

DB-Access

User's Manual

Informix Extended Parallel Server, Version 8.3
Informix Dynamic Server.2000, Version 9.2

December 1999
Part No. 000-6523

Published by Informix® Press

Informix Corporation
4100 Bohannon Drive
Menlo Park, CA 94025-1032

© 1999 Informix Corporation. All rights reserved. The following are trademarks of Informix Corporation or its affiliates, one or more of which may be registered in the United States or other jurisdictions:

Answers OnLine™; C-ISAM®, Client SDK™; DataBlade®, Data Director™; Decision Frontier™; Dynamic Scalable Architecture™; Dynamic Server™; Dynamic Server™, Developer Edition™; Dynamic Server™ with Advanced Decision Support Option™; Dynamic Server™ with Extended Parallel Option™; Dynamic Server™ with MetaCube®; Dynamic Server™ with Universal Data Option™; Dynamic Server™ with Web Integration Option™; Dynamic Server™, Workgroup Edition™; Dynamic Virtual Machine™; Extended Parallel Server™; Formation™; Formation Architect™; Formation Flow Engine™; Gold Mine Data Access®, IIF.2000™; i.Reach™; i.Sell™; Illustra®, Informix®, Informix® 4GL; Informix® InquireSM; Informix® Internet Foundation.2000™; InformixLink®; Informix® Red Brick® Decision Server™; Informix Session Proxy™; Informix® Vista™; InfoShelf™; Interforum™; I-Spy™; Mediazation™; MetaCube®; NewEra™; ON-Bar™; OnLine Dynamic Server™; OnLine/Secure Dynamic Server™; OpenCase®, Orca™; PaVER™; Red Brick® and Design; Red Brick® Data Mine™; Red Brick® Mine Builder™; Red Brick® Decisionscape™; Red Brick® Ready™; Red Brick Systems®, Regency Support®, Rely on Red BrickSM; RISQL®, Solution DesignSM; STARindex™; STARjoin™; SuperView®, TARGETindex™; TARGETjoin™; The Data Warehouse Company®; The one with the smartest data wins.™; The world is being digitized. We're indexing it.SM; Universal Data Warehouse Blueprint™; Universal Database Components™; Universal Web Connect™; ViewPoint®, Visionary™; Web Integration Suite™. The Informix logo is registered with the United States Patent and Trademark Office. The DataBlade logo is registered with the United States Patent and Trademark Office.

Documentation Team: Linda Briscoe, Evelyn Eldridge, Laura Kremers, Ann McKillop, Barbara Nomiyama

GOVERNMENT LICENSE RIGHTS

Software and documentation acquired by or for the US Government are provided with rights as follows:

- (1) if for civilian agency use, with rights as restricted by vendor's standard license, as prescribed in FAR 12.212;
- (2) if for Dept. of Defense use, with rights as restricted by vendor's standard license, unless superseded by a negotiated vendor license, as prescribed in DFARS 227.7202. Any whole or partial reproduction of software or documentation marked with this legend must reproduce this legend.

Table of Contents

Introduction

In This Introduction	3
About This Manual	3
Types of Users	3
Software Dependencies	4
Assumptions About Your Locale	4
New Features	5
Documentation Conventions	5
Typographical Conventions	6
Icon Conventions	7
Command-Line Conventions	8
Sample-Code Conventions	10
Additional Documentation	11
On-Line Manuals	11
Printed Manuals	11
On-Line Help	12
Error Message Documentation	12
Documentation Notes, Release Notes, Machine Notes	12
Related Reading	14
Compliance with Industry Standards	14
Informix Welcomes Your Comments	14

Chapter 1

Getting Started with DB-Access

In This Chapter	1-3
What Is DB-Access?	1-3
Using DB-Access	1-4
Setting Up DB-Access	1-6
Pre-DB-Access Installation	1-6
Environment Variables	1-7

Creating and Working with the Demonstration Databases	1-8
Demonstration Databases	1-8
Demonstration Installation	1-9
Command-Line Syntax.	1-10
Privileges for the Demonstration Database	1-12
Permissions for the SQL Command Files	1-13
Invoking DB-Access	1-13
Starting the DB-Access Menu Interface	1-14
Executing a Command File	1-22
Viewing the Information Schema	1-22
Checking for ANSI Compliance.	1-23
Activating the XLUF Feature for Nonprintable Characters	1-24
Using DB-Access Interactively in Non-Menu Mode	1-24
Related Manuals	1-28

Chapter 2

Using the Full-Screen Menu Interface

In This Chapter	2-3
Using the DB-Access User Interface	2-3
Using the Keyboard with DB-Access	2-4
Using Menus	2-5
Using Response Screens	2-7
Using the HELP Screen	2-8
Using an Editor Screen	2-9
Alternative Approaches	2-9
Database-Level Tasks	2-9
Table-Level Tasks.	2-12
Related Manuals	2-14

Chapter 3

The Query-language Option

In This Chapter	3-3
Overview of the SQL Menu	3-3
Using a System Editor	3-4
The Query-language Option	3-5
Entering a New SQL Statement	3-7
Using the Editing Keys.	3-7
Editing Restrictions	3-8
Running an SQL Statement	3-10
Statements That the Run Option Supports	3-11
Viewing Successful Results	3-15
What Happens When Errors Occur	3-16

Modifying an SQL Statement	3-17
Redirecting Query Results	3-19
Sending Output to a Printer	3-19
Sending Output to a File.	3-19
Sending Output to a Pipe	3-21
Choosing an Existing SQL Statement	3-21
Saving the Current SQL Statement	3-23
Displaying Table Information	3-24
Dropping an SQL Statement	3-26
Support for SPL Routines	3-28
Related Manuals	3-30

Chapter 4 The Database Option

In This Chapter	4-3
Selecting a Database Menu Option	4-4
Selecting a Database	4-5
List of Available Databases	4-5
Specifying a Database	4-6
Creating a Database	4-7
Specifying a Dbspace	4-8
Specifying Logging	4-8
Exiting the CREATE DATABASE Menu	4-9
Displaying Database Information	4-9
Retrieving Information on Dbspaces	4-11
Retrieving Nondefault Locale Information	4-11
Retrieving Information on Routines	4-12
Selecting a Different Database	4-14
Deleting a Database	4-15
The DROP DATABASE Screen	4-15
Confirming Your Decision to Delete a Database.	4-16
Closing a Database	4-16
Related Manuals	4-18

