes.

Specify where you want the copy image saved by selecting an option under Specify The Options Used In Saving The Changes:

- *ADSM* is available only when the Save A Copy Of The Changes Made check box is selected. Use this option to specify that the copy of changes made be stored by using

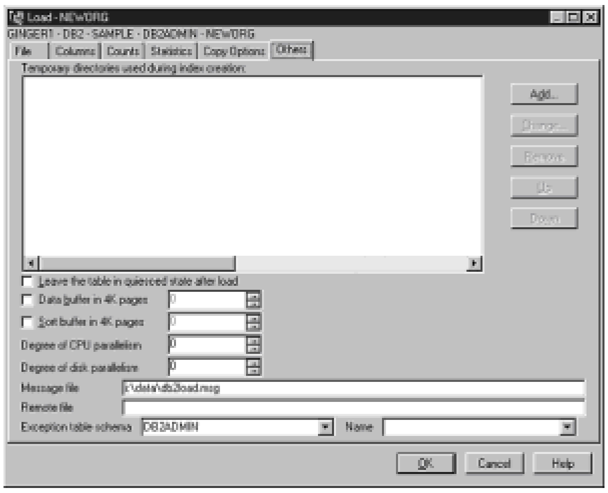
ADSTAR Distributed Storage Manager (ADSM). The Number Of Sessions box contains the number of I/O sessions to be used with ADSM.

- *Vendor Shared Library* is available only when the Save a Copy Of The Changes Made check box is selected. Select this option to specify that the copy of the changes be stored by using a vendor product. The Number Of Sessions box contains the number of I/O sessions to be used with the vendor product. (The default is `1`, meaning that one I/O session will be used.) In the Vendor Library Name text box, type the name of the shared library containing the vendor backup and restore I/O functions to be used.

- *Save To Devices Or Directories* specifies that a copy of changes made be stored on devices or directories. Click Add to open the Add window so that you can specify those devices and directories.

Specifying Other Settings

Select the Others tab to specify various options for the load process (see Figure 13.17).


Load - NEWORG
GINGER1 - DB2 - SAMPLE - DB2ADMIN - NEWORG

Temporary directories used during index creation:

☐ Leave the table in quiesced state after load
☐ Data buffer in 4K pages: 0
☐ Sort buffer in 4K pages: 0
Degree of CPU parallelism: 0
Degree of disk parallelism: 0
Message file: c:\data\db2load.msg
Remote file:
Exception table schema: DB2ADMIN
Name:

Figure 13.17: The Load notebook's Others page.

In the Temporary Directories Used During Index Creation list box, specify locations where temporary files used during index creation will be written. Click the Add button; use the Add window to add directories to the Temporary Directories Used During Index Creation list. If you don't specify directories, the files are created in the `sqllib\tmp` directory of the `DB2INSTANCE` owner. If you specify more than one directory, each directory should be on a different file system, and each file system should be on a different disk in order to optimize performance. Make sure that enough space is on these directories to hold all index keys for the data being loaded.

Select the Leave The Table In Quiesced State After Load check box to specify that the table be left in exclusive mode after the load is completed. In this mode, you have exclusive access to the table and its table spaces; others can't read or update the table data. This way, you can ensure that the load worked as expected before others begin using the data. By default, this option isn't selected.

The Data Buffer In 4K Pages option specifies the number of pages of memory for data buffers used during the load. The default is `0`, meaning that the minimum number of buffers to be used will be calculated according to a particular algorithm. Each buffer is at least 16 pages, each page being 4KB of memory. If you specify a value less than the algorithmic minimum, the minimum required number of buffers is used, and no warning is returned.

The Sort Buffer In 4K Pages option specifies the number of pages of memory for sorting the index keys during the load. The default is `0`, meaning that a minimum amount of storage is used. This minimum amount depends on the number of indexes, the number of keys in the index, the size of the keys, and the number of temporary directories specified in the Temporary Directories Used During Index Creation list.

Tip The sort buffer size greatly affects sort performance. Therefore, for very large tables, this buffer should be set as large as possible.

The Degree Of CPU Parallelism option specifies the degree of CPU parallelism to be used by the Load notebook for operations such as formatting data. A degree of parallelism of `3` means that three formatters are spawned. The default value of `0` specifies that the degree of CPU parallelism is based on the number of CPUs now online. This option is most useful when you're running on a system with multiple CPUs. It enables for optimal parallel loading while preserving the order of records in the source data. This is especially useful when you're loading presorted data, because record order in the source data is preserved.

The Degree Of Disk Parallelism option specifies the degree of disk parallelism to be used by the Load notebook for such operations as writing data to disk. A degree of parallelism of `3` means that three disk writers are spawned. The default value of `0` specifies that the degree of disk parallelism is based on the number of system disks now online.

The Message File option contains the name of the file where warning and error messages that occur during the load are written. By default, the messages are written to the file `db2load.msg`.

In the Remote File text box, type the base name to be used for creating temporary files during the load. This name must be fully qualified according to the server node. The remote file is a base filename that DB2 appends with different extensions to create temporary files for storing messages and consistency points and to delete phase information. The remote file resides on the server and is accessed by the DB2 instance exclusively. Any filename qualification must reflect the directory structure of the server, not the client, and the DB2 instance owner must have read and write permission on this file. You must ensure that only one load is issued with the same fully qualified remote filename.

Caution If you're on the same machine as the database instance and don't provide a name, DB2 uses the name `db2utmp` and qualifies it with your current working directory. If you're loading data in more than one load process using this directory and don't specify remote filenames for each load process, the loads will clash.

If you're on a different machine from the database instance and don't provide a name, DB2 generates a name that will reside in the database directory. The generated filename isn't guaranteed to be unique; therefore, clashes between loads can occur.

Select the Exception Table Schema check box to specify the schema of the exception table. The schema must exist.

Setting Constraints

Referential integrity and check constraints on a table are normally enforced automatically. In some situations, you might need to manually turn on the constraint checking for a table. Typically, you need to set constraint checking manually after loading data into a table.

The load operation causes a table to be put into check pending state automatically if the table has check or foreign key constraints defined on it. When the load operation is completed, you can turn on constraint checking for the table.

Use the Set Constraints window to turn off constraint checking for a table and to turn it back on. Follow these steps:

1. Log on as a user with `SYSADM` or `DBADM` authority, unless you have `CONTROL` privilege on the table and on all tables whose foreign keys can be traced back to the primary key of the table. These tables are its dependents and descendants.

2. Start the Control Center.

3. Expand the object tree until you find the Tables folder. Click that folder. The available tables are listed on the right side of the window.

4. Right-click the table you want to set constraints, and select Set Constraints from the pop-up menu. The Set Constraints window opens.

5. To turn off constraint checking, select Off. This puts the table in check pending state. Click OK to turn off constraint checking for this table.

   Select On With Checking to turn on constraint checking for referential and check constraints and to check the table immediately for constraint violations.

   Select the On Without Checking radio button to turn on constraint checking for a table without checking existing table data. The table can't be a system table.

When you select On With Checking, you can specify an exception table that will collect any row that's in violation of a foreign key or check constraint before it's deleted from the table. Specify an exception table by using the Exception Table Schema and Exception Table Name options. The constraints are turned back on again, and the table is taken out of check pending state, regardless of whether errors are detected. A warning is issued to indicate that one or more rows were moved to the exception table. If you don't specify an exception table and any constraints are violated, only the first violation detected is returned to you. The table is left in check pending state.

When you select On Without Checking, you can use the Referential Constraints and Check Constraints check boxes to specify the constraint type for which you're turning on constraint checking. When you turn on constraint checking without performing the checking, this is recorded in the database catalog. (The value in the `CONST_CHECKED` column in the `SYSCAT.TABLES` view is set to `U`.) This indicates that you've assumed responsibility (over the database management system) for ensuring that the data complies with the constraints. This value remains the same until the table is put back into check pending state or all unchecked constraints for the table are dropped.

Summary

In today's lesson, you saw that you can export data from a DB2 database to four different formats: nondelimited ASCII, delimited ASCII, worksheet, and integrated exchange. You can import data into a DB2 database from an input file by using the same four formats supported for export. You can import data into an existing table or create a new table.

You can load data into a DB2 database from a file, tape, or named pipe. Loading data can be faster than importing data if you have a large amount of data that needs to be moved.

Informational and error messages are kept during export, import, and load operations, which can be helpful if you run into a problem with the operations.

What Comes Next?

On Day 14, "Replicating Data," you learn how to replicate data from one database to another.

Q&A

Q If I want to create a new database by importing data, what format should the data be in?

A The integrated exchange format (IXF) must be used if you want to import into a new table. IXF is a format that contains the relational database structure. This format is recommended when exchanging data between different versions of DB2.

Q Where are messages kept?

A When you're setting up the export, import, or load operation, you must indicate the name of the file you want the messages to be kept in. You should specify an entire path for the filename. If the file doesn't exist, it will be created. If it does exist, messages will be appended to the file. You can't start the operation if you don't indicate a message file.

Q Can I import flat text?

A If you use an editor such as Notepad, you can create flat or ASCII text files, and these can be imported. It's easiest if you use delimiters such as commas between each data column. You'll then need to enclose character strings with double quotes ("). If you create a text file like this, you'll import it by using the delimited ASCII format.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. To create a new table by importing IXF data, which import mode should you use?
2. If you want to create a new table, what format should you use? Would you use the Load notebook or the Import notebook to create a new table?
3. How do you export only a subset of the data in a table?
4. What must take place before you can save a copy of the changes when using the Load notebook?

Exercise

Use the Export notebook to export data from the SAMPLE database, choosing the file formats that you didn't try during the lesson, and then use the Import and Load notebooks to insert this data into new or existing tables. Experiment with some available options that were not used in the lesson.

Day 14: Replicating Data

Overview

Replication is the process of taking changes stored in the database log at thesource server and applying them to the target server. You can use replication to define, synchronize, automate, and manage copy operations for data across your enterprise. You can automatically deliver the data from a host system to target sites. For example, you can copy data and applications to branch offices and even sales representatives' laptops.

To manage the replication environment, you define source tables, known as *replication sources*, and specifications for copying changed data, known as *replication subscriptions*. Two programs, Capture and Apply, take the changes from a source table, stage them until the target replication program is ready to receive them, and then replicate the changes to one or more target tables.

The Capture and Apply programs perform the ongoing replication operations. When your application updates the source table, Capture reads the database log or journal and copies the changes to a staging area for replication at a later time. The Apply program reads the changes and replicates them to the target table.

With the Control Center, you can do the setup required for replication by using the `Define as replication source` and `Define subscription` actions. The replicationprograms, Capture and Apply, run outside the DB2 Administration Tools.

Replication administrators can perform the following tasks from the Control Center:

- Define replication sources
- Define replication subscriptions
- Create control tables and target tables
- Specify SQL to enhance data during the Apply process

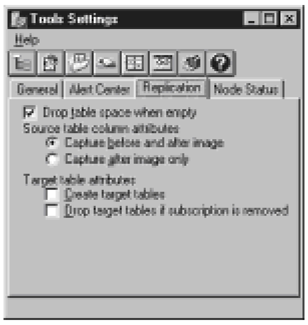
During this lesson, you'll learn the concepts of replication by defining several replication subscriptions that replicate several tables in the CDLIB database to another database.

Setting Defaults for Replication

Before you begin using the replication tools, you should set the defaults in the Tools Settings notebook. The defaults you set are used in future steps required to set up replication. In fact, these settings are used in the `Quick` method of defining replication sources. When using Custom replication, you can override these settings. Perform the following steps:

1. Open the Tools Settings notebook by clicking the Tools Settings icon in the Control Center's toolbar or by choosing Start | Programs | DB2 for Windows NT | Administration Tools | Tools Settings.

2. Click the Replication tab to display the Replication page, where you specify the settings for replication (see Figure 14.1).



[✓] Drop table space when empty

Source table column attributes

(●) Capture before and after image

( ) Capture after image only

Target table attributes

[ ] Create target tables

[ ] Drop target tables if subscription is removed

Figure 14.1: The Replication page of the Tools Settings notebook.

3. Select the Drop Table Space When Empty check box to specify that table spaces for target and change data tables be dropped when the table spaces are empty. This check box is selected by default. If this check box isn't selected, you're responsible for dropping the empty table spaces.

4. Specify to save the source table columns by selecting a source table column attribute:

   - *Capture Before and After Image* saves the source table columns before and after they're captured. Use this option to keep a record of all changes for rollback or auditing purposes.

   - *Capture After Image Only* (the default) saves a copy of source table columns after they're updated.

5. Specify preferences for the target table attributes by selecting one or more of the check boxes under Target Table Attributes:

   - *Create Target Tables* automatically creates target tables every time you define a replication subscription. If this check box isn't selected, you must check the Create Table check box in the Replication Subscription panel.

   - *Drop Target Tables if Subscription Is Removed* drops target tables if the associated subscription is removed. If this check box isn't selected, you're responsible for dropping the target tables.

6. Close the Tools Settings notebook.

Setting Up Replication Sources

A *replication source* is a table used as a source for copying data to a target table. You can define replication sources by using the `Quick` or `Custom` methods. `Quick` enables you to define a replication source by using the defaults set in place. `Custom` enables you to customize the defaults, such as specifying any columns that shouldn't be captured for update replication.

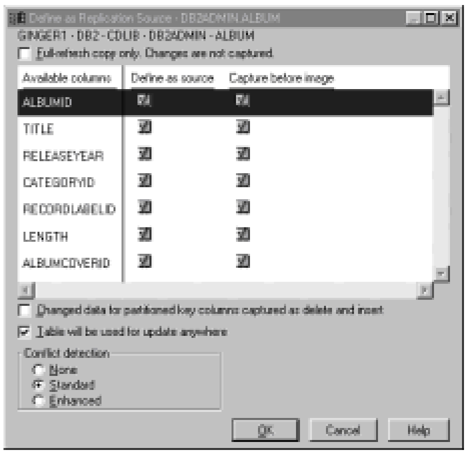
Using the `Custom` Method

To see how the replication tools work, replicate the `ALBUM` table in the CDLIB database to the SAMPLE database. Follow these steps to define replication sources with customized settings:

1. Start the Control Center.

2. Expand the object tree until you see the Tables folder within the CDLIB database. Click the Tables folder to see a list of available tables in the right side of the window.

3. Right-click the `ALBUM` table to define it as a replication source, and select Define as Replication Source | Custom from the pop-up window. The Define as Replication Source window opens (see Figure 14.2).



Define as Replication Source - DB2ADMIN.ALBUM

GINGER1 - DB2 - CDLIB - DB2ADMIN - ALBUM

☐ Full-refresh copy only. Changes are not captured.

Available columns	Define as source	Capture before image
ALBUMID	✓	✓
TITLE	✓	✓
RELEASEYEAR	✓	✓
CATEGORYID	✓	✓
RECORDLABELID	✓	✓
LENGTH	✓	✓
ALBUMCOVERID	✓	✓

☐ Changed data for partitioned key columns captured as delete and insert

☑ Table will be used for update anywhere

Conflict detection: None [✓ Standard] Enhanced

Figure 14.2: The Define as Replication Source window.

4. Make sure that the Full-Refresh Copy Only check box isn't selected, so only changes are captured.

Tip When you want to copy all data from the source table to the target table, select the Full-Refresh Copy Only check box; the Capture program won't log changes for this table, and copying just the updated data isn't possible. Use full-refresh copying when

- You don't want to use the Capture program.
- You want all rows of the source table replicated to the target table.
- Your table is small and has many frequent updates.
- You want the data to be completely up-to-date.

Don't use full-refresh copying when you have a large table with a few or moderate number of updates, when you don't want all rows of the source table copied to the target table every time the table is replicated, or when you're using Capture to capture and replicate only changed data to thetarget table.

Note When you use full-refresh copying, you don't need to make any more changes in this notebook. Skip to step 10.

5. Use the check boxes in the Define as Source column to specify table columns for replication. These boxes are selected by default; ensure that all columns are selected if the table might be used for update-anywhere replication. (For this example, you want to replicate all columns of the `ALBUM` table, so ensure that all these check boxes are selected.)

   If you don't want to replicate a column, clear its Define as Source check box. That way, the column name becomes unavailable as a source column during subscription and can't be replicated.

6. The primary keys in the `ALBUM` table aren't likely to change; therefore, make sure that the Changed Data for Partitioned Key Columns Captured as Delete and Insert check box isn't selected.

   Tip If your applications update one or more columns of the target primary key, select this check box so that any update to primary key columns is captured as a `DELETE` and `INSERT` in the change data table rather than as an `UPDATE`. Suppose that the primary key for a table is the employee's last name, Visser, and the employee changes her name to Weckworth. Without logical partitioning key support, when the Apply program updates the employee's last name with a new primary key value, it doesn't find the value `Weckworth` and inserts a new row. The row with the primary key value `Visser` isn't deleted.

7. Select the Table Will Be Used for Update Anywhere check box to specify that this source table can be used in update-anywhere replication. This way, the table can have changes replicated to one or more replica target tables. The replica table can, in turn, be updated and have its changes replicated back to the replication source table. All columns are defined as source, all before-image columns are captured, and conflict detection is enabled with standard detection.

8. If you want to save a column from the source table before it's updated (known as *before-image*), select the Capture Before Image check box for that column in the list box. This option is useful for recovery and auditing. When selected, a column is replicated to the target table containing the values of the source table before the values were updated. By default, the name of the before-image column begins with the prefix character `X`. (This check box is selected automatically when the Table Will Be Used for Update Anywhere Replication check box is selected.)

9. If you selected the Table Will Be Used for Update Anywhere check box, specify the level of Conflict Detection for user tables and all associated replicas:

   - Use None if you don't want to monitor conflict detection. Conflicting updates between the origin table and the replica won't be detected. Use this option if it's acceptable to sometimes lose updates.

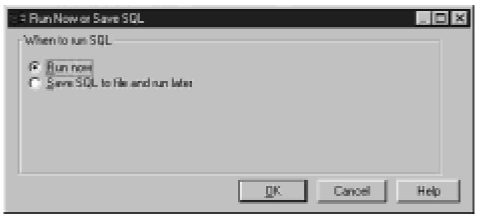
   - For this example, use the default, Standard, to moderate conflict detection monitoring. The Apply program searches rows already captured in the replica's change table for conflicts.

   - Use Enhanced if you want conflict detection monitoring that guarantees data integrity among all replicas and the origin table. The Apply program locks all replicas in the subscription set against further transactions and begins detection after all changes before locking are captured.

   The replication source is the primary or global table. It can receive updates from replicas, but if a conflict occurs, the source table wins, and the replica's transaction is rejected. Conflict detection is used to detect an out-of-date row in a replica table updated by a user application. When a conflict is detected, the transaction that caused the conflict is rejected.

   Note Conflict detection doesn't guarantee data integrity for read operations. For example, an application reads the replica's updated row before it's replicated back to the source table. If the transaction for the update is later rejected as a conflict, conflict detection can't detect this conflict.

10. Click OK to define a custom replication source. In the Run Now or Save SQL window, you can choose to run your SQL request now or later (see Figure 14.3). For this example, choose to run the SQL request now.



[✓ Run now]
Save SQL to file and run later

Figure 14.3: The Run Now or Save SQL window.

11. Refresh the screen by pressing F5.

12. To verify your object, click the Replication Sources folder, and ensure that the table you set up as a replication source is listed.

13. Start or reinitialize the Capture program from your command-line processor so that it will recognize the new replication source custom definition.

14. To begin capturing changes to the source table, start the Capture program from the command-line processor. See the later section "Configuring and Running the Capture Program" for information.

Using the `Quick` Method

By using the Define as Replication Source | Quick option, you can define multiple tables as replication sources. The `Quick` method uses the default values to define tables as sources for replication:

- All source columns are available for data capture.

- Changed data for partitioned key columns is captured as updates.

- If the table will be used for update-anywhere replication, the conflict detection level is set to `NONE`.

Now define the `ARTIST` table to be used for replication as well. Follow these steps to define replication sources by using the default values:

1. In the Control Center, with the Tables folder expanded for the CDLIB database, right-click the `ARTIST` table and select Define as Replication Source | Quick from the pop-up menu.

2. In the Run Now or Save SQL window, choose to begin the replication now.

3. Refresh the screen by pressing F5, and then click the Replication Sources folder to ensure that the table you set up as a replication source is listed.

Defining Replication Subscriptions

After you define at least one replication source, you specify how it's replicated to target tables by defining a replication subscription. A *replication subscription* tells the Apply program how and when one or more source tables are replicated to their respective target tables and defines the structure of the target tables (columns and rows). Information about the replication subscription is stored in control tables at the control server, a logical server you specify when you define the subscription.

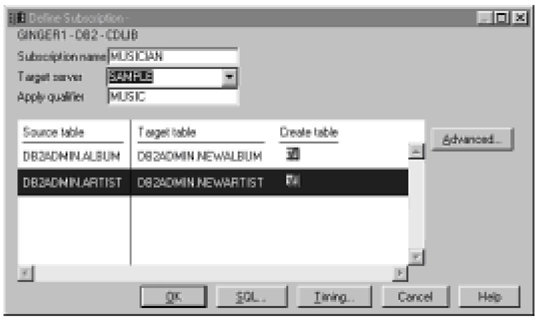
You can use the Define Subscription window to define basic or customized subscription sets to be replicated. Defining basic subscription sets involves setting up source and target table combinations with the appropriate target server and Apply qualifier. Defining customized subscription sets involves adding more specifications, such as specifying the target table type, timing, and runtime SQL statements.

Defining a Basic Subscription

To define a subscription that indicates when and how the `ARTIST` and `ALBUM` tables are replicated, follow these steps:

1. In the Control Center, expand the object tree until you see the Replication Sources folder within the CDLIB database. Click this folder to see a list of existing replication sources on the right side of the window.

2. Select the `ARTIST` and `ALBUM` replication sources, right-click, and select Define Subscription from the pop-up window. The Define Subscription window opens (see Figure 14.4).



GINGER1 - DB2 - CDLIB

Subscription name: MUSICIAN
Target server: SAMPLE
Apply qualifier: MUSIC

Source table	Target table	Create table
DB2ADMIN.ALBUM	DB2ADMIN.NEWALBUM	✓
DB2ADMIN.ARTIST	DB2ADMIN.NEWARTIST	✓

Figure 14.4: The Define Subscription window.

3. Specify a name for the subscription in the Subscription Name text box—for example, **MUSICIAN**. The name can be up to 18 characters long.

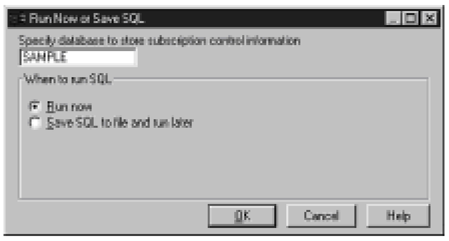
4. In the Target Server combo box, specify the alias of the target server—for this example, select the SAMPLE database from the drop-down list. The default is the alias of the source server.

5. In the Apply Qualifier text box, type the name of the Apply instance that will run the current subscription—for example, **MUSIC**. The name can be up to 18 characters long.

6. The `Target Table` field lists default target tables that match the corresponding source tables. Type an existing or new target table name in this field. For this example, change the names to `NEWARTIST` and `NEWALBUM`, and select their corresponding Create Table check boxes. (To change the name, click the target table name and type a new name.)

   Note When you use a new name, you need to select its corresponding Create Table check box to create the table.

7. Click OK to submit the subscription. The Run Now or Save SQL window opens (see Figure 14.5).


Run Now or Save SQL

Specify database to store subscription control information: SAMPLE

When to run SQL: [✓ Run now] Save SQL to file and run later

Figure 14.5: The Run Now or Save SQL window.

8. Specify a database to store subscription control information. For this example, use the SAMPLE database.

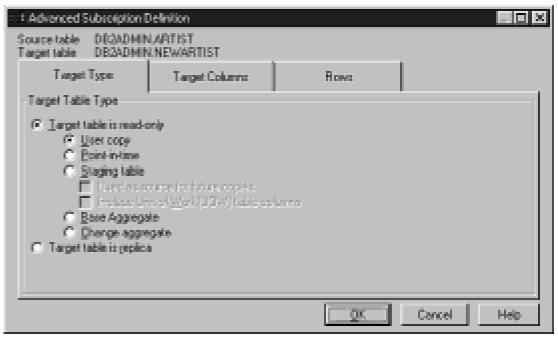
9. You can choose to run the subscription request now or to schedule it for a later time and date. For this example, choose to run the subscription request now, and then click OK to have the subscription created.

10. Refresh the screen by pressing F5, and then click the Replication Subscriptions folder in the Control Center to ensure that the table you set up as a replication source is listed.

When the subscription is created, you must run the Apply program from the command line. The Apply program runs independently of the Control Center. Generally, you should run the Apply program at the target server, but you can run it on any server in the network that can connect the source, target, and control servers. See "Configuring and Running the Apply Program" for information.

Defining an Advanced Subscription

To define an advanced subscription, click the Advanced button in the Define Subscription window to open the Advanced Subscription Definition notebook (see Figure 14.6). In this notebook, you can define a subscription that accepts updates from replica tables or that replicates only some of the rows and columns. You must first complete a basic subscription, as explained in the preceding section, before you can use the following steps to define an advanced subscription.


Advanced Subscription Definition

Source table DB2ADMIN.ARTIST
Target table DB2ADMIN.NEWARTIST

Target Table Type

[✓ Target table is read-only]
- [✓ User copy] Point-in-time / Staging table / Base Aggregate / Change aggregate
- Staging table: ☐ Used as source for future copies; ☐ Include Unit-of-Work (UOW) table columns

Target table is replica

Figure 14.6: The Advanced Subscription Definition notebook.

Specifying the Target Table Type

On the Target Type page of the Advanced Subscription Definition notebook, you specify whether the target table is an updatable replica (the target table is used to replicate data back to the origin table) or read-only. For the example, select Target Table Is Read-Only to specify that the target table can't be updated.

Note You use the Target Table Is Replica option to specify that the target table is used to replicate data back to the origin table. Replica tables default to the same after-image columns and primary key fields as the source table.

Because you select a read-only target table, you must choose one of the following read-only target types:

- *User Copy* (the default) contains a snapshot of the table when copied. This target type must have a primary key. Use this target type for the example.

- *Point-in-Time* contains a complete, condensed copy of a source table at a certain point in time. A point-in-time table includes an added system column that identifies the approximate time when the particular row was inserted or updated at the source table. This table must have a primary key.

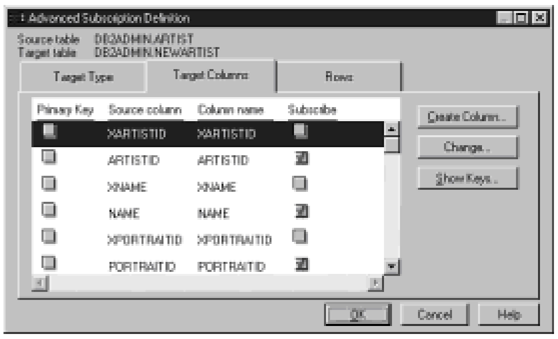
- *Staging Table* can be used as a source for further copies, supports a consistent data environment, provides flexibility in data distribution across a network, and becomes the source for a copy operation. If you select this target type, you can optionally append columns that contain committed unit-of-work information to the target table by selecting the Unit of Work (UOW) Table Columns check box. You can also use a staging table as a source for future copies.

- *Base Aggregate* is a history table in which a new row is appended for each changed row in the source table. Base aggregate tables can include calculations against stable base data. This history type is useful for tracking broad indicators that have relatively low volatility.

- *Change Aggregate* is a history table in which a new row is appended for each changed row in the source table with calculations made against changed data. Each calculation ranges over the recent changes since the time of the last calculation. You can track insert, update, and delete operations.

Creating New Columns in the Target Table

On the Target Columns page (see Figure 14.7), use the list box to create new columns in the target table. A column's value is derived from a SQL expression involving one or more after-image columns of the source table. You can compute average values, minimum and maximum values, sums, and so on. The new column is called a *computed column*. For this example, leave the columns as they are.



Source table DB2ADMIN.ARTIST
Target table DB2ADMIN.NEWARTIST

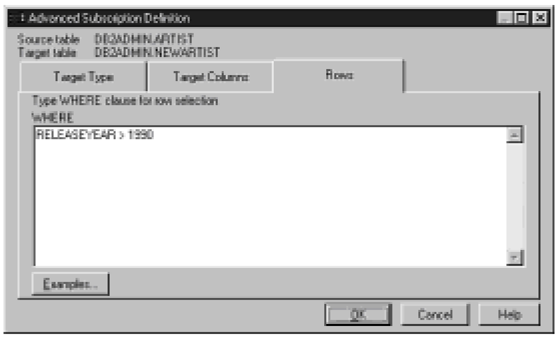
Primary Key	Source column	Column name	Subscribe
☐	XARTISTID	XARTISTID	☐
☐	ARTISTID	ARTISTID	☑
☐	XNAME	XNAME	☐
☐	NAME	NAME	☑
☐	XPORTRAITID	XPORTRAITID	☐
☐	PORTRAITID	PORTRAITID	☑

Figure 14.7: The Advanced Subscription Definition notebook's Target Columns page.

To change a SQL expression for a computed column, click the Change button. To see which columns are defined as primary or unique keys in the source table, click the Show Keys button.

Defining Rows

On the Rows page of the Advanced Subscription Definition notebook (see Figure 14.8), you create a `WHERE` clause to specify that only a subset of rows be copied to the target table. Click Examples for a list of sample predicates. Type any valid SQL predicate in the `WHERE` box. Only source table rows meeting the conditions of the `WHERE` clause are copied to the target table. For this example, replicate only those rows in which an album was released after 1990. To do this, enter **RELEASEYEAR > 1990** in the `WHERE` text box.



Source table DB2ADMIN.ARTIST
Target table DB2ADMIN.NEWARTIST

Type WHERE clause for row selection
WHERE
RELEASEYEAR > 1990

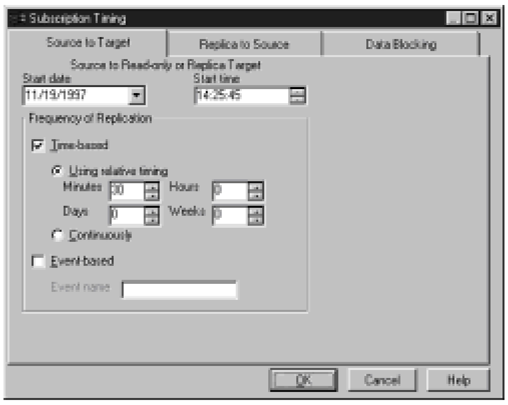
Figure 14.8: The Advanced Subscription Definition notebook's Rows page.

Defining Runtime SQL Statements

Click the SQL button in the Define Subscription window to open the SQL window, in which you can define SQL statements to be processed during runtime. You can add, change, or remove runtime SQL processing statements that will be submitted before or after the subscription is processed.

Scheduling Subscription Replication

In the Define Subscription window, click the Timing button to open the Subscription Timing notebook so that you can specify when and how often to replicate the subscription (see Figure 14.9).



Source to Read-only or Replica Target
Start date 11/19/1997
Start time 14:25:45
Frequency of Replication
[✓ Time-based]
[✓ Using relative timing]
Minutes 30 Hours 0
Days 0 Weeks 0
Continuously
Event-based
Event name

Figure 14.9: The Subscription Timing notebook.

Follow these steps to use this notebook:

1. On the Source to Target page, specify the date and time when you want the replication to start and whether you want to replicate this subscription based on time,

an event, or both. Select the Time-Based check box to apply changes to the target table or source on a weekly, daily, hourly, by-the-minute, or continuous basis. Select the Event-Based check box to specify that the table will be replicated whenever the event you specify occurs. The replication program reads the Event Name in your job-scheduling program to initiate the Apply process for the target table.

For this example, select to have the data replicated every 30 minutes.

2. On the Replica to Source page, specify when and how often you want to replicate this subscription from a replica target table to a source table. You can use this page only when a target table in a subscription is a replica.

3. On the Data Blocking page, specify the number of minutes at a time that Apply will copy committed data.

4. Click OK to return to the Define Subscription window.

Configuring and Running the Capture Program

Before you can use the Capture program, you first prepare the database by setting it up for roll-forward recovery and binding the Capture utilities to the database. Follow these steps:

1. Log on to the source server system with a username that has execute privilege on the Capture packages and `DBADM` or `SYSADM` privileges for the database.

2. Start the Control Center.

3. Connect to the source server database (CDLIB, for example) by right-clicking the database and selecting Connect from the pop-up menu.

4. Ensure that the source server database is configured for roll-forward recovery. Right-click the source server database, and select Configure from the pop-up menu. On the Logs page, ensure that the Retain Logs for Roll-Forward Recovery parameter is set to Yes.

   Note If you have to change the Retain Logs for Roll-Forward Recovery parameter from No to Yes, you must perform a backup of the database before proceeding. If the parameter was already set to Yes and you have a recent backup of the database, you can proceed with the steps.

5. Change to the drive that contains the Capture bind files (usually `\sqllib\bnd`), and bind the Capture program packages to the source database. Enter the following commands from the DB2 Command window:

   Input
   ```
   db2 connect to CDLIB
   db2 bind @capture.lst isolation ur blocking all
   ```

   `CDLIB` is the source database server, and `ur` specifies the list in uncommitted read format for greater performance.

   Note Perform this last step only once for each database.

To start the Capture program, enter the following command from a Windows NT command window:

Input
```
asnccp CDLIB
```

where `CDLIB` is the name of the source server database.

After you start the Capture program, it runs continuously until you stop it or it detects an error. Enter the **asncmd stop** command when you want to stop the Capture program.

Tip You might want to create a script or a service for the `asnccp` command. Use the Script Center as described in Day 11, "Using System Administration Tools," to create a script to start the `asnccp` command.

Configuring and Running the Apply Program

Before you can use the Apply program, you must first bind the Apply utilities to both the target and source databases. Follow these steps:

1. Log on to the source server system with a username that has execute privilege on the Apply packages and `DBADM` or `SYSADM` privileges for the database.

2. Start the Command Center.

3. Connect to the source server database by entering

   Input
   ```
   db2 connect to CDLIB
   ```

   where `CDLIB` is the name of the source server database.

4. Create and bind the Apply package to the source server database by entering the following commands from the DB2 Command window:

   Input
   ```
   db2 bind @applycs.lst isolation cs blocking all
   db2 bind @applyur.lst isolation ur blocking all
   ```

   `cs` specifies the list in cursor stability format, and `ur` specifies the list in uncommitted read format.

5. Connect to the target server database by entering

   Input
   ```
   db2 connect to SAMPLE
   ```

   where `SAMPLE` is the name of the target server database.

6. Create and bind the Apply package to the target server database by entering the following commands:

   Input
   ```
   db2 bind @applycs.lst isolation cs blocking all grant
   public
   db2 bind @applyur.lst isolation ur blocking all grant
   public
   ```

   `cs` specifies the list in cursor stability format, and `ur` specifies the list in uncommitted read format.

7. Repeat the connect and bind steps for each server that the Apply program connects to. You must bind the Apply program to the source, target, and control servers.

To start the Apply program, enter the following command from a Windows NT command window:

Input `asnapply MUSIC SAMPLE`

`MUSIC` is the Apply qualifier, and `SAMPLE` is the name of the target database.

After you start the Apply program, it runs continuously until you stop it or it detects an error. Enter the **asnastop** command when you want to stop the Apply program.

Summary

In today's lesson, you saw that you can replicate data from one database to another. When the replication is set up, logs are made during data transactions. These logs are then used to update the replica database, minimizing the amount of copying required to keep the databases in sync.

You use the Control Center to define the replication sources and the subscription actions. When these are set up, you use the Capture and Apply programs outside the Control Center to read and replicate the changes.

You can set up many types of replication. You can choose to have the entire contents of the database copied or just the changes.

You have the option of setting up a more advanced scenario in which the source database is replicated to many sites and any changes made to the replicated databases at these individual sites are then captured and replicated back to the source database.

What Comes Next?

On Day 15, "Ensuring Data Security," you learn how to set up your database so that your database is protected against unauthorized use.

Q&A

Q When should I use the full-refresh copying option?

A Because the full-refresh copying option replicates the entire database table, regardless of the number of changes made to the table, you should use this option when you frequently update a relatively small table. This ensures that the table is completely up-to-date and eliminates the use of the Capture program that logs all database transactions.