Chapter 5 The Table Option

In This Chapter	5-3
The TABLE Menu	5-3
Creating or Altering a Table	5-4
The CREATE TABLE Screen	5-4
The ALTER TABLE Screen	5-5
Using the Schema Editor	5-7

Adding Columns to a Table (Add Option)	5-8
Building the Table (Exit Option)	5-14
Modifying Columns (Modify Option).	5-15
Deleting Columns (Drop Option)	5-16
Arranging Storage and Locking (Table_options)	5-17
Defining Constraints	5-32
Displaying Table Information	5-43
Displaying Column Information	5-47
Displaying Index Information	5-51
Displaying Table-Level Privileges	5-52
Displaying References Privileges	5-53
Displaying Column Constraints and Defaults	5-54
Displaying Triggers	5-57
Displaying Fragmentation Information	5-59
Dropping a Table	5-60
Related Manuals	5-62

Chapter 6 The Connection and Session Options

In This Chapter	6-3
Choosing the Connection Option	6-3
Connecting to a Database Environment	6-4
Disconnecting from a Database Environment	6-8
Transaction Processing	6-8
Choosing the Session Option	6-9

Appendix A How to Read On-Line Help for SQL Statements

Appendix B Demonstration SQL

Index

Introduction

In This Introduction	3
About This Manual.	3
Types of Users	3
Software Dependencies	4
Assumptions About Your Locale.	4
New Features.	5
Documentation Conventions	5
Typographical Conventions	6
Icon Conventions	7
Comment Icons	7
Feature, Product, and Platform Icons	7
Command-Line Conventions	8
Sample-Code Conventions	10
Additional Documentation	11
On-Line Manuals	11
Printed Manuals	11
On-Line Help	12
Error Message Documentation	12
Documentation Notes, Release Notes, Machine Notes	12
Related Reading	14
Compliance with Industry Standards	14
Informix Welcomes Your Comments.	14

In This Introduction

This Introduction provides an overview of the information in this manual and describes the conventions it uses.

About This Manual

This manual describes how to use the DB-Access utility to access, modify, and retrieve information from Informix database servers. Chapter 1 of this manual explains how to create and work with the demonstration databases provided with your Informix database server.



Important: Use DB-Access with the current version of an Informix database server. If you use DB-Access with a database server from a different version, you might obtain inconsistent results, such as when you use a version that does not support long identifiers with a version that does.

Types of Users

This manual is written for the following users:

- Database users
- Database administrators
- Database-application programmers

This manual assumes that you have the following background:

- A working knowledge of your computer, your operating system, and the utilities that your operating system provides
- Some experience working with relational databases or exposure to database concepts
- Some experience with computer programming

If you have limited experience with relational databases, SQL, or your operating system, see the following Informix manuals:

- [Getting Started](#)
- The [Informix Guide to GLS Functionality](#)
- The [Informix Guide to SQL: Reference](#)
- The [Informix Guide to SQL: Syntax](#)
- The [Informix Guide to SQL: Tutorial](#)

Software Dependencies

This manual assumes that you are using one of the following database servers:

- Informix Extended Parallel Server, Version 8.3
- Informix Dynamic Server 2000, Version 9.2

Assumptions About Your Locale

Informix products can support many languages, cultures, and code sets. All culture-specific information is brought together in a single environment, called a Global Language Support (GLS) locale.

This manual assumes that you use the U.S. 8859-1 English locale as the default locale. The default is **en_us.8859-1** (ISO 8859-1) on UNIX platforms or **en_us.1252** (Microsoft 1252) for Windows NT environments. This locale supports U.S. English format conventions for dates, times, and currency, and also supports the ISO 8859-1 or Microsoft 1252 code set, which includes the ASCII code set plus many 8-bit characters such as é, è, and ñ.

If you plan to use nondefault characters in your data or your SQL identifiers, or if you want to conform to the nondefault collation rules of character data, you need to specify the appropriate nondefault locale.

For instructions on how to specify a nondefault locale, additional syntax, and other considerations related to GLS locales, see the [Informix Guide to GLS Functionality](#).

New Features

For a comprehensive list of new features for your database server, see the release notes.

Documentation Conventions

This section describes the conventions that this manual uses. These conventions make it easier to gather information from this and other volumes in the documentation set.

The following conventions are discussed:

- Typographical conventions
- Icon conventions
- Command-line conventions
- Sample-code conventions

Typographical Conventions

This manual uses the following conventions to introduce new terms, illustrate screen displays, describe command syntax, and so forth.

Convention	Meaning
KEYWORD	All primary elements in a programming language statement (keywords) appear in uppercase letters in a serif font.
<i>italics</i> italics <i>italics</i>	Within text, new terms and emphasized words appear in italics. Within syntax and code examples, variable values that you are to specify appear in italics.
boldface boldface	Names of program entities (such as classes, events, and tables), environment variables, file and pathnames, and interface elements (such as icons, menu items, and buttons) appear in boldface.
monospace <i>monospace</i>	Information that the product displays and information that you enter appear in a monospace typeface.
KEYSTROKE	Keys that you are to press appear in uppercase letters in a sans serif font.
◆	This symbol indicates the end of one or more product- or platform-specific paragraphs.
→	This symbol indicates a menu item. For example, “Choose Tools→Options ” means choose the Options item from the Tools menu.



Tip: When you are instructed to “enter” characters or to “execute” a command, immediately press RETURN after the entry. When you are instructed to “type” the text or to “press” other keys, no RETURN is required.

Icon Conventions

Throughout the documentation, you will find text that is identified by several different types of icons. This section describes these icons.

Comment Icons

Comment icons identify three types of information, as the following table describes. This information always appears in italics.

Icon	Label	Description
	<i>Warning:</i>	Identifies paragraphs that contain vital instructions, cautions, or critical information
	<i>Important:</i>	Identifies paragraphs that contain significant information about the feature or operation that is being described
	<i>Tip:</i>	Identifies paragraphs that offer additional details or shortcuts for the functionality that is being described

Feature, Product, and Platform Icons

Feature, product, and platform icons identify paragraphs that contain feature-specific, product-specific, or platform-specific information.