Q When do I need to worry about conflict detection?

A Conflict detection is necessary only when you're using update-anywhere replication. *Update-anywhere replication* means that you've replicated your table to many sites and enables these changes to be replicated back to the source database. Conflict detection is necessary to ensure that changes made to the same data at two separate sites don't jeopardize the integrity of the data.

Q When do I need to use the advanced options when creating a subscription definition?

A You need to use the advanced options when you're defining a table for update-anywhere replication, to subscribe to only a subset of the columns in the table, or to replicate only a subset of the rows in the table.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. How can you set some defaults when using the `Quick` method of defining replication sources?
2. Where must the Capture program be run? Where must the Apply program be run?
3. What is *capture before-image*? When would you use this option?

Exercise

When you have set up the replication as described in this lesson, add data to the `ARTIST` table to see that the data is replicated to the `NEWARTIST` table.

Week 2

In Review

You've now completed Week 2! This week covered most of the core tasks involved in using DB2. You learned how to create databases, tables, and table spaces. You learned various ways to access DB2, including ODBC applications and Java applications. You learned how to use many useful administration tools, such as the Script Center and the Journal. Backing up and recovering data as well as importing, loading, replicating, and exporting data were also taught in the past week.

Week 3: At a Glance

Overview

During Week 3, you'll learn how to set up your system to be secure, administer your clients, work with DB2 instances, ensure that you're getting the best performance from your system, and diagnose problems that you may encounter. Here's a day-by-day summary of Week 3:

- On Day 15, "Ensuring Data Security," you'll learn ways to ensure that your data is protected against unauthorized use.
- Day 16, "Administering Clients with the CCA," teaches how you can use the Client Configuration Assistant to administer a large number of clients.
- Day 17, "Working with DB2 Instances," steps through the ways you can work with and manipulate the DB2 instance.
- Day 18, "Performance Aids," introduces performance aids such as Visual Explain, Snapshot Monitor, and Event Monitor. You'll also learn how you can use these aids to explore the performance of your databases and applications.
- Day 19, "Design Considerations," covers several ways you can affect the performance of your system simply through the design of your databases, tables, and applications.

- On Day 20, "Tuning DB2 Universal Database Performance," you'll learn how to customize configuration parameters to ensure that your system is finely tuned for your specific needs.
- Day 21, "Diagnosing Problems," gives you guidance on solving problems that you may encounter.

Day 15: Ensuring Data Security

Overview

Two security levels control access to DB2 Universal Database data and functions. Access to DB2 Universal Database is managed by facilities specific to the operating environment, whereas access within it is managed by the database manager.

As a database administrator, you might need to control the type of access people have to data, or restrict their view of the data. Today's lesson describes the methods for controlling access to the data, including access to DB2, the privileges to work with objects within DB2, and authentication, which provides the security for data access.

Access to DB2 Universal Database

DB2 uses external facilities to provide a set of user and group validation and management functions. Users must log on through the external facilities by providing a username and a password. The security facilities validate the username and password to ensure that access for this user is allowed.

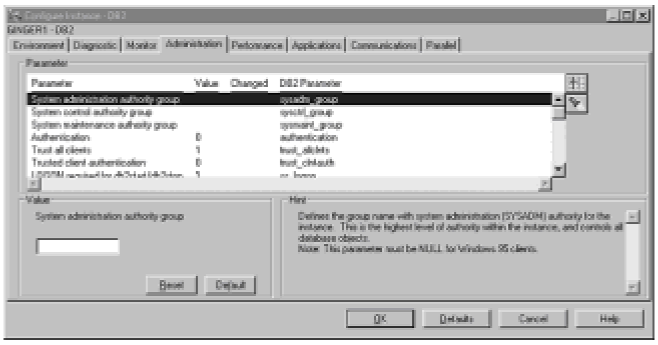
You need to have a Windows NT username in order to administer DB2. The username must belong to the Administrators group and be a valid DB2 username. In many cases, DB2 creates a username during installation, called `db2admin`, for administering DB2 and setting up the security for the users on your system.

> **Note** By default, system administration (`SYSADM`) privileges are granted toany valid DB2 username that belongs to the Administrators group on Windows NT.

You can change the users who have administrator privileges for each DB2 instance by changing the `SYSADM_GROUP` parameter. Before you do, however, you need to ensure that the group exists. To check this, use the Windows NT User Manager administrative tool (choose Start | Programs | Administrative Tools | User Manager). If the group exists, it will be listed in the lower section of the User Manager window.

To use another group as the system administrative group (`SYSADM_GROUP`), update the database manager configuration file. To change `SYSADM_GROUP` on the server instance, follow these steps:

1. In the Control Center, click the plus sign (+) beside the Systems icon to list all the systems known to your workstation, and then click the plus sign (+) for the system containing the instance you want to update.
2. Right-click the instance that you want to change—for example, DB2—and select Configure from the pop-up menu. The Configuration Instance notebook opens.
3. The Administration page shows the configuration parameters associated with administration (see Figure 15.1). In the System Administration Authority Group text box, type the name of an existing group to which you want to assign thisprivilege.



Parameter	Value	Changed	DB2 Parameter
System administration authority group			sysadm_group
System control authority group			sysctrl_group
System maintenance authority group			sysmaint_group
Authentication	0		authentication
Trust all clients	1		trust_allclnts
Trusted client authentication	0		trust_clntauth
[illegible]	1		[illegible]

System administration authority group

Defines the group name with system administration (SYSADM) authority for the instance. This is the highest level of authority within the instance, and controls all database objects.
Note: This parameter must be NULL for Windows 95 clients.

Figure 15.1: The Configuration Instance notebook.

4. Click OK.

5. Stop all applications that are using DB2, including the Control Center. When the application or the Control Center is restarted, the new value for SYSADM_GROUP is used.

You can use these same steps to change the SYSCTRL_GROUP and SYSMAINT_GROUP parameters.

Access Within DB2

Access within DB2 is managed by the administrative authorities and user privileges within the database manager. Database authorities involve actions on a database as a whole. When a database is created, some authorities are automatically granted to anyone who accesses the database. For example, CONNECT, CREATETAB, BINDADD, and IMPLICIT_SCHEMA authorities are granted to all users.

Authorities

DB2 Universal Database uses authorities as a way of grouping privileges and control over database objects. You can have administrative authorities that give full privileges for a set of objects, or you can have system authority that gives full privileges for managing the system but doesn't allow access to the data. A user or group can have one or more of the following levels of authority: SYSADM, DBADM, SYSCTRL, and SYSMAINT.

SYSADM

SYSADM, the highest level of administrative authority, controls all resources created and maintained by DB2 Universal Database. It includes all the privileges on all databases within the DB2 instance, as well as the authority to grant or revoke authority. When using DB2 on a Windows NT system, you assign and manage SYSADM authority with the Windows NT security facilities.

SYSADM authority is assigned to the group specified by the SYSADM_GROUP configuration parameter.

DBADM

The second highest level of administrative authority, DBADM applies only to a specific database and lets users run utilities, issue database commands, and access any data in any table in the database. When DBADM authority is granted, database privileges (described shortly) are granted as well.

Only users with SYSADM authority can grant or revoke DBADM authority. Users with DBADM authority can grant privileges on the database to others and can revoke any user's privilege, regardless of who granted it. Privileges include loading tables, reading log files, and creating, dropping, and activating event monitors.

SYSCTRL

SYSCTRL, the highest level of system control authority, allows operations that affect system resources only. Direct access to data isn't allowed. SYSCTRL authority is meant for users to administer a database instance containing sensitive data.

SYSCTRL authority is assigned to the group specified in the SYSCTRL_GROUP configuration parameter. Privileges include creating and dropping databases and table spaces; updating directory, history, and configuration files; and dropping, creating, or altering table spaces.

SYSMAINT

SYSMAINT, the second level of system control authority, allows system maintenance operations on all databases associated with an instance. Direct access to data isn't allowed. SYSMAINT authority is meant for users who administer a database instance that contains sensitive data.

SYSMAINT authority is assigned to the group specified in the SYSMAINT_GROUP configuration parameter. Privileges include updating database configuration files; backing up and restoring databases and table spaces; and issuing roll-forward, trace, and database monitor commands.

Privileges

Database privileges involve actions on specific objects within the database. When a database is created, some privileges are automatically granted to anyone who accesses the database. For example, the SELECT privilege is granted on catalog views, and EXECUTE and BIND privileges on each successfully bound utility are granted to all users.

A *privilege* is the right to access a specific database object in a specific way. These rights are controlled by users with SYSADM or DBADM authority or by creators of objects. Authorized users can create database objects and access the objects they own. They can also give privileges for their own objects to other users by using the Control Center. Only users with SYSADM or DBADM authority can grant and revoke these privileges.

Privileges can be held on the following database objects: databases, tables, views, packages, and indexes. The following privileges, for example, can be held on a database and are granted to PUBLIC when a database is created:

- Connect to the database (CONNECT).
- Create packages (BINDADD).
- Create tables (CREATETAB).
- Create a schema implicitly when creating another object (IMPLICIT_SCHEMA).
- Query data in the system catalog views (SELECT).

Privileges not requiring specific authority include reading system and database configuration files, directories, and history files, and attaching to a DB2 instance.

Individual privileges can be set to allow users to perform specific functions on selected objects.

You can grant implicit privileges to users who have the privilege to execute a package. Although users can run the application, they don't necessarily require explicit privileges on the database objects used within the package.

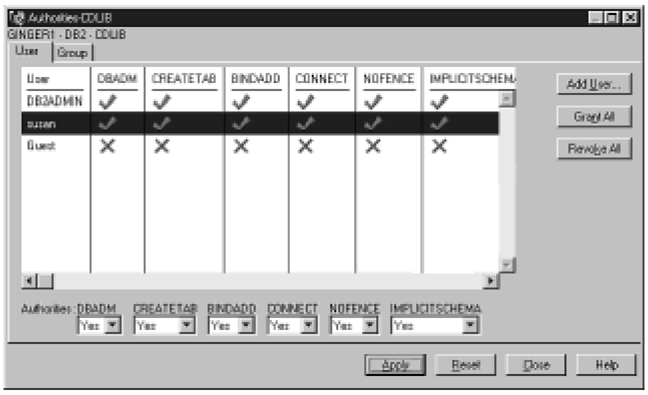
Together, privileges and authorities act to control access to an instance and its database objects. Users can access only those objects for which they have the appropriate authorization—that is, the required privilege or authority.

Granting and Revoking Privileges and Authorities

You can grant authorities at the database level and privileges for most other objects. To grant privileges and authorities to other users on the system, or to revoke them, follow these steps:

1. In the Control Center, expand the objects until you see the CDLIB database.
2. Right-click the CDLIB database icon, and select Authorities from the pop-up menu. The Authorities – CDLIB window opens, listing the users that now have privileges assigned to work with objects in the CDLIB database. By default, the instance owner has privileges on all objects.
3. To add additional users to the list, click Add User, select one or more users from the list, and click Add. Click Close when you've finished selecting users, to return to the Authorities CDLIB window.

The Authorities window shows the users you've just added, along with each one's privileges. A check mark in a column means that the user has the privilege; an `X` means that the user doesn't have the privilege (see Figure 15.2). You can grant access to `DBADM`,`CREATETAB`, `BINDADD`, `CONNECT`, `NOFENCE`, and `IMPLICITSCHEMA`.



User	DBADM	CREATETAB	BINDADD	CONNECT	NOFENCE	IMPLICITSCHEMA
DB2ADMIN	✓	✓	✓	✓	✓	✓
susan	✓	✓	✓	✓	✓	✓
Guest	✗	✗	✗	✗	✗	✗

Authorities: DBADM Yes, CREATETAB Yes, BINDADD Yes, CONNECT Yes, NOFENCE Yes, IMPLICITSCHEMA Yes

Figure 15.2: The Authorities window.

Tip You can also grant privileges to an entire group. First, use the Windows NT User Manager to assign users to a group. Then, use the Group page of the Authorities window to grant access to the group.

Security for Data Access

Access to a database or instance is first validated outside DB2. This process, known as *authentication*, verifies that the user is who he or she claims to be.

An authentication type is specified for a database instance when it's created or cataloged at the server and when it's cataloged on a remote node that will be accessing the database. The authentication type determines where and how users are verified for access to databases on the server. The authentication type is stored in the database manager configuration file at the server and can be `CLIENT`, `SERVER`, or `DCS`.

SERVER

This is the default authentication type. If the type is `SERVER`, the user will be verified at the server where the database resides, using local operating-system security. If a username and password are specified during the connection attempt, they're compared to the valid username and password combinations at the server to determine if the user is permitted to access the instance.

CLIENT

If `CLIENT` is specified, it's assumed that the user was verified on the client, where the application resides. No further authentication will take place at the server.

Specify Trusted Clients in the database manager configuration file to choose whether to trust all clients (the default) or trust only those clients from systems where security is inherent in the operating system. Trusted clients are clients that have a reliable, local security system. Specifically, all clients are trusted except Macintosh, Windows 3.1, and Windows 95 clients.

DCS

If `DCS` is specified, verification will be passed to the host database management system. If `DCS` is specified, but DB2 Connect isn't involved in the data access, verification will occur where the database resides.

DB2 supports the Open Software Foundation's (OSF) Distributed Computing Environment (DCE) Security component for use in authentication of database users. This provides a more secure authentication mechanism and more central management of users, passwords, and groups using the DCE architecture.

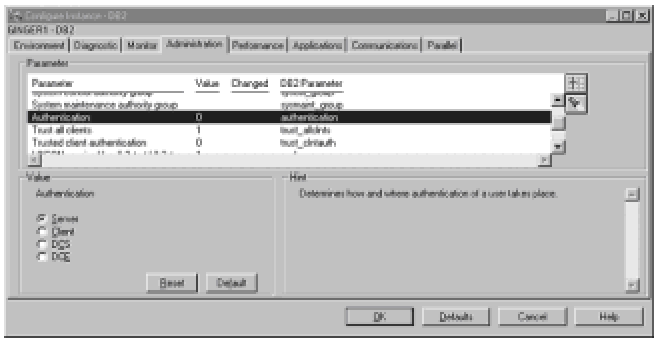
Setting the Authentication Level

The authentication level is set at the instance level. To view or modify the authentication level for the DB2 instance, follow these steps:

1. In the Control Center, right-click the DB2 instance object, and select Configure from the pop-up menu. The Configure Instance – DB2 notebook opens.

2. Click the Administration tab to view the Administration-related configuration parameters now set for the instance.

3. Scroll down to the Authentication parameter and select it. A set of radio buttons give you a choice of Server (the default), Client, DCS, or DCE authentication (see Figure 15.3). For this example, use Client authentication.



Parameter

Parameter	Value	Changed	DB2 Parameter
System maintenance authority group			sysmaint_group
Authentication	0		authentication
Trust all clients	1		trust_allclnts
Trusted client authentication	0		trust_clntauth

Value

Authentication

[✓ Server] Client DCS DCE

Hint

Determines how and where authentication of a user takes place.

Figure 15.3: The Administration page of the Configure Instance – DB2 notebook.

4. Click OK to save your settings, and close the notebook.

5. For your changes to take effect, you must stop the instance and restart it. Right-click the instance, and select Stop from the pop-up menu. To restart the instance, right-click the instance, and select Start from the pop-up menu.

From the Parameter list, you can also select the Trust All Clients parameter to specify whether to trust all the clients. You can select Trusted Client Authentication to specify whether a trusted client is authenticated at the client or the server. (Trusted Client Authentication is available only when Client authentication is selected.)

Controlling Access to Data with Views

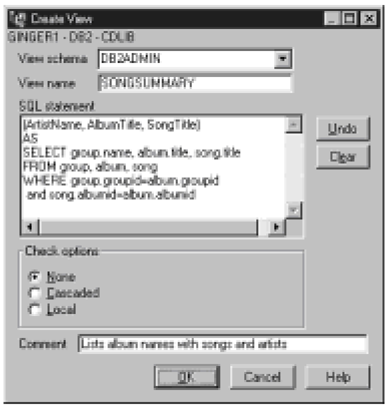
You might also want to create a view to restrict access. You can define a view so that it includes some or all rows in the table. For example, you can create a view that allows each department manager to look at the salary information for employees. This is done by specifying the department number for each manager when you define each view.

To create a view, use the Control Center as follows:

1. Expand the object until you see the View folder within the CDLIB database.

2. Right-click the View folder, and select Create from the pop-up menu.

3. In the Create View window (see Figure 15.4), select a schema for the view from the View Schema drop-down list. For the example, select the `DB2ADMIN` schema.



GINGER1 - DB2 - CDLIB

View schema: DB2ADMIN

View name: SONGSUMMARY

SQL statement

```
(ArtistName, AlbumTitle, SongTitle)
AS
SELECT group.name, album.title, song.title
FROM group, album, song
WHERE group.groupid=album.groupid
 and song.albumid=album.albumid
```

Check options

[✓ None] Cascaded Local

Comment: Lists album names with songs and artists

Figure 15.4: Creating a view.

4. In the View Name text box, enter a name for the view— for example,**SONGSUMMARY**. The name must be 18 characters or fewer and must be unique within the view's schema.

5. Enter a SQL statement that defines the view. For example, use the following query to view the songs that appear on an album and the group that released the album:

Input
```
(ArtistName, AlbumTitle, SongTitle)
AS SELECT group.name, album.title, song.title
FROM group, album, song
WHERE group.groupid = album.groupid and
song.albumid = album.albumid
```

6. In the Check Options section, specify how the constraint conforms to the definition of the view. A row that doesn't conform to the view's definition does not satisfy the view's search conditions:

   - Use None (the default) if the definition of the view isn't used in the checking of insert or update operations. For this example, stay with the default.

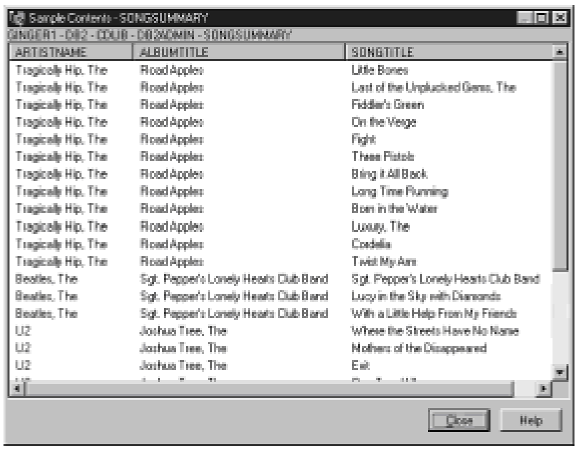
   - Select Cascaded if the constraint on a new view inherits the search conditions as constraints from an updatable view on which the new view depends.

   - Select Local to specify that the search condition used to create the view is used as a constraint for insert or update operations.

7. Click OK to create the view.

8. To sample the contents of a view, right-click the view you've just created, and select Sample Contents from the pop-up menu.

A view provides a means of controlling access or extending privileges to a table by allowing access to only designated columns of a table or to only a subset of the rows in a table. In Figure 15.5, you see only three columns and only the rows that satisfy the condition of the `WHERE` clause.


Sample Contents - SONGSUMMARY
GINGER1 - DB2 - CDLIB - DB2ADMIN - SONGSUMMARY
Close
Help

ARTISTNAME	ALBUMTITLE	SONGTITLE
Tragically Hip, The	Road Apples	Little Bones
Tragically Hip, The	Road Apples	Last of the Unplucked Gems, The
Tragically Hip, The	Road Apples	Fiddler's Green
Tragically Hip, The	Road Apples	On the Verge
Tragically Hip, The	Road Apples	Fight
Tragically Hip, The	Road Apples	Three Pistols
Tragically Hip, The	Road Apples	Bring it All Back
Tragically Hip, The	Road Apples	Long Time Running
Tragically Hip, The	Road Apples	Born in the Water
Tragically Hip, The	Road Apples	Luxury, The
Tragically Hip, The	Road Apples	Cordelia
Tragically Hip, The	Road Apples	Twist My Arm
Beatles, The	Sgt. Pepper's Lonely Hearts Club Band	Sgt. Pepper's Lonely Hearts Club Band
Beatles, The	Sgt. Pepper's Lonely Hearts Club Band	Lucy in the Sky with Diamonds
Beatles, The	Sgt. Pepper's Lonely Hearts Club Band	With a Little Help From My Friends
U2	Joshua Tree, The	Where the Streets Have No Name
U2	Joshua Tree, The	Mothers of the Disappeared
U2	Joshua Tree, The	Exit

Figure 15.5: Sample contents of the view.

Using a Backup Domain Controller

In a Windows NT LAN environment, you can have users authenticated at the primary or backup domain controller. A company with a large distributed LAN might have an environment with one central primary domain controller and one or more backup domain controllers at each site. This way, users can be authenticated on the backup domain controller at their site, rather than with a required call to the primary domain controller. The advantage of a backup domain controller, in this case, is that users are authenticated faster and the LAN isn't as congested as it would be, were there no backup domain controller.

For DB2 authentication at the backup domain controller, you must do the following:

1. Install your DB2 server on the backup domain controller.

2. Set the `DB2DMNBCKCTLR` profile registry value. To perform DB2 authentication on the backup domain controller, set this registry value as follows:

   ```
   db2set db2dmnbckctlr = ?
   ```

3. Ensure that the Cached Primary Domain under the Registry Editor is the domain at which the machine is located. (You can find this setting under `HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\Current Version\WinLogon`.)

4. Ensure that Server Manager shows the backup domain controller as active and available.

The Registry for the DB2 for Windows NT server indicates that the system is a backup domain controller on the specified domain.

Authentication with Groupsand Domain Security

DB2 supports several types of Windows NT groups, including local groups, global groups, and global groups as members of local groups. DB2 enumerates the local and global groups that the user is a member of, using the security database where the user was found. DB2 provides an override that forces group enumeration to occur on the local Windows NT server where DB2 is installed, regardless of where the user account was found.

To achieve this override globally, enter the following from a command window:

Input **`db2set -g db2_grp_lookup=local`**

To achieve this override for a specific instance, like DB2, enter the following from a command window:

Input **`db2 -i DB2 db2_grp_lookup=local`**

For DB2 for Windows NT to work with domain security, you must grant authority and privileges to a local group. To be authenticated correctly, usernames within the local and global groups must be defined on the same domain as the local or global group. Update the `SYSADM_GROUP` parameter in the database manager configuration file on the server to reflect the local group that was defined on the domain controller.

If the `db2_grp_lookup` variable is set to local, DB2 tries to find a user on the local machine only. If the user is found on the local machine or isn't defined as a member of the local or global group, authentication fails. DB2 doesn't try to find the user on another machine in the domain or on the domain controllers.

If the `db2_grp_lookup` variable isn't set (the default), DB2 tries to find the user on the same machine. If the username is defined locally, the user is authenticated locally. If the user isn't found locally, DB2 attempts to find the username on its domain and then on trusted domains.

Summary

In this lesson, you saw that DB2 includes levels of security that control the access to the database and within the database. Access to DB2 for Windows NT is controlled by the Windows NT operating system security mechanism. Access within DB2 is controlled through authorities and privileges that are set for most objects in the database.

Through authorities, DB2 groups privileges and control over database objects.

Authentication is the process used to ensure that a user is who he or she claims to be. You set an authentication type at the instance level in the database manager configuration file.

Also, you can restrict access to data through the use of views.

What Comes Next?

On Day 16, "Administering Clients with the CCA," you learn how to administer a large number of clients by using the Client Configuration Assistant.

Q&A

Q What are the four levels of authority?

A DB2 uses four levels of authority to group privileges and control over database objects. `SYSADM`, the highest level of authority, provides control over all resources created and maintained by DB2. `DB2ADM` authority provides the second highest level of authority and applies only to a specific database. The `SYSCRTL` and `SYSMAINT` authorities are provided to allow access to the system resources but not the data.

Q What authentication types can I set for an instance?

A The default authentication type is `SERVER`. This means that users are verified at the server where the database resides. `CLIENT` authentication can be used to verify users at the client machine, with no further authentication occurring at the server. `DCS` authentication is used when you have DB2 Connect installed and are trying to access data on a host database system. This means that the verification will occur where the database resides.

Q Why should I use views?

A One reason is to restrict access to certain rows or columns in a database. For example, you have a personnel database that contains each employee with employee salaries. You'll want only authorized people to see the salary for each employee. You might want to set it up so that managers can view the salaries of their own employees but not the salaries of employees outside their departments. You might also want a single person in the human resources department to have access to all salaries in the company. You can also create views so that each employee can see his own salary but no one else's in the company. You can restrict the data in many ways so that sensitive information is available to only those with the proper authority.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. If you want to create databases and assign authorities to other users, what authority should you have?
2. Describe the importance of implicit versus explicit privileges.
3. What authorities are granted to all users when the database is created?
4. What is the default for the Trusted Clients option? What does this mean?

Exercise

Create several users who have different levels of authorities and attempt to access data from each user. Set up different views for each user to see how you can restrict access to certain pieces of information.

Day 16: Administering Clients with the CCA

Overview

In this lesson, you see how to use the Client Configuration Assistant (CCA) to perform administration tasks on client workstations. When you install the DB2 server, if your protocols are set up and configured, DB2 server communications are automatically configured to enable DB2 to accept requests from remote clients.

With the CCA, you can do the following:

- Configure database connections that applications can use.
- Update or delete existing configured database connections.
- Display the information for existing configured connections.
- Test a connection to a database.
- Enable or disable database connections to be configured as CLI/ODBC data sources.
- Import or export client profiles, which contain information for the setup of a client.
- Update client configuration settings.
- Discover remote databases (if enabled).
- Bind user applications and utilities to databases.

Tip If the CCA isn't installed on your system, you can install it by reinstalling the DB2 product and choosing to install the CCA. Installing the CCA is covered in Day 6, "Installing DB2 Clients."

Configuring Database Connections

You can configure database connections through the Add Database SmartGuide by any of the following:

- Using a profile as a source of information to add database connections
- Searching the network for databases
- Adding database connections manually

Each of these techniques is covered in more detail in the following sections.

> **Note** To complete the steps in the following sections, you must be logged on to the local system as a user with system administration (`SYSADM`) authority on the instance.

Using an Access Profile

An *access profile* is a file containing all the necessary information for a client to access a remote server. Using a profile to set up access enables you to create a single profile for access and easily propagate this information to other clients in your network. You can use a server profile or a client profile to configure database connections on a client.

Server Profiles

Server profiles can be generated for a DB2 server. They contain information about instances on the server system and databases within each instance. The information for each instance includes the protocol information required to connect a client to databases in that instance.

To generate server profiles, use the Generate Access Profile function provided in the Control Center. When a profile is generated for a DB2 server system, it includes server instances that have the `DISCOVER_INST` configuration parameter set to `ENABLE` and databases with the `DISCOVER_DB` configuration parameter set to `ENABLE`. In the Administration Server, the `DISCOVER` parameter must be set to `SEARCH` or `KNOWN` to generate a profile for a server system.

For information on setting the `DISCOVER_INST`, `DISCOVER_DB`, and `DISCOVER` configuration parameters, see "Setting Discovery Parameters" later in this chapter.

Generating a Server Profile You need to generate a server profile only if you want to set up clients to access this server. At least one database should be created on your server before creating a profile.

To generate an access profile, perform the following steps on the machine where your databases are located:

1. Start the Control Center.
2. Click the plus sign (+) beside the Systems folder to list the systems.
3. Right-click the system to be profiled, and select Generate Access Profile from the pop-up menu. The Generate Access Profile window appears (see Figure 16.1).

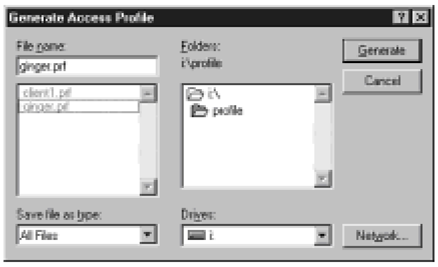


Figure 16.1: The Generate Access Profile window.

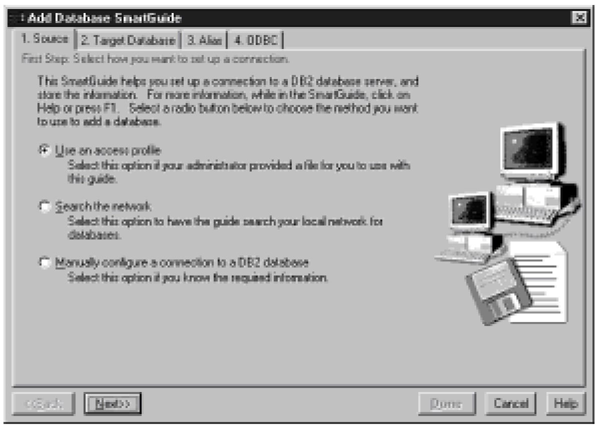
4. Select the path, type in a filename for the profile, and then click the Generate button. You should put the file in a location that your clients can access, such as a network drive or an FTP site. For the example in this lesson, I used the name `i:\profile\ginger.prf`.

To process a server profile and add its databases to the client's connection configuration list, use the Client Configuration Assistant's Add functions on the remote client machine to select the database connection data in the server profile that you want to add to the client. (This function invokes the Add Database SmartGuide, covered in Day 6, "Installing DB2 Clients.")

Using a Server Profile on the Client After you create an access profile on theserver, you need to add the information to each client system that you want to give access to. Use the Add function to add the database. Follow these steps:

1. Start the CCA by choosing Start | Programs | DB2 for Windows NT | Client Configuration Assistant.

2. Click the Add button to open the Add Database SmartGuide (see Figure 16.2).



First Step: Select how you want to set up a connection.

This SmartGuide helps you set up a connection to a DB2 database server, and store the information. For more information, while in the SmartGuide, click on Help or press F1. Select a radio button below to choose the method you want to use to add a database.

[✓ Use an access profile]
Select this option if your administrator provided a file for you to use with this guide.

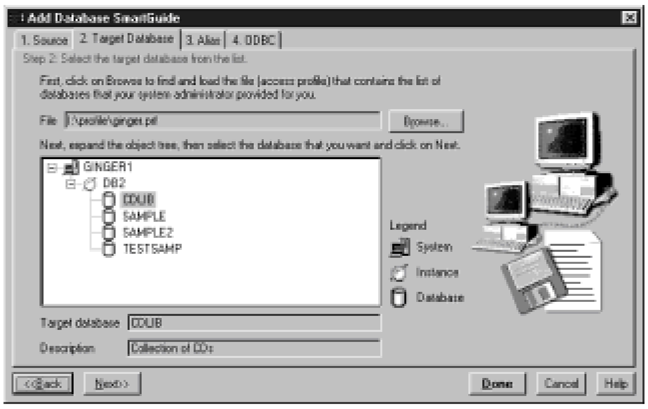
Search the network
Select this option to have the guide search your local network for databases.

Manually configure a connection to a DB2 database
Select this option if you know the required information.

Figure 16.2: The Add Database SmartGuide.

3. Select Use an Access Profile, and click the Next button.

4. On the Target Database page (see Figure 16.3), click the Browse button to select the server profile that you want to access, or enter the path and filename in the File field. Select the server profile that you created in the preceding section. Click OK to return to the Add Database SmartGuide, where you'll now see a listing of the available databases for the server system where the profile was created.



Step 2: Select the target database from the list.

First, click on Browse to find and load the file (access profile) that contains the list of databases that your system administrator provided for you.

File: I:\profile\ginger.prf

Next, expand the object tree, then select the database that you want and click on Next.

```
GINGER1
  DB2
    CDLIB
    SAMPLE
    SAMPLE2
    TESTSAMP
```

Target database: CDLIB

Description: Collection of CDs

Figure 16.3: The Target Database page of the Add Database SmartGuide.

5. In the object tree, select a database to be added.

6. If desired, you can select the Alias and ODBC tabs to specify a database alias name for the database or to select CLI/ODBC options for the database.

7. Click Done after you complete all the necessary information.

8. In the confirmation window that appears, test the connection by clicking the Test Connection button. If the connection is successful, you'll receive information about the connection and the server.

Client Profiles

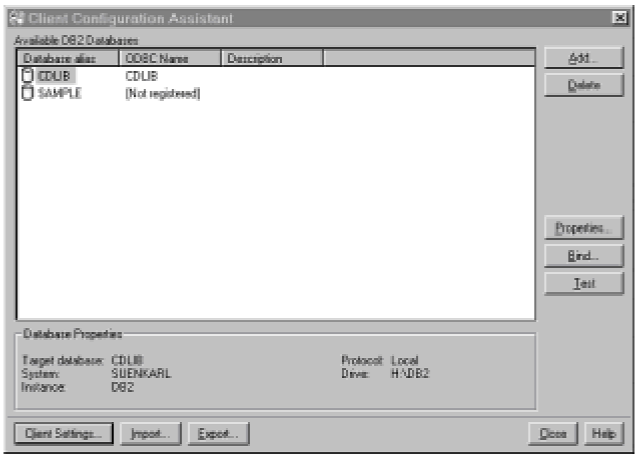
You generate client profiles from clients by using the CCA's Export function. Database connections configured on one client are exported and used to set up one or more additional clients. Export can be used to generate a customized profile that can be imported on another client to set it up initially, or to update it.

The information contained in a client profile is determined during the export process. Depending on the settings chosen, it can contain the following for the existing clients:

- Database connection information, including CLI/ODBC settings
- Client settings, including database manager configuration parameters
- CLI/ODBC common parameters
- Configuration data for the local APPC communications subsystem

Exporting a Client Profile Follow these steps to export the information from your fully configured client:

1. Enter **db2cca admin** at a command prompt to start the CCA in administrator mode. (You must be an administrator on the local system to run this command.) Figure 16.4 shows the Client Configuration Assistant open in administrator mode.


Client Configuration Assistant

Available DB2 Databases

Database alias	ODBC Name	Description
CDLIB	CDLIB	
SAMPLE	[Not registered]	

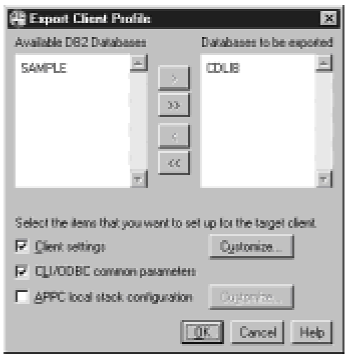
Database Properties

Target database: CDLIB
System: SUENKARL
Instance: DB2
Protocol: Local
Drive: H:\DB2

Figure 16.4: The Client Configuration Assistant in administrator mode.

Note The Export function of the CCA is available only when the CCA is startedin administrator mode. The CCA can be started in administrator mode by adding the parameter `admin` to the Client Configuration Assistant shortcut properties or by issuing the `db2cca admin` command in a commandwindow.

2. Click the Export button to open the Export Client Profile window(see Figure 16.5).


Export Client Profile

Available DB2 Databases: SAMPLE

Databases to be exported: CDLIB

Select the items that you want to set up for the target client.

- [✓] Client settings
- [✓] CLI/ODBC common parameters
- [ ] APPC local stack configuration

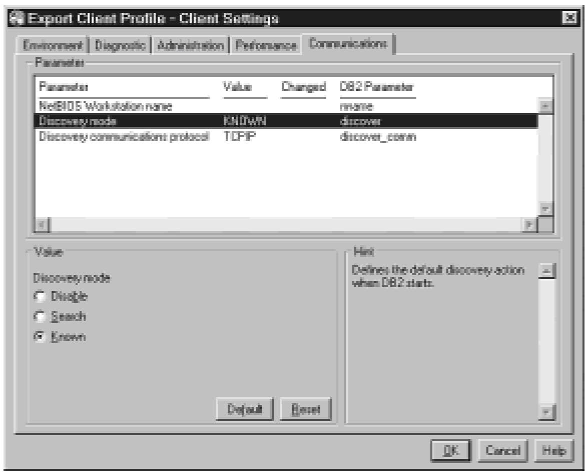
Figure 16.5: The Export Client Profile window.

3. Select the databases to be exported from the Available DB2 Databases list, and click the right arrow to add them to the Databases to Be Exported list.