Icon	Description
	Identifies information that relates to the Informix Global Language Support (GLS) feature
	Identifies information that is specific to Informix Dynamic Server 2000
	Identifies information that is specific to a Native Language Support (NLS) database or application

(1 of 2)

Icon	Description
UNIX	Identifies information that is specific to UNIX platforms
WIN NT	Identifies information that is specific to the Windows NT environment
XPS	Identifies information or syntax that is specific to Informix Extended Parallel Server

(2 of 2)

These icons can apply to an entire section or to one or more paragraphs within a section. If an icon appears next to a section heading, the information that applies to the indicated feature, product, or platform ends at the next heading at the same or higher level. A ♦ symbol indicates the end of feature-, product-, or platform-specific information that appears within one or more paragraphs within a section.

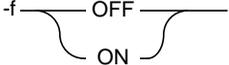
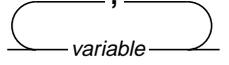
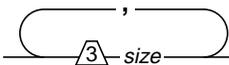
Command-Line Conventions

This section defines and illustrates the format of commands that are available in Informix products. These commands have their own conventions, which might include alternative forms of a command, required and optional parts of the command, and so forth.

Each diagram displays the sequences of required and optional elements that are valid in a command. A diagram begins at the upper-left corner with a command. It ends at the upper-right corner with a vertical line. Between these points, you can trace any path that does not stop or back up. Each path describes a valid form of the command. You must supply a value for words that are in italics.

You might encounter one or more of the following elements on a command-line path.

Element	Description
command	This required element is usually the product name or other short word that invokes the product or calls the compiler or preprocessor script for a compiled Informix product. It might appear alone or precede one or more options. You must spell a command exactly as shown and use lowercase letters.
<i>variable</i>	A word in italics represents a value that you must supply, such as a database, file, or program name. A table following the diagram explains the value.
-flag	A flag is usually an abbreviation for a function, menu, or option name, or for a compiler or preprocessor argument. You must enter a flag exactly as shown, including the preceding hyphen.
.ext	A filename extension, such as .sql or .cob , might follow a variable that represents a filename. Type this extension exactly as shown, immediately after the name of the file. The extension might be optional in certain products.
(., ; + * - /)	Punctuation and mathematical notations are literal symbols that you must enter exactly as shown.
' '	Single quotes are literal symbols that you must enter as shown.
	A reference in a box represents a subdiagram. Imagine that the subdiagram is spliced into the main diagram at this point. When a page number is not specified, the subdiagram appears on the same page.
	
	A shaded option is the default action.
	Syntax within a pair of arrows indicates a subdiagram.
	The vertical line terminates the command.

Element	Description
	A branch below the main path indicates an optional path. (Any term on the main path is required, unless a branch can circumvent it.)
	A loop indicates a path that you can repeat. Punctuation along the top of the loop indicates the separator symbol for list items.
	A gate ($\sqrt{3}$) on a path indicates that you can only use that path the indicated number of times, even if it is part of a larger loop. You can specify <i>size</i> no more than three times within this statement segment.

(2 of 2)

Sample-Code Conventions

Examples of SQL code occur throughout this manual. Except where noted, the code is not specific to any single Informix application development tool. If only SQL statements are listed in the example, they are not delimited by semicolons. For instance, you might see the code in the following example:

```
CONNECT TO stores_demo
...

DELETE FROM customer
    WHERE customer_num = 121
...

COMMIT WORK
DISCONNECT CURRENT
```

To use this SQL code for a specific product, you must apply the syntax rules for that product. For example, if you are using DB-Access, you must delimit multiple statements with semicolons. If you are using an SQL API, you must use EXEC SQL at the start of each statement and a semicolon (or other appropriate delimiter) at the end of the statement.



Tip: *Ellipsis points in a code example indicate that more code would be added in a full application, but it is not necessary to show it to describe the concept being discussed.*

For detailed directions on using SQL statements for a particular application development tool or SQL API, see the manual for your product.

Additional Documentation

For additional information, you might want to refer to the following types of documentation:

- On-line manuals
- Printed manuals
- On-line help
- Error message documentation
- Documentation notes, release notes, and machine notes
- Related reading

On-Line Manuals

An Answers OnLine CD that contains Informix manuals in electronic format is provided with your Informix products. You can install the documentation or access it directly from the CD. For information about how to install, read, and print on-line manuals, see the installation insert that accompanies Answers OnLine.

Informix on-line manuals are also available on the following Web site:

www.informix.com/answers

Printed Manuals

To order printed manuals, call 1-800-331-1763 or send email to moreinfo@informix.com. Please provide the following information when you place your order:

- The documentation that you need
- The quantity that you need
- Your name, address, and telephone number

WIN NT

On-Line Help

Informix provides Help screens with each interactive interface that displays information about the interfaces and the functions that they perform. For more information about the on-line help that is available with DB-Access, see [“Using the HELP Screen”](#) in [Chapter 2](#) and [Appendix A](#).

Error Message Documentation

Informix software products provide ASCII files that contain all of the Informix error messages and their corrective actions.

To read error messages and corrective actions on UNIX, use one of the following utilities.

Utility	Description
finderr	Displays error messages on line
rofferr	Formats error messages for printing



To read error messages and corrective actions in Windows environments, use the **Informix Find Error** utility. To display this utility, choose **Start→Programs→Informix** from the Task Bar. ◆

Instructions for using the preceding utilities are available in [Answers OnLine](#). [Answers OnLine](#) also provides a listing of error messages and corrective actions in HTML format.

Documentation Notes, Release Notes, Machine Notes

In addition to printed documentation, the following sections describe the on-line files that supplement the information in this manual. Please examine these files before you begin using your database server. They contain vital information about application and performance issues.

UNIX

WIN NT

UNIX

On UNIX platforms, the following on-line files appear in the `$INFORMIXDIR/release/en_us/0333` directory. Replace `x.y` in the filenames with the version number of your database server.