4. Select the check boxes that correspond to the options that you want to set up for the target client:

- Client settings
- CLI/ODBC common parameters
- APPC local stack configuration

5. Click the Customize push button to open the Export Client Profile – Client Settings notebook (Figure 16.6 shows the Communications page). The settings that you customize affect only the profile to be exported; no changes will be made to your workstation. Use the onscreen hints to help you set the various client settings. Click OK to return to the Export Client Profile window when you complete any customization.



Parameter

Parameter	Value	Changed	DB2 Parameter
NetBIOS Workstation name			nname
Discovery mode	KNOWN		discover
Discovery communications protocol	TCPIP		discover_comm

Value

Discovery mode
- ( ) Disable
- ( ) Search
- (●) Known

Hint

Defines the default discovery action when DB2 starts.

Figure 16.6: The Export Client Profile – Client Settings with the Communications page displayed.

6. Click OK in the Export Client Profile window when you've customized the settings. A dialog box for the profile name and location appears.

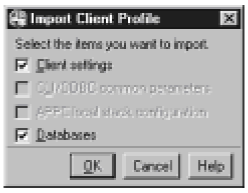
7. In the filename text box, enter a path and filename for the client profile—for example, `i:\profile\client1.prf`. Click OK to export the profile to the specified name and location.

Importing a Client Profile Perform the following steps at the client that you want to set up. You can use this process to initially set up a new client or to update an existing one:

1. Start the CCA.

2. Click the Import button to open the Import Client Profile window.

3. Select the path and filename of the client profile you want to import, and click OK. For example, to import the profile that was exported in the previous section, select `i:\profile\client1.prf`. The Import Client Profile window opens (see Figure 16.7).



Select the items you want to import.
- [✓] Client settings
- [ ] CLI/ODBC common parameters
- [ ] APPC local stack configuration
- [✓] Databases

Figure 16.7: The Import Client Profile window.

4. Select the items you want to import:

   • **Client Settings.** Configuration parameters found in the exported file replace existing ones on the client; all other parameters are set to their default values.

   • **CLI/ODBC Common Parameters.** Values found in the exported file replace those on the client.

- **Local Configuration Data for the APPC Communications Subsystem.** If it's not already configured, this information is used to configure the APPC communications subsystem.

5. Click OK to import the settings.

6. If databases are contained in the client profile that you're importing, and you select to import them, the Add Database SmartGuide opens, enabling you to import the database catalog information.

7. Test the connection to the database by clicking the Test button in the main Client Configuration Assistant window.

Searching the Network for Databases

Rather than enter protocol information to make a connection to remote database servers, you can use the CCA to find all the databases on your local network.

> **Note** The following scenario assumes that the installation defaults on the client and the server haven't been changed and that messages used by the `SEARCH` method of discovery aren't filtered by your network.

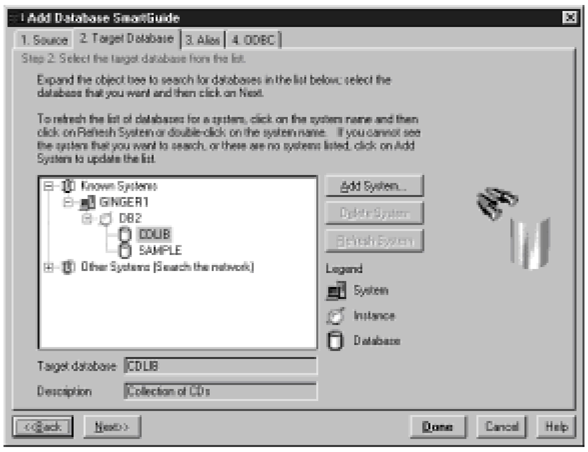
Follow these steps:

1. Start the CCA.

2. Click the Add button to start the Add Database SmartGuide.

3. Select Search the Network, and click the Next button.

4. Click the + sign beside the Known Systems icon to list all the systems known to your client, and then click the plus sign (+) beside the system to get a list of the instances and databases on it (see Figure 16.8). Select the database that you want to add.


Step 2: Select the target database from the list.

Expand the object tree to search for databases in the list below; select the database that you want and then click on Next.

To refresh the list of databases for a system, click on the system name and then click on Refresh System or double-click on the system name. If you cannot see the system that you want to search, or there are no systems listed, click on Add System to update the list.

Target database: CDLIB

Description: Collection of CDs

Figure 16.8: The Add Database SmartGuide's known systems.

5. If desired, on the Alias and ODBC pages you can specify a database alias name for the database or select CLI/ODBC options for the database.

6. Click the Done button.

7. In the confirmation window that appears, test the connection that has been added by clicking the Test Connection button.

Note If the system that contains the database you want isn't listed, click the plus sign beside Other Systems (Search the Network) to search the network for additional systems. Then click the plus sign beside the system to get a list of the instances and databases on it, and select the database you want to add.

If the system you want still isn't listed, you can add it to the list of systems by clicking the Add System button. Select a protocol you want to use to connect to the system, and complete the protocol information.

Choosing a Discovery Method

Network searches can be customized to meet the needs of individual organizations. Network searching uses the DB2 Discovery facility to obtain information from DB2 servers. This information is used to configure clients for database connections. Two discovery methods are available for searching the network: Known and Search.

Known discovery enables you to find instances and databases on systems known to your client and allows you to add new systems so that their instances and databases can be discovered. However, it doesn't support searching the network for servers.

When you click the + sign beside the Known Systems icon, you see a list of knownDB2 server systems. Click the + sign beside the system for a list of the instances and databases on it.

Initially, the list of systems will be blank; however, if you're running the CCA on the server, an entry for the local server will be shown. Add systems to the list by clicking the Add System button. To use this option, you must know a few details about the DB2 Administration Server on the system to be searched:

- A protocol configured and running on the DB2 Administration Server
- The protocol's configuration information

To see the set protocols and the discovery method used, run the command `get admin cfg` from the Command Center. The DB2 Administration Server will listen for `KNOWN` discovery requests from clients on the protocols specified by the `DB2COMM` registry value in the Administration Server.

The *Search discovery* mode provides all the functions described for the Known discovery mode but enables your local network to be searched for DB2 servers. Searching doesn't require information about the Administration Server. When you click the + sign beside the Other Systems (Search the Network) icon, a list of DB2 server systems appears. Click the + beside the system for a list of the instances and databases on it.

Tip The Search discovery mode might appear to be a simpler method. In larger networks, however, network routers and bridges can filter the messages Search uses to find DB2 servers on the network, resulting in an incomplete or even empty list. In this case, use the Add System method; its messages aren't filtered by routers and bridges. If in doubt, contact your network administrator for assistance.

To have the server support Known discovery, set the `DISCOVER` parameter in the Administration Server to `KNOWN`. To have it support Search discovery, set this parameter to `SEARCH`. To prevent discovery of the server, and all its instances and databases, set `DISCOVER` to `DISABLE`.

On the client, enabling discovery is also done by using the `DISCOVER` parameter;however, in this case, the `DISCOVER` parameter is set in the client instance (or a server acting as a client) as follows.

Tip To see the values now set, start the CCA, click the Client Settings button, select the Communications tab, and look at the `DISCOVER` parameter.

`KNOWN` The `KNOWN` setting enables the CCA to refresh systems in the known list and to add new systems to the list through the Add Systems button. When `DISCOVER` is set to `KNOWN`, the CCA can't search the network.

Note Servers configured with `DISCOVER` set to `KNOWN` won't respond to search requests from clients. It's important that you consider this when changing the `DISCOVER` parameter, which was set to `SEARCH` during installation.

`SEARCH` The `SEARCH` setting enables all the facilities of the `KNOWN` setting, plus enables network searching.

`DISABLE` The `DISABLE` setting disables discovery. In this situation, the Search the Network option isn't available in the Add Database SmartGuide.

Setting Discovery Parameters

The following sections explain how each discovery parameter is set and used.

Additional Settings for Search Discovery Search discovery requires that the configuration parameter `DISCOVER_COMM` be set on the server (in the Administration Server's configuration file) and the client (in the database manager configuration file).

The `DISCOVER_COMM` parameter is used to control the communication protocols that the server will listen on for search requests from clients and that clients will use to send out search requests. The `DISCOVER_COMM` parameter can be any combination of TCP/IP and NetBIOS.

Note Search discovery operates only on TCP/IP and NetBIOS.

On the server, the values specified by `DISCOVER_COMM` must be equal to or a subset of the values set by `DB2COMM` for the Administration Server.

Tip To see the current value for the `DB2COMM` parameter, issue the `get admin cfg` command in the Command Center.

On the server, `DISCOVER_COMM` is set in the Administration Server. On the client (or a server acting as a client), `DISCOVER_COMM` is set in the instance.

Note When using Search discovery mode, at least one protocol specified by the `DISCOVER_COMM` parameter on the client must match those specified by `DISCOVER_COMM` on the Administration Server. If there's no match, theserver won't respond to the client's requests.

Also, you can use two DB2 profile registry values to tune Search discovery on the client: `DB2DISCOVERYTIME` and `DB2NBDISCOVERRCVBUFS`. The default values should be suitable in most cases.

Hiding Server Instances and Databases from Discovery You can have multiple instances, and multiple databases within these instances, on a server. You might want to hide some instances or databases from the discovery process.

To allow clients to discover server instances on a system, set the `DISCOVER_INST` database manager configuration parameter in each server instance on the system to `ENABLE` (the default value). Set this parameter to `DISABLE` to hide this instance and its databases from discovery.

To allow a database to be discovered from a client, set the `DISCOVER_DB` database configuration parameter to `ENABLE` (the default value). Set this parameter to `DISABLE` to hide the database from discovery.

`DISCOVER` and `DISCOVER_COMM` The `DISCOVER` and `DISCOVER_COMM` parameters are set in the Administration Server on the server system and in the client instance.

To view the current settings on the Administration Server, issue the `get admin db configuration` command from the Command Center. Update the Administration Server's configuration file in the command line processor as follows:

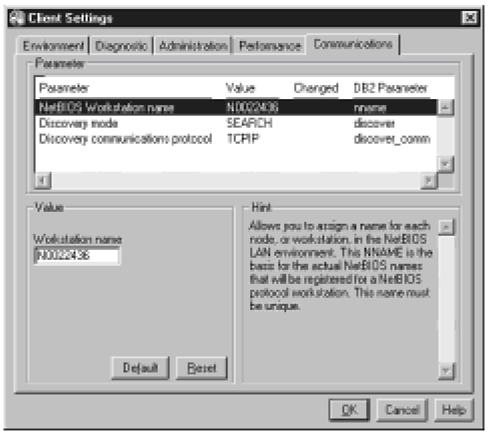
```
update admin cfg using discover [ DISABLE | KNOWN | SEARCH ]
update admin cfg using discover_comm [ NETBIOS | TCPIP ]
db2admin stop
db2admin start
```

On the client, set these parameters as follows:

1. Start the CCA.

2. Click the Client Settings button.

3. On the Communications page, select and change the parameters that you want to modify in the Parameter section (see Figure 16.9).



Parameter	Value	Changed	DB2 Parameter
NetBIOS Workstation name	N0022436		nname
Discovery mode	SEARCH		discover
Discovery communications protocol	TCPIP		discover_comm

Workstation name: N0022436

Allows you to assign a name for each node, or workstation, in the NetBIOS LAN environment. This NNAME is the basis for the actual NetBIOS names that will be registered for a NetBIOS protocol workstation. This name must be unique.

Figure 16.9: The Communications page for client settings.

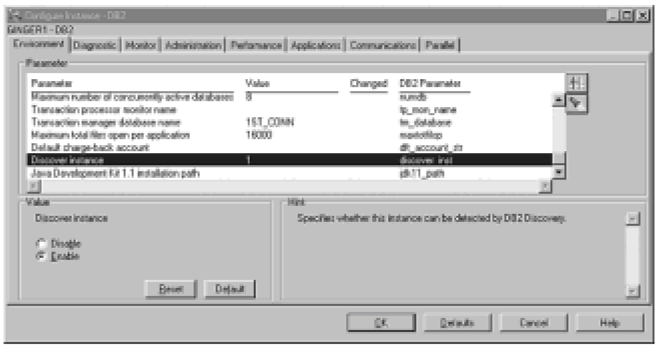
If `DISCOVER_COMM` includes NetBIOS, you must ensure that the workstation name (`nname`) parameter is set for the client and the Administration Server. Also, you must ensure that the `DB2NBADAPTERS` registry value is set to the adapter number that you want to use.

`DISCOVER_INST` and `DISCOVER_DB` Use the Control Center to set the `DISCOVER_INST` and `DISCOVER_DB` parameters on the server system. To set these parameters, follow these steps:

1. Start the Control Center.

2. Right-click the instance that you want to configure, and select Configure from the pop-up menu.

3. In the Configure Instance notebook, select the Environment tab, and select the `DISCOVER_INST` parameter (see Figure 16.10).


Configure Instance - DB2
GINGER1 - DB2
Environment | Diagnostic | Monitor | Administration | Performance | Applications | Communications | Parallel
Parameter
Value
Hint
Reset
Default
OK
Defaults
Cancel
Help

Parameter	Value	Changed	DB2 Parameter
Maximum number of concurrently active databases	8		numdb
Transaction processor monitor name			tp_mon_name
Transaction manager database name	1ST_CONN		tm_database
Maximum total files open per application	16000		maxtotfilop
Default charge-back account			dft_account_str
Discover instance	1		discover_inst
Java Development Kit 1.1 installation path			jdk11_path

Discover instance

Disable [✓ Enable]

Specifies whether this instance can be detected by DB2 Discovery.

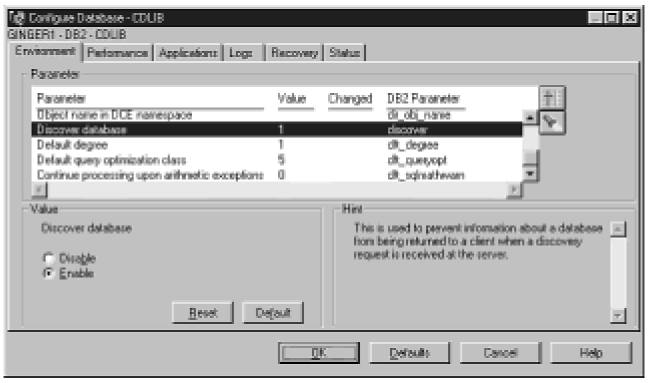
Figure 16.10: The Environment page of the Configure Instance notebook.

4. Choose to enable or disable the Discover instance (Enable is selected by default). Then click OK.

5. Right-click the database that you want to configure, and select Configure from the pop-up menu.

6. On the Environment page of the Configure Database window, select the `DISCOVER` parameter (see Figure 16.11).

7. Choose to enable or disable the Discover database (Enable is selected by default). Then click OK.


Configure Database - CDLIB
GINGER1 - DB2 - CDLIB
Environment | Performance | Applications | Logs | Recovery | Status
Parameter
Value
Hint
Reset
Default
OK
Defaults
Cancel
Help

Parameter	Value	Changed	DB2 Parameter
Object name in DCE namespace			dir_obj_name
Discover database	1		discover
Default degree	1		dft_degree
Default query optimization class	5		dft_queryopt
Continue processing upon arithmetic exceptions	0		dft_sqlmathwarn

Discover database

Disable [✓ Enable]

This is used to prevent information about a database from being returned to a client when a discovery request is received at the server.

Figure 16.11: The Environment page of the Configure Database – CDLIB notebook.

`DB2DISCOVERYTIME` and `DB2NBDISCOVERRCVBUFS` The `DB2DISCOVERYTIME` and `DB2NBDISCOVERRCVBUFS` profile registry values are set in the client instance (or a server acting as a client). Set these parameters as follows:

- To set the `DB2DISCOVERYTIME` registry value to 35 seconds, enter the following command to specify that the searched discovery wait 35 seconds for a response from servers:

Input `db2set db2discoverytime=35`

- To set the `DB2NBDISCOVERRCVBUFS` registry value to `10`, enter the following command:

  Input `db2set db2nbdiscoverrcvbufs=10`

  This specifies the number of NetBIOS buffers that will be allocated for response messages from discovered servers.

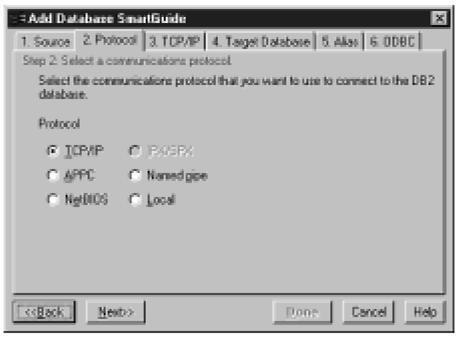
Manually Configuring a Connectionto a DB2 Database

Manually configuring a database connection requires you to know the following:

- One of the protocols supported by the server instance containing the database
- The protocol connection information required to configure the connection to the server instance
- The name of the database on the server system

With this information, the SmartGuide will guide you through the steps necessary to add the database connection. Follow these steps:

1. Start the CCA.
2. Click the Add button on the CCA's main panel to start the Add Database SmartGuide.
3. Select Manually Configure a Connection to a DB2 Database, and click the Next button.
4. Select the protocol that you'll use to connect to the database (Figure 16.12 shows TCP/IP selected). Click the Next button.



Step 2: Select a communications protocol.

Select the communications protocol that you want to use to connect to the DB2 database.

Protocol

[✓ TCP/IP] IPX/SPX APPC Named pipe NetBIOS Local

Figure 16.12: Manually configuring a connection with the Add Database SmartGuide.

5. Type the required protocol parameters, and click Next.
6. Type the name of the target database.
7. If desired, you can use the Alias and ODBC pages to specify a database alias name for the database or to select CLI/ODBC options for the database.

8. Click the Done button.

9. In the confirmation window that appears, test the connection by clicking the Test Connection button.

Summary

In this lesson, you learned about the Client Configuration Assistant (CCA). The CCA comes with DB2 servers or clients that run on Windows NT, Windows 95, or OS/2.

The CCA enables you to automatically catalog remote databases by selecting from a list of known databases on the network. You can generate profiles of database information that are used to simplify the setting up of multiple clients.

If you are an administrator, you can set the discovery level to be used at the clients to determine whether the network will be searched for databases.

What Comes Next?

On Day 17, "Working with DB2 Instances," you learn how to create, delete, or specify database instances.

Q&A

Q What are the two available discovery methods?

A You can choose the `KNOWN` or `SEARCH` discovery methods. With `KNOWN`, your client can find only the instances and databases on the network that are known to the client. You can add systems if you know what protocol is used and the specific protocol information of the remote server. Using the `SEARCH` discovery method enables your client to search all the systems on the local network for databases.

Q When would I use a server profile?

A A server profile is generated at the server workstation and contains information about the databases cataloged on the server. After you generate a profile of the databases on the server, you can store the profile on a network drive that all your clients can access. Each client user can import this profile so that each client can easily set up the information needed to connect to the database. This ensures that each client uses the same database information, thereby reducing problems when connecting clients to servers.

Q What three methods can I use to catalog databases on my client with the CCA?

A You can use a pregenerated profile generated at the server or the client. The profile contains information about the database. You can search the network for systems that contain DB2 databases. You can manually enter the specific protocol information for the server system that you want to access.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. What types of information can you export to a profile from a client workstation?
2. If you don't want your clients to be able to search the network for databases, how can you turn off this option?
3. What protocols are supported by the `SEARCH` discovery method?
4. What profile registry values can be used to tune the Search discovery on the client?

Exercise

Change the discovery parameters to disable network searching, and test that it works as expected.

Day 17: Working with DB2 Instances

Overview

Today's lesson describes DB2 server instances and how to work with them. An *instance* is a logical database environment that can be customized for the group of databases contained within the instance. For example, it's common to have an instance for test databases and an instance for production databases.

In this lesson, you'll see how to create and set up instances, remove instances, start or stop instances, and set the instance that's used as the default. You'll also learn how to set up the DB2 Administration Server, a special instance that enables you to perform remote administration.

DB2 program files are physically stored in one location on a particular machine. Each created instance points back to this location, so the program files aren't duplicated for each instance created. Several related databases can be located within a single instance.

Tip If possible, you should create each database in a separate instance so you can specifically tune the instance for a single database.

Setting Up Instances

A default instance, known as `DB2`, is created when you install DB2. The instance name is used to set up the directory structure. To support the immediate use of this instance, the following are set during installation:

- The environment variable `DB2INSTANCE` is set to DB2.
- The DB2 Registry value `DB2INSTDEF` is set to DB2.

These settings establish DB2 as the default instance. You can change the instance that's used by default, but first you have to create an additional instance.

Tip To list all the database instances available on a system, run the `db2ilist` command at a command prompt.

Creating Additional Instances

If you're a user in the Administrators group, you can create additional DB2 instances by using the `db2icrt` command at a command prompt. The command syntax is

Syntax `db2icrt` *`instance_name`*
in which *`instance_name`* is a string up to eight alphanumeric characters long.
For example, to create an instance called *`examples`*, use the following command:
`db2icrt examples`

How DB2 Selects an Instance

The value of the `DB2INSTANCE` variable determines which instance is started or stopped by default. During installation, DB2 sets the `DB2INSTANCE` variable in the Windows NT System Properties dialog box. This sets the variable for all sessions on the system. You can also set the `DB2INSTANCE` variable for an individual session or globally for thesystem.

Setting `DB2INSTANCE` for the Current Session

The current session's environment variables are examined to see if `DB2INSTANCE` is set. You can set it in a session by issuing the following at a command prompt:

```
set db2instance=new_instance_name
```

To see the value that is currently set for the session, issue the following command at a command prompt:

```
set db2instance
```

Setting `DB2INSTANCE` for All Sessions

If `DB2INSTANCE` isn't set in the current session, DB2 examines the system environment variables to see whether `DB2INSTANCE` is set for all sessions. On Windows NT, system environment variables are set in the System Properties dialog box. The `DB2INSTANCE` variable is usually set during installation; however, if the variable doesn't exist, you can add the variable as follows:

1. Choose Start | Settings | Control Panel, and double-click the System icon.

2. In the System Properties dialog box, select the Environment tab, and select any environment variable in the System Variables list.

3. Change the name in the Variable text box to the name of the environment variable you want to set—for example, `DB2INSTANCE`.

4. Change the Value text box to the instance name—for example, `examples`.

5. Click the Set button.

6. Click OK.

If the variable already exists in the System Variables list, you can set a new value as follows:

1. Choose Start | Settings | Control Panel, and double-click the System icon.

2. In the System Properties dialog box, select the Environment tab and then the environment variable you want to append—for example, `DB2INSTANCE`.

3. Change the Value text box to the instance name—for example, `examples`.

4. Click the Set button.

5. Click OK.

You might have to reboot your system for these changes to take effect.

Setting `DB2INSTANCE` Globally

If `DB2INSTANCE` isn't set by either environment variable, you must use the DB2 Profile Registry value, `DB2INSTDEF`. It's set at the Registry's global level. For more information on setting it at this level, see the next section "Setting the Default Instance."

To determine which database instance applies in the current session, enter the following at a command prompt:

Input **`db2set db2instance`**

Setting the Default Instance

You can change the setting of the default instance by setting the `DB2INSTDEF` Registry value. It's set at the global level of the Registry with the following command:

```
db2set db2instdef=new_instance_name -g
```

Remember, a value set for the environment variable `DB2INSTANCE` at the session or system level will override the `DB2INSTDEF` setting.

Starting and Stopping a DB2 Server Instance

You must start a DB2 server instance before you can perform the following tasks:

- Connect to a database on the instance.

- Precompile an application.

- Bind a package to a database.

To start a single database instance, log in with a username that has `SYSADM` authority on the instance. Then use one of these methods to start the database instance:

- From the Control Center, right-click the instance that you want to start, and select Start from the pop-up menu.

- From a command line, enter the **db2start** command.

The `db2start` command will resolve the `DB2INSTANCE` value, as described earlier in the section "Setting Up Instances."

To stop an instance, use one of these methods:

- From the Control Center, right-click the instance you want to stop, and select Stop from the pop-up menu.
- From a command line, enter the **db2stop** command.

  Note If command-line processor sessions are attached to an instance, you must issue the `terminate` command to end each session, before issuing the `db2stop` command.

Running Multiple Instances Concurrently

You can start multiple DB2 instances as long as they belong to the same level of code (such as DB2 version 5). To run multiple instances concurrently, use one of the following methods:

- From the Control Center, right-click another instance that you want to start, and select Start.
- From the command line, set the `DB2INSTANCE` variable to the name of the other instance that you want to start, by using the command `set db2instance=`*`another_instance_name`*. Start the instance by entering the `db2start` command, and then Stop and restart the DB2 Administration Server by entering the following commands at a command prompt:

  Input
  ```
  db2admin stop
  db2admin start
  ```

See Day 15, "Ensuring Data Security," for information on configuring each instance with different environment variables.

If instances aren't shown when you refresh the Control Center, you must right-click the Instances folder and select Add from the pop-up menu. In the Add Instances window, click Refresh to get a list of instances. From the drop-down list, select an instance, and then click Apply.

If you had databases that no longer appear, right-click the Databases folder and select Add. Click Refresh to see a list of databases that you can recatalog.

Attaching to Instances

You can attach to other instances to perform maintenance and utility tasks that can be done only at an instance level, such as creating a database, forcing off applications,monitoring a database, or updating a database manager configuration. When you attempt to attach to an instance that's not your default instance, the node directory is used to determine how to communicate with that instance.

To attach to another instance, which might be remote, use the `attach` command:

```
attach to examples
```

This command will attach you to the instance called `examples` that was previously cataloged in the node directory.

To use the Control Center to attach to another instance, which can be remote, follow these steps:

1. In the Control Center, right-click the Instances folder, and select Attach from the pop-up menu.

2. Enter the username and password for the system, and click OK.

After performing maintenance activities for the `examples` instance, you can then detach from that instance by using the `detach` command.

Removing Instances

To remove a DB2 instance, follow these steps:

1. End all applications that are now using the instance.

2. Stop the instance by right-clicking the instance and selecting Stop from the pop-up menu.

3. Back up files in the `\sqllib\`*`instance_name`* directory, if needed. For example, you might want to save the database manager configuration file, `db2systm`.

   Note If the `DB2INSTPROF` environment variable is set, these files will be in a different location than the one used in this example.

4. Drop the instance by right-clicking the instance and selecting Drop from thepop-up menu.

Creating the DB2 Administration Server

DB2 Administration Server is a DB2 instance that enables remote administration of DB2 servers. The Administration Server instance is created and used in a similar fashion to any other DB2 instance. You can have only one DB2 Administration Server on a machine.

The DB2 Administration Server is automatically created during installation. You should re-create the Administration Server only if you have a problem with your DB2 Administration Server.

To create the DB2 Administration Server, follow these steps:

1. Remove the DB2 Administration Server that's now on your system. First, stop the DB2 Administration Server with the `db2admin stop` command, and then drop the DB2 Administration Server with the `db2admin drop` command.

2. Enter **db2admin create** at a command prompt. If an error message appears, stating that a DB2 Administration Server already exists, issue the `db2admin drop` command to remove the existing one, and then reissue the `db2admin create` command.

3. Optionally, provide a username and a user password. If they're valid, the username and password will identify the owner of the DB2 Administration Server.

4. To establish or modify the ownership of the DB2 Administration Server after it's created, enter **db2admin setid**, and provide a username and password.

Starting and Stopping the Administration Server

To start the DB2 Administration Server, use the `db2admin start` command. To stop the

DB2 Administration Server, use the `db2admin stop` command. To automatically start or stop the Administration Server, see Day 4, "Getting Started."

Customizing Configuration Parametersfor the Administration Server

To view the current values for the database manager configuration parameters relevant to the DB2 Administration Server, enter the **get admin cfg** command. To update individual entries in the database manager configuration file relevant to the DB2 Administration Server, enter the **update admin cfg using** command. To reset the configuration parameters to the recommended database manager defaults, enter **db2reset admin cfg**.

Changes to the database manager configuration file become effective only after they're loaded into memory—that is, when `db2start` is executed.

Cataloging the Administration Server

The DB2 Administration Server must be cataloged before you can attach to the DB2 Administration Server. The only difference from the normal `catalog node` command is the addition of the keyword `admin` after the keyword `catalog`—that is, `catalog admin node`.

When cataloging locally, you must know the instance name of the DB2 Administration Server. Use the `db2set` command to view the instance name: `db2set db2adminserver`.

Summary

In this lesson, you saw that you can create one or more instances of DB2 on your system. DB2 program files are physically stored in one location on a particular machine, and each created instance points back to this location.

Databases are stored in instances. You can have several related databases located within a single instance.

A default instance called `DB2` is created when you install the product. The instance name is used to set up the directory structure where the data files are stored.

What Comes Next?

On Day 18, "Performance Aids," you learn how to use the various performance tools that ship with DB2.

Q&A

Q The default instance is determined by the value of the `DB2INSTANCE` environment variable. What are the different ways this variable can be set?

A You can set the `DB2INSTANCE` environment variable for an individual session by entering set db2instance=***`new_instance_name`*** at a command prompt. The environment variable can be set for all sessions by using the System Properties dialog box. Finally, you can use the `DB2INSTDEF` profile Registry value to set the `DB2INSTANCE` environment variable globally. If you have all three values set, the two environment variable settings override the registry value.

Q Can I run multiple instances at one time?

A If you have multiple instances defined on your system, and they're all at the same level of product code, you can run multiple instances at one time. You might want to do this if you have a production database in one instance and a test database in another, and you want to query the data in both instances at the same time.

Q What's the difference between starting an instance and attaching to an instance?

A You attach to an instance when you want to perform maintenance such as creating a database, forcing off applications, or updating a database manager configuration file. You can perform these same actions if you start the instance as well. Attaching to an instance enables you to perform maintenance on several instances that might not be defined as your default instance.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. What's the command that you use if you want to view the configuration parameters set for the DB2 Administration Server?

2. What drive and subdirectory structure is created when you create an instance?

3. What's the command to use if you want to change the ownership of the DB2 Administration Server after it's created?

4. How do you change the instance that's started as a default?

Exercise

Create a few instances, attach to the instances, and create databases in each. Look at the directory structure that's created when an instance is created. Start multiple instances, and connect to more than one database at a time. Drop the instances that you no longer need.

Day 18: Performance Aids

Overview

DB2 provides several tools to help you determine your specific performance requirements. These tools—Database System Monitor, Performance Monitor, and Visual Explain—help you find out what's going on in the database manager and can help you tune the performance on your system. You can learn specifics about who is using the database, how many statements are being issued, and the exact SQL statements being entered.

This lesson also shows you how to reorganize your tables after you've had many inserts and deletions and how to gather statistics on the tables to ensure that DB2 is finding the quickest path to your data.

Database System Monitor

The database system monitor is a set of commands and APIs used to provide a wide variety of statistical information about the operation of the database manager, and activity information such as counters and other measurements of database processing. This information can help you do the following:

- Better understand how the database manager works.
- Improve database and application performance.
- Tune configuration parameters.
- Determine the source and cause of problems.
- Understand user and application activity within the database manager.

The Performance Monitor is a front-end tool to the database system monitor. It includes the Snapshot Monitor and Event Monitor to help you gather information about your databases. You can choose to enter database system monitor commands at a command prompt, write your own applications with the database system monitor APIs, or simply use the tools provided with DB2, all to perform the same basic functions.

The information gathered by the monitor is generated by the internal components of the database manager. When DB2 is started, the database manager level of information begins to be recorded and continues until DB2 is stopped. Database information begins being recorded when the first connection is made to the database and continues until the last application disconnects from it. Application information is recorded from the time an application connects to a database until it disconnects.

Some basic monitoring information will always be collected, and you can also selectively collect other statistical information. This flexibility is important because additional monitoring increases overhead, which can affect performance.

The database system monitor commands can be run from the command-line processor to quickly gather statistical information. Applications can be written to take advantage of the rich set of APIs and to provide some automated analysis of the vast amount of monitor data available from the database system monitor.

There are two monitoring methods: *event* and *snapshot*. Event monitoring is used to query the operational status of a database over a period of time. Snapshot monitoring is used to query the operational status of a database at any particular point in time.

The Snapshot Monitor and Event Analyzer provide the following benefits:

- **Comprehensive, flexible data collection.** More than 200 performance attributes are supported, including buffer pool and I/O, lock and deadlock, sorting, communication, agent, and logging information. Data is shown for database managers,databases, table spaces, tables, buffer pools, connections, transactions, and SQL statements.
- **Easy-to-use, intuitive viewing.** Snapshot data can be viewed in real time with easy-to-read graphs or textual views conveniently organized into logical groups. Details and summary views are provided, with the capability to access more detailed information.
- **Powerful data analysis capabilities.** You can customize measurements by allowing spreadsheet-like formulas. Rather than monitor an absolute measurement, for example, you can monitor a ratio calculated from two related measurements.

- **Robust alerting capabilities.** For any performance measurement, you can define exception conditions by specifying a threshold value. When the threshold value is reached, you can specify any or all of the following actions: notification through the Alert Center, audible alarm, or execution of a script, program, or message. Records are logged by default for alarm and warning thresholds on the Alerts page of the Journal.

Monitoring Databases with DB2 Performance Monitor

An installable option of the Control Center, the Performance Monitor is a graphical interface that provides comprehensive performance data collection, viewing, reporting, analysis, and alerting capabilities for your DB2 system. With the DB2 Performance Monitor, you can do the following:

- Identify and analyze performance problems in the database applications or the database manager.

- Identify exception conditions, based on thresholds you define.

- Use the early warning system to detect potential problems.

- Automate actions to correct discovered problems.

- Define your own statistics, in addition to the default set provided.

- Use additional performance monitoring views to monitor parallel objects with the Control Center.

You can choose to monitor snapshots or events.

Tip You can also use the Windows NT Performance Monitor to monitor database and system performance.

Event Monitoring

You can create an event monitor to record data about the occurrence of specific milestones for a period of time. An *event monitor* enables you to collect information about transient events that would be difficult to monitor through snapshots, such as deadlocks, transaction completion, and completion information that includes how long a transaction took or how much CPU a statement used. This information is a report on the activities occurring in the database.

Monitoring a database manager event results in information being returned when that event occurs. The information provides a good summary of the activity of a particular event—for example, a database connection or a SQL statement.

You use the Event Analyzer to view the information collected by an event monitor and stored in event files. The Event Analyzer is used with an event monitor file to provide information about database events that have taken place. For example, you can use an event monitor to record information on database activities, such as connections, transactions, statements, and deadlocks. After the event monitor creates its file, you can look at the collected performance information by using the Event Analyzer.

You can create event monitors that help you manage your databases and the applications that connect to them—for example:

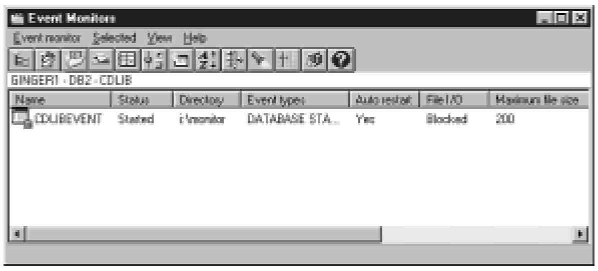
- Deadlock analysis is possible with the collection of deadlock information.

- You can enable usage accounting by gathering data at the connection or application level.
- Capacity planning can be improved. You can use statistical data collected for database objects and applications in trend analysis to predict future project needs.
- You can tune applications—through the analysis of connection and transaction data or by analyzing SQL statement event data—with a focus on heavy SQL statements.
- You can tune databases by using data collected on database objects, such as buffer pools, table spaces, and tables. This tuning can include an evaluation of configuration parameters and sort activity.

Creating an Event Monitor

Use the Create Event Monitor window to create an event monitor that will collect performance information on database activities in a file. Follow these steps:

1. Start the Control Center.
2. Expand the object tree until you find the database that you want to monitor—for example, the CDLIB database.
3. Right-click the database, and then select Monitor Events from the pop-up menu. The Event Monitors window opens (see Figure 18.1).

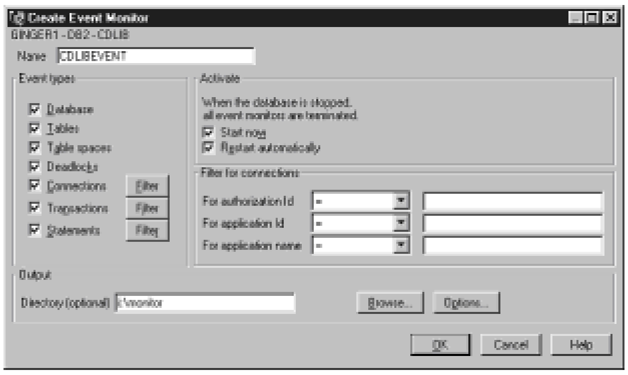


GINGER1 - DB2 - CDLIB

Name	Status	Directory	Event types	Auto restart	File I/O	Maximum file size
CDLIBEVENT	Started	i:\monitor	DATABASE STA...	Yes	Blocked	200

Figure 18.1: The Event Monitors window.

4. Choose Event Monitor | Create from the menu.
5. In the Create Event Monitor window (see Figure 18.2), specify a name, in the Name text box, for the event monitor you're creating. This new event monitor can't have the same name as any existing monitor and must be fewer than 18 characters long. Blank spaces aren't allowed in the name. For this example, use **`CDLIBEVENT`**.



GINGER1 - DB2 - CDLIB

Name: CDLIBEVENT

Event types:
- ✓ Database
- ✓ Tables
- ✓ Table spaces
- ✓ Deadlocks
- ✓ Connections
- ✓ Transactions
- ✓ Statements

Activate:
When the database is stopped, all event monitors are terminated.
- ✓ Start now
- ✓ Restart automatically

Filter for connections:

For authorization Id	=	
For application Id	=	
For application name	=	

Output:
Directory (optional): i:\monitor

Figure 18.2: The Create Event Monitor window.

6. Select one or more of the Event types check boxes to indicate the type of events that you want to monitor (for this example, select all the check boxes):

   - *Database* records a database event when the last application disconnects from the database.
   - *Tables* records a table event for each active table when the last application disconnects from the database.
   - *Table Spaces* records a table space event for each active table space when the last application disconnects from the database.
   - *Deadlocks* (the default) records a deadlock event whenever a deadlock occurs.
   - *Connections* records a connection event when an application disconnects from the database. To control monitoring, use a filter to specify the conditions that will generate connection events.
   - *Transactions* records a transaction event whenever a transaction completes (a commit or rollback). To control monitoring, use a filter to specify the conditions that will generate transaction events.
   - *Statements* records a statement event whenever a SQL statement finishes executing. To control monitoring, use a filter to specify the conditions that will generate statement events.

7. In the Activate section, indicate when you want this monitor to start. The default is Start Now, for monitors to start as soon as they're defined. Keep the Start Now check box selected for this example.

   If you start a statement event monitor after a statement starts, the monitor will start collecting information when the next SQL statement starts. As a result, the event monitor won't return information about statements that the database manager is executing when the monitor was started. This is also true for transaction and connection information.

   Select Restart Automatically to have monitoring automatically restarted when the database starts up. Candidates for auto-started monitors might include monitors that collect statistical data for tuning or those gathering information for usage accounting systems. When the database is stopped, any running event monitor is terminated; only auto-started monitors start up again when the database comes back online.

   Tip You can create any number of event monitors, turning them on or off as required. This flexibility allows you to maintain a collection of various event monitors and lets you turn on an appropriate subset of these to achieve your desired monitoring. The maximum number of active event monitors for a database at any given time is 32.

8. Use the settings in the Filter for Connections section to define one or more conditions that will cause connection, statement, or transaction monitoring to occur. You can specify the application ID, authorization ID, or application name of each database connection that should be compared with the condition specified in the second column.

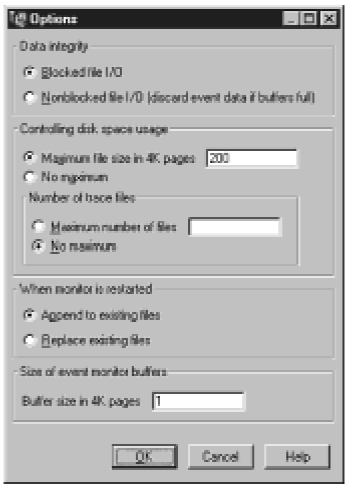
   It's optional to use filters to define conditions. If you don't use a filter, all data for the selected event type will be kept.

9. In the Directory text box, identify a path where the monitor will write the event data files. You can type a pathname in the Directory text box or click the Browse button to choose from a list of existing directories. For the example shown in the figures, the files are stored in `i:\monitor`.

   Note Event monitor paths must be unique. An event monitor can't have the same path as another monitor. If the directory you enter doesn't exist, it will be created automatically.

10. If you don't want to specify file options, click OK to create the monitor. Otherwise, click the Options button, make your selections, and then click OK twice to save your settings and close the Create Event Monitor window.

The file options determine how monitor output is handled and can affect the performance of your event monitor (see Figure 18.3). The following sections explain the settings available.


Options

Data integrity
[✓ Blocked file I/O]
Nonblocked file I/O (discard event data if buffers full)

Controlling disk space usage
[✓ Maximum file size in 4K pages] 200
No maximum
Number of trace files
Maximum number of files
[✓ No maximum]

When monitor is restarted
[✓ Append to existing files]
Replace existing files

Size of event monitor buffers
Buffer size in 4K pages 1

Figure 18.3: The Options window.

Data Integrity Setting In the Data Integrity section, indicate the type of file I/O you want. Event monitors have two I/O buffers: blocked or nonblocked file I/O. The default is blocked file I/O.

If you select blocked file I/O and both I/O buffers are full, the database agents wait for the event monitor to write one of its buffers to disk before attempting to write new data to the buffer. If you select nonblocked file I/O, the database agents don't wait for the monitor to write to disk; if both buffers are full when an event occurs, the event data is discarded.

Blocked I/O provides better data integrity, but nonblocked I/O provides better performance because database activities don't slow down because of waiting.

Controlling Disk Space Usage Setting In the Controlling Disk Space Usage section, specify the maximum size for the event monitors, which enables you to break up the event monitor data into smaller, more manageable pieces. Alternatively, you can specify that there be no limit to the size of the file. Your file size selection is closely related to the number of files kept by the event monitor.

The size of the file (broken into 4KB pages) that you specify must be greater than the size of the event monitor buffers. The default file size for the event monitor appears in the text box.

Number of Trace Files Setting In the Number of Trace Files section, specify the maximum number of files an event monitor can create. By specifying a maximum, you can control the amount of disk space used by the event monitor. The space available for event monitor data is a combination of the number of files and the file size. If you specify No Maximum for the file size, your maximum number of files must be 1. (The default for this option is no maximum.)

When a file is full, output automatically flows into the next file. When the number of files reaches the specified maximum, the event monitor stops recording information. To ensure that recording doesn't stop, remove the trace files from the File Path field.

When Monitor Is Restarted Setting In the When Monitor Is Restarted section, indicate whether new monitor data appends to existing data or replaces it. Use Append to Existing Files (the default) to have new data appended to the existing stream of data files. Use Replace Existing Files to overwrite any data files that already exist.

Size of Event Monitor Buffers Setting In the Size of Event Monitor Buffers section, specify the size of the event monitor buffers. Each event monitor has two buffers: One fills while the other is being written to disk.

Allocating a large buffer is recommended for systems with high activity. Doing so reduces file I/O, the possibility of data overflows for nonblocked output, and potential slowdowns for blocked output.

Note Event data can be lost if buffers are very large and there is a systemwide failure.

The default size for each buffer is one page. The maximum size is limited by the size of the database heap. If you're running multiple monitors, you might want to increase the size of the database heap configuration parameter.

Starting an Event Monitor

As soon as you create an event monitor, you must start it. To do so, follow these steps:

1. In the Control Center, expand the object tree until you find the database you want to monitor.

2. Right-click the database, and select Monitor Events from the pop-up menu.

3. In the Event Monitors window (refer to Figure 18.1), check the Status column to see whether the event has started. To start it, right-click the monitor you want to start, and select Start Event Monitoring from the pop-up menu.

To restart a monitor, you must stop it and then start it again. You might want to restart a monitor to reset the values.

Stopping an Event Monitor

To stop an event monitor, follow these steps:

1. From the list of monitors in the Event Monitors window, select the event monitor that you want to stop.

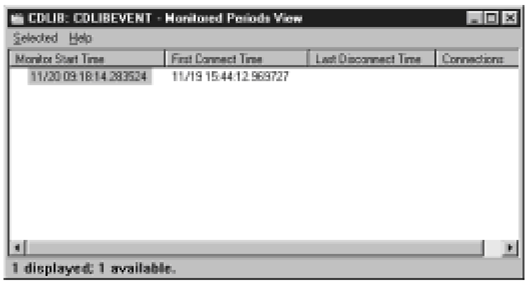
2. Choose Selected | Stop Event Monitoring from the menu.

Viewing Monitored Events

To view data files collected by an event monitor, use the Monitored Periods View window. Event data is available at different times, depending on the event type. To view event monitor files, follow these steps:

1. From the list of monitors in the Event Monitors window, click the event monitor that you want to view.

2. Choose Selected | View Event Monitor Files from the menu. The Monitored Periods View window opens (see Figure 18.4).

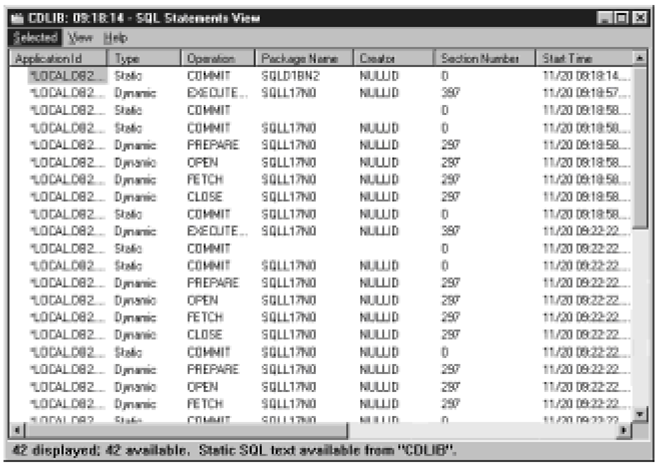


Monitor Start Time	First Connect Time	Last Disconnect Time	Connections
11/20 09:18:14.283524	11/19 15:44:12.969727		

1 displayed; 1 available.

Figure 18.4: The Monitored Periods View window.

3. Choose Selected | Open As from the menu. The monitored database objects appear in a list. Because you selected to have all objects monitored when you created the sample event monitor, you can select from Connections, Deadlocks, Deadlock Connections, Overflows, Transactions, or Statements. By selecting one of these objects, you'll see the details recorded for the event. For this example, click Statements to see the SQL statements (see Figure 18.5).

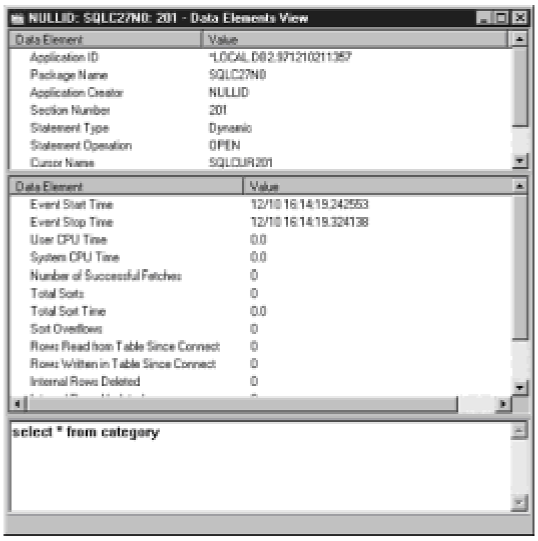


Application Id	Type	Operation	Package Name	Creator	Section Number	Start Time
*LOCAL.DB2...	Static	COMMIT	SQLD1BN2	NULLID	0	11/20 09:18:14...
*LOCAL.DB2...	Dynamic	EXECUTE...	SQLL17N0	NULLID	397	11/20 09:18:57...
*LOCAL.DB2...	Static	COMMIT			0	11/20 09:18:58...
*LOCAL.DB2...	Static	COMMIT	SQLL17N0	NULLID	0	11/20 09:18:58...
*LOCAL.DB2...	Dynamic	PREPARE	SQLL17N0	NULLID	297	11/20 09:18:58...
*LOCAL.DB2...	Dynamic	OPEN	SQLL17N0	NULLID	297	11/20 09:18:58...
*LOCAL.DB2...	Dynamic	FETCH	SQLL17N0	NULLID	297	11/20 09:18:58...
*LOCAL.DB2...	Dynamic	CLOSE	SQLL17N0	NULLID	297	11/20 09:18:58...
*LOCAL.DB2...	Static	COMMIT	SQLL17N0	NULLID	0	11/20 09:18:58...
*LOCAL.DB2...	Dynamic	EXECUTE...	SQLL17N0	NULLID	397	11/20 09:22:22...
*LOCAL.DB2...	Static	COMMIT			0	11/20 09:22:22...
*LOCAL.DB2...	Static	COMMIT	SQLL17N0	NULLID	0	11/20 09:22:22...
*LOCAL.DB2...	Dynamic	PREPARE	SQLL17N0	NULLID	297	11/20 09:22:22...
*LOCAL.DB2...	Dynamic	OPEN	SQLL17N0	NULLID	297	11/20 09:22:22...
*LOCAL.DB2...	Dynamic	FETCH	SQLL17N0	NULLID	297	11/20 09:22:22...
*LOCAL.DB2...	Dynamic	CLOSE	SQLL17N0	NULLID	297	11/20 09:22:22...
*LOCAL.DB2...	Static	COMMIT	SQLL17N0	NULLID	0	11/20 09:22:22...
*LOCAL.DB2...	Dynamic	PREPARE	SQLL17N0	NULLID	297	11/20 09:22:22...
*LOCAL.DB2...	Dynamic	OPEN	SQLL17N0	NULLID	297	11/20 09:22:22...
*LOCAL.DB2...	Dynamic	FETCH	SQLL17N0	NULLID	297	11/20 09:22:22...
*LOCAL.DB2...	Static	COMMIT	SQLL17N0	NULLID	0	11/20 09:22:22...

42 displayed; 42 available. Static SQL text available from "CDLIB".

Figure 18.5: The SQL Statements View window.

4. Double-click the entries to see detailed information, including the SQL statement issued and the type of statement issued (see Figure 18.6).



NULLID: SQLC27N0: 201 - Data Elements View

Data Element	Value
Application ID	*LOCAL.DB2.971210211357
Package Name	SQLC27N0
Application Creator	NULLID
Section Number	201
Statement Type	Dynamic
Statement Operation	OPEN
Cursor Name	SQLCUR201

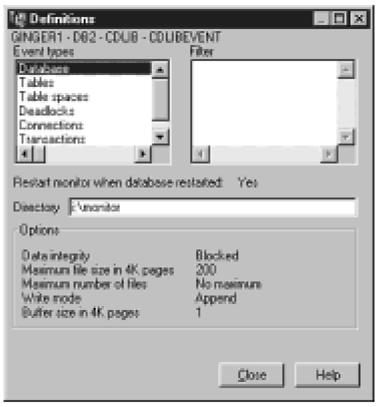
Data Element	Value
Event Start Time	12/10 16:14:19.242553
Event Stop Time	12/10 16:14:19.324138
User CPU Time	0.0
System CPU Time	0.0
Number of Successful Fetches	0
Total Sorts	0
Total Sort Time	0.0
Sort Overflows	0
Rows Read from Table Since Connect	0
Rows Written in Table Since Connect	0
Internal Rows Deleted	0

```
select * from category
```

Figure 18.6: Detailed information for the SQL statements.

Tip To view event monitor files from the Event Analyzer, click the Event Analyzer icon on the Control Center's toolbar. The Event Analyzer window opens. In the Path field, identify the path where the data files are stored.

To see how an existing event monitor was defined, choose Selected | View Definitions from the menu. The selections in the Definitions window were made when the monitor was created (see Figure 18.7).



Definitions

GINGER1 - DB2 - CDLIB - CDLIBEVENT

Event types: Database, Tables, Table spaces, Deadlocks, Connections, Transactions

Filter

Restart monitor when database restarted: Yes

Directory: f:\monitor

Options

Data integrity	Blocked
Maximum file size in 4K pages	200
Maximum number of files	No maximum
Write mode	Append
Buffer size in 4K pages	1

Figure 18.7: The Definitions window.

Snapshot Monitoring

Taking a snapshot of data gives you information for the point in time that the snapshot was requested. This information provides a picture of the current state of all activity in a database instance. Use the Snapshot Monitor when you have a problem that's occurring now or if you want to query the status of a database at a particular instance in time. You can analyze long-running transactions for which you might want interim information, before they complete.

Some snapshot information provides cumulative information in the form of *counters*, to cover activity from the time monitoring started to the time the snapshot was requested. Other information is returned only if that particular event has completed before the snapshot was requested.

With the Snapshot Monitor, for a given point in time a snapshot is returned for a performance variable. The Snapshot Monitor displays these points for comparison over time.

Each point on the graph represents a data value. This information can be used to do the following:

- Detect performance problems.
- Analyze performance trends.
- Tune database instance and database configuration parameters.
- Analyze the performance of database applications.

Performance information is available for the following database objects:

- Database instance
- Database
- Table
- Table space
- Database connections

The Snapshot Monitor has a graphical interface that lets you set the profile sampling frequency of performance snapshots, view the results of performance calculations, define threshold values and alert actions, and generate alerts.

Monitoring a Database

Use the following instructions to start monitoring a database:

1. In the Control Center, expand the object tree until you find the database you want to monitor. For this example, monitor the CDLIB database.
2. Right-click the CDLIB database icon, and select Snapshot Monitoring | Start Monitoring from the pop-up menu. The snapshot monitoring session begins and continues until you select Stop Monitoring.

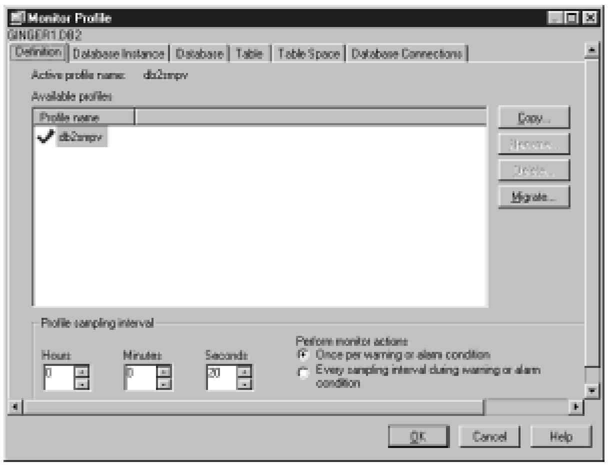
To start the monitoring for a table, table space, or connection, select the database object icon, and then move to the contents pane on the right side of the Control Center. Right-click the specific database object, and select Snapshot Monitoring | Start Monitoring from the pop-up menu.

Working with Performance Variables

When you start monitoring the CDLIB database, use the Monitor Profile notebook to work with your performance variables and performance variable profiles. Follow these steps:

1. Right-click the CDLIB database icon, and select Snapshot Monitoring | Show Monitor Profile from the pop-up menu.
2. On the Definition page of the Monitor Profile notebook (see Figure 18.8), set the

profile sampling interval to indicate how often the database system monitor returns a performance snapshot. For this example, set the sampling interval to 20 seconds. Also, specify when a monitored action is to occur: once per warning or alarm condition, or every sampling interval during a warning or alarm condition.



Active profile name: db2smpv

Available profiles

Profile name
✓ db2smpv

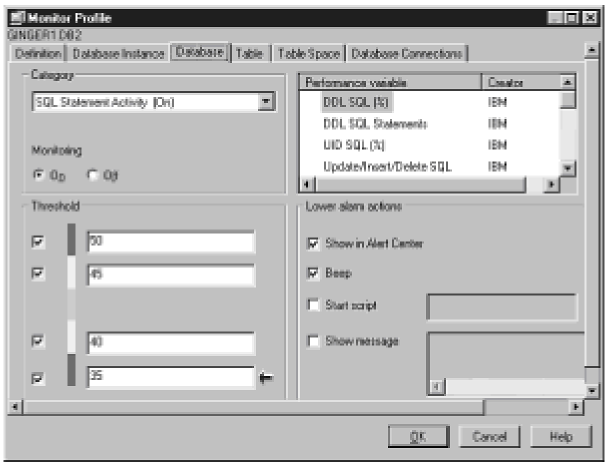
Profile sampling interval

Hours: 0 Minutes: 0 Seconds: 20

Perform monitor actions
- [✓ Once per warning or alarm condition]
- Every sampling interval during warning or alarm condition

Figure 18.8: The Monitor Profile notebook's Definition page.

3. On the Database Instance, Database, Table, Table Space, or Database Connections pages, you can select a category of performance variables to enable for each object: Agents, Connections, or Sort. Depending on the enabled category, you can select or add a performance variable and set its threshold targets. Figure 18.9 shows an example of the Database page, where the category is set to SQL statement activity, the performance variable is set to DDL SQL (%), and various threshold levels are set.



Category: SQL Statement Activity (On)

Monitoring: [✓ On] Off

Performance variable	Creator
DDL SQL (%)	IBM
DDL SQL Statements	IBM
UID SQL (%)	IBM
Update/Insert/Delete SQL	IBM

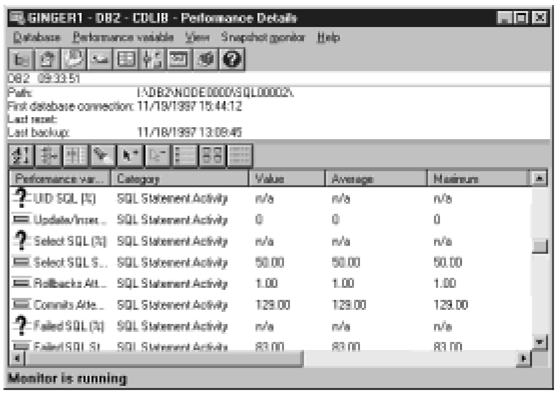
Threshold
- ✓ 50
- ✓ 45
- ✓ 40
- ✓ 35

Lower alarm actions
- ✓ Show in Alert Center
- ✓ Beep
- Start script
- Show message

Figure 18.9: The Monitor Profile notebook's Database page.

After you set the threshold values, enter SQL statements to cause activity against the database. For example, select to sample the contents of various tables in the CDLIB database, or enter simple SQL statements in the Command Center.

4. If you selected to have an event recorded in the Alert Center each time a threshold was exceeded, the Alert Center will open when this occurs. Double-click an event to open the Performance Details page to analyze the event that exceeded its threshold. Figure 18.10 shows sample performance details based on the SQL statement activity.



GINGER1 - DB2 - CDLIB - Performance Details

Database Performance variable View Snapshot monitor Help

DB2 09:33:51

Path: I:\DB2\NODE0000\SQL00002\
First database connection: 11/19/1997 15:44:12
Last reset:
Last backup: 11/18/1997 13:09:45

Performance var...	Category	Value	Average	Maximum
UID SQL (%)	SQL Statement Activity	n/a	n/a	n/a
Update/Inser...	SQL Statement Activity	0	0	0
Select SQL (%)	SQL Statement Activity	n/a	n/a	n/a
Select SQL S...	SQL Statement Activity	50.00	50.00	50.00
Rollbacks Att...	SQL Statement Activity	1.00	1.00	1.00
Commits Atte...	SQL Statement Activity	129.00	129.00	129.00
Failed SQL (%)	SQL Statement Activity	n/a	n/a	n/a
Failed SQL St...	SQL Statement Activity	83.00	83.00	83.00

Monitor is running

Figure 18.10: The Performance Details window.

5. Double-click a performance variable to view and set the alarm and warning threshold values.

6. Right-click a performance variable to view a graph of the performance variable you've chosen (see Figure 18.11).

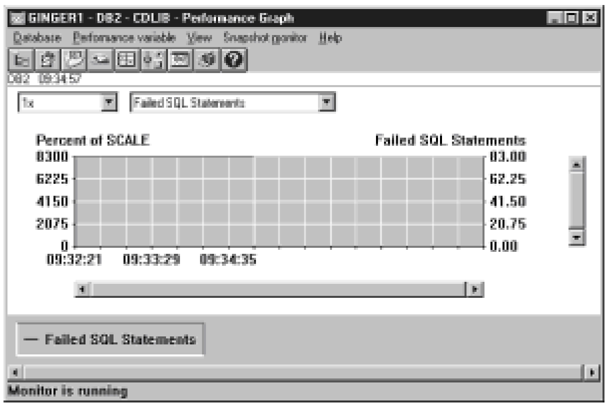


Figure 18.11: A graph of performance variables.

Viewing Performance Details

You also can open the Performance Details window from the Control Center. Use the performance details view for a selected database object to see all the details. When you open a performance details view from the Control Center, monitoring of the object begins automatically. Follow these steps to open a performance details view:

1. In the Control Center, expand the object tree until you find the database object you want to monitor.

2. For an instance or a database, right-click the database object and select Snapshot Monitoring | Show Monitor Details from the pop-up menu. For a table, table space, or connection, select the database object icon, and then move to the contents pane on the right side of the Control Center. Right-click the specific database object, and select Snapshot Monitoring | Show Monitor Details from the pop-up menu.

Viewing Performance Summaries

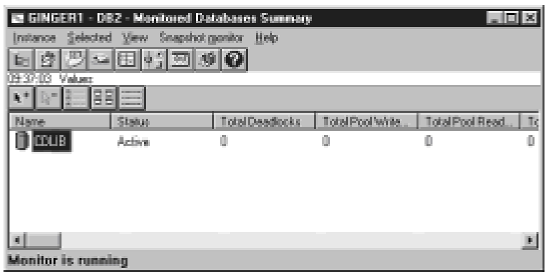
Use the monitored summary view to see the performance information on all the database objects being monitored in a database object folder. Follow these steps to open a monitored summary view:

1. In the Control Center, expand the object tree until you find the database object you want. For example, for a summary view of monitored databases, select the Databases object folder, not a specific database name.

2. If you're already monitoring some objects in the folder, right-click the database object folder, and select Show Monitor Summary from the pop-up menu.

   If you aren't already monitoring the objects, go to the contents pane and select Snapshot Monitoring | Start Monitoring from the pop-up menu. Do this for all the database objects you want in the monitored summary view. Return to the database object folder in the main object tree, right-click the database object folder, and select Show Monitor Summary from the pop-up menu.

The monitored summary view for the monitored object(s) opens (see Figure 18.12).


GINGER1 - DB2 - Monitored Databases Summary
Instance Selected View Snapshot monitor Help
09:37:03 Values:

Name	Status	Total Deadlocks	Total Pool Write...	Total Pool Read...	To
CCLIB	Active	0	0	0	0

Monitor is running

Figure 18.12: The Monitored Databases Summary window.

Stopping a Snapshot Monitor

Follow these steps to stop monitoring:

1. In the Control Center, expand the object tree until you find the database object you've been monitoring.

2. For an instance or a database, right-click the database object, and select Snapshot Monitoring | Stop Monitoring from the pop-up menu. (If you're monitoring several database objects and want to stop them all at the same time, select Stop All Monitoring.)

   For a table, table space, or connection, select the database object icon, and then move to the contents pane on the right side of the Control Center. Right-click the specific database object, and select Snapshot Monitoring | Stop Monitoring from the pop-up menu.

Using the Alert Center

Use the Alert Center to work with alerts generated by the snapshot monitor. The icons displayed in the Alert Center represent alarms and warnings generated by the performance variable threshold values set in the Snapshot Monitor.

To view or work with objects in the Alert Center, they must be monitored and in a warning or alarm state. You must also have tools settings set up to show alerts in the Alert Center:

1. Open the Tools Settings notebook by clicking the Tools Settings icon on the Control Center's toolbar.

2. On the Alert Center page, specify when you want the Alert Center to open:

   - Select Never to have the Alert Center never open automatically. If this option is selected, you must explicitly open the Alert Center window from the tool bar or product folder to see monitored objects in alarm or warning states.

   - Select On New Warnings and Alarms to start the Alert Center automatically every time there is a new warning or alarm. If the window is minimized, it will open; if the window is open but covered by other windows, it will surface on top of the other windows.

   - Select On New Alarms Only to surface the Alert Center automatically each time there is a new alarm.

To work with objects in the Alert Center, follow these steps:

1. Open the Alert Center window (see Figure 18.13) by clicking the Alert Center icon on the Control Center's toolbar. (Depending on the settings you made in the Tools Settings notebook, the Alert Center might open automatically when there's a new warning or alarm.)

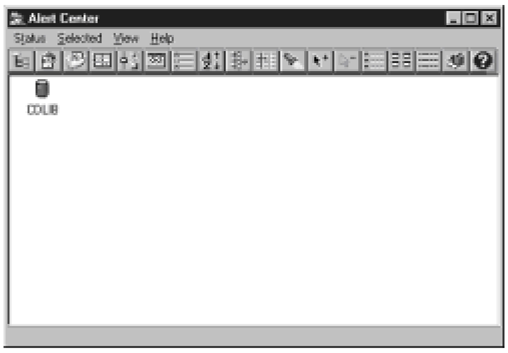
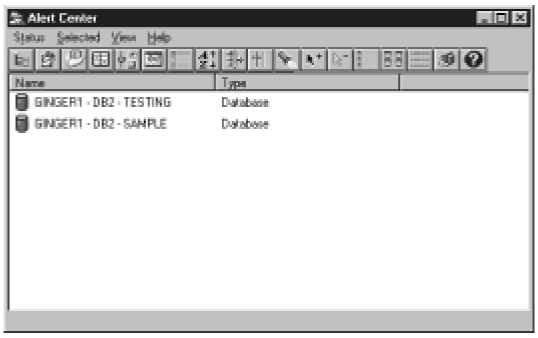


Figure 18.13: The Alert Center window.

2. Use the View menu to look at the information in the Alert Center in differentformats:

   - Select Details to see more details about the displayed object. The details include the icon, name, type of object, and severity of the alert (see Figure 18.14).



Name	Type
GINGER1 - DB2 - TESTING	Database
GINGER1 - DB2 - SAMPLE	Database

Figure 18.14: The Alert Center window's details view.

   - Select Icon to see only the icons that represent the database objects whose alerts have reached their alarm or warning values (see Figure 18.15).

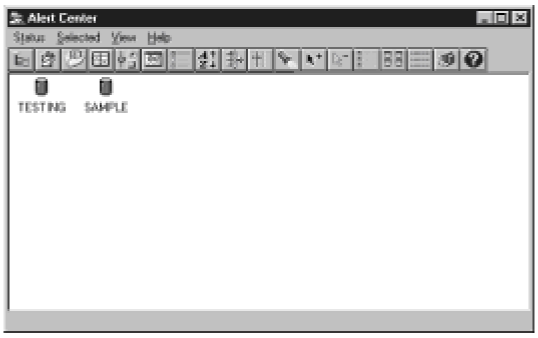


Figure 18.15: The Alert Center window's icon view.

- Select Text to see the name and severity level of the displayed database objects (see Figure 18.16).

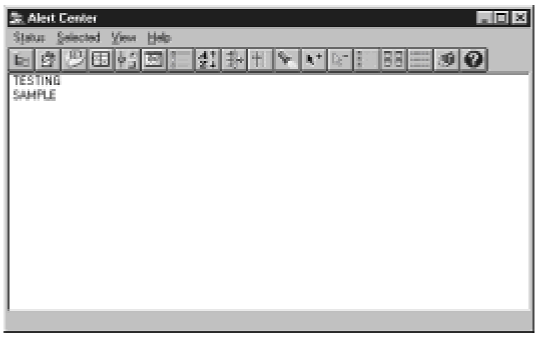


Figure 18.16: The Alert Center window's text view.

3. Use the Sort window to specify how you want the contents to be sorted (see Figure 18.17). You can choose which columns to view and whether the columns are sorted in ascending or descending order. You can access the Sort window by clicking the Sort icon in the Alert Center's toolbar or by choosing View | Sort from the menu.

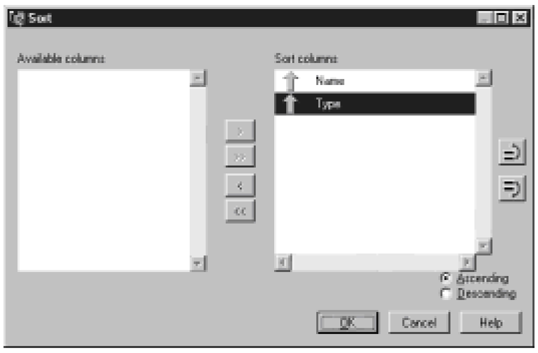
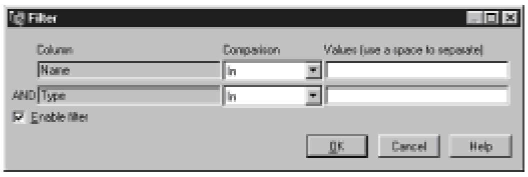


Figure 18.17: The Sort window in the Alert Center.

4. Use the Filter window to specify an operator that you want applied to the comparison value (see Figure 18.18). The possible operator values are In, Not In, Equal To, and Not Equal. You can access the Filter window by clicking the Filter icon in the toolbar or by choosing View | Filter from the menu.

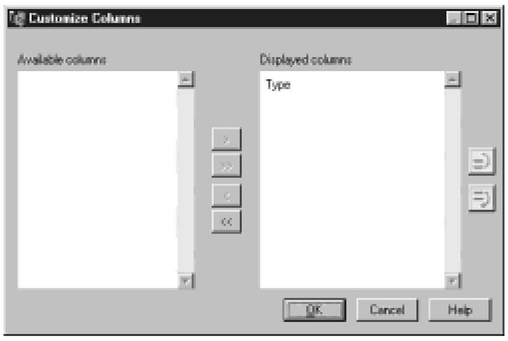


Column	Comparison	Values (use a space to separate)
Name	In	
AND Type	In	

[✓] Enable filter

Figure 18.18: The Alert Center's Filter window.

5. Use the Customize Columns window to specify which columns are displayed (see Figure 18.19). Move the columns you want displayed from the Available Columns box to the Displayed Columns box. You can also change the order of the specified columns by using the reposition buttons. You can access the Customize Columns window by clicking the Customize Columns icon in the toolbar or choosing View | Customize Columns from the menu.



Available columns

Displayed columns: Type

Figure 18.19: The Alert Center's Customize Columns window.

6. To temporarily disable alerts, choose Status | Suspend Alerts to stop alerts from being reported.

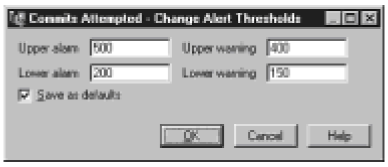
7. To remove a database object from the Alert Center, select the database object you want to remove, and choose Selected | Remove. The object is removed from the Alert Center, but monitoring continues. Alerts for the object no longer appear in the Alert Center, but a full history of the database object can be found in the Journal.

8. To stop monitoring of a selected database object, right-click the object you want, and select Stop Monitoring from the pop-up menu.

9. Double-click the entry to open the Performance Details window. A red arrow is shown next to the performance variable that has exceeded its threshold.

10. Right-click one of the entries, and select Change Alert Thresholds to change the limits for alarms and warnings (see Figure 18.20).

11. Right-click one of the entries, and select Show Performance Graph to see a graph of the selected performance variable.



Upper alarm 500 Upper warning 400
Lower alarm 200 Lower warning 150
[✓] Save as defaults

Figure 18.20: The Commits Attempted – Change Alert Thresholdswindow.

Using Visual Explain

The Visual Explain tool graphically shows you the access plans for explained SQL statements. You can use the information available from the graph to tune your SQL queries for better performance.

An access plan graph shows details of the following:

- Tables, their associated columns, and indexes
- Operators such as table scans, sorts, and joins
- Tables spaces and functions

You can also use Visual Explain to do the following:

- View the statistics used at the time of optimization. You can then compare these statistics to the current catalog statistics to help you determine whether rebinding the package might improve performance.
- Obtain information about each operation in the access plan, including the total estimated cost and number of rows retrieved.
- Model the effects of performing various tuning techniques by comparing before and after versions of the access plan graph for a query.
- Determine whether an index was used to access a table. If an index wasn't used, Visual Explain can help you determine which columns might benefit from being indexed.
- View the access plan chosen by the database manager's optimizer for a given SQL statement.
- Tune SQL statements for better performance.
- Design application programs and databases.
- View all the details of the access plan, including the statistics in the system catalogs.
- Decide whether to add an index to a table.
- Identify the source of problems by analyzing the access plan or performance of SQL statement execution.
- Use the portable snapshot function to view snapshots from any remote DB2 server.
- Display access plans for parallel and SMP systems.