On-Line File	Purpose
DBACCDOC_x.y	The documentation-notes file for your version of this manual describes topics that are not covered in the manual or that were modified since publication.
SERVERS_x.y	The release-notes file describes feature differences from earlier versions of Informix products and how these differences might affect current products. This file also contains information about any known problems and their workarounds.
IDS_x.y or XPS_x.y	The machine-notes file describes any special actions that you must take to configure and use Informix products on your computer. Machine notes are named for the product described.

**WIN NT**

The following items appear in the **Informix** folder. To display this folder, choose **Start→Programs→Informix** from the Task Bar.

Program Group Item	Description
Documentation Notes	This item includes additions or corrections to manuals, along with information about features that might not be covered in the manuals or that have been modified since publication.
Release Notes	This item describes feature differences from earlier versions of Informix products and how these differences might affect current products. This file also contains information about any known problems and their workarounds.

Machine notes do not apply to Windows environments. ◆

Related Reading

For a list of publications that provide an introduction to database servers and operating-system platforms, refer to your *Getting Started* manual.

Compliance with Industry Standards

The American National Standards Institute (ANSI) has established a set of industry standards for SQL. Informix SQL-based products are fully compliant with SQL-92 Entry Level (published as ANSI X3.135-1992), which is identical to ISO 9075:1992. In addition, many features of Informix database servers comply with the SQL-92 Intermediate and Full Level and X/Open SQL CAE (common applications environment) standards.

Informix Welcomes Your Comments

Let us know what you like or dislike about our manuals. To help us with future versions of our manuals, we want to know about any corrections or clarifications that you would find useful. Include the following information:

- The name and version of the manual that you are using
- Any comments that you have about the manual
- Your name, address, and phone number

Send electronic mail to us at the following address:

`doc@informix.com`

The **doc** alias is reserved exclusively for reporting errors and omissions in our documentation.

We appreciate your suggestions.

Getting Started with DB-Access

In This Chapter	1-3
What Is DB-Access?	1-3
Using DB-Access	1-4
Setting Up DB-Access	1-6
Pre-DB-Access Installation	1-6
Environment Variables	1-7
Creating and Working with the Demonstration Databases	1-8
Demonstration Databases	1-8
Demonstration Installation	1-9
Command-Line Syntax	1-10
Privileges for the Demonstration Database.	1-12
Permissions for the SQL Command Files	1-13
Invoking DB-Access	1-13
Starting the DB-Access Menu Interface	1-14
Displaying the Main Menu	1-14
Displaying Other Menus or Options	1-15
Menu Suboptions.	1-18
Examples of Command-Line Syntax	1-21
Executing a Command File.	1-22
Viewing the Information Schema	1-22
Checking for ANSI Compliance	1-23
Activating the XLUF Feature for Nonprintable Characters	1-24
Using DB-Access Interactively in Non-Menu Mode	1-24
Reading from the Keyboard or Standard Input Device	1-24
Connecting to a Database Environment in Non-Menu Mode.	1-26
Related Manuals.	1-28



In This Chapter

This chapter introduces a new user to DB-Access. It provides information about how to set up your DB-Access environment and the demonstration database.

This chapter discusses the following topics:

- What is DB-Access?
- Setting up DB-Access
- Creating and working with the demonstration databases
- Setting the DB-Access permissions
- Invoking DB-Access

What Is DB-Access?

DB-Access provides a user interface for entering, executing, and debugging Structured Query Language (SQL) statements and Stored Procedure Language (SPL) routines.

SQL and SPL allow you to perform data-definition tasks, such as specifying the number and type of data columns in a table, and data-management tasks, such as storing, viewing, and changing table data. The DB-Access interface allows you to apply powerful Informix extensions to SQL and SPL.

You can use DB-Access for the following aspects of database processing:

- Using ad hoc queries that you execute infrequently
- Connecting to one or more databases, transferring data between the database and external text files, and displaying information about a database

- Displaying system catalog tables and the Information Schema, which are explained in the *Informix Guide to SQL: Reference*
- Practicing the statements and examples provided in the *Informix Guide to SQL: Tutorial* or the *Informix Guide to Database Design and Implementation*
- Testing applications that you intend to store for use in a production environment



Important: *DB-Access is not intended as an application-development environment. You branch conditionally or loop through SQL statements when you run them within DB-Access.*

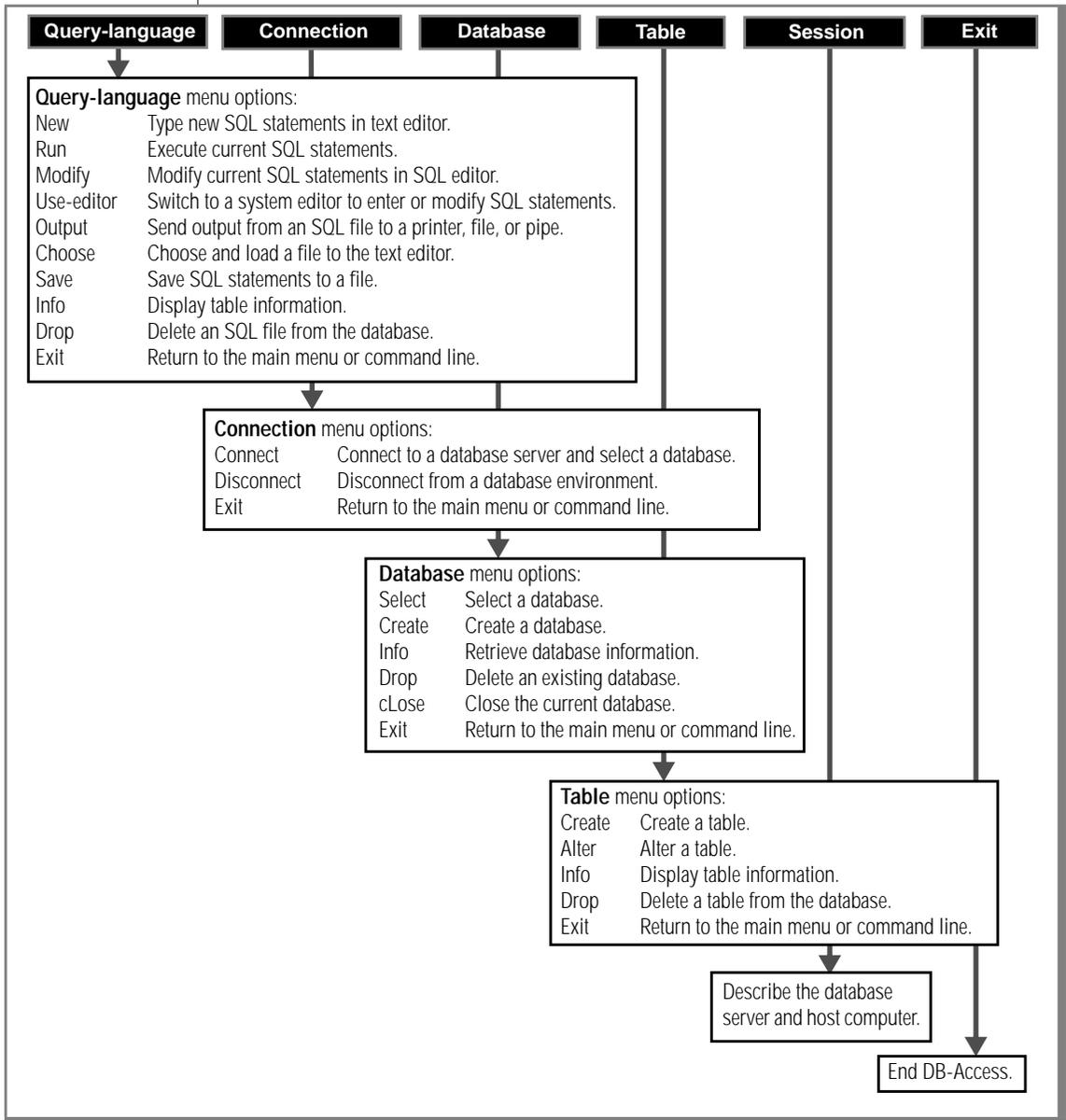
Using DB-Access

You can use the DB-Access user interface to:

- run statements interactively, discarding them after you achieve the desired results, or saving them in a file for repetitive execution.
- type statements directly in the DB-Access text-entry screen or your preferred text editor.
- start DB-Access in menu mode and select options from the menus.

[Figure 1-1 on page 1-5](#) illustrates the top two levels of the DB-Access menu hierarchy. The view of available options summarizes the types of database tasks that you can accomplish with DB-Access.

Figure 1-1
The DB-Access Menu Hierarchy



For detailed submenu instructions, see the following chapters.

Option	Instructions
Query-language	Chapter 3, “The Query-language Option”
Database	Chapter 4, “The Database Option”
Table	Chapter 5, “The Table Option”
Session	Chapter 6, “The Connection and Session Options”
Connection	Chapter 6, “The Connection and Session Options”

Setting Up DB-Access

This section reviews the procedure for setting up the DB-Access environment.

Pre-DB-Access Installation

Before you set up the DB-Access environment, you must perform the following preparatory steps:

1. Install the database server and set environment variables.
2. Set up the Global Language Support (GLS) locale, where language localization requires it. ♦
3. Start the database server.

You can then run the initialization script for the demonstration database (optional) and invoke the DB-Access program, as described in [“Invoking DB-Access” on page 1-13](#).

GLS

Environment Variables

As part of the installation and setup process, the system or database administrator sets certain environment variables that enable Informix products to work within a particular operating-system environment. This section lists the environment variables that affect your ability to use DB-Access.



Important: You must have `$INFORMIXDIR/bin` in your path if you use DB-Access on a UNIX platform or `%INFORMIXDIR%\bin` in your path if you use DB-Access on a Windows NT platform. Your operating system uses the path to locate the initialization script and the `dbaccess` executable.

UNIX

In a UNIX environment, the database server must have the appropriate terminal set up from among those listed in `INFORMIXTERM`. ♦

IDS

You can set the following optional environment variables:

- `DBACCNOIGN` rolls back an incomplete transaction if you execute the `LOAD` command in menu mode. ♦
- `DBACCNOIGN` rolls back an incomplete transaction if an error occurs while you are in nonmenu mode. ♦
- `DBCENTURY` lets you choose the appropriate expansion for `DATE` and `DATETIME` values that have only a two-digit year, such as `04/15/98`.
- `DBDATE` specifies the end-user formats of `DATE` values. See [Informix Migration Guide](#) for more information on this variable.
- `DBEDIT` sets the default DB-Access text editor without changing the default text editor associated with the operating-system shell.

For more information on how DB-Access uses the text editor that you specify as default, see “[Entering a New SQL Statement](#)” on page 3-7.

- `DBFLTMASK` sets the default floating-point values of data types `FLOAT`, `SMALLFLOAT`, and `DECIMAL` within a 14-character buffer. The effect of this variable is limited to the DB-Access display size for numbers.
- `DELIMIDENT` causes the database server to interpret double quoted (“”) text as identifiers rather than strings.
- `IFX_LONGID` determines whether a client application can handle long identifiers. ♦

IDS

For more information on environment variables, see the [Informix Guide to SQL: Reference](#).

Creating and Working with the Demonstration Databases

You can practice using DB-Access with a demonstration database or with a database that you create. If you use an Informix demonstration database, you can add, delete, or change the provided data and scripts, then reinitialize the database to its original condition. This manual provides examples of statements run against the demonstration data, with illustrations that show the results of these statements.

Demonstration Databases

The DB-Access utility, which is provided with your Informix database server products, includes one or more of the following demonstration databases:

- The **stores_demo** database illustrates a relational schema with information about a fictitious wholesale sporting-goods distributor. Many examples in Informix manuals are based on the **stores_demo** database.
- The **sales_demo** database illustrates a dimensional schema for data-warehousing applications. For conceptual information about dimensional data modeling, see the [Informix Guide to Database Design and Implementation](#). ♦
- The **superstores_demo** database illustrates an object-relational schema. The **superstores_demo** database contains examples of extended data types, type and table inheritance, and user-defined routines. ♦

For descriptions of the databases and their contents, see the [Informix Guide to SQL: Reference](#).