Producing an Access Plan Graph

Follow these steps to produce an access plan graph by using the Command Center:

1. If the statistics in the system catalog table aren't current, update them with the `runstats` operation. See the later section "Collecting Statistics" for information on running statistics.

2. Enter a SQL statement into the Script page of the Command Center. The following is a sample query that uses the SAMPLE database:

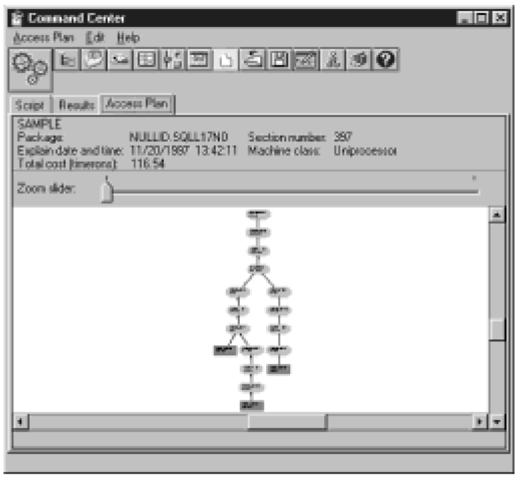
   ```
   select s.id, s.name, o.deptname, salary+comm
   from org o, staff s
   where
      o.deptnumb = s.dept and
      s.job < 'Mgr' and
      s.salary+s.comm  > (select st.salary*.9
                              from staff st
                              where st.job='Mgr')
      order by s.name
   ```

   This query lists the name, department, and earnings for all nonmanager employees who earn more than 90% of the highest-paid manager's salary.

3. Click the gears button to see whether the statement is accurate. If the statement is accurate, you'll see some columns of data. If the statement isn't accurate, an error message is returned.

4. When you've corrected any typos or other errors in the query, return to the Script page, select the entire query, and choose Script | Create Access Plan from the menu. An access plan graph will appear in the Access Plan page of the Command Center (see Figure 18.21).

5. Use the zoom slider to see details of the graph. Double-click a node in the graph to see details.



SAMPLE
Package: NULLID.SQLL17N0 Section number: 397
Explain date and time: 11/20/1997 13:42:11 Machine class: Uniprocessor
Total cost (timerons): 116.54

Figure 18.21: The Access Plan page of the Command Center.

Tip You can also create access plans through the Control Center. See the online help for information.

To see the statistics being used for the `ORG` table, double-click the `ORG` table node in the graph. The Table Statistics window opens (see Figure 18.22).

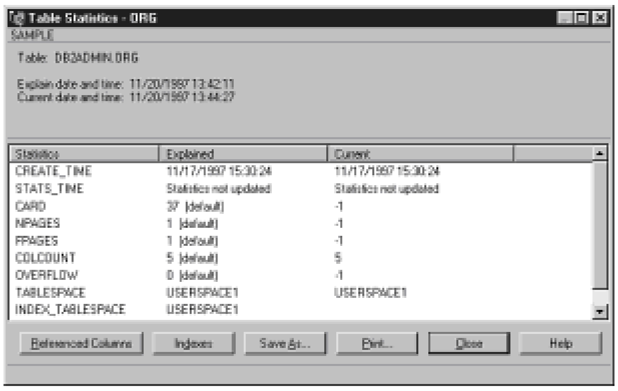


Table: DB2ADMIN.ORG

Explain date and time: 11/20/1997 13:42:11
Current date and time: 11/20/1997 13:44:27

Statistics	Explained	Current
CREATE_TIME	11/17/1997 15:30:24	11/17/1997 15:30:24
STATS_TIME	Statistics not updated	Statistics not updated
CARD	37 [default]	-1
NPAGES	1 [default]	-1
FPAGES	1 [default]	-1
COLCOUNT	5 [default]	5
OVERFLOW	0 [default]	-1
TABLESPACE	USERSPACE1	USERSPACE1
INDEX_TABLESPACE	USERSPACE1	

Figure 18.22: The Table Statistics window.

Tip To view table statistics, you need a minimum of `SELECT` privilege on the explain tables and the system catalog tables. To view statistics for multiple tables in a graph (or indexes, if any), select each table node by clicking it (the color changes); then right-click, and select Show Statistics from the pop-up menu.

You can perform some other tasks while viewing the table statistics:

- To display a list of referenced columns (see Figure 18.23), click the Reference Columns button.

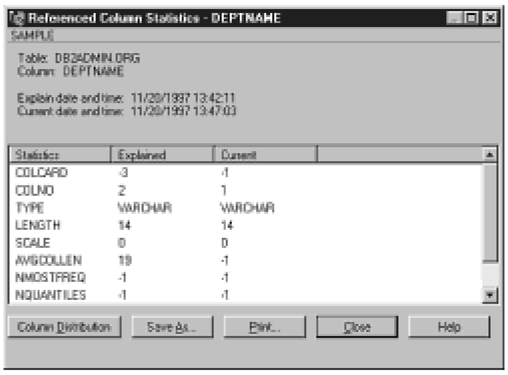


Table: DB2ADMIN.ORG
Column: DEPTNAME

Explain date and time: 11/20/1997 13:42:11
Current date and time: 11/20/1997 13:47:03

Statistics	Explained	Current
COLCARD	-3	-1
COLNO	2	1
TYPE	VARCHAR	VARCHAR
LENGTH	14	14
SCALE	0	0
AVGCOLLEN	19	-1
NMOSTFREQ	-1	-1
NQUANTILES	-1	-1

Figure 18.23: The Referenced Column Statisticswindow.

- To display a list of indexes defined on the table (see Figure 18.24), click the Indexes button.

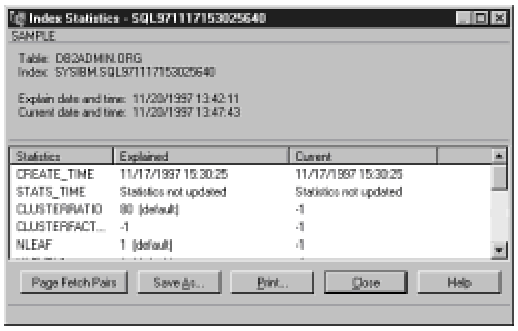


Table: DB2ADMIN.ORG
Index: SYSIBM.SQL971117153025640

Explain date and time: 11/20/1997 13:42:11
Current date and time: 11/20/1997 13:47:43

Statistics	Explained	Current
CREATE_TIME	11/17/1997 15:30:25	11/17/1997 15:30:25
STATS_TIME	Statistics not updated	Statistics not updated
CLUSTERRATIO	80 [default]	-1
CLUSTERFACT...	-1	-1
NLEAF	1 [default]	-1

Figure 18.24: The Index Statistics window.

- To save the contents of the window to a file, click the Save As button.
- To print the contents of the window, click the Print button.

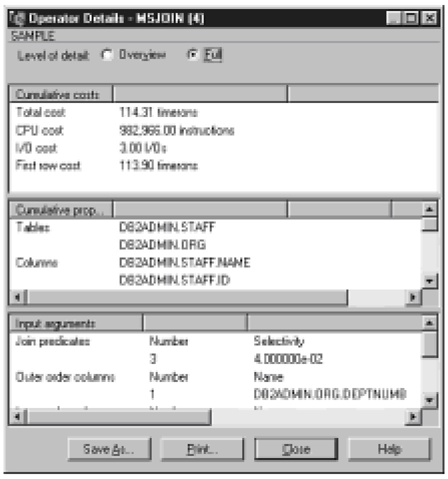
Analyzing an Access Plan Graph

The Table Statistics window displays the following statistics:

- `CREATE_TIME`—The date and time that the table was created.
- `STATS_TIME`—The last time a change was made to any recorded statistic for the table. If the statistics are out-of-date, the optimizer uses default values, which can result in an inefficient access plan. Use the `runstats` command to update thestatistics, and then rebind the package.
- `CARD`—The number of rows in the table. If the cardinality is zero, ensure that you used `runstats` after making your table updates.
- `NPAGES`—The number of pages of the table that contain one or more rows.
- `FPAGES`—The number of columns in the table.
- `OVERFLOW`—The number of overflow pages used by the table.
- `TABLESPACE`—The name of the table's primary table space.
- `INDEX_TABLESPACE`—The name of the table space that contains all indexes for the table.
- `LONG_TABLESPACE`—The name of the table space that contains all long data (`LONG` or `LOB` column types) for the table.

An *operator* is an action that must be performed on data or is the output from a table or an index, when the access plan for a SQL statement is executed. Examples of operators include `DELETE`, `FETCH`, `FILTER`, `GRPBY`, and `INSERT`.

To view the details for an operator, double-click an operator node in the diagram to open the Operator Details window (see Figure 18.25). To view detailed information in the window, select Full; to view less detailed information, select Overview. The following sections explain each bit of detail shown in this window.



Level of detail: Overview [✓ Full]

Cumulative costs		
Total cost	114.31 timerons	
CPU cost	992,966.00 instructions	
I/O cost	3.00 I/Os	
First row cost	113.90 timerons	

Cumulative prop...			
Tables	DB2ADMIN.STAFF		
	DB2ADMIN.ORG		
Columns	DB2ADMIN.STAFF.NAME		
	DB2ADMIN.STAFF.ID		

Input arguments		
Join predicates	Number	Selectivity
	3	4.000000e-02
Outer order columns	Number	Name
	1	DB2ADMIN.ORG.DEPTNUMB

Figure 18.25: The Operator Details window.

Cumulative Costs

The costs shown represent the estimated cumulative costs up to and including the point where the action represented by the operator is performed. These costs are estimated by the optimizer:

- *Total Cost* shows the estimated cumulative cost of executing the current access plan (in timerons).
- *CPU Cost* shows the estimated cumulative CPU cost (in number of instructions).
- *I/O Cost* shows the estimated cumulative input/output (I/O) cost (in number of seeks and page transfers).
- *First Row Cost* shows the estimated cumulative effort (in timerons) required to produce the first row in the set of rows that result when the action represented by the operator is performed.

Cumulative Properties

Several properties are included in this section:

- *Tables* lists the tables that have been accessed thus far in the access plan.
- *Columns* lists the columns that have been accessed thus far in the access plan.
- *Order Columns* shows the columns on which this stream is ordered. They can be ascending (`ASC`) or descending (`DESC`). A value of –1 indicates that they aren't ordered.
- *Predicates* shows the set of predicates applied (including an estimate of their selectivity). A *predicate* is a condition placed on the data, usually in the form of a comparison operation. Predicates are included in clauses beginning with `WHERE` or`HAVING`.

  Selectivity refers to the probability that any row will satisfy a predicate, making it `true`. A highly selective predicate (with a selectivity of 10% or less) is desirable. Such predicates return fewer row for future operators to work on, thereby requiring less CPU and I/O to satisfy the query.
- *Cardinality* shows the estimated number of rows to be returned. If the value is `0`, the table appears empty to the optimizer, and you should rerun `runstats`.
- *Total Buffer Pool Pages Used* shows the estimated number of pages in the buffer pool that will be required during processing of this operator and its inputs.

Input Arguments

The arguments in this section affect the operator's behavior. The details vary, depending on the type of operator and the level of detail chosen. For example, if you're using the `DELETE` operator, the Deleted Table item shows the name of the table from which the rows are to be deleted. Information regarding other operators is available in the online help.

Collecting Statistics

Statistics describe the physical and logical characteristics of a table and its indexes. You must periodically collect table and index statistics for each table. DB2 uses these statistics to determine a good way to access the data. If the data has changed significantly, to the extent that the information last collected no longer reflects the actual table data, performance can begin to deteriorate when users are accessing data.

You should rebind application programs that use static SQL after collecting statistics, because the SQL optimizer might choose a different access plan given new statistics. In particular, you should rebind those programs that reference tables for which new statistics are available.

Use the Run Statistics window to update system catalog statistics on the data in a table, the data in the table's indexes, or the data in the table and indexes. The optimizer uses these statistics to choose which path will be used to access the data. In general, you want to update statistics if extensive changes have been made to the data in the table.

To collect statistics, follow these steps:

1. In the Control Center, click the Tables icon to see a list of tables available.

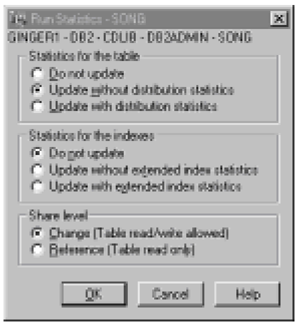
2. In the contents pane, right-click the table you want to collect statistics for, and select Run Statistics from the pop-up menu.

3. In the Run Statistics dialog box (see Figure 18.26), specify the level of statistics you want to gather for the table by selecting an option under Statistics for the Table:

   - Select Do Not Update to update statistics for the table's index but not for the table.

   - Select Update Without Distribution Statistics to update the basic-level statistics, such as the number of pages being used by the table, on the table. Distribution statistics on values in table columns aren't collected.

   - Select Update with Distribution Statistics to update the basic-level statistics and collect the statistics on the distribution of data values in table columns. The query optimizer uses these statistics to more accurately estimate the number of rows in a column that satisfy given equalities or ranges. (An example of a distribution statistic is the frequency with which a particular data value occurs in a column.)



Run Statistics - SONG
GINGER1 - DB2 - CDLIB - DB2ADMIN - SONG
Statistics for the table
Do not update
[✓ Update without distribution statistics]
Update with distribution statistics
Statistics for the indexes
[✓ Do not update]
Update without extended index statistics
Update with extended index statistics
Share level
[✓ Change (Table read/write allowed)]
Reference (Table read only)
OK
Cancel
Help

Figure 18.26: The Run Statistics dialog box.

4. Specify the level of statistics you want to gather for the table's index by selecting an option under Statistics for the Indexes:

   - Select Do Not Update to update statistics for the table but not for the table's index.

- Select Update Without Extended Index Statistics to update the basic-level statistics, such as the number of index leaf pages.

- Select Update with Extended Index Statistics to update the basic-level and detailed statistics for the indexes. The detailed statistics can help the optimizer better estimate the I/O cost of an index scan. (An example of a detailed statistic is the number of I/Os needed to read the data pages into buffer pools of various sizes.)

5. Use the Share Level options to specify the type of access you want others to have to the table while the statistics are being gathered:

   - Select Change (Table Read/Write Allowed) to allow other users to read from and write to the table while you're collecting statistics.

   - Select Reference (Table Read Only) to make the table read-only while you're collecting statistics.

6. Click OK to begin collecting statistics.

Reorganizing a Table

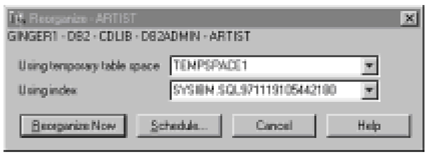
You can reorganize a table immediately or schedule it for a specific date and time. When you reorganize a table, the table data is rearranged into a physical sequence, usually according to a specified index. As a result, SQL statements on that data can be processed more efficiently. Also, the reorganization process removes unused, empty space from the table, and the reorganized table is stored more compactly.

Use the following instructions to reorganize your table immediately:

1. Before you reorganize tables, collect the statistics of the data, as described in the previous section.

2. In the Control Center, click the Tables icon to see a list of tables available.

3. In the contents pane, right-click the table you want to reorganize, and select Reorganize from the pop-up menu. The Reorganize dialog box opens (see Figure 18.27).



Reorganize - ARTIST

GINGER1 - DB2 - CDLIB - DB2ADMIN - ARTIST

Using temporary table space: TEMPSPACE1

Using index: SYSIBM.SQL971119105442180

Figure 18.27: The Reorganize dialog box.

4. In the Using Temporary Table Space combo box, change the name of the table space where the table being reorganized can be temporarily stored. The default is none, meaning that the temporary copy is stored in the table space where the table now resides. To specify a different table space, type the name of an existing table space, or select an existing table space from the drop-down list. The reorganization process uses a temporary copy of the table to do the reorganization.

Tip It's generally recommended that you specify a temporary SMS table space. If the table you're reorganizing resides in a DMS table space, and you think the temporary space will fit in the same table space, don't specify a table space in this combo box. The reorganization runs faster in this case.

Specifying a DMS table space as the temporary table space isn't generally recommended. If you specify a DMS table space in this field, you can reorganize only one table at a time in that table space.

5. In the Using Index combo box, specify an index to use to reorganize the table rows. If you specify none, the table rows are reorganized without regard to the order.

6. Click Reorganize Now to reorganize the table immediately.

   Alternatively, click Schedule to open the Schedule window to schedule thereorganization for a specific time or date.

   Tip You might want to schedule this activity because reorganizing data canbe time-consuming and users won't be able to access the table beingorganized.

7. When the tables are reorganized, collect the statistics again, as explained earlier in the section "Collecting Statistics." This provides the most up-to-date statistics for the data so that you can have the fastest access to the data, based on the new organization of the data and indexes.

You should rebind applications that use static SQL after collecting statistics because the SQL optimizer might choose a different access plan, given the new statistics. In particular, you should rebind those programs that reference tables for which the new statistics are available. (See Day 10, "Accessing the Data.")

Summary

In today's lesson, you saw that DB2 provides database system monitor commands and APIs that enable you to find out more about the operation of the database manager. You can use this information to obtain a better understanding of how the database manager works, to tune configuration parameters, or to determine the source and cause of problems.

The Performance Monitor, a graphical interface to the database system monitor functions, allows you to analyze snapshots through the Snapshot Monitor and events through the Event Monitor. These tools enable you to identify and analyze performance problems and to identify exception conditions based on thresholds.

Visual Explain graphically shows you the access plans for explained SQL statements. By using the Visual Explain tool, you can view the statistics used during optimization, determine whether an index was used to access a table, and view all the details of the access plan, including statistics in the system catalogs.

You can collect statistics to update system catalog statistics on the data in the table. The optimizer uses these statistics to choose which path to use when accessing data.

What Comes Next?

On Day 19, "Design Considerations," you're introduced to the options you should consider when designing a database or applications.

Q&A

Q When do I use an event monitor versus a snapshot monitor?

A Use the event monitor when you want a good summary of the activity of a particular event. When using event monitors, data is returned when the event occurs. An example of an event is a database connection.

A snapshot monitor, on the other hand, provides a picture of the current state of all activities in a database instance. When using snapshot monitors, data is returned for the point in time that the snapshot was requested. You would typically use a snapshot monitor at the time you're experiencing a problem. An object that you might want to obtain snapshot information about is a database.

Q Can I create more than one event monitor at a time?

A You can start up to 32 event monitors at a time. Each monitor can be turned on and off as required. This flexibility gives you the opportunity to achieve your desired monitoring.

Q When using snapshot monitoring, where can I find out whether a performance variable exceeded its threshold?

A The Performance Details page gives you details on exceeded thresholds. Typically, you get to the Performance Details page through the Alert Center. If you selected to have an event recorded in the Alert Center each time a threshold is exceeded, the Alert Center automatically opens, showing you the object that has exceeded a threshold. Double-click the object to see the performance details. You can also view the performance details directly through the Control Center.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. What do you need to create an access plan graph?
2. What's an operator? How is it shown on an access plan graph?
3. When is it recommended that you collect statistics and reorganize a table?
4. What should you do immediately following a Run Statistics or a Reorganize Table operation?

Exercise

Create several event monitors and snapshot monitors for your database. Change the performance variables to make it easier to exceed the thresholds. Perform several actions on the database to cause the thresholds to be exceeded. Look at the graphs and data provided. Create an Access Plan Graph, perform a Run Statistics and Reorganize Table operation on the table you're querying, and re-create the Access Plan Graph. Did any of the values change?

Day 19: Design Considerations

Overview

Today's lesson gives an overview of what you need to think about when designing your databases or applications. You'll learn, for example, how to use your indexes to ensure that you can access your data quickly, how to use pointers to access your multimedia objects, and how to separate data into partitions to improve performance. You'll also learn the basics of managing concurrency with isolation levels and how row and table locking affect the concurrency of your data. Also covered are some programming techniques that you can use to reduce network traffic between a database client and server. These techniques include using stored procedures, blocking SQL statements, and retrieving groups of rows.

Note The tips covered in this lesson should be considered when you're designing your database. However, you can use some of these tips to help improve performance after your database is designed.

Index Considerations

An index is a table's key value. It provides pointers to the rows in the table, which enables more efficient access by creating a direct path to the data. An index optimizes data retrieval without performing a lengthy sequential search—that is, you can avoid having the entire table scanned when data is queried. An index doesn't have to be defined for a database table, but it can improve system performance by establishing a high-speed path to frequently accessed information.

You don't decide when an index should be used to improve performance; DB2 makes this decision, based on the available table and index information. However, you play an important role in the process by defining keys that create the necessary indexes. It's also important for you to collect statistics on your indexes—when you create them and on an ongoing basis.

Indexes can also be separated from data to further improve performance. See the later section "Separating Different Data Types" for further information.

Tip The best index on a table is one that uses high-speed disks, is highlyclustered, and is made up of only a few narrow columns.

Large Object Considerations

When using large objects in your database, you have special performance considerations. DB2 Universal Database provides the flexibility for you to

- Access only those portions of a large object that are needed by the application—for example, using the `SUBSTR` scalar function.

- Use LOB locators to defer or avoid materialization of large objects.

- Choose whether changes to a column will be logged when defining a large object column.

  Tip As a rule of thumb, you might not want to log `LOB` columns larger than 10MB. If the `LOB` column is larger than 1GB, logging must be turned off. To turn logging off, specify the `NOT LOGGED` clause when the table is created. You can specify the NOT LOGGED option when you're adding a column with the Create Table notebook. Day 8, "Creating Databases," covers the details of using the Create Table notebook.

- Store large object data in separate database partitions specially architected for large object handling.

Database Partitioning

Database administrators can partition a database into parts known as *table spaces*. A table space has one or more *containers* (preallocated system files or directories)associated with it. Containers are separate identifiers of storage space to be used by the table space. (How to create table spaces was covered in detail in Day 9, "Creating Table Spaces.")

Separating data into table spaces provides a number of advantages, including separating different data types and enabling prefetch. Each is described in the following sections.

Separating Different Data Types

When you create a table, you should consider keeping different types of data, indexes, and large object (LOB) data separate from the rest of the table data. These separate table spaces can possibly be on different media than the rest of the data. This flexibility enables you to choose the table space characteristics and the physical devices supporting those table spaces to best suit the type of data. You can also allocate a table space to a specific buffer pool.

Note Separating data is possible only if you're using DMS table spaces.

You might, for example, have tables containing infrequently used historical data. As a result, end users might be willing to accept a longer response time for queries executed against this data. In this situation, you could use a different table space for the historic tables and assign this table space to less expensive physical devices with slower access rates.

You might be able to identify some essential tables that require high availability and fast response time. You might want to put these tables into a table space assigned to a fast physical device that can help support these important data requirements.

Indexes stored in a different table space from that used to store other table data can allow for more efficient use of disks by reducing the movement of read/write heads. You can also allocate faster devices to indexes, which can speed index access.

Long column data stored in a different table space enables you to allocate a special device to this type of data. For example, you can choose to store large object data on devices that have a very large capacity but might be slow and to store the rest of your data on a faster device.

Enabling Prefetch

Enabling prefetch means that index and data pages are copied into the buffer pool from disk before they're actually needed. This can help improve performance by reducing the time spent waiting for the I/O to complete. By splitting data across devices, the prefetch operations become more effective at improving performance. See Day 20, "Tuning DB2 Universal Database Performance," for more information on prefetch operations.

Allocating Additional Space

New containers can be added when you alter a table space. The database manager will then automatically rebalance the tables in the table space across all available containers. During rebalancing, data in the table space remains accessible. Refer to Day 11, "Using System Administration Tools," for information on how to add new containers to table spaces.

If you're adding more than one container to the table space, add all the containers at one time to save the overhead of performing the balancing step more than once. It isn't required, but it's recommended that all containers be the same size. Data is added to containers in a striping manner to ensure that data is evenly distributed across all containers. Data is first added to one container, then the next, and so on. If one container is much smaller than the other containers, you might receive a message that the containers are full. For improved performance, assign each container to a separate disk to enable data to be accessed from all containers at the same time.

Managing Concurrency

Whenever you have multiple concurrent users, you have to manage the interaction of their activities. That is, you have to determine to what degree each user appears to be acting independently. What you manage is the degree of independence or isolation of one user from the others. DB2 offers four degrees of isolation, ranging from very isolated to no isolation:

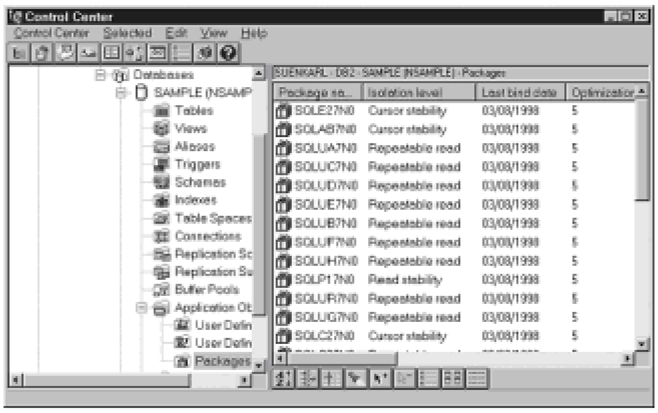
- *Repeatable Read* keeps a lock on all rows accessed by the application program since the last commit point and holds these locks until the next commit point. If the application program reads the same row again, the values won't have changed. The effect of this isolation level is that one application program can prevent other application programs or users from changing tables. As a result, overall concurrency can decrease.

- *Read Stability* keeps a lock on all qualifying rows accessed by the application program until the unit of work is complete. Changes made by other applications can't be read until the changes are committed by those applications. If an application issues the same query more than once, it can read the new or changed rows.

- *Cursor Stability* holds a row lock only while the cursor is positioned on that row. When the cursor moves to another row, the lock is released. If the data is changed, however, the lock must be held until the data is committed. Cursor stability applies only to read data. All changed data remains locked until a `COMMIT` or `ROLLBACK` is processed.

- *Uncommitted Read* enables you to view rows without waiting for locks. It applies only to read-only and `SELECT` statements. For other operations, it performs the same way as cursor stability. An application program using this isolation level reads and returns all rows, even if they contain uncommitted changes made by other application programs. Because this isolation level doesn't wait for concurrency locks, overall performance can increase.

The isolation level is specified at precompile time or when an application is bound to the database. If no isolation level is specified, the default of cursor stability is used. To see the isolation level used for a package, select the package folder in the object tree in the Control Center. The right panel provides details on the package, including the isolation level. Figure 19.1 shows the isolation levels set for the packages in the SAMPLE database.



Package na...	Isolation level	Last bind date	Optimization
SQLE27N0	Cursor stability	03/08/1998	5
SQLAB7N0	Cursor stability	03/08/1998	5
SQLUA7N0	Repeatable read	03/08/1998	5
SQLUC7N0	Repeatable read	03/08/1998	5
SQLUD7N0	Repeatable read	03/08/1998	5
SQLUE7N0	Repeatable read	03/08/1998	5
SQLUB7N0	Repeatable read	03/08/1998	5
SQLUF7N0	Repeatable read	03/08/1998	5
SQLUH7N0	Repeatable read	03/08/1998	5
SQLP17N0	Read stability	03/08/1998	5
SQLUR7N0	Repeatable read	03/08/1998	5
SQLUG7N0	Repeatable read	03/08/1998	5
SQLC27N0	Cursor stability	03/08/1998	5

Figure 19.1: Isolation level used by packages.

On Day 14, "Replicating Data," you were required to enter the following commands to see up the Apply tool:

```
db2 bind @applycs.lst isolation cs blocking all grant public
db2 bind @applyur.lst isolation ur blocking all grant public
```

These bind commands are setting the isolation levels for the packages. The first has the option `isolation cs`, which sets the isolation level for the package to cursor stability. The second command has `isolation ur`, which sets the isolation level for the package to uncommitted read.

Row and Table Locking

Concurrent processing means that multiple processes or applications can access the same database at the same time. DB2 uses *locking* to maintain data integrity during concurrent processing. Locking guarantees that a transaction maintain control over a database row until it has finished, and locking prevents another application from changing a row before the ongoing change is complete.

DB2 Universal Database provides *table-level* and *row-level locking*. Row-level locking provides finer granularity and better concurrency support. If extensive changes are required to the database, you may want to avoid contention by using the `LOCK TABLE` statement in an application instead, to lock the entire table until the transaction is committed or rolled back. Figure 19.2 summarizes the two lock-level methods.

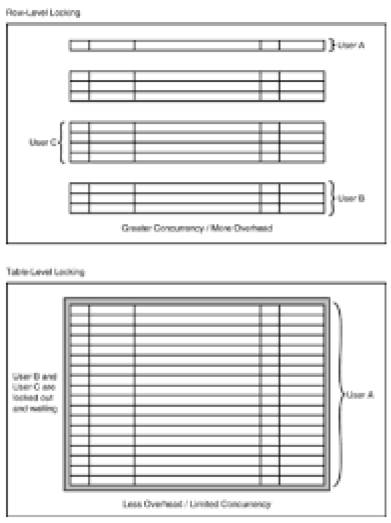

Row-Level Locking
User A
User C
User B
Greater Concurrency / More Overhead
Table-Level Locking
User B and User C are locked out and waiting
User A
Less Overhead / Limited Concurrency

Figure 19.2: Row-level and table-level locking.

Tables and rows can be locked in share or exclusive mode. If share mode is chosen, other applications can retrieve data as read-only.

DB2 locks a single row as it's accessed, meaning that only the rows being updated, inserted, deleted, or read (unless the isolation level is cursor stability or uncommitted read) are locked. These rows remain locked until the transaction is committed or rolled back. If concurrency is less of a consideration, you can avoid the overhead cost ofrow-level locking by using the `LOCK TABLE` statement to lock an entire table. The lock isn't released until the transaction is committed or rolled back.

When the number of locks issued exceeds the capacity specified in the databaseconfiguration file, a *lock escalation* condition occurs. During a lock escalation, locks are freed by converting locks on rows of a table into one lock on a table. This is repeated until enough locks are freed by one or more processes to meet the capacity specified in the database configuration file.

DB2 Universal Database uses isolation levels to determine the basic locking scheme for cursors in an application. A *cursor* is a SQL mechanism that enables an application to retrieve a set of rows from the database.

A *deadlock* can occur when two or more transactions are each waiting for data locked by the other. A *deadlock detector* is periodically activated in the background to check the locks now in the system to determine whether there's a deadlock situation. If there is, the detector selects a transaction involved in the deadlock and stops it. This transaction will be rolled back, and the other transactions will be able to proceed. The frequency of the deadlock detector activity is controlled by a parameter in the database configuration file.

You also can choose to have locks *time out*. If locks haven't been released in the time specified in the timeout interval configuration parameter, the waiting application will time out, and a failure return code will be passed to the application or user.

Stored Procedures

In a database application environment, many situations are repetitive—for example, receiving or returning a fixed set of data or performing the same multiple requests against a database.

A *stored procedure* is a technique that enables an application running on a client to call a procedure stored on a database server. The server procedure executes and accesses the database locally and returns information to the client application. To use this technique, an application must be written in two separate procedures:

- The calling procedure is contained in a client application and executes on the client.
- The server procedure executes at the location of the database on the databaseserver.

Applications using stored procedures have the following advantages:

- Reduced network traffic
- Improved performance of server-intensive work
- Access to features that exist only on the database server

The use of stored procedures is applicable in all DB2 environments. Network traffic will be reduced because only the initial request and final result flow across the network.

Row Blocking

DB2 Universal Database uses row blocking to provide transmission of data in blocks, instead of one row at a time. A *block* is a group of rows returned from a local or remote database in response to a `FETCH` request from an application. Performance is usually enhanced by reducing the number of requests made against the database manager.

The three types of row blocking are `UNAMBIGUOUS`, which causes blocking for read-only requests; `UNAMBIGUOUSALL`, which causes blocking for read-only and ambiguous requests; and `NO UNAMBIGUOUS`, which doesn't block any requests.

Row blocking is specified at precompile time or when an application is bound to the database. If no blocking method is specified, the default of `UNAMBIGUOUS` is used. To see the blocking used for a package, select the package folder in the object tree in the

Control Center. The right panel provides details on the package, including the blocking type used.

On Day 14, "Replicating Data," you were required to enter the following commands to see up the Apply tool:

```
db2 bind @applycs.lst isolation cs blocking all grant public
db2 bind @applyur.lst isolation ur blocking all grant public
```

Both `bind` commands specify to use `BLOCKING ALL`, meaning that all read-only and ambiguous requests are blocked.

Compound SQL

Compound SQL enables you to group several SQL statements into a single executable block. The SQL statements contained within the block (substatements) can be executed individually; however, by creating and executing a block of statements, you reduce the database manager overhead. For remote clients, compound SQL also reduces the number of requests that have to be transmitted across the network.

Global SQL Cache

The aim of the Global SQL Cache is to minimize the amount of catalog access required for sections of static SQL statements and to maximize the sharing of sections fordynamic DML statements by eliminating many previous restrictions. This is done by establishing a global cache shared by all agents connected to the same database, in which sections for static and dynamic SQL statements will be placed. This global cache acts as a public repository, or library, for different sections being used on the database at any given time.

SQL Optimization

DB2 includes a sophisticated cost-based SQL Optimizer, which provides superior complex query performance. The purpose of the SQL Optimizer is to select the most efficient way to access the data. Relational databases offer many ways of accessing the data, each with a cost associated with it. The optimizer considers these potential paths, assigns a cost to each one by using its model of the database, and selects the most efficient access path by choosing the one with the lowest estimated overall cost.

You can specify optimization levels at precompile time or when an application is bound to the database. If no optimization level is specified, the default value set in the database configuration parameter `DFT_QUERYOPT` is used. You can set this value by right-clicking the database in the Control Center and selecting Configure from the pop-up menu. The `DFT_QUERYOPT` parameter is set on the Enviroment page of the Configure Databasenotebook.

To see the optimization level used for a package, select the package folder in the object tree in the Control Center. The right panel provides details on the package, including the optimization level used.

The following sections cover a number of methods for influencing the optimizer. These include query rewrite, catalog statistics, distribution statistics, and optimization classes.

Query Rewrite

The optimizer considers a number of things that can affect the path selected. One thing it considers is the SQL statement. How a statement is written can affect the chosen path by the optimizer.

The optimizer includes a very sophisticated query rewrite phase that transforms queries into forms that can be optimized more effectively. This ensures that you have the fastest route to your data. For example, it adds logically implied predicates, merges views, converts subqueries into joins, and eliminates redundant references to tables.

Catalog Statistics

One way you can influence the path selection is to update the catalog statistics. Statistics about the physical characteristics of tables and indexes are stored in the catalog and used to determine the best way to access the data.

You may want to update catalog statistics for several reasons:

- **To update current statistics that better reflect how data is stored.** This isusually done when many changes have been made to the database. Use the Run Statistics dialog box (discussed on Day 18, "Performance Aids") to update thecurrent statistics in the system catalog tables.
- **To model query performance on a development system that has production system statistics applied.** To model production system statistics, you can update the catalog statistics and make them permanent by using a `COMMIT` statement. This enables you to ensure that your production system is operating at peak performance without the need to interrupt production systems.
- **To perform "what-if" query performance analysis.** Visual Explain has built-in function to perform this type of analysis. (Visual Explain was covered on Day 18, "Performance Aids.") It ensures that the changes remain temporary by doing a rollback at the end of the analysis. This way, you can test assumptions without affecting production systems.

Reorganizing Tables in a Database

A table can become fragmented because of many updates, causing performance to deteriorate. If you collected statistics and didn't notice a visible performance improvement, reorganizing table data may help. When you reorganize table data, you're rearranging the data of a table into a physical sequence according to a specified index and removing the free space that's inherent in fragmented data. This can provide faster access to the data and thereby improve performance.

See Day 18 for details on reorganizing tables.

Distribution Statistics

If the distribution of your data isn't uniform, you can ensure that the optimizer considers the actual distribution of your data. A set of column-level statistics enables the optimizer to recognize many non-uniform data distributions. This allows for the calculation of more accurate selectivity estimates by the optimizer, which in turn aids the optimizer in choosing better access plans.