The scripts that you use to install the demonstration databases reside in the **\$INFORMIXDIR/bin** directory on UNIX platforms and in the **%INFORMIXDIR%\bin** directory in Windows environments.

XPS

IDS

Figure 1-2 lists the databases available for each database server. To set up or reinitialize the demonstration database, run the corresponding initialization script that Figure 1-2 shows.

Figure 1-2 Demonstration Databases

Server Type	Database Name	Model	Initialization Script
All Informix database servers	stores_demo	Relational database	dbaccessdemo
Informix Extended Parallel Server	sales_demo	Dimensional data warehouse	dbaccessdemo -dw
Informix Dynamic Server 2000	superstores_demo	Object-relational database	dbaccessdemo_ud

Demonstration Installation

When you run the installation script for a demonstration database, the script asks you if you would like to copy sample SQL command files. Command files that the demo includes have a **.sql** extension and contain sample SQL statements that you can use.

Always initialize or run DB-Access from the directory in which you want to store SQL command files for the following reasons:

- Each time you create or reinitialize the demonstration database, the installation script prompts you to ask if you want a copy of the sample command files provided with the demonstration database saved in the current directory.
- DB-Access lists only the files that end in the extension **.sql** in the current directory.

Command files are described in Appendix B of this manual.

UNIX



To prepare a demonstration database

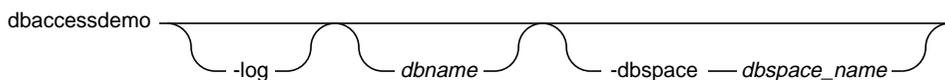
1. Create a new directory.
You must have UNIX read and execute permissions for each directory in the pathname that you create. ♦
2. Change directories to the new directory and run the initialization script.
[Figure 1-2 on page 1-9](#) describes the various database models available for each database server version. For options that you can use with the initialization script, see [“Command-Line Syntax” on page 1-10](#).
3. The initialization script displays a series of messages on the screen as the database is created. The final message of the script prompts you to make a choice. If you want to copy the command files into the directory that you created, press Y.

Tip: If you want to discard changes that you made to your database or to the command files, rerun the demonstration initialization script. When the script prompt message displays, press Y to replace the command files with the original versions.

Command-Line Syntax

The illustrations in this section show the syntax of the initialization scripts. For assistance in reading the diagrams, see [“Command-Line Conventions” on page 8](#) in the Introduction to this manual.

To create stores_demo



Examples

- The following command creates a database named **stores_demo**:

```
dbaccessdemo
```

- The following example creates an instance of the **stores_demo** database named **demo_db**:

```
dbaccessdemo demo_db
```

- The following command initializes the **stores_demo** database and also initiates log transactions:

```
dbaccessdemo -log
```

- The following command creates an instance of the **stores_demo** database named **demo_db** in **dbspace_2**:

```
dbaccessdemo demo_db -dbspace dbspace_2
```

Privileges for the Demonstration Database

When you run the initialization script to create an instance of the demonstration database under your login, the database server recognizes your login as the owner and database administrator (DBA) of that database. As DBA, you automatically acquire some exclusive privileges over objects in your database. You can extend those privileges to others with the GRANT statement, which is described in the [Informix Guide to SQL: Syntax](#).

UNIX

Permissions for the SQL Command Files

Some operating systems require that you have execute permissions to run SQL command files, read permissions to display these files or their contents in DB-Access, or write permissions to save modified or new files.

Use the UNIX **chmod** command to enable execution of the SQL files that the initialization script installed. ♦

Invoking DB-Access

For more information on invoking DB-Access, see the following references:

- To display all the menus, start DB-Access at its main menu. See [“Displaying the Main Menu” on page 1-14](#).
- To start and exit from a specific DB-Access menu or screen, see [“Displaying Other Menus or Options” on page 1-15](#).
- To execute a file that contains SQL statements without displaying the DB-Access menus, see [“Executing a Command File” on page 1-22](#).
- To type DB-Access options at the command line, without the full-screen menu interface, see [“Using DB-Access Interactively in Non-Menu Mode” on page 1-24](#).
- To check the DB-Access version or transfer nonprintable characters in hexadecimal form, see [“Activating the XLUF Feature for Nonprintable Characters” on page 1-24](#).

WIN NT

You can set up the DB-Access program icon to perform any of the commands that this chapter shows. ♦

UNIX

If the **TERM**, **TERMCAP**, or **TERMINFO** environment variables do not enable DB-Access to recognize the type of terminal you use, the main menu does not appear. Instead, a message similar to the following text appears:

```
Unknown terminal type.
```

♦

WIN NT



If you use a Windows NT terminal to run DB-Access on a UNIX database server, the terminal-emulation window must emulate a terminal type that DB-Access can recognize, or the database server displays an unknown terminal-type message in the terminal-emulation window.

For more information about **INFORMIXTERM** and the appropriate terminal setup, see your *Installation Guide* or the *Informix Guide to SQL: Reference*. ♦

*Tip: If your operating system cannot find **dbaccess**, place the full path before the program name, as follows:*

```
$INFORMIXDIR/bin/dbaccess
```

Starting the DB-Access Menu Interface

For information on how to read the diagrams in this section, see “[Command-Line Conventions](#)” on page 8 in the Introduction to this manual. To familiarize yourself with the DB-Access menu structure, see [Figure 1-1](#) on page 1-5.

Displaying the Main Menu

Typically, you start DB-Access with the main menu as the top-level menu from which you reach submenus and their options.

If you use a window interface, such as OpenWindows, on a UNIX terminal, issue the `dbaccess` command from a nonscrolling console window. ♦

If your path includes `$INFORMIXDIR/bin`, the following command shows the simplest way to start DB-Access:

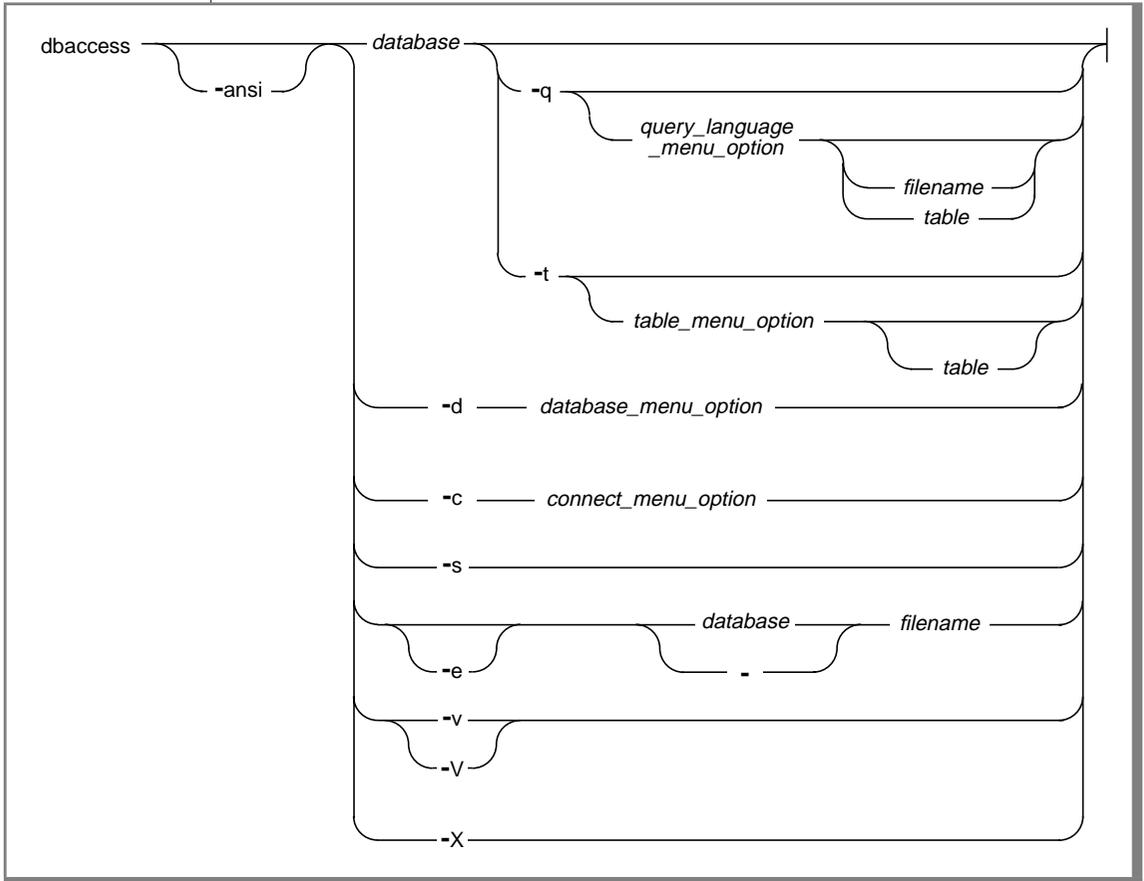
```
dbaccess
```

Without arguments, the single word **dbaccess** starts the main menu with no database selected and no options activated. You can then select submenus from the main menu.

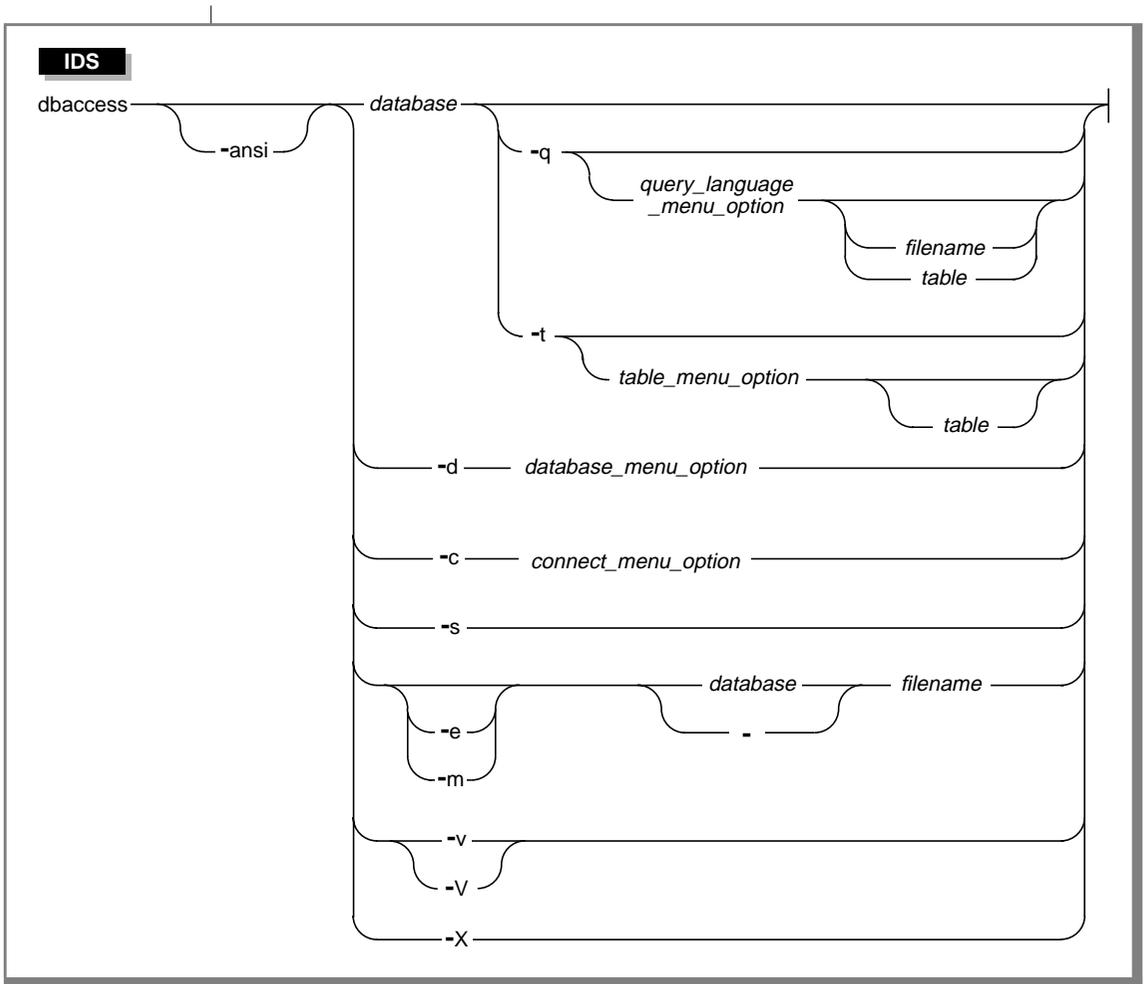
UNIX

Displaying Other Menus or Options

You can specify the options shown in the following diagram to access menus directly.