Optimization Classes

The compiler has several optimization classes. The highest level of optimization might result in the best route to your data, but the compilation can take longer and more space might be required to store the additional statistics that will be kept. You can set the compiler's optimization class to control the resources and performance characteristics of query compilation.

In general, use higher optimization classes with complex SQL. The overhead of query optimization when processing large amounts of data is negligible, but enhancedoptimization may produce a noticeably better access plan.

Summary

In this lesson, you learned that indexes can be created to improve performance. However, it's DB2 that decides when the index will be used.

You can partition your data into table spaces to separate different types of data to help improve performance.

DB2 provides four degrees of isolation to help you manage multiple concurrent users that access data in your databases. DB2 also includes a sophisticated cost-based SQL optimizer used to select the most efficient path to your data.

What Comes Next?

On Day 20, "Tuning DB2 Universal Database Performance," you learn how to improve the performance of your system by adjusting the configuration parameters, using directory caching, and tuning I/O operations.

Q&A

Q What's a requirement for using table spaces to separate different types of data?

A The two types of table spaces are database managed space (DMS) or system managed space (SMS). If you want to separate different types of data, you must use DMS table spaces. You may want to separate indexes or `LOB` data into separate table spaces to improve the performance.

Q What are the four degrees of isolation?

A DB2 provides repeatable read, read stability, cursor stability, and uncommitted read as ways to control how locks are handled when multiple concurrent users access the same data.

Repeatable Read enables you to lock the rows you're accessing until you commit your changes. This prevents all other users from changing the data in the rows you're accessing until your changes are completed.

Read Stability also locks the data that you're accessing until you complete your changes but enables other applications to change rows that might affect the result table that you create.

Cursor Stability locks the row you're accessing while the cursor is positioned on the row. The lock remains in effect until the next row is fetched or changes are committed.

Uncommitted Read enables an application to access uncommitted changes of other transactions. The application also doesn't lock other applications out of the row it's reading unless the other application attempts to drop or alter the row.

Q What are some ways to improve performance if I have a large amount ofnetwork traffic?

A Stored procedures can reduce the amount of network traffic. A stored procedure enables an application running on a client to call a procedure stored on the database server. The server executes and accesses the database locally and returns information to the client application.

Row blocking enables a group of rows to be returned from the database in response to a `FETCH` operation. Without row blocking, each row is transmitted separately.

You can use compound SQL to group several SQL statements in a single executable block. Having a group of statements reduces the number of requests that have to be transmitted across the network.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. When would you use table locking versus row locking?
2. What's a deadlock? How can you break out of a deadlock situation?
3. How can you ensure that your catalog statistics are up-to-date? Why is thisimportant?
4. When should you use the highest level of optimization?

Exercise

On Day 18, you learned how to create an access plan graph. For today's exercise, create an access plan graph and determine whether an index is used when accessing the data. Create several indexes on the table and rerun the access plan graph to see whether any of the new indexes are used.

Day 20: Tuning DB2 Universal Database Performance

Overview

The easiest way to improve database performance is by tuning configuration parameters. DB2 enables you to control configuration parameters at two levels: for the entire environment (known as *database manager configuration*) or for a single database (known as *database configuration*). You can set these parameters by using commands at a command prompt, or you can use the Client Configuration Assistant or the Control Center. This lesson shows how to change your parameters through the graphical tools and provides descriptions for several of the parameters.

You can further tune your environment with environment variables and registry values. This lesson teaches you how to set these values and explains whether to set a value at an instance or global level.

DB2 provides the Performance Configuration SmartGuide to help you determine how many of your configuration parameters should be set. This lesson guides you through using this tool to tune the performance of your system.

Also covered in this lesson are several I/O tuning considerations. You can createadditional buffer pools and set configuration parameters to ensure I/O tasks areperformed in parallel.

Controlling Your DB2 Environment

Registry values, environment variables, and configuration parameters control yourdatabase environment.

You can update DB2 registry values for an instance, provided that you have system administration (`SYSADM`) authority for that instance. Use the `db2set` command to update DB2 registry values without rebooting your system. The DB2 registry applies the updated information to DB2 server instances and DB2 applications started after the changes are made.

DB2 configures its operating parameters by checking for variable values according to the following search order:

- The environment variable settings
- Profile registry values set with the `db2set` command in the instance-level profile
- Profile registry values set with the `db2set` command in the global-level profile

See the later section "DB2 Registry Values and Environment Variables" for descriptions of the subset of registry values and environment variables that you may want to adjust.

Controlling the DB2 Profile Registry

The DB2 profile registry stores DB2 registry values. The *levels* of registry values are as follows:

- The *DB2 instance-level profile* contains instance-level variable settings and overrides. Values defined in this level will override their settings in the global level.
- The *DB2 global-level profile* contains systemwide variable settings. Any variable not defined at the instance level is evaluated at the global level.

To modify registry variable values, use the `db2set` command. The syntax of the `db2set` command is as follows:

Syntax

- To set a parameter for the current instance, use db2set parameter=value.
- To set a parameter's value for a specific instance, use db2set parameter=value -i intant–name.
- To set a parameter at the global profile level, use db2set parameter=value -g.

Note Some parameters, including `DB2SYSTEM` and `DB2INSTDEF`, will always default to the global-level profile. They can't be set at the instance- or node-level profiles.

Syntax

- To delete a parameter's value at a specified level, you can use the same command syntax to set the parameter but specify nothing for the parameter value. For example, to delete the parameter's setting at the global level, use db2set parameter= -g.
- To explicitly unset a parameter's value at a specified level and prevent evaluating the parameter at the next level, use `db2set parameter= –`

`null -i` *`intant-name`*. The `-null` option lets you set the value of a parameter globally but unset the value for a specific instance. For example, you can set `DB2COMM` to `TCPIP` at the global level. If you have four instances on your system, you can set three of the instances to default to the global setting but use `-null` to unset `DB2COMM` on the fourth instance.

- To evaluate the current session's parameter's value, use `db2set parameter`.
- To evaluate the parameter's value at all levels, use `db2set parameter -all`.
- To view a list of all values defined in the profile registry, use `db2set -all`.

Setting Your Environment

It's strongly recommended that all DB2-specific registry values be defined in the DB2 profile registry. If DB2 variables are set outside the DB2 registry, remote administration of those variables won't be possible, and the workstation will have to be rebooted for the variable values to take effect.

DB2 on Windows NT has one system environment variable, `DB2INSTANCE`, that can be set only outside the DB2 profile registry. You aren't required to set `DB2INSTANCE`, however; the DB2 profile registry variable `DB2INSTDEF` can be set in the global-level profile to specify the instance name to use if `DB2INSTANCE` isn't defined.

Syntax To determine the setting of an environment variable, use

```
echo %variable-name%
```

Set system environment variables in the Windows NT System Properties dialog box. If the variable doesn't exist, do the following:

1. Choose Start | Settings | Control Panel, and double-click the System icon.
2. In the System Properties dialog box, click the Environment tab, and select any environment variable in the System Variables list.
3. Change the name in the Variable text box to the name of the environment variable you want to set—for example, `DB2INSTANCE`.
4. Change the Value text box to the instance name—for example, `db2inst`.
5. Click the Set button.
6. Click OK.

You might have to reboot your system for these changes to take effect.

If the variable already exists in the System Variables list, you can set a new value asfollows:

1. Select the environment variable you want to append—for example, `DB2INSTANCE`.
2. Change the Value text box to the instance name—for example, `db2inst`.

3. Click the Set button.

4. Click OK.

You might have to reboot your system for these changes to take effect.

Tip The environment variable `DB2INSTANCE` can also be set at the session level. For example, if you want to start a second DB2 instance called `EXAMPLES`, issue the following commands in a command window:

```
set db2instance=EXAMPLES

db2start
```

DB2 Registry Values and Environment Variables

Table 20.1 shows the subset of the DB2 registry values and environment variables that you may need to know about in order to get up and running. Each has a brief description; some may not apply to your environment.

Tip To view a list of all supported variables, use the following command:

```
db2set -lr
```

Table 20.1. DB2 registry values and environment variables.

Parameter	Values	Description
	General	
`DB2DBDFT`	Default not set	Specifies the database alias name of the database that will be implicitly connected to when applications are started and no implicit connect has been done. This keyword is ignored if it's set.
`DB2DISCOVERYTIME`	Default=40 seconds, minimum=20 seconds	Specifies the amount of time that SEARCH discovery will search for DB2 systems.
`DB2INSTDEF`	Default=`DB2`	Sets the value to be used if `DB2INSTANCE` isn't defined.
	System Environment	

`DB2INSTANCE`	Default=`db2instdef`	The environment variable used to specify the instance that's active by default.
	Default not set	The environment variable used to specify the location of the instance directory, if different from `DB2PATH`.
`DB2PATH`	Default= `x:\sqllib`	The environment variable used to specify the directory where the product is installed.
	Communications	
`DB2COMM`	Default=null, values=any combination of `APPC`, `IPXSPX`, `NETBIOS`, `NPIPE`, `TCPIP`	Specifies the communication managers that start when the database manager is started. If this isn't set, no DB2 communications managers are started at the server.
`DB2NBADAPTERS`	Default=0, range=0–15; multiple values should be separated by commas.	Used to specify the loca adapters to use for DB2 NetBIOS LAN communications. Each local adapter is specified with its logical adapter number.

Configuration Parameters

DB2 has been designed with an extensive array of tuning and configuration parameters. Configuration parameters affect the operating characteristics of a database or database management system. The parameters' default values might be sufficient to meet your needs; however, you may not be able to achieve maximum performance with these default values. Specifically, you may need to modify the default parameter values if your environment has any of the following:

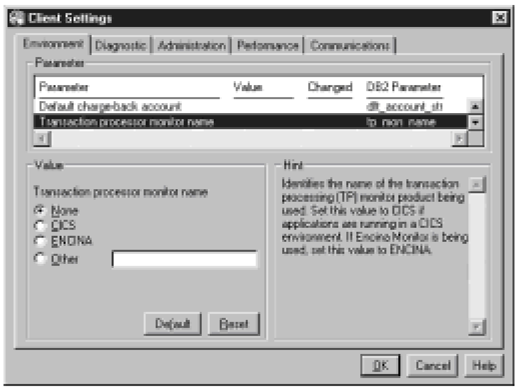
- A large amount of memory (greater than 64MB)
- Large databases (greater than 1GB)
- Large numbers of connections
- High performance for a specific application
- Different machine configuration/use
- Unique query or transaction loads or types

Each transaction processing environment has one or more unique aspects within it; these differences can have a profound effect on the performance of DB2 when using the default configuration. For this reason, you're strongly advised to take the time to customize DB2 for your environment.

Database manager configuration parameters exist on servers and clients. However, only a subset of the database manager configuration parameters can be set on a client. Database configuration parameters, on the other hand, can be set only on the server.

Setting Database Manager Configuration Parameters on a Client Instance

You can control database management configuration parameters on a client instance by using the Client Configuration Assistant. To access the client settings, choose Start | Programs | DB2 for Windows | Client Configuration Assistant. In the Client Configuration Assistant window, click the Client Settings button. Figure 20.1 shows the Client Settings window of the Client Configuration Assistant.



Parameter	Value	Changed	DB2 Parameter
Default charge-back account			dft_account_str
Transaction processor monitor name			tp_mon_name

Transaction processor monitor name

[✓ None] CICS ENCINA Other

Identifies the name of the transaction processing (TP) monitor product being used. Set this value to CICS if applications are running in a CICS environment. If Encina Monitor is being used, set this value to ENCINA.

Figure 20.1: The Client Settings dialog box.

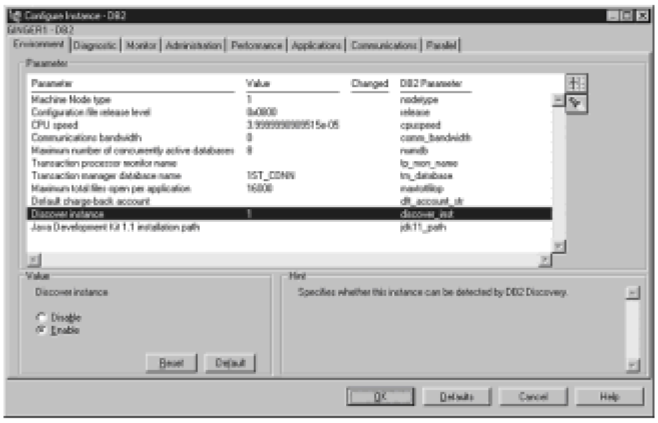
Follow the hints provided with the CCA in the Client Settings dialog box to modify the database manager and database configuration parameters for your environment. For each parameter that you select, recommendations for setting the parameter appear in the dialog box's hint section.

Setting Database Manager Configuration Parameters on a Server Instance

To control database manager configuration parameters on a server instance, follow these steps:

1. In the Control Center, expand the object tree until you see the instance you want to configure. For example, to configure the default instance, look for DB2.

2. Right-click the instance you want to configure, and select Configure from the pop-up menu. The Configure Instance notebook opens (see Figure 20.2).


Configure Instance - DB2

GINGER1 - DB2

Parameter	Value	Changed	DB2 Parameter
Machine Node type	1		nodetype
Configuration file release level	0x0800		release
CPU speed	3.99999999999515e-05		cpuspeed
Communications bandwidth	0		comm_bandwidth
Maximum number of concurrently active databases	8		numdb
Transaction processor monitor name			tp_mon_name
Transaction manager database name	1ST_CONN		tm_database
Maximum total files open per application	16000		maxtotfilop
Default charge-back account			dft_account_str
Discover instance	1		discover_inst
Java Development Kit 1.1 installation path			jdk11_path

Value: Discover instance ( ) Disable [✓ Enable]

Hint: Specifies whether this instance can be detected by DB2 Discovery.

Figure 20.2: The Configure Instance notebook.

The Configure Instance notebook contains eight tabbed pages: Environment, Diagnostic, Monitor, Administration, Performance, Applications, Communications, and Parallel. On each page you'll see the configuration parameters that correspond to each category. Select a parameter to be able to change its value.

3. Click Help to see a description of the parameter you've selected. Click Defaults to have each parameter value set to its default value. Click a parameter to see the hint text to help you set the value.

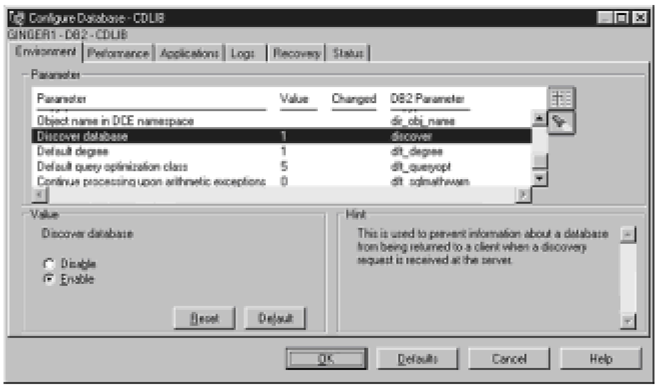
4. Click OK when you've finished changing the parameter values. The changes will take effect when you restart the instance.

Setting Database Configuration Parameterson a Server Instance

To control database configuration parameters on a server instance, use the Control Center:

1. In the Control Center, expand the object tree until you see the database that you want to configure—for example, the SAMPLE database.

2. Right-click the database you want to configure, and select Configure from the pop-up menu. The Configure Database notebook opens (see Figure 20.3).


Configure Database - CDLIB

GINGER1 - DB2 - CDLIB

Parameter	Value	Changed	DB2 Parameter
Object name in DCE namespace			dir_obj_name
Discover database	1		discover
Default degree	1		dft_degree
Default query optimization class	5		dft_queryopt
Continue processing upon arithmetic exceptions	0		dft_sqlmathwarn

Value: Discover database ( ) Disable [✓ Enable]

Hint: This is used to prevent information about a database from being returned to a client when a discovery request is received at the server.

Figure 20.3: The Configure Database notebook.

The Configure Database notebook contains six tabs: Environment, Performance, Applications, Logs, Recovery, and Status. On each page you'll see the configuration parameters corresponding to each category. Select a parameter to be able to change its value. You'll also see a tip that can help you set the parameter.

3. Click Help to see a description of the parameter you've selected. Click Defaults to have each parameter value set to its default value.

4. Click OK when you've finished changing the parameter values. The values will take effect when you restart the database.

Summary of Configuration ParametersCovered in This Book

All the configuration parameters introduced in this book are as follows:

- `LOGRETAIN`. Activates archive logging when the `LOGRETAIN` parameter is set to `YES`. When archive logging is activated, a full offline backup is required. If `LOGRETAIN` is turned off, logging reverts to circular, and the online logs are automatically deleted. Set this parameter on the server in the Control Center at an instance level.

- `DISCOVER`. Determines whether you can search the network for databases. On the server, update the Administration Server's configuration file in the command-line processor as follows:

  ```
  update admin cfg using discover [ DISABLE | KNOWN | SEARCH ]

  db2admin stop
  db2admin start
  ```

  On the client, update this parameter in the CCA.

- `DISCOVER_COMM`. Determines the protocol used during discovery functions. On the server, update the Administration Server's configuration file in the command-line processor as follows:

  ```
  update admin cfg using discover_comm [ NETBIOS | TCPIP ]

  db2admin stop
  db2admin start
  ```

  On the client, update this parameter in the CCA.

- `DISCOVER_INST`. Determines whether clients can discover databases in the instance. Set this parameter on the server in the Control Center at an instance level.

- `DISCOVER_DB`. Determines whether a database is available to be discovered by clients. Set this parameter on the server in the Control Center at an instance level.

- `DB2DISCOVERYTIME`. Specifies that the searched discovery wait 35 seconds for a response from servers. Set this parameter on the client in the CCA.

- `DB2NBDISCOVERRCVBUFS`. Specifies the number of NetBIOS buffers to be allocated for response messages from discovered servers. Set this parameter on the client in the CCA.

- `LOGPATH`. The roll-forward operation checks for logs in the location specified by the `LOGPATH` database configuration parameter.

Using the Performance Configuration SmartGuide

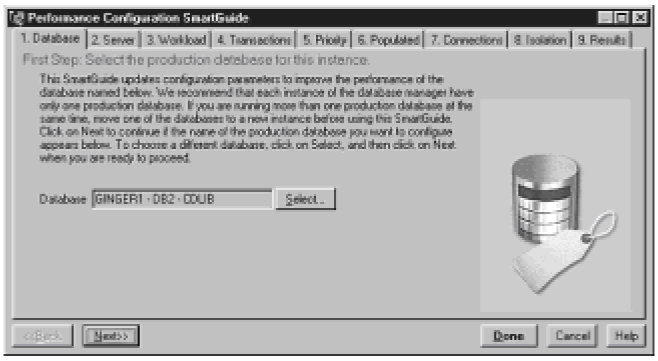
The DB2 Performance Configuration SmartGuide helps you tune performance and balance memory requirements for a single database per instance by suggesting which configuration parameters to modify and providing suggested values for them.

Note Many configuration parameters come with default values but may need to be updated to achieve optimal performance for your database. In most cases, the values recommended by the Performance Configuration SmartGuide will provide better performance than the default values because they're based on information about your workload and your own particular server. Note, however, that the values are designed to improve the performance of, though not necessarily optimize, your database system. They should be thought of as a starting point on which you can make further adjustments to obtain optimized performance.

Here are the steps to use the SmartGuide:

1. In the Control Center, right-click the database for which you want to configure performance, and select Configure Performance from the pop-up menu.

2. On the first page of the Performance Configuration SmartGuide (see Figure 20.4), select the database you want to configure.



First Step: Select the production database for this instance.

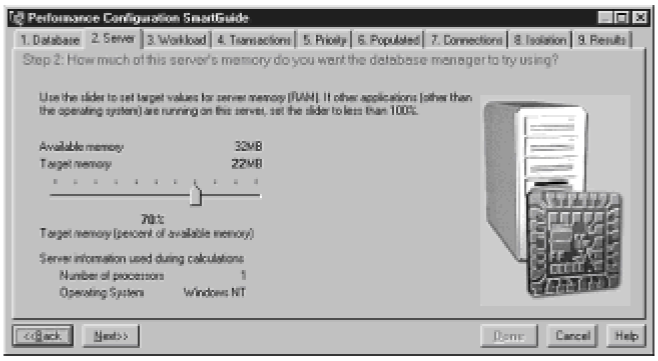
This SmartGuide updates configuration parameters to improve the performance of the database named below. We recommend that each instance of the database manager have only one production database. If you are running more than one production database at the same time, move one of the databases to a new instance before using this SmartGuide. Click on Next to continue if the name of the production database you want to configure appears below. To choose a different database, click on Select, and then click on Next when you are ready to proceed.

Database GINGER1 - DB2 - CDLIB Select...

Figure 20.4: The Performance Configuration SmartGuide's Database page.

Tip It's recommended that each instance of the database manager have only one production database. It's difficult to tune a system with more than one production database per instance running concurrently.

3. On the Server page (see Figure 20.5), use the slider to indicate the portion of the server's memory (RAM) that can be used by this database (not including the operating system). If other applications are running on this server, set the slider to less than 100%.



Step 2: How much of this server's memory do you want the database manager to try using?

Use the slider to set target values for server memory (RAM). If other applications (other than the operating system) are running on this server, set the slider to less than 100%.

Available memory 32MB
Target memory 22MB

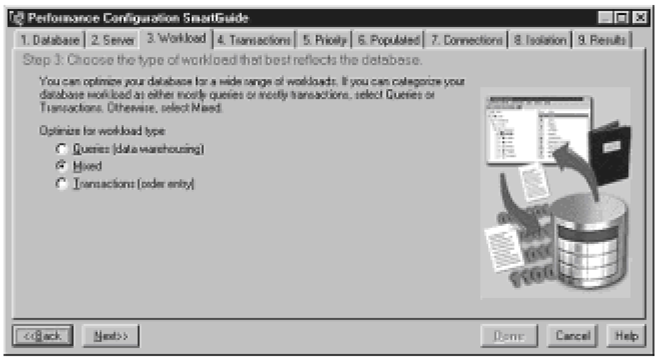
70%
Target memory (percent of available memory)

Server information used during calculations
Number of processors 1
Operating System Windows NT

Figure 20.5: The Performance Configuration SmartGuide's Server page.

4. On the Workload page (see Figure 20.6), select the option that best reflects your database workload type:

   - If there are many queries of the data, select Queries (Data Warehousing).
   - If the data is updated frequently, select Transactions (Order Entry).
   - If the data is updated and queried frequently, select Mixed.



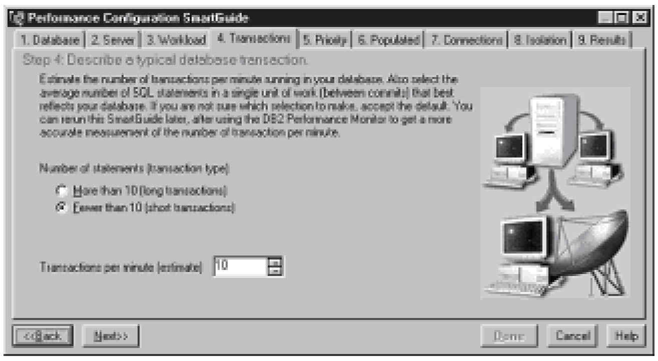
Step 3: Choose the type of workload that best reflects the database.

You can optimize your database for a wide range of workloads. If you can categorize your database workload as either mostly queries or mostly transactions, select Queries or Transactions. Otherwise, select Mixed.

Optimize for workload type
Queries (data warehousing) [✓ Mixed] Transactions (order entry)

Figure 20.6: The Performance Configuration SmartGuide's Workload page.

5. On the Transactions page (see Figure 20.7), indicate the approximate number of SQL statements in a single unit of work (between commits) that best reflects the database. Also, estimate the number of transactions per minute running in your database. If you aren't sure which value to use, accept the default provided by the SmartGuide.



Step 4: Describe a typical database transaction.

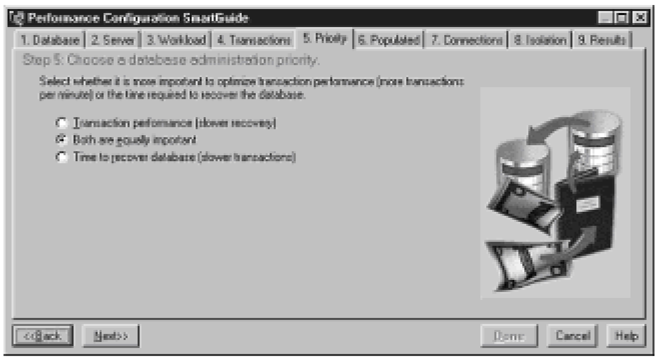
Estimate the number of transactions per minute running in your database. Also select the average number of SQL statements in a single unit of work (between commits) that best reflects your database. If you are not sure which selection to make, accept the default. You can rerun this SmartGuide later, after using the DB2 Performance Monitor to get a more accurate measurement of the number of transaction per minute.

Number of statements (transaction type)
More than 10 (long transactions) [✓ Fewer than 10 (short transactions)]

Transactions per minute (estimate) 10

Figure 20.7: The Performance Configuration SmartGuide's Transactions page.

6. On the Priority page (see Figure 20.8), indicate whether it's more important to optimize transaction performance or the time required to recover the database.



Step 5: Choose a database administration priority.

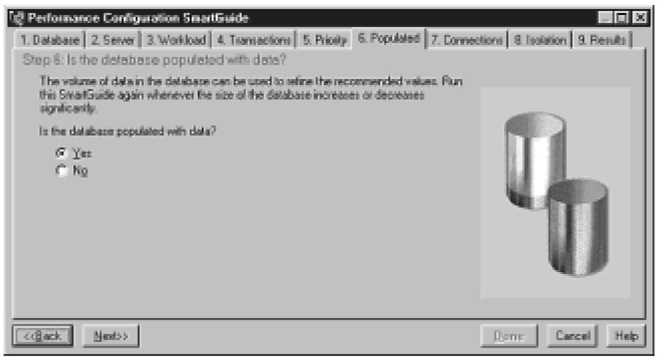
Select whether it is more important to optimize transaction performance (more transactions per minute) or the time required to recover the database.

- Transaction performance (slower recovery)
- [✓ Both are equally important]
- Time to recover database (slower transactions)

Figure 20.8: The Performance Configuration SmartGuide's Priority page.

- Select Transaction Performance to indicate that the number of transactions per minute is more important than the time it takes to recover the database.
- Select Time to Recover Database to indicate that the time it takes to recover the database is more important than the number of transactions per minute.
- Select Both Are Equally Important to indicate that you want a balance between the number of transactions per minute and the time it takes to recover the database.

7. On the Populated page (see Figure 20.9), indicate whether the database contains production data. If this database is new, you should rerun the SmartGuide after you insert data.



Step 6: Is the database populated with data?

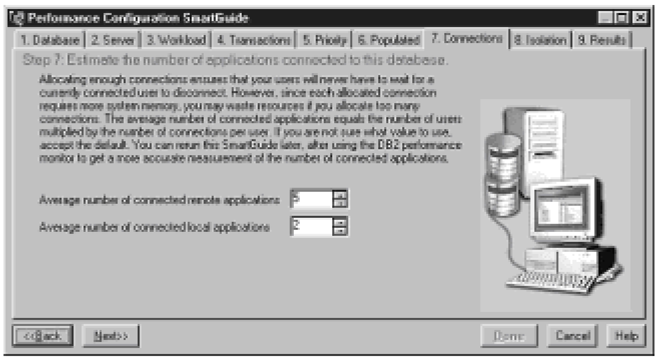
The volume of data in the database can be used to refine the recommended values. Run this SmartGuide again whenever the size of the database increases or decreases significantly.

Is the database populated with data?

[✓ Yes] No

Figure 20.9: The Performance Configuration SmartGuide's Populated page.

8. On the Connections page (see Figure 20.10), estimate the number of applications likely to connect to this database. You can indicate an approximate value for local and remote applications.



Step 7: Estimate the number of applications connected to this database.

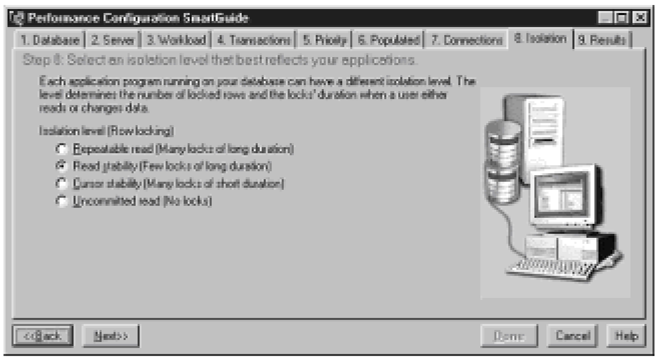
Allocating enough connections ensures that your users will never have to wait for a currently connected user to disconnect. However, since each allocated connection requires more system memory, you may waste resources if you allocate too many connections. The average number of connected applications 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. You can rerun this SmartGuide later, after using the DB2 performance monitor to get a more accurate measurement of the number of connected applications.

Average number of connected remote applications: 5

Average number of connected local applications: 2

Figure 20.10: The Performance Configuration SmartGuide's Connections page.

9. On the Isolation page (see Figure 20.11), select an isolation level that best reflects your application. The isolation level determines the number of locked rows and lock duration when users read or change data. DB2 uses locking to maintain data integrity during concurrent processing. Locking guarantees that a transaction will maintain control over a database row until it has finished and prevent another application from changing a row before the ongoing change is complete.



Step 8: 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.

Isolation level (Row locking)

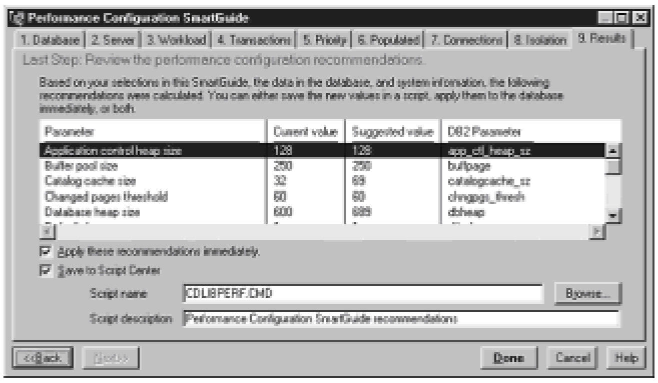
- Repeatable read (Many locks of long duration)
- [✓ Read stability (Few locks of long duration)]
- Cursor stability (Many locks of short duration)
- Uncommitted read (No locks)

Figure 20.11: The Performance Configuration SmartGuide's Isolation page.

You can select from the following isolation levels:

- Select Repeatable Read if you expect to have many locks of long duration.
- Select Read Stability if you expect to have few locks of long duration.
- Select Cursor Stability if you expect to have many locks of short duration.
- Select Uncommitted Read if you expect to have no locks.

10. On the Results page (see Figure 20.12), you can review the performance configuration recommendations. If you want to change any values, return to the preceding pages to make the changes. On the Results page, you can choose to apply the changes immediately or to save this to the Script Center to run at a later time. You can select both options and save the changes to a script to use on other systems or as a record of the changes you made.



Last Step: Review the performance configuration recommendations.

Based on your selections in this SmartGuide, the data in the database, and system information, the following recommendations were calculated. You can either save the new values in a script, apply them to the database immediately, or both.

Parameter	Current value	Suggested value	DB2 Parameter
Application control heap size	128	128	app_ctl_heap_sz
Buffer pool size	250	250	buffpage
Catalog cache size	32	69	catalogcache_sz
Changed pages threshold	60	60	chngpgs_thresh
Database heap size	600	689	dbheap

☑ Apply these recommendations immediately.
☑ Save to Script Center

Script name: CDLIBPERF.CMD
Script description: Performance Configuration SmartGuide recommendations

Figure 20.12: The Performance Configuration SmartGuide's Results page.

11. Click Done when you're satisfied with the changes that will be made to your configuration parameters.

For database configuration parameter updates to take effect, all applications must first disconnect from the database; the changes will take effect only at the first connect to the database. For database manager configuration parameter updates to take effect, you must first stop and then restart the instance.

Rerun the Performance Configuration SmartGuide if the size of the database significantly increases (for example, by more than 20%) or if the machine characteristics change (for example, more memory is available). An increase in the size of the database or adding memory to your system can significantly change the recommendations for various performance values, and you should rerun the Performance Configuration SmartGuide to take advantage of these changes.

Note The next time you use the SmartGuide, you're given the option of backing out of the options you selected the preceding time the SmartGuide was used.

Directory Caching

DB2 directories can be cached in memory to improve `CONNECT` performance. A configuration option can be set to turn on this feature. A shared directory cache is built during startup processing, and a private application directory cache is built when an application issues its first connect. Each cache provides an image of the system database directory, the distributed connection services directory, and the node directory. The use of the directory cache reduces `CONNECT` costs by eliminating directory file I/O and minimizing directory searches required to retrieve directory information.

Input/Output Performance

Typically, performing I/O is the slowest function a database needs to complete. DB2 has several features to eliminate I/O bottlenecks, such as prefetching data pages into memory in anticipation of the pages being needed, performing I/O in parallel by prefetching several pages of data at a time, and fetching blocks of related pages during the prefetch. You also can choose to validate constraints after loading data rather than while the data is being loaded. You can also create more than one buffer pool to store data pages and set configuration parameters to ensure that data pages are written to disk in a timely manner.

Prefetching Data Pages

Prefetching of data pages into the buffer pool can help improve performance by reducing the time spent waiting for I/O to complete. To *prefetch* pages means that one or more pages are retrieved from disk in anticipation of their use. Prefetching is started when DB2 determines that sequential I/O is appropriate and prefetching can help improve performance. There are two types of prefetch:

- *Sequential prefetch* reads consecutive pages into the buffer pool before the pages are required by the application. When the access plan looks sequential, the DB2 engine automatically begins sequential prefetch. Having DB2 activate or deactivate sequential prefetching as necessary is known as *sequential detection*. To specifically enable sequential prefetching, set the `SEQDETECT` database parameter in the database configuration file to `YES`.

- *List prefetch*, or list sequential prefetch, is a way to access data pages efficiently, even when the data pages needed aren't consecutive.

If access to the tables includes many queries or transactions that process large quantities of data, prefetching data from the tables can provide significant performance benefits. A DMS table space using multiple device containers in which each container is on a separate disk offers the best potential for efficient prefetching.

Parallel Input/Output

Many I/O operations can be performed in parallel on behalf of a single query. This can improve application response time, especially when the application performs a table scan. Parallel I/O is enabled for a database by setting the `NUM_IOSERVERS` parameter that controls the number of prefetch servers created (this parameter is set in the database configuration file). Configuring enough I/O servers can greatly enhance the performance of queries. Having extra I/O servers won't hurt performance because extra I/O servers aren't used and their memory pages are paged out.

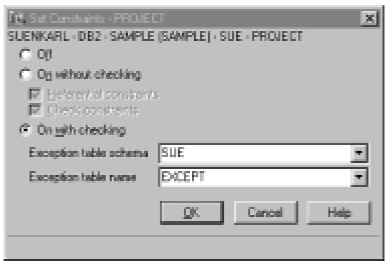
Big Block Reads

This capability enables several disk pages to be read by using a single I/O operation. Reading several disk pages at a time reduces the CPU use and elapsed time for I/O and improves query response time. Big block reads are used by the database manager every time prefetch is done. They can also be used for list prefetch if multiple pages being read belong to the same table space extent. DB2 determines when it is necessary to use big block reads.

Check Pending

Check pending is a *state* into which a table can be put. Only limited activity is allowed on the table when it's in this state, and constraints aren't checked while the table is updated. The load utility (discussed on Day 13, "Moving Data") exploits this function by automatically setting check pending off and then doesn't check referential constraints when loading data. Users are protected from using the loaded data before the constraints are verified.

When the data is loaded, you must use the Set Constraints dialog box to turn the constraints back on and to carry out the deferred checking (see Figure 20.13). You may also want to use check pending to improve performance when altering a table—for example, when adding foreign keys to a table. Checking of these keys will be postponed until all the changes are complete and the set constraints are turned back on. To access the Set Constraints dialog box, in the Control Center right-click the table for which you want to set constraints, and select Set Constraints from the pop-up menu.


Set Constraints - PROJECT

SUENKARL - DB2 - SAMPLE (SAMPLE) - SUE - PROJECT

Off

On without checking

[✓] Referential constraints

[✓] Check constraints

[✓ On with checking]

Exception table schema: SUE

Exception table name: EXCEPT

Figure 20.13: The Set Constraints dialog box.

To turn off constraint checking for a table, select Off. If you turn off constraint checking for a parent table, the foreign key constraints of all its dependent and descendent tables are put in check pending state.

You can turn on constraint checking for a table without checking the existing table data. Select On Without Checking to specify that the table not be checked immediately for constraint violations when constraint checking is turned on. Optionally, you can select the Referential Constraints and Check Constraints check boxes to specify the constraint type for which you are turning on constraint checking.

When you turn on constraint checking without performing the checking, this setting is recorded in the database catalog (the value in the `CONST_CHECKED` column in the `SYSCAT.TABLES` view is set to `U`). This value indicates that you've assumed responsibility (over the database management system) for ensuring the integrity of the data, with respect to the constraints turned on. This value remains the same until the table is put back into check pending state (by selecting Off in the Set Constraints dialog box) or until all unchecked constraints for the table are dropped.

You can also turn on constraint checking for a table and check the existing table data by selecting On with Checking. Normally, referential integrity and check constraints on a table are enforced automatically. You need to turn on constraint checking manually for a table if you've turned it off to postpone the checking.

You can optionally specify an exception table by using the Exception table schema and Exception table name fields. Any row that violates a referential or check constraint is deleted from your table and copied to the exception table. If you don't specify an exception table, when a constraint is violated, only the first violation detected is returned to you, and the table is left in the check pending state.

Buffer Pools

Each table is associated with a buffer pool. When you create a table space, the default buffer pool (`IBMDEFAULTBP`) is used. You can modify the size of the default buffer pool.

The default of one buffer pool will usually suffice, but for a higher level of database tuning, you can create additional buffer pools and assign table spaces to individual buffer pools. For example, you could create four buffer pools, one each for catalogs, temporary space, indexes, and data. The extended buffer pool enables DB2 to take advantage of large amounts of memory.

You can create additional buffer pools in the Control Center by right-clicking the Buffer Pools folder and selecting Add from the pop-up menu. Day 9, "Creating Table Spaces," provides details on creating buffer pools.

To alter the size of a buffer pool beyond what's recommended by the Performance Configuration SmartGuide, select the Buffer Pool folder from the object list in the Control Center. Right-click `IBMDEFAULTBP`, and select Alter from the pop-up menu. Modify the

Size in 4KB pages field, and click OK. The change is effective the next time the database is started. table is associated with a buffer pool. When you create

Asynchronous Buffer Writer

To read table is associated with a buffer pool. When you create or modify data, pages of data are moved from disk to the buffer pool. If a page has been modified, it should be written back to the disk. The purpose of the buffer writers is to write out most changed pages to disk, so regular database agents always find empty slots and don't have to hold the transaction to write out pages. This means that agents won't wait for additional I/O and queries should run faster. The buffer writers run in parallel with the database agents.

Pages are written from the buffer pool to disk when the percentage of space occupied by changed pages in the buffer pool exceeds the value specified by the `CHNGPGS_THRESH` database configuration parameter. You can also configure the number of page cleaners used for the database. *Page cleaner agents* perform I/O that normally would be done by database agents. As a result, your applications can run faster because transactions aren't forced to wait while their database agents write pages to disk. To set the number of page cleaner agents, use the `NUM_IOCLEANERS` database configuration parameter.

Summary

In this lesson, you saw that DB2 provides registry values, environment variables, and configuration parameters to help you control your database environment.

There are configuration parameters that affect the entire database environment or only a specific database. Database manager configuration parameters affect the entire database system as a whole, whereas database configuration parameters affect only a single database.

The Performance Configuration SmartGuide helps you tune the performance of your system by suggesting which configuration parameters you should modify.

Input/output performance is usually the slowest function a database needs to complete. DB2 provides several features, including prefetching, parallel input/output, big block reads, check pending, and asynchronous buffer writers, to help you eliminate bottlenecks in your system.

What Comes Next?

On Day 21, "Diagnosing Problems," you learn the basics in troubleshooting your DB2 system.

Q&A

Q What are the two levels of registry values?

A The DB2 instance-level profile contains settings for the instance. The DB2 global-level profile contains settings that control the entire system. The values set in the instance-level profile override values in the global-level profile. These values are set by using the `db2set` command.

Q How do I update configuration parameters?

A To update configuration parameters on your client system, use the Client Configuration Assistant. To update configuration parameters on your server system, use the Control Center. Both tools group the different types of configuration parameters on separate tabbed pages and provide hints to help you determine the best value for the parameter. On the server, you can also use the Performance Configuration SmartGuide to help you decide whether parameters should be changed.

Q What must be done for configuration parameter changes to take effect?

A For database manager configuration parameters to take effect, all applications must first disconnect from the database. The changes will take place when the next application connects to the database. For database configuration parameters to take effect, you must first stop and then restart the instance.

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. How many buffer pools should you define for your system?

2. When do you need to change your configuration parameters?

3. Can database configuration parameters be set on the client?

4. If you expect to have no locks on the data, which isolation level should you use?

Exercise

Use the Client Configuration Assistant and Control Center to view the current settings of the various configuration parameters. Read the provided hint text to understand the purpose of the parameters. When you have a good understanding of the settings, run the Performance Configuration SmartGuide to see whether any changes are recommended.

Day 21: Diagnosing Problems

Overview

Today, you learn how you can find the cause of an error you might have encountered. If an error is encountered, DB2 provides an error message, such as `SQL30081`, that you can look up in the Messages Reference, or you can scan the tips and techniques page on the DB2 Web site. The Message Reference gives you guidance as to what caused the error message you encountered and possible solutions to the problem. The DB2 Web site contains tips to solve commonly reported problems.

This lesson also shows how to determine whether you have corrupt data inyour database and introduces the DB2DART tool to help you correct manycorruption problems.

To search the DB2 product library, you must use the Net.Question Search Server, which is installed when you install DB2. This tool requires that you set up TCP/IP and your browser in a specific manner. In this lesson, you see how you can ensure that this tool works, enabling you to quickly find information in the vast product library.

Using Diagnostic Tools

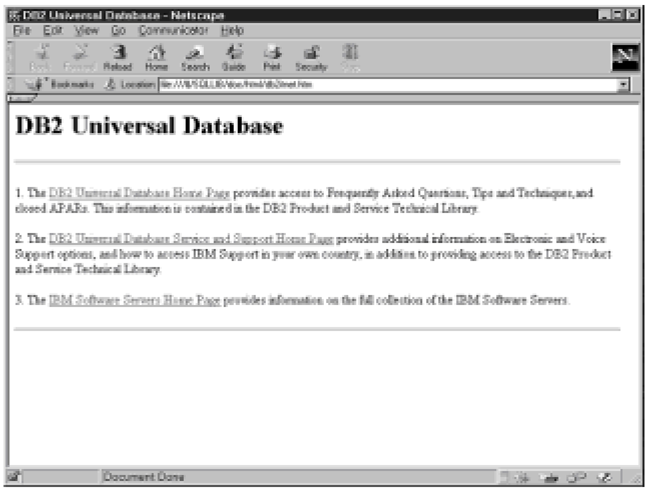
The following diagnostic tools gather and format information to assist in problem|determination and problem source identification. You can scan the Internet for known problems that might be similar to a problem you have encountered. You can also search through DB2 product information to help find a topic you need more information on and download the latest fix from the DB2 Web site. Alternatively, you can use DB2DART to identify and correct data corruption problems in your database, take a trace of theproblem you're having to help DB2 Technical Support personnel uncover the cause of a specific problem, or scan the online messages to obtain information on the meaning of a SQL error message you have encountered.

Accessing Internet Information

DB2 provides direct Internet access to the latest DB2 information, such as tips andtechniques, DB2 publications in HTML format, fix pack information, and frequently asked questions. To access this information, follow these steps:

1. Establish your Internet connection.

2. Choose Start | Programs | DB2 for Windows NT | Problem Determination Tools | Support Through Internet. Your Web browser will launch, showing a page that provides access to the DB2 information on the Web (see Figure 21.1).


DB2 Universal Database - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location
Document Done

DB2 Universal Database

1. The DB2 Universal Database Home Page provides access to Frequently Asked Questions, Tips and Techniques, and closed APARs. This information is contained in the DB2 Product and Service Technical Library.

2. The DB2 Universal Database Service and Support Home Page provides additional information on Electronic and Voice Support options, and how to access IBM Support in your own country, in addition to providing access to the DB2 Product and Service Technical Library.

3. The IBM Software Servers Home Page provides information on the full collection of the IBM Software Servers.

Figure 21.1: Accessing DB2 information on the Web.

If your Web browser is set up to access external Web pages, simply click a link to access this information.

Online Messages

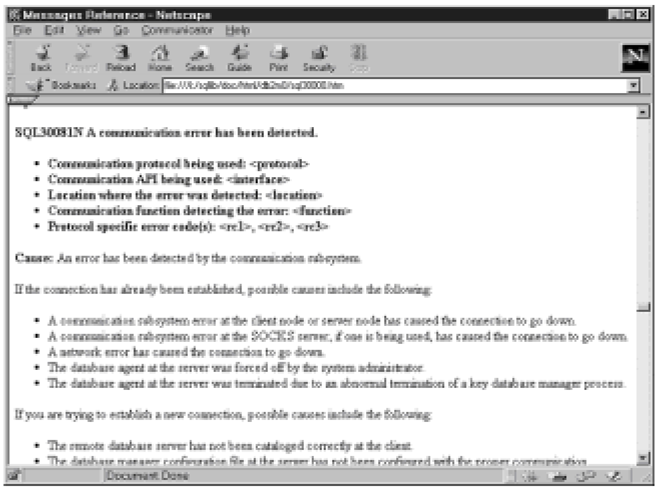
DB2 provides online help that contains error-message information about the cause of errors and the user action to take. See Appendix C, "Road Map to DB2 Information," for a listing of the different types of error messages you can receive while using DB2. To access the help, try the following:

- When using a DB2 tool, click the Help button to activate the help.

- For a description of SQL codes and other messages, type **? *message*** from the Command Center— *message* is the SQL code or message number.

- For a description and the syntax of DB2 commands, type **? *command***— *command* is the name of the command.

- For a description of any DB2 error message, open the Information Center and access message information directly on the Troubleshooting page of the notebook. On the Books page of the notebook, you can also access the Messages Reference that contains full descriptions of each DB2 error message. The error message text contains information on the possible causes for the problem and provides several suggestions on how the problems can be resolved. Figure 21.2 shows an example of the message text you see for `SQL30081`.


Messages Reference - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location:
Document Done

SQL30081N A communication error has been detected.

- **Communication protocol being used: <protocol>**
- **Communication API being used: <interface>**
- **Location where the error was detected: <location>**
- **Communication function detecting the error: <function>**
- **Protocol specific error code(s): <rc1>, <rc2>, <rc3>**

Cause: An error has been detected by the communication subsystem.

If the connection has already been established, possible causes include the following:

- A communication subsystem error at the client node or server node has caused the connection to go down.
- A communication subsystem error at the SOCKS server, if one is being used, has caused the connection to go down.
- A network error has caused the connection to go down.
- The database agent at the server was forced off by the system administrator.
- The database agent at the server was terminated due to an abnormal termination of a key database manager process.

If you are trying to establish a new connection, possible causes include the following:

- The remote database server has not been cataloged correctly at the client.
- The database manager configuration file at the server has not been configured with the proper communication

Figure 21.2: Message text for error message `SQL30081`.

Error-Logging Facilities

The DB2 Universal Database products use a number of error-logging facilities. DB2 automatically captures diagnostic information and stores it in the following logs:

- The messages log contains a list of error messages and their severity. To view these messages, open the Journal and go to the Messages page. See Day 11, "Using System Administration Tools," for information on using the Journal.

- The `db2.log` file stores general information and error messages resulting from install and uninstall activities. By default, the file is located in the *`x`*`:\db2log` directory; *`x`*`:` is the drive on which Windows NT is installed.

- The `db2diag.log` file is updated with information each time an error occurs. You use the `DIAGLEVEL` and `DIAGPATH` parameters to set the level of informationcaptured and the location of the file. If you want detailed messages to appear in `db2diag.log`, you must set the `DIAGLEVEL` parameter to level 4. To do so, enter **update dbm cfg using diaglevel 4** in the Command Center. Be sure to stop the database manager (`db2stop`) and restart (`db2start`) it before re-creating yourproblem. Starting and stopping DB2 is covered in Day 4, "Getting Started."

- Error conditions when using the Control Center are recorded in the Control Center Administration Engine Log (`db2cc.log`). This log is always active while the Control Center is active. The log file is kept in the home directory of the executable that invokes the Control Center. Typically, this is *`x`*`:\sqllib\bin;` *`x`*`:` is the drive where DB2 is installed. You can view and update the file by using an ASCII file editor.

  The log file records the error message type, a time stamp, a process identifier (PID), a thread identifier (TID), and a SQL error message. The PID and TID are used to identify the operating system that generated the log. Combined with the Control Center trace information, DB2 Service and Support personnel can determine which Control Center task caused the error. The information is of use to only DB2 Service and Support personnel.

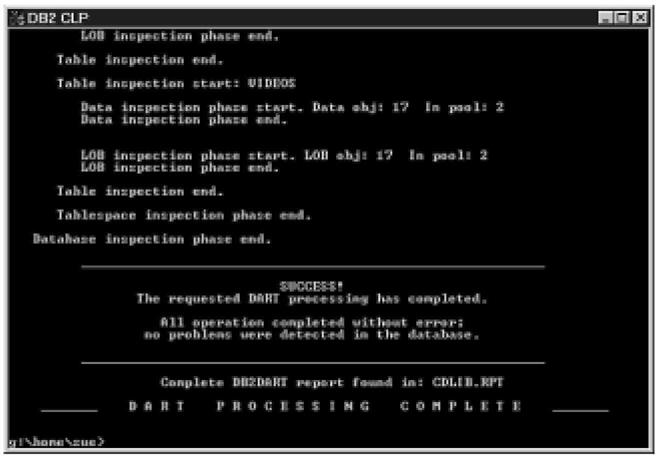
DB2 Database Repair Tool

Use the DB2 Database Repair Tool (DB2DART) to verify that the architectural integrity of a database is correct. This tool confirms the following:

- The control information is correct.
- There are no discrepancies in the format of the data.
- The data pages are the correct size and contain the correct column types.
- Indexes are valid.

You must run this tool on the DB2 server where the database resides. Also, ensure that there are no active connections to the database before running this tool.

To use DB2DART, enter **db2dart** and the name of the database you're checking. For example, to check the CDLIB database, enter **db2dart cdlib**. A report about the integrity of the data is created in a file named `cdlib.rpt`. Figure 21.3 shows the result of running DB2DART against the CDLIB database.



```
        LOB inspection phase end.

     Table inspection end.

     Table inspection start: VIDEOS

        Data inspection phase start. Data obj: 17  In pool: 2
        Data inspection phase end.

        LOB inspection phase start. LOB obj: 17  In pool: 2
        LOB inspection phase end.

     Table inspection end.

     Tablespace inspection phase end.

  Database inspection phase end.

        ____________________________________________________________

                               SUCCESS!
               The requested DART processing has completed.

                  All operation completed without error;
                no problems were detected in the database.

        ____________________________________________________________

                  Complete DB2DART report found in: CDLIB.RPT
    _______       D A R T    P R O C E S S I N G    C O M P L E T E    _______

g:\home\sue>
```

Figure 21.3: Results of running DB2DART for the CDLIB database.

Configuration Files

Many DB2 configuration files kept on your system can be useful if you have a problem to debug:

- The database manager configuration file that contains overall configurationinformation for your system. To produce a listing of this file, enter **get dbm cfg > dbm.cfg** from the Command Center. View the `dbm.cfg` file to see the configuration settings for your system.
- The database configuration file that contains configuration information for anindividual database. To produce a listing of this file, enter **get db cfg > db.cfg** from the Command Center. View the `db.cfg` file to see the configuration settings for one of your databases.

- The CLI configuration file that contains keywords to set the environment when using the CLI driver. The file, `db2cli.ini`, is located in the `sqllib` directory where you installed DB2.

- The `odbc.ini` file that contains the ODBC keywords to set the environment when using ODBC applications. The file is located in the same drive where Windows NT is installed. If you optimized the settings for the ODBC application you use, you may have the file `db2cli.opt` in the `sqllib` directory that contains all theseoptimized settings.

- The `db2cfg.txt` file in the `sqllib` directory. This file contains the protocolconfiguration information used for your default instance and the DB2 Administration Server.

- If you performed imports, exports, or loads, you'll have files that containoperational information. You choose a name and a location for these files when you set up the import, export, or load operation. Day 13, "Moving Data," discussed the import, export, and load operations.

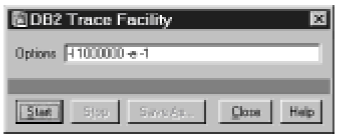
Performing DB2 Traces

DB2 Customer Service may request that you take a DB2 *trace*, which provides detailed information about DB2 to help the service representative determine the cause of aproblem you may be experiencing. When the trace information is collected, you send it to DB2 Customer Service for analysis.

To collect trace information, follow these steps:

1. Choose Start | Programs | DB2 for Windows NT | Problem Determination Tools | Trace.

2. Click Start to activate the tracing of DB2 activities (see Figure 21.4). Default trace parameters are already specified.

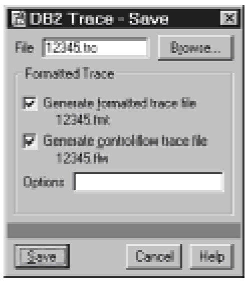


Options -l 1000000 -e -1

Figure 21.4: Activating a trace.

3. Re-create the problem you're diagnosing by running through the same steps that led to the original problem.

4. After you finish re-creating the problem, click the Save As button to open the DB2 Trace – Save dialog box (see Figure 21.5).



File 12345.trc

☑ Generate formatted trace file 12345.fmt

☑ Generate controlflow trace file 12345.flw

Options

Figure 21.5: DB2 Trace – Save dialog box.

5. Enter a filename for the trace file that matches your problem record number. Select Generate Formatted Trace File if you want the output formatted so that it's readable (the formatted trace file that's created will have the file extension `.FMT`). Select Generate Control-Flow Trace File if you want some of the trace information formatted into a nested format (the file that's created will have the file extension `.FLW`). Specify options for the trace output format in the Options text box. If you don't specify any options, the default is to trace everything. Click the Save button to save the trace output, and return to the DB2 Trace Facility dialog box.

6. Stop the tracing process by selecting Stop.

7. Send the file to IBM, as instructed by the DB2 Technical Support person you're working with.

   Note You send your trace information to this FTP server only if you have an open problem with IBM and are working with a DB2 Support Analyst.

Performing CLI, ODBC, and JDBC Traces

All CLI, ODBC, and JDBC programs access DB2 through the same driver. You can enable a trace by using the `TRACE`, `TRACEFILENAME`, `TRACEPATHNAME`, and `TRACEFLUSH` `db2cli.ini` keywords (which can be set by using the Client Configuration Assistant). These keywords apply to all applications that use the driver and can't be set for aspecific data source.

Having the trace turned on can slow applications, and the trace files can grow quite large. Therefore, the trace should be used for problem solving, database and application tuning, and gaining an understanding of what an application is doing.

Use the following steps to enable the trace:

1. Start the Client Configuration Assistant. (Day 16, "Administering Clients with the CCA," covers the Client Configuration Assistant in more detail.)

2. Select a database and click Properties. The trace applies to all connections through the CLI/ODBC/JDBC driver, so it doesn't matter which database you select.

3. If the database isn't already registered as an ODBC data source, register it now. In most cases, you should use the default As a System Data Source so that all users on the machine can access the database in ODBC.

4. Click the Settings button. The Connect to Database window opens. Enter a valid username and password, and click OK to connect to the database.

5. Click the Advanced button.

6. Select the Service tab.

7. Select Trace (Common), and click the Enable Trace button.

8. To use just one file to store all CLI/ODBC/JDBC trace information for each thread, select the File radio button and enter a specific filename. To use multiple files (one per thread), select the Path radio button and enter a subdirectory.

9. By default, the driver stores a number of trace entries in memory before writing them to disk. In some cases, the application can crash, and the buffered trace information

will be lost. Enable Flush After Each Entry to force the driver to write each entry to disk as soon as it's complete.

10. Close the dialog boxes.

The trace of the CLI/ODBC/JDBC driver is enabled the next time an application uses the driver. You don't need to restart the machine or DB2, but you must restart the applications you want to trace.

Remember to disable the CLI/ODBC/JDBC trace when you're finished with it.

Updating DB2 Products

DB2 provides product updates through *fix packs* (also known as *FixPaks*, *patches*, *updates*, or *PTFs*). DB2 provides an Internet and FTP site that you can use to download the latest fix. To access these sites:

- Go to the DB2 Services Web site by choosing Start | Programs | DB2 for Windows NT | Problem Determination Tools | Support Through Internet. Your browser will open to the DB2 Service page, from which you can read about the available fixes and download the one you want.

- Use FTP to go to the anonymous server `ftp.software.ibm.com`. Then go to the directory `/ps/products/db2/fixes`.

After you download a fix pack, use its readme file (`readme.txt`) for prerequisite and installation instructions.

Several fixes are currently available for the 5.0 version of the product. The productprovided on the CD-ROM is at the version 5 level and includes the WRO9014 fixes.

Installation Errors

The following sections list the problems that might occur during installation and provide information on how to deal with these errors.

Handling Insufficient Space

If the space required to install selected components exceeds the space found in the path you specify for installing the components, the setup program issues a warning about the insufficient space. You can continue with the installation, but it will stop when there's no more space. At this time, you'll have to manually stop the setup program if you can't free up space.

Path Length

The `PATH` environment variables can't exceed 512 bytes. Therefore, before installing DB2, do the following:

1. Remove the values from the path.

2. Install the product.

3. Determine which values can be added or removed to make the path an acceptable length.

DB2 requires the following in the path:

- `BIN` in all cases
- `HELP` for graphical tools
- `SAMPLES-REPL` for replication
- `FUNCTION` for fenced DARIs (stored procedures, described on Day 19, "Design Considerations") and UDFs (user-defined functions, described on Day 11).

Solving Net.Question Problems

The entire DB2 product library in HTML format is installed on your system. To view information in the DB2 product library, you must have a Web browser such as Netscape installed on your system. To search for information in the DB2 product library, DB2 uses a component known as Net.Question. Net.Question Search Server is installed on your system when DB2 is installed. For the Net.Question component to work correctly, you must have the following prerequisites on your system:

- On Windows NT 4.0, for best results you should have installed Service Pack 3. For information, see `http://www.microsoft.com/ntserver/info/servicepack3.htm`.
- You need a browser, such as Netscape 3.0. Be sure to turn off proxy handling for localhost in the browser you use.
- You must have TCP/IP version 3 or higher installed and configured on your machine for the DB2 Universal Database Search Server to function properly.

Stopping Any Previously Installed Versionof the Search Server

If the Search Server was previously installed by another product (for example, VisualAge for Java), the Search Server must be stopped. To stop the search server, choose Start | Programs | DB2 for Windows NT | Stop HTML Search Server.

Locating the Search Server Directory

The Search Server is stored in its own directory because it can be used by other products. For example, if you installed DB2 Universal Database along with the Search Server on drive G and later installed VisualAge for Java on drive H, only one Search Server is installed—the one that came first.

Some instructions in this document require you to specify the location of this directory. To locate this directory, use this command:

```
echo %IMNINSTSRV%
```

Changing to a Different Port Numberfor the Search Server

The search server is assigned to port 49213, a number beyond the public ports assigned for TCP/IP. If you have another product that uses this port, you can change the search server port by following these steps:

1. Edit the `httpd.cnf` file in the Search Server directory to change the port number to

one you know is available, preferably above 49000.

2. Determine the location of `db2path` by issuing the command `db2set db2path`.

3. Go to the DB2 Universal Database directory. In the `db2path/doc/html` directory, use a text editor to edit the `db2srch.htm` file. Change the following line to the port you selected in the previous step:

   ```
   http://localhost:49213/cgi-bin/db2srsen.exe" method="POST"
   ```

   in which `49213` is the new port for localhost.

4. Stop and start the search server. To stop the search server, click Stop HTML Search Server in the DB2 menu item of the Start menu. To start the search server, choose Start | Programs | DB2 for Windows NT | Start HTML Search Server.

Diagnosing Search Server Install Errors

During a DB2 Universal Database install, if you encounter a failed search server installation or initialization, the DB2 Universal Database install will proceed to completion.

Look in the `temp\imnnq\install` directory (where `temp` is your system's temp directory) to find the `imnnq.err` file. If it doesn't exist, reboot and try installing the product again.

If `imnnq.err` does exist, here are the possible contents:

- `1` indicates that the current path is too long, and adding the search server to the path will cause the entire path to be erased. The current path limit on Windows NT 4.0 is 512. If your path is too long, it's recommended that you remove any values you no longer need in the path and run the product install again to install the Search Server properly.

- `2` means miscellaneous error; contact IBM service.

- `3` is the out-of-disk-space error. Ensure that you have at least 4.5MB of disk space for the Search Server. If `imnnq.err` contains a message saying `XXX.EXE DOES NOT EXIST`, the Search Server executables couldn't be found. Try rerunning the Search Server install/initialization program again.

Starting and Stopping the Search Server

After you install the Search Server, it will start automatically after you reboot. If the Search Server starts without any errors but you encounter the following error message in your browser, click Stop HTML Search Server to stop the search server and then Start HTML Search Server to restart it.

```
A network error occurred: unable to connect to server.
The server may be down or unreachable.
Try connecting again later.
```

After you finish searching, stop the Search Server to reclaim the memory it used by clicking Stop HTML Search Server.

Searching with Proxies Enabled in Netscapeor Internet Explorer

If you use Netscape or Internet Explorer with proxies enabled manually, you can speed a search significantly by modifying your proxy information.

If you use Netscape 3 as a browser, follow these steps:

1. Select Options | Network Preferences from Netscape's menu.
2. On the Proxies page, select the Manual Proxy Configuration check box, and click View next to the Manual Proxy Configuration selection.
3. In the No Proxies For text box, type **localhost:49213**. If you have other entries here, separate them with commas.
4. Click OK to close the Manual Proxy Configuration dialog box.
5. Click OK to exit the Preferences dialog box.

If you're using Netscape 4 (Communicator) as a browser, follow these steps:

1. Choose Edit | Preferences from Netscape's menu.
2. Double-click Advanced in the Category tree.
3. Click Proxies in the Advanced subtree to open the Proxies window.
4. Select Manual Proxy Configuration, and click View next to the Manual Proxy Configuration selection.
5. In the Do Not Use Proxy Servers for Domains Beginning With text box, type **localhost:49213**. If you have other entries here, separate them with commas.
6. Click OK to close the Manual Proxy Configuration window.
7. Click OK to exit the Preferences Window.

If you're using Internet Explorer 3 as a browser, follow these steps:

1. Choose View | Options from Internet Explorer's menu.
2. On the Connection page, select the Connect Through a Proxy Server check box, and then click the Settings button.
3. In the Do Not Use Proxy Servers for Domains Beginning With text box, type **localhost:49213**. If you have other entries here, separate them with commas.
4. Select the Do Not Use Proxy Server for Local (Intranet) Addresses check box.
5. Click OK to exit the Proxy Settings dialog box.
6. Click OK to exit the Options dialog box.

If you're using Internet Explorer 4 as a browser, follow these steps:

1. Choose View | Internet Options from Internet Explorer's menu.

2. On the Connection page, select the Access the Internet Using a Proxy Server check box, and then click the Advanced button.

3. In the Do Not Use Proxy Server for Addresses Beginning With text box, type **localhost:49213**. If you have other entries here, separate them with commas.

4. Click OK to exit the Proxy Settings dialog box.

5. Select the Bybass Proxy Server for Local (Intranet) Addresses check box.

6. Click OK to exit the Internet Options dialog box.

If Search Doesn't Work

If the product installation worked but searching results in error 500, follow these steps:

1. Check that the Search Server was properly installed. The environment variables `IMNINST` and `IMNINSTSRV` should be set, and `IMNINSTSRV` should point to the Search Server directory (see "Locating the Search Server Directory" earlier in this lesson).

2. The Search Server directory should contain these files:

   - A `db2srsen` executable file
   - `db2head.htm` and `db2foot.htm`

3. Ensure that the Search Server is registered with product documentation. Invoke the `nqmap -a` command to list all the documentation registered with the Search Server. The documentation for DB2 Universal Database is called `db2admen`, `db2apden`, or `db2conen` (`en` is the two-character identifier for the English documents). One or more of these names should appear in the list of names that `nqmap` returns.

Tip With the list of indexes the `nqmap` command provides, you can obtainadditional details on each index with the following command:

```
imnixsta index_name
```

This command provides details such as the status of the index and thenumber of documents in the index.

If any preceding conditions aren't true, you can rerun the DB2 Universal Database product installation program. If the files `db2srch.exe`, `db2head.htm`, and `db2foot.htm` are missing, you can copy them over from the directory `db2path\misc` into Search Server's directory (for example, `e:\imnnq_nt`). The product installation program will rerun the search server's installation and initialization.

`File not found` Errors

The Search Server searches predefined indexes. All these indexes are installed with the Search Server, regardless of the DB2 Universal Database products you install. Therefore, when you conduct a search, some results might return a `File not found` error because they pertain to a product not installed on your system or a document that you chose not to install during installation.

Summary

In this lesson, you saw how DB2 provides several problem-determination tools, including an Internet link to DB2 information, product fixes, certification news, and technical notes.

Online messages can help you debug your own problems. Several logs are kept to track various kinds of information during the operation of DB2.

Tools are also available to verify the integrity of your databases, and trace tools help you gather additional information in case of a problem you can't diagnose on your own.

Many configuration files are available to help you determine how your system wasconfigured.

What Comes Next?

Appendix A contains the answers to all quiz questions covered in the book. (No peeking allowed for today's quiz!)

Q&A

Q What message logs are kept by DB2?

A DB2 keeps `db2.log` for installation warnings and messages, `db2diag.log` forgeneral error messages, and `db2cc.log` for errors that occur when using the Control Center.

Q If I suspect that I have corrupted data in my database, what tool should I use to check and possibly fix the problem?

A Use the DB2 Database Repair Tool (DB2DART) to verify the architecturalintegrity of your database. This tool can help discover and correct possible data corruption problems.

Q How can I look at the current configuration of my system?

A There are three main configuration files: the database configuration file, thedatabase manager configuration file, and the administration configuration file.The easiest way to view these configuration files is through DB2 commands:

```
get dbm cfg > dbm.cfg
get db cfg for SAMPLE > sample.cfg
get admin cfg > admin.cfg
```

Workshop

The purpose of the Workshop is to enable you to test your knowledge of the material covered in the lesson. See whether you can successfully answer the questions in the quiz and complete the exercises before you continue with the next lesson.

Quiz

1. How do you acquire fixes for the DB2 products?

2. When would you do a trace on your system?

3. What type of information is available on DB2 Web sites?

4. What parameter must be set to obtain more detailed information in the `db2diag.log`?

Exercise

Browse through the various logs to see whether you have any errors. Look through the online Messages Reference for a description of the problem and to see whether there are any suggested actions to overcome the problem. Look at the configuration files set up for your system and database, and see whether you understand how these values are set. Look through the online information to develop an understanding of each parameter you can set.

Week 3

In Review

Congratulations—you've completed the final week of this book. This week covered many important topics that you need to use DB2 effectively. You learned how to set up data security. You learned various ways to administer clients using the Client Configuration Assistant. You learned how to work with DB2 instances and how to use several performance aids such as the event and snapshot monitors. Tuning and design considerations were also taught in the past week. To finish the week off, you learned how to find the error information that is captured by DB2 to help you diagnose problems.

Appendix A: Answers to Quiz Questions

Day 1, "What Can DB2 Do For You?"

1. What's a local application? What's a remote application?

   When you're running an application on the server where the database is located, it's referred to as a *local application*. The application could be using the Control Center, the command line, or an ODBC application. If you're running an application on the server but accessing a database located on a different machine, you're running a *remote application*. If you are on a client machine, you can run only remote applications because you can't have databases on a client machine.

2. What are the two ways to use Java Database Connectivity to access DB2 data?

   You can create JDBC applications or applets to access DB2 data. If you're using an applet, all you need is a Java-enabled browser and the applet on your system. If you're using an application, you must have the DB2 Client Application Enabler code installed on your system, along with the application. A Web browser and Web server aren't required when using JDBC applications.

3. What's the name of the DB2 product that provides a parallel, multinode database environment?

   The product that provides a parallel, multi-node environment is known as the DB2 Universal Database Enterprise – Extended Edition. Previous versions of this product were called DB2 Parallel Edition. The Enterprise – Extended Edition is available for AIX, Solaris, and Windows NT platforms.

4. Name the interfaces that you can use when creating applications with the DB2 Software Developer's Edition.

When using the DB2 Software Developer's Edition, you can create applications that contain embedded SQL in a program written in languages such as C, C++, COBOL, Call Level Interface (CLI), ODBC, JDBC, and DB2 APIs. This gives you a great deal of flexibility to create the application with the tools that best suit your environment.

Day 2, "Exploring the Capabilitiesof DB2 Universal Server"

1. What's the advantage of using a view?

   By using views, you can set up different presentations of the same data. Each view is derived from the actual table data, but each user will see a subset of the data. The main benefit to using views is that they allow you to control the access your users have to restricted data.

2. Give some reasons why you would want to define a primary key for a table.

   Defining a primary key for a table enables you to guarantee that each row in the table is unique. Defining a primary key can also improve performance because table access is quicker when each row is uniquely identified. If you're using an ODBC application to update your data, you need to have a primary key defined for the table. An index created automatically for the primary key helps DB2 find an efficient path to the data.

3. What's an instance? What's the advantage of using an instance?

   An instance is a logical database manager environment. Having an instance enables you to catalog databases and set specific configuration parameters to the environment. For example, you can set up a test environment and a production environment by creating two instances. You can change the catalog information and the configuration parameters in the test instance without affecting the data in the production instance. Using instances can also help you protect access to sensitive information because each instance has a separate assignment of authorized users.

4. What's the difference between static and dynamic SQL?

   When you write a SQL statement for an application program and know the entire statement, including the SQL statement type (`UPDATE`, `INSERT`, and so on) and the table and column names that the statement acts on, you're using static SQL. Static SQL is a SQL statement that's fully written in an application program before the program is compiled. Dynamic SQL are statements that the application builds and executes at runtime. Generally dynamic SQL is used in an interactive program in which the user is prompted for key parts of the SQL statement.

Day 3, "Installing and ConfiguringDB2 Server"

1. What's the one difference concerning the default instance when using the Custom install type rather than the Typical install type?

   If you do a Typical install, you're prompted only for a few key items needed during the installation; you can't select which program components are installed, nor can you customize the protocol information. If you do a Custom install, you must answer many prompts to determine exactly which components to install on your system.

2. Under what circumstances do you need to manually invoke the installation program?

   You need to manually invoke the installation program when you want to install a language other than U.S. English on your system. Also, if the installation program doesn't start automatically when you insert the CD-ROM, you need to manually start it.

3. How do you know which subcomponents are installed on your system when using the Custom install type?

   Click the Details button in the window showing the components to see the subcomponents.

4. When would you want the Control Center to start automatically each time you boot your system?

   If you're planning to use the Control Center often, you should have it started each time you boot your system. Typically, if you are a database administrator, you'll spend most of your time working with databases through the Control Center and should choose to start it automatically.

Day 4, "Getting Started"

1. Why would you want to create and use a schema?

   Using a schema helps you group database objects under a single name. If you don't create a schema, any database objects you create will, by default, be stored under a schema name that's the same as your username. If your database will be shared with several people in your organization, you are best off creating a schema with a generic name such as the department name and creating all related objects within this schema. If everyone uses his default schema, you may end up with many related objects having different schema names.

2. What authority do you need to create databases?

   You must have `SYSADM` authority to create a database. `SYSADM` authority—the most powerful authority used within DB2—is given to all users in the Administrators group.

3. What's the name of the tool that you use to view the DB2 product library?

   The DB2 Information Center is the tool provided with DB2 products in Windows NT, Windows 95, and OS/2 for you to view DB2 product information. The DB2 Information Center also requires that a Web browser be installed on your system because the product information is provided in HTML format. The Netscape browser is provided on the CD-ROM shipped with this book.

4. What are the two ways you can start administration tools such as the Script Center?

   You can start administration tools such as the Script Center through the Administration Tools folder or through toolbar icons in the Control Center.

Day 5, "Configuring Server Communications with the Control Center"

1. What properties can you change if you're modifying the TCP/IP protocol?

   You can change the service name and the port number associated with the TCP/IP protocol. You can't change the hostname because it's a systemwide parameter.

2. Do you configure communications for an entire instance or for each individual database?

Communications are defined on an instance basis. Each database within this instance takes on these communication properties.

Day 6, "Installing DB2 Clients"

1. What three configuration methods are available with the Client Configuration Assistant?

   You can use the Client Configuration Assistant to search your network for databases, to use a profile generated from a server system, or to manually enter database and protocol information.

2. When should the client's `DISCOVER` parameter be set to `SEARCH`?

   It's set to `SEARCH` when you want the client to search the network for available databases.

3. Why would you want to register a database for ODBC?

   You would register your database for ODBC when you want to use ODBC applications such as Lotus Approach or Microsoft Access.

4. If you can't connect to the server, what's the most likely problem?

   Before you can make a connection between a server and a client, the database manager on the server must be started. Start the database manager by issuing the `db2start` command at the server.

Day 7, "Designing a CDLIB Database"

1. What different types of relationships can you represent in a table?

   You can have one-to-one, one-to-many, and many-to-many relationships represented in a table.

2. Where are foreign keys defined?

   Foreign keys are defined on the dependent table and rely on the data in the parent table. A foreign key references a primary key or a unique key in the same table or another table. A foreign key assignment indicates that referential integrity is to be maintained according to the specified referential constraints.

3. Why should you normalize your tables?

   You should normalize your table to avoid redundancies and inconsistencies in your data.

4. Is it a good idea to use a phone number as a primary key? Why or why not?

   Using a phone number as a primary key isn't a great idea, mainly because a phone number might not be unique (several people might share a phone, for example). Also, phone numbers change when people move or when the phone company needs to change the area code.

Day 8, "Creating Databases"

1. Can you define a primary key column to be nullable? Why or why not?

   Primary keys must not be nullable. There must be a value for each primary key column, and each row must be unique. If a column contains a null value, it's impossible to reference the row.

2. What are some differences between the data types `VARCHAR` and `CHARACTER`?

   The `VARCHAR` and `CHARACTER` data types hold the same types of data, but `VARCHAR` is more efficient. Suppose that you create two 10-character columns, defining one column as `VARCHAR` and the other as `CHARACTER`. Ten bytes are reserved for the `CHARACTER` column, regardless of the data inserted in the column. For the `VARCHAR` column, only the amount of space required to store the data is used for the column. `CHARACTER` columns are fixed-length and can contain up to 254 bytes per column. `VARCHAR` columns are variable-length and can contain up to 4,000 bytes per column.

3. What are the rules for naming a database?

   The name you choose for the database must be different from that of any other database stored on the same drive. The name must contain a maximum of eight characters—A–Z, 0–9, @, #, or $—and can't begin with a number. If your database will be used outside North America, you should avoid using special characters in the name to eliminate potential code-page problems.

4. What are the differences between the Create Table SmartGuide and the Create Table notebook?

   The Create Table SmartGuide provides more basic information on the pages than the Create Table notebook, but it doesn't allow you to set up referential constraints while creating the table. If you use the Create Table SmartGuide, you must usethe Alter Table option to add referential constraints to the table after the table is created.

Day 9, "Creating Table Spaces"

1. What's the difference between the Create Table Space SmartGuide and the Create Table Space notebook?

   The Create Table Space SmartGuide provides more instructions on the panels to help you choose the type of table space to define for the data that will be stored in it. The terminology on the pages is also easier to understand. When you understand the terminology, the Table Space notebook becomes easy to use. If you want to set up performance parameters for the table space while you're creating the table space, you must use the Create Table Space notebook.

2. How many buffer pools should you define for each database?

   You can define as many buffer pools as you like, but usually one buffer pool is sufficient.

3. At what point in the process do you specify the tables to be stored in a table space?

   When you create a table, you have the option of defining a table space for the table to reside in. If you don't specify a table space, the default table spaces are used.

4. How can you view the defined table spaces?

In the Control Center, click the Table Spaces folder to see the list of defined table spaces in the contents pane.

Day 10, "Accessing the Data"

1. What's a benefit of using Net.Data to create Web applications?

   With Net.Data, you can create interactive Web pages that use HTML and SQL to access live data from local or remote databases. Net.Data simplifies the writing of Web applications through the use of macros.

2. What's the main difference between a JDBC applet and a JDBC application?

   To run a JDBC applet, all you need is a Java-enabled browser. To run a JDBC application, you need to have DB2 Client Application Enabler installed on your system, but you don't need a browser.

3. If you receive a message that the table you're opening in an ODBC application is read-only, what's the most likely problem?

   Most likely, you need to define a unique index on the table before you can open it in read/write mode.

4. When entering commands in the Command Center, do you need to end each line with a semicolon (`;`)?

   If you're entering a single command, you don't need to end the command with a semicolon. If you have a series of commands, you must end each command with a semicolon, except the last command in the series.

Day 11, "Using System Administration Tools"

1. How do you create an index? Why do you want to create an index?

   An index is automatically created when you define a unique key, a primary key, or a foreign key. You can also define an index when the table is defined or at any time afterward. Creating an index produces pointers to rows in the table and allows more efficient access by creating a direct path to the data.

2. If you don't see hover help while you're using the DB2 tools, how can you turn on this feature?

   Use the Tools Settings notebook to turn on hover help. (Hover help is turned on by default.) To open the Tools Settings notebook, click the Tools Settings icon on the toolbar in the Control Center.

3. If you create several distinct types based on the same existing type, can you directly compare these values?

   To compare distinct types, you can't directly compare these values. For example, you can't directly compare two currencies. You must first convert one currency to another and then compare. Usually, a user-defined function is created to compare distinct types.

4. Can you modify scripts that are created as a result of using the Backup Database SmartGuide?

   No, you can't. You can, however, run the script or schedule the script. If you need to make changes beyond the schedule, you must create a new script.

Day 12, "Backing Up and Recovering Data"

1. How can you ensure that you have exclusive access to a database if you want to do a full offline backup of your data?

   If you try to perform a full offline backup of your database while users are still attached to it, you'll receive an error. To gain exclusive access to the database, use Force Applications. You'll maintain exclusive access to the data during the entire time it takes to perform the offline backup.

2. In what cases are you required to use the Backup Database notebook instead of the Backup Database SmartGuide?

   If you want to back up only a portion of your data by backing up individual table spaces, you must use the Backup Database notebook. The first time you back up your database, you should use the Backup Database SmartGuide.

3. How does using redirected restore give you additional flexibility when using SMS table spaces?

   If you're using SMS table spaces, you must define the size of the containers when the table space is defined, and you can't change the size of the containers. You can redefine the layout of the table space containers by using redirected restore.

4. How can you use roll-forward recovery to undo an unexpected error?

   If you know the precise time the error occurred, you can restore your database and then roll forward your logs until the moment before the error occurred. This means that your data will be returned to the state it was in just before the error.

Day 13, "Moving Data"

1. To create a new table by importing IXF data, which import mode should you use?

   You must use the `CREATE` import mode.

2. If you want to create a new table, what format should you use? Would you use the Load notebook or the Import notebook to create a new table?

   Integrated exchange format (IXF) is used to create a new table; the DEL, ASC, and WSF formats can be used only to import data into an existing table. IXF format is the format used by database managers. Load doesn't support creating a new table; the table must exist before you can load data into it. To create a new table, use the Import utility.

3. How do you export only a subset of the data in a table?

   Click the Columns tab in the Export notebook to specify only a subset of the columns to export. Exporting a subset of columns is supported only when using WSF or IXF formats.

4. What must take place before you can save a copy of the changes when using the Load notebook?

   Forward recovery must be turned on to enable you to save a copy of the changes when using the Load utility. On the Copy Options page of the Load notebook, specify that a copy of the changes be saved.

Day 14, "Replicating Data"

1. How can you set some defaults when using the quick method of defining replication sources?

   The defaults for the quick method of defining replication sources are set in the Tools Settings notebook.

2. Where must the Capture program be run? Where must the Apply program be run?

   Run the Capture program from a command line on the server where the source database is located. Run the Apply program from a command line on the server where the target database is located.

3. What is the capture before-image option? When would you use this option?

   The capture before-image option in replication enables you to save the column in the source table before it's updated. This option is useful for recovery and auditing.

Day 15, "Ensuring Data Security"

1. If you want to create databases and assign authorities to other users, what authority should you have?

   `SYSADM` authority is required to create databases and to assign authorities to other users. Because this is the highest level of authority, you should be cautious when assigning `SYSADM` authority to users.

2. Describe the importance of implicit versus explicit privileges.

   You can grant implicit privileges to a user to run an application that performs actions on database objects, and the user will obtain the privileges necessary to perform the actions while running the application. This means you can grant your users explicit privileges to execute applications, and they will be implicitly granted the privileges they need for manipulating the database objects.

3. What authorities are granted to all users when the database is created?

   `CONNECT`, `BINDADD`, `CREATETAB`, `IMPLICIT_SCHEMA`, and `SELECT` privileges are granted to `PUBLIC` when a database is created.

4. What is the default for the Trusted Clients option? What does this mean?

   The default for the trusted clients option is to trust all clients. This means that you'll allow all clients to access the database, even if reliable local security systems aren't built into the clients.

Day 16, "Administering Clients with the CCA"

1. What types of information can you export to a profile from a client workstation?

   You can export database connection information, CLI and ODBC settings, database manager configuration settings, CLI and ODBC common parameters, and APPC configuration information.

2. If you don't want your clients to be able to search the network for databases, how can you turn off this option?

   To turn off the search capabilities, you must set the `DISCOVER` parameter to`DISABLE` on the client and the server.

3. What protocols are supported by the `SEARCH` discovery method?

   The Administration Server must be set up to use NetBIOS or TCP/IP for the `SEARCH` discovery method to work.

4. What profile registry values can be used to tune the search discovery on the client?

   You can set the `DB2DISCOVERYTIME` or `DB2NBDICOVERRCVBUFS` parameters to tune the search discovery.

Day 17, "Working with DB2 Instances"

1. What command do you use when you want to view the configuration parameters set for the DB2 Administration Server?

   Enter the command **get admin cfg**.

2. What drive and subdirectory structure is created when you create an instance?

   When you create an instance, a subdirectory with the same name as the instance is created under the `SQLLIB` directory in the path where DB2 is installed. (If you set the `DB2INSTPROF` environment variable, the instance subdirectory is created in the directory specified by this variable.)

3. What's the command for changing the ownership of the DB2 Administration Server after it's created?

   To change the ownership of the DB2 Administration Server, use the command `db2admin setid`, and provide a username and password.

4. How do you change the instance that's started as a default?

   Set the `db2instdef` Registry value to define the instance to start by default.

Day 18, "Performance Aids"

1. What do you need to create an access plan graph?

   Before you can create an access plan graph, you need a working SQL statement.

2. What's an operator? How is it shown on an access plan graph?

   An operator is an action that must be performed on the output from a table or index. Examples include `DELETE`, `FETCH`, `FILTER`, `GRPBY`, and `INSERT`. The rectangular boxes in the graph are operators.

3. When is it recommended that you collect statistics and reorganize a table?

You should collect statistics when the data in your tables changes significantly. You should also collect statistics after a table is reorganized. Tables should be reorganized to remove all the unused space and to write the table and indexes in contiguous pages.

4. What should you do immediately following a Run Statistics or a Reorganize Table operation?

   You should rebind application programs that use static SQL.

Day 19, "Design Considerations"

1. When do you use table locking versus row locking?

   Use table locking if you want to lock the table while you make extensive changes to it and want to ensure that no other users have access to it while you're making the changes. Use row locking when it's important that many users have access to data in the table at the same time.

2. What's a deadlock? How can you break out of a deadlock situation?

   A deadlock occurs when two or more transactions are each waiting for data locked by the other. You can break out of a deadlock situation by using the deadlock detector, which will run periodically to determine whether there's a deadlock and will stop one of the transactions in the deadlock. You can also use lock timeouts to have locks released in a specified time limit.

3. How can you ensure that your catalog statistics are up-to-date? Why is this important?

   Catalog statistics are updated when you perform the Run Statistics operation. This is important because the optimizer uses the statistics to find the most efficient path to the data in your tables. If you've made significant changes to your data without updating the statistics, the optimizer won't be able to find the most efficient path to your data, and performance may suffer.

4. When should you use the highest level of optimization?

   The highest level of optimization should be used with very complex SQL statements.

Day 20, "Tuning DB2 Universal Database Performance"

1. How many buffer pools should you define for your system?

   Usually, one buffer pool is sufficient, but you can create additional buffer pools for a higher level of database tuning.

2. When do you need to change your configuration parameters?

   The default values defined for the configuration parameters are usually sufficient to suit your needs, but to achieve maximum performance for your system, you should analyze and tune these parameters.

3. Can database configuration parameters be set on the client?

   Database configuration parameters exist only on the server where the database is located. On the client, you can set database manager configuration parameters.

4. If you expect to have no locks on the data, which isolation level should you use?

   You should use the uncommitted read isolation level when you expect to have no locks on your data.

Day 21, "Diagnosing Problems"

1. How do you find fixes for the DB2 products?

   You can see the fixes available for the various DB2 products on the`ftp.software.ibm.com` FTP server. Each fix pack comes with a readme file describing the problems the fix pack fixes. After you read the information pro-vided, you can download the fix pack to your system and install it on your system.

2. When would you do a trace on your system?

   You would do a trace on your system when you're having a problem and are requested by DB2 Support personnel to gather trace information to help debug the problem.

3. What information is available on the DB2 Web site?

   The DB2 Web site contains the entire DB2 library in HTML format, many tips and techniques to help you use the product, frequently asked questions, fix pack information, and information on how to become a certified user of DB2.

4. What parameter must be set to find more detailed information in the `db2diag.log`?

   The `DIAGLEVEL` parameter in the database manager configuration file should be set to the level 4 to capture additional diagnostic information in the `db2diag.log` file.

Appendix B: Uninstalling DB2 Products

Overview

Before you can uninstall a DB2 product, you must stop all programs that have DB2 files open. Some programs, such as the following, can have DB2 files open without this being obvious:

- Communications Manager for Windows NT
- NetFinity

You must also back up the databases you want to keep before uninstalling DB2. Although databases aren't dropped or deleted when you uninstall DB2, you should back up the databases you want to keep to make sure that nothinghappens to the data.

Also, drop any databases you no longer want. For example, to drop theSAMPLE database, use the following command in the Command Center:

Input `drop database sample;`

Then you need to stop all services by doing the following:

1. Click Start | Settings | Control Panel, and double-click the Services icon.

2. In the Services dialog box, click a DB2 service that's now started, and click Stop. Repeat this step for each started DB2 service. For example, if you have DB2-DB2, DB2-DB2DAS00, DB2 Governor, and DB2 Security Server started on your system, stop each service.

3. Close the Services dialog box.

Running the Uninstall Program

Now you are ready to uninstall the DB2 products. Follow these steps:

1. Click Start | Programs | DB2 for Windows NT | Uninstall to start the DB2 uninstall program.

2. Click Yes to begin removing the DB2 product from your system.

3. A progress indicator is visible until all the DB2 files are gone. Click OK when the procedure is finished.

   Note The uninstall program doesn't remove the file `\db2log\db2.log`, which is located in the same directory where Windows NT is installed. You can remove this file and directory only after DB2 is uninstalled; don't remove this file while DB2 is installed on your system.

If you can't run the uninstall program (for example, if it was deleted or the install was stopped and left in an inconsistent state), do the following:

1. Remove any partially installed files and the install directory—for example, `c:\sqllib`.

2. With a Registry editor such as `regedt32`, clean up the Registry by deleting the Registry control information for the DB2 node:

   ```
   HKEY_LOCAL_MACHINE
   SOFTWARE
   IBM
   DB2
   ```

   Also, remove any installed DB2 services:

   ```
   HKEY_LOCAL_MACHINE
   SYSTEM
   CurrentSetservices
   DB2NETSERVER
   IBM
   DB2

   HKEY_LOCAL_MACHINE
   SYSTEM
   Current
   Setservices
   DB2
   ```

Uninstalling the HTML Search Server

Before uninstalling the DB2 Universal Database product, the Search Server must be stopped. To stop the Search Server, choose Start | Programs | DB2 for Windows NT | Stop HTML Search Server. To uninstall the Search Server with the DB2 Universal Database product, choose Start | Programs | DB2 for Windows NT | Uninstall.

If the Search Server isn't uninstalled, the cause might be that DB2 Universal Database or another product is using the Search Server.

To determine which products are still registered with the Search Server, issue the following command:

Input **`nqmap -a`**

If the list contains indexes that don't belong to DB2 Universal Database (that is, their names begin with something other than `db2`), you can't remove the Search Server. If the list contains any of the DB2 index filenames (`db2admen`, `db2apden`, `db2conen`), DB2 Universal Database couldn't unregister the indexes, therefore causing the Search Server uninstall to fail. This occurs if DB2 Universal Database was incorrectly uninstalled (for example, the `SQLLIB` folder was deleted). In this case, you have to manually unregister the indexes and remove the Search Server directory by following these steps:

1. Issue the **`imnss start server`** command to ensure that the search server is running.
2. Issue the **`nqmap -d index_name`** command for each index file.
3. Issue **`nqdelet index_name`** for each index file.
4. Stop the search server with the command **`imnss stop server`**.
5. Issue the command **`nqmap -a`**, and verify that no DB2 indexes (`db2`*`xxxxx`*) are remaining. If there are, contact IBM Technical Support.
6. Issue the **`nqcounti <Search Server directory`** command to verify that no other indexes are active. (See "Locating the Search Server Directory" to determine where the Search Server is installed.) If this command returns data that indicates one or more indexes are still active, the Search Server can't be removed because other products are still registered. Don't perform the remaining steps. If the Search Server returns `0 indices active`, run the command `uninstnq.exe`.

If the Search Server still doesn't uninstall, you can try removing the product manually:

- Remove the Registry entry `\\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\NetQuestion` and all its subtrees.
- Remove the Search Server's directory and all its subtrees (for example, `D:\IMNNQ_NT`).
- Remove the environment variables `IMNINST` and `IMNINSTSRV`, and remove the Search Server path from the `PATH` environment variable.

If you still can't uninstall the Search Server, call IBM Technical Support.

Rebooting After Uninstalling the Product

After a DB2 Universal Database uninstall, it's very important to reboot before doing another install, because some Search Server DLLs can be held by the operating system and aren't removed until the next reboot. If a Search Server install happens before the reboot, the newly installed Search Server DLLs will be deleted on the next reboot, rendering the Search Server unusable.

Appendix C: Road Map to DB2 Information

Overview

This appendix lists the types of information shipped with DB2. Use this appendix to find information that will help you get the most use out of your product. The DB2 Universal Database library consists of SmartGuides, online help, and books.

To help you access product information online, DB2 provides the Information Center for OS/2, Windows 95, and the Windows NT operating systems. You can view task information, DB2 books, troubleshooting information, sample programs, and DB2 information on the Web. The section "About the Information Center" has more details.

SmartGuides

SmartGuides help you complete administration tasks by taking you through each task, one step at a time, and are available for OS/2, Windows 95, and Windows NT:

- The *Add Database SmartGuide* helps you catalog a database on a client workstation. To access this SmartGuide, click Add in the Client Configuration Assistant. Day 16, "Administering Clients with the CCA," explains how to use this SmartGuide.

- The *Create Database SmartGuide* helps you create a database and perform some basic configuration tasks. From the Control Center, right-click the Databases icon, and select Create | New. Day 8, "Creating Databases," explains how to use this SmartGuide.

- The *Performance Configuration SmartGuide* helps you tune the performance of a database by updating configuration parameters to match your business requirements. From the Control Center, right-click the database you want to tune, and select Configure Performance. The use of this SmartGuide is discussed on Day 20, "Tuning DB2 Universal Database Performance."

- The *Backup Database SmartGuide* helps you determine, create, and schedule a backup plan. From the Control Center, right-click the database you want to back up, and select Backup | Database Using SmartGuide. The use of this SmartGuide is discussed on Day 12, "Backing Up and Recovering Data."

- The *Restore Database SmartGuide* helps you recover a database after a failure. It also helps you understand which backup type to use and which logs to replay. From the Control Center, right-click the database you want to restore, and select Restore | Database Using SmartGuide. The use of this SmartGuide is discussed on Day 12, "Backing Up and Recovering Data."

- The *Create Table SmartGuide* helps you select basic data types and create a primary key for the table. From the Control Center, right-click the Tables icon, and select Create | Table Using SmartGuide. Day 8, "Creating Databases," covers the use of this SmartGuide.

- The *Create Table Space SmartGuide* helps you create a new table space. From the Control Center, right-click the Table Spaces icon, and select Create | Table Space Using SmartGuide. The use of this SmartGuide is covered on Day 9, "Creating Table Spaces."

Online Help

Online help of various types is available with all DB2 components:

- *Command Help* explains the syntax of commands in the command-line processor. With the command-line processor in interactive mode, enter **`? command`**, in which `command` is a keyword or the entire command. For example, `? catalog` displays help for all the `CATALOG` commands, whereas `? catalog database` displays help for the `CATALOG DATABASE` command.

- *Control Center Help* explains the tasks you can perform in a window or notebook. This help includes prerequisite information you need to know and describes how to use the window or notebook controls. To access this help from a window or notebook, click the Help button or press the F1 key.

- *Message Help* describes the cause of a message number and any action you should take. With the command-line processor in interactive mode, enter **`? message_number`**, in which `message_number` is a valid message number. For example, `? SQL30081` displays help about the `SQL30081` message.

  To view message help one screen at a time, enter **`? XXXnnnnn | more`**, in which `XXX` is the message prefix, such as `SQL`, and `nnnnn` is the message number, such as `30081`. To save message help in a file, enter **`? XXXnnnnn > filename.ext`**, in which `filename.ext` is the file where you want to save the message help.

- *SQL Help* explains the syntax of SQL statements. With the command-line processor in interactive mode, enter **`help statement`**, in which `statement` is a SQL statement. For example, `help SELECT` displays help about the `SELECT` statement.

- *SQLSTATE Help* explains SQL states and class codes. With the command-line processor in interactive mode, enter **`? sqlstate`** or **`? class-code`**, where `sqlstate` is a valid five-digit SQL state and `class-code` is a valid two-digit class code. For example, `? 08003` displays help for the `08003` SQL state, whereas `? 08` displays help for the `08` class code.

DB2 Books

Table C.1 lists the DB2 books. Use this table to determine which DB2 book you need to read to perform a task. The books are divided into the following groups:

- **Installation Books.** These books introduce the DB2 product and provide detailed information on how to install and configure the DB2 server and client products. Because DB2 is available on multiple platforms, a separate Quick Beginnings book is available for each server platform—for example, OS/2, Windows NT, and UNIX-based operating systems.

- **Getting Started Books.** The three Getting Started books help you learn about the DB2 product in a specific area: SQL, administration, or programming. Before you begin using the DB2 product or the more advanced books, you should read one or more of these books.

- **Advanced Administration Books.** After you learn the basics of the DB2 product, you might need to refer to the books in this category to help you perform more advanced tasks. In many cases, the Getting Started books refer to the advanced topics in the administration and replication books.

- **Programming Guide Books.** The books in this category guide you through the programming tasks you can perform with DB2.

- **Programming Reference Books.** After you know how to perform a programming task, you might need to search in one or more of these books for detailed syntax or programming options.
- **General Reference Books.** The books in this category include overall product references, such as the Glossary, Master Index, What's New, and Problem Determination Guide.
- **DB2 Connect Books.** These books are specific for the DB2 Connect product. You need to read these books only if you have DB2 Connect set up in your environment.

Table C.1. Books available on DB2's CD-ROM.

Book	Book Description	Filename
	Installation Books	
Quick Beginnings for OS/2 (S10J-8147)	Provides planning, installing, configuring, and using information for DB2 Universal Database on OS/2. Also contains installing and setup information for all supported clients.	`db2i2e50`
DB2 Personal Edition Quick Beginnings (S10J-8150)	Provides planning, installing, configuring, and using information for DB2 Universal Database Personal on OS/2, Windows 95, and Windows NT.	`db2i1e50`
Quick Beginnings for UNIX (S10J-8148)	Provides planning, installing, configuring, and using information for DB2 Universal Database on UNIX-based platforms. Also contains installing and setup information for all supported clients.	`db2ixe50`
Quick Beginnings for Windows NT (S10J-8149)	Provides planning, installing, configuring, and using information for DB2 Universal Database on the Windows NT operating system. Also contains installing and setup information for all supported clients.	`Db2i6e50`
	Installation Books	

Installing and Configuring DB2Clients (no form number)	Provides installation and setup information for all DB2 Client Application Enablers and DB2 Software Developer's Kits. (Available only in HTML and PostScript formats.)	`db2iye50`
DB2 Extended Enterprise Edition QuickBeginnings (S72H-9620)	Provides planning, installing, configuring, and using information for DB2 Universal Database Extended Enterprise Edition for AIX.	`db2v3e50`

Getting Started Books

Administration Getting Started (S10J-8154)	Introduces basic DB2 database adminis- tration concepts and tasks, and walks you through the primary administrative tasks.	`db2k0e50`
SQL Getting (S10J-8156)	Introduces SQL concepts and provides examples for many constructs and tasks.	`db2y0e50`*Started*
Road Map to DB2 Programming (S10J-8155)	Introduces the different ways your applications can access DB2, describes key DB2 features you can use in your applications, and points to detailed sources of information for DB2 programming	`db2u0e50`

Advanced Administration Books

Administration Guide (S10J-8157)	Contains information required to design, implement, and maintain a database to be accessed locally or in a client/server environment.	`db2d0e50`
Replication Guide and Reference (S95H-0999)	Provides planning, configuring, administering, and using information for the IBM Replication tools supplied with DB2.	`db2e0e50`

Programming Guide Books

CLI Guide and Reference	Explains how to develop applications that access DB2	`db2l0e50`

(S10J-8159)	databases by using the DB2 Call Level Interface, a callable SQL interface that's compatible with the Microsoft ODBC specification.	
Embedded SQL ProgrammingGuide(S10J-8158)	Explains how to develop applications that access DB2 databases by using embedded SQL. Includes discussions about programming techniques and performance considerations.	`db2a0e50`
Building Applicationsfor UNIXEnvironments (S10J-8161)	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a UNIX system.	`db2axe50`
Building Applications for Windows and OS/2 Environments (S10J-8160)	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a Windows or OS/2 system.	`db2a1e50`
DB2 SDK for Macintosh Building Your Applications (S50H-0528)	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a Macintosh system. (Available inPostScript format and hard copy only for DB2 version 2.1.2.)	`sqla7e02`
DB2 SDK for SCO OpenServer Building Your Applications (S89H-3242)	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on an SCO OpenServer system. (Available only for DB2 version 2.1.2.)	`sqla9e02`
DB2 SDK for Silicon Graphics IRIX Building Your Applications (S89H-4032)	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a Silicon Graphics system. Available only in PostScript format and hard copy for DB2 version 2.1.2.	`sqlaae02`

Programming Guide Books

DB2 SDK for SINIX Building Your Applications (S50H-0530)	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a SINIX system. (Available only in PostScript format and hard copy for DB2 version 2.1.2.)	`sqla8e00`

Programming Reference Books

Command Reference (S10J-8166)	Explains how to use the command-line processor and describes the DB2 commands you can use to manage your database.	db2n0e50
API Reference (S10J-8167)	Describes the DB2 application programming interfaces (APIs) and data structures you can use to manage your databases. Explains how to call APIs from your applications.	db2b0e50
SQL Reference (S10J-8165)	Describes SQL syntax, semantics, and the rules of the language. Also includes information about release-to-release incompatibilities, product limits, and catalog views.	db2s0e50
System Monitor Guide and Reference (S10J-8164)	Describes how to collect different kinds of information about your data-base and the database manager. Explains how you can use the information to understand database activity, improve performance, and determine the cause	db2f0e50
Message Reference (S10J-8168)	Lists messages and codes issued by DB2 and describes the actions you should take.	db2m0e50

General Reference Books

What's New (no form number)	Describes the new features, functions, and enhancements in DB2 Universal Database. (Available only in HTML and PostScript formats.)	db2q0e50

General Reference Books

Glossary (no form number)	Provides a comprehensive list of all DB2 terms and definitions. (Available only in HTML format.)	db2t0e50
Master Index (S10J-8170)	Contains a cross-reference to the major topics covered in the DB2	db2w0e50

	library. (Available only in PostScript format and hard copy.)	
Troubleshooting Guide (S10J-8169)	Helps you determine the source of errors, recover from problems, and use diagnostic tools in consultation with DB2 Customer Service.	`db2p0e50`
	DB2 Connect Books	
DB2 Connect Enterprise Edition Quick Beginnings (S10J-7888)	Provides planning, installing, configuring, and using information for DB2 Connect Enterprise Edition. Also contains installation and setup information for all supported clients.	`db2cye50`
DB2 Connect Personal Edition Quick Beginnings (S10J-8162)	Provides planning, installing, configuring, and using information for DB2 Connect Personal Edition.	`db2c1e50`
DB2 Connect User's Guide (S10J-8163)	Provides concepts, and programming and general usage information about the DB2 Connect products. DRDA Application Servers with DB2 Connect (formerly DDCS) application requesters. (Available only in HTML and PostScript formats.)	`db2c0e50`
DB2 Connectivity Supplement (no form number)	Provides setup and reference infor-mation for customers who want to use DB2 for AS/400, DB2 for OS/390, DB2 for MVS, or DB2 for VM as DRDA Application Requesters with DB2 Universal Database servers, and customers who want to use	`db2h1e50`

Most books are available in HTML and PostScript format, as well as in hard copy that you can order from IBM (exceptions are noted in the table). To order a hard-copy book from IBM, use the form number (in parentheses under the book title). The PostScript versions of the books are found on the CD-ROM with a file type of `.EXE` in the directory `x:\doc\en\ps\`.

Viewing Books

If you don't have your own Web browser, you can install Netscape from the CD-ROM shipped with the book. To do this, insert the CD-ROM into the CD-ROM drive and change to the `netscape` directory. Run the `N32E404.EXE` file to begin the Netscape installation process. Follow the instructions on the installation panels to proceed.

Searching for Information

To search for information in the HTML books, you can do the following:

- Click Search the DB2 Books at the bottom of any page in the HTML books. Use the search form to find a specific topic.
- To use the index to find a specific topic, click Index at the bottom of any page in an HTML book.
- Display the Table of Contents or Index of the HTML book, and then use the Web browser's `find` function to find a specific topic in the book.
- Use the `bookmark` function of the Web browser to quickly return to a specific topic.
- Use the `search` function of the Information Center to find specific topics. The later section "About the Information Center" has more details.

Printing Books

To print a book on a PostScript printer, look for the filename shown in the table. These files are in compressed format and have a file type of `.EXE`. They're located on the CD-ROM in the `x:\doc\en\ps\` directory.

To generate the `.PS` files from the CD-ROM, change to a drive where you have write access, and run the `.EXE` file as follows:

```
x:\doc\en\ps\db2s0e50.exe
```

Here *x* is the CD-ROM drive, `en` is the directory where the English-language books reside, and `db2s0e50.exe` is the book executable for the SQL Reference.

After you extract the PostScript book, a `.PS` file is created. To print this file, use

```
print filename.ps
```

in which *`filename`* is the name of the book—for example, `db2s0e50`.

The character in the sixth position of the filename indicates the language of a book. For example, the filename `db2d0e50` indicates that the *Administration Guide* is in English. Table C.2 lists the letters that indicate the language of a book.

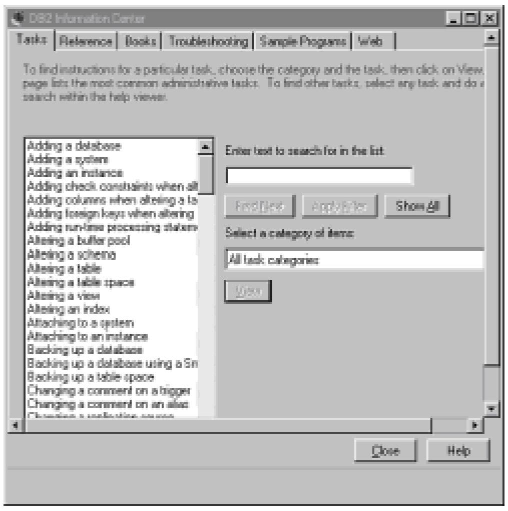
Table C.2. Filename letters indicating a book's language.

Language	Identifier	Language	Identifier	Language	Identifier
Brazilian Portuguese	B	German	G	Polish	P
Bulgarian	U	Greek	A	Russian	R

Czech	X	Hungarian	H	Simplified Chinese	S
Danish	D	Italian	I	Slovenian	L
English	E	Japanese	J	Spanish	Z
Finnish	Y	Korean	K	Swedish	S
French	F	Norwegian	N	Traditional Chinese	T

About the Information Center

The Information Center provides quick access to DB2 product information (see Figure C.1) and is available on OS/2, Windows 95, and Windows NT. To see it, you must install the DB2 Administration Tools.



To find instructions for a particular task, choose the category and the task, then click on View. page lists the most common administrative tasks. To find other tasks, select any task and do a search within the help viewer.

Adding a database
Adding a system
Adding an instance
Adding check constraints when alt
Adding columns when altering a ta
Adding foreign keys when altering
Adding run-time processing statem
Altering a buffer pool
Altering a schema
Altering a table
Altering a table space
Altering a view
Altering an index
Attaching to a system
Attaching to an instance
Backing up a database
Backing up a database using a Sn
Backing up a table space
Changing a comment on a trigger
Changing a comment on an alias

Enter text to search for in the list:

Select a category of items:

All task categories

Figure C.1: The Information Center.

Depending on your system, you can access the Information Center from the main product folder, the toolbar in the Control Center, or the Windows Start menu.

The Information Center provides the following kinds of information. Click the appropriate tab to look at the information:

- *Tasks* lists tasks you can perform by using DB2.
- *Reference* lists DB2 reference information, such as keywords, commands, and APIs.
- *Books* lists DB2 books.
- *Troubleshooting* lists categories of error messages and their recovery actions.
- *Sample Programs* lists sample programs that come with the DB2 Software Developer's Kit. If the Software Developer's Kit isn't installed, this tab isn't displayed.
- *Web* lists DB2 information on the World Wide Web. To access this information, you must have a connection to the Web from your system.

When you select a listed item, the Information Center launches a viewer to display the information. The viewer might be the system help viewer, an editor, or a Web browser, depending on the kind of information you select.

The Information Center provides search capabilities so that you can look for specific topics and filter capabilities to limit the scope of your searches.

Readme Files

For late-breaking information that couldn't be included in the DB2 books, see the readme files. Each DB2 product includes a readme file that you can find in the directory where the product is installed. The following readme files are available:

- *Installation Notes* contains product installation instructions that aren't found in the product library. This file is `x:\doc\en\install.txt`; *x* represents your CD-ROM drive and `en` the directory name used to represent the English language. This file is found only on the CD-ROM; it's not installed with the product.

- *Release Notes* contains general information about the product, which you'll need after the product is installed. This file is `e:\sqllib\release.txt`; *e* is the drive where the product is installed and `sqllib` the directory. This readme is available only after the product is installed, and it's more than 50 pages long.

- *HTML Search Readme* provides special instructions regarding the availability, installation, and use of the HTML search system. The file is `x:\doc\nqreadme.txt`; *x* is the CD-ROM drive.

- *DB2 Connect Readme* provides additional information regarding the DB2 Connect Personal and DB2 Connect Enterprise Edition products as well as the DB2 Application Server function. The file is `x:\doc\readdcs.txt`; *x* is the CD-ROM drive.

  Note Because this book was written after the product library was created, it contains much of the information found in the readme files.