Starting the DB-Access Menu Interface



-ansi	causes DB-Access to generate a warning whenever it encounters an Informix extension to ANSI-compliant syntax. For more information, see “Checking for ANSI Compliance” on page 1-23 .
-c	starts with the CONNECTION menu as the top-level menu.
-d	starts with the DATABASE menu as the top-level menu.
-e	echoes each line from a command file designated by <i>filename</i> .
-m	displays all error messages generated by multiple levels of the server that pertain to an SQL statement in command files. This option is available only for Informix Dynamic Server 2000.
-q	starts at the query-language menu (SQL-menu) as the top-level menu.
-s	connects you to the main DB-Access menu and displays information about the current session. This information includes database server name, database server type, the host computer, server capabilities, and other settings.
-t	starts at the TABLE menu as the top-level menu.
-v or -V	displays the version number and serial number for DB-Access without launching the application. You cannot use any other options with -v or -V .
-X	activates the hexadecimal format for LOAD and UNLOAD statements.
-	indicates that you are not specifying a database in the command line because one is specified in a DATABASE statement in the command file.
<i>database</i>	name of the database that you want DB-Access to connect to at the startup of your current session.
<i>filename</i>	names a command file to load with the SQL menu.

<i>table</i>	specifies a table in the database
<i>connect_menu_option</i>	option or suboption on the connect menu. See “ CONNECTION Menu Options. ”
<i>database_menu_option</i>	option or suboption on the database menu. See “ DATABASE Menu Options. ”
<i>query_language_menu_option</i>	option or suboption on the query-language menu. See “ QUERY-LANGUAGE Menu Options. ”
<i>table_menu_option</i>	option or suboption on the table menu. See “ TABLE Menu Options ” on page 1-20.

If you exit from a submenu or option that you specified from the command line, you will exit directly to the operating-system command line.

Menu Suboptions

The following menu suboptions allow you to access submenus directly.

CONNECTION Menu Options

- cc** chooses the Connect option on the CONNECTION menu.
- cd** chooses the Disconnect option on the CONNECTION menu.

DATABASE Menu Options

- dc** chooses the Create option on the DATABASE menu.
- dcl** takes you to the LOG option on the CREATE DATABASE menu
- dd** chooses the Drop option on the DATABASE menu.
- di** chooses the Info option on the DATABASE menu. With this option, you can add another letter as follows to go to the next menu level and view:
 - dib** dbspaces information for the current database (OnLine only)

-din NLS information for the current database

-dip stored procedures in the current database

If you do not include a database name before any **-di** option, you must choose a current database from the SELECT DATABASE screen.

-dl chooses the cLose option on the DATABASE menu.

-ds chooses the Select option on the DATABASE menu.

QUERY-LANGUAGE Menu Options

-qc chooses the Choose option on the SQL menu.

-qd chooses the Drop option on the SQL menu.

-qi chooses the Info option on the SQL menu. With this option, you can add another letter as shown in the following list (and specify a table) to go to the next menu level and view:

-qic columns in the table

-qif information on fragmentation strategy for the table

-qig information on triggers in the table

-qii indexes on the table

-qio table constraints

-qip access privileges on the table

-qir table-level references privilege on the table

-qis table status information

If you do not include a table name with the **-qi** option, you must choose one from the INFO FOR TABLE screen.

-qm chooses the Modify option on the SQL menu.

-qn chooses the New option on the SQL menu.

-qs chooses the Save option on the SQL menu.

-qu chooses the Use-editor option on the SQL menu.

If you do not include a database name before a **-q** option, you must choose a current database from the SELECT DATABASE screen.

When you select the Modify option on the QUERY-LANGUAGE menu, you must first select a command file to modify from the CHOOSE menu. The MODIFY screen then appears and displays the text.



Tip: You cannot go directly to the Run or Output option on the SQL menu. Trying to do so results in an error message.

TABLE Menu Options

-ta chooses the Alter option on the TABLE menu.

-tc chooses the Create option on the TABLE menu.

-td chooses the Drop option on the TABLE menu.

-ti chooses the Info option on the TABLE menu. With this option, you can add another letter as shown in the following list (and specify a table) to go to the next menu level and view:

-tic columns in the table

-tif information on fragmentation strategy for the table

-tig information on triggers in the table

-tii indexes on the table

-tio table constraints

-tip access privileges on the table

-tir table-level references privilege on the table

-tis table status information

If you do not include a table name with the **-ti** option, you must choose one from the INFO FOR TABLE screen.

If you do not include a database name before a **-t** option, you must choose a current database from the SELECT DATABASE screen.

Examples of Command-Line Syntax

Assume that the database server that you have on-line contains a database named **mystores**. To make the **mystores** database the current database, start DB-Access with the following command:

```
dbaccess mystores
```

You can specify a database on a database server that is not on-line. For example, either of the following commands selects the **newstores** database on the **xyz** database server:

```
dbaccess newstores@xyz  
dbaccess //xyz/newstores
```

When DB-Access starts, the database and database server name that you specify appear on the dashed line, as [Figure 1-3](#) shows.

```
DB-Access:  Query-language  Connection  Database  Table  Session  Exit  
  
----- newstores@xyz -----Press CTRL-W for Help ---
```

Figure 1-3
*The DB-Access
Main Menu
with Database and
Database Server
Name*

Executing a Command File

When you invoke DB-Access from the command line, you can specify a database as current, execute a file that contains one or more SQL statements, and display multilevel error messages.

The following sample command executes the SQL statements in a file named **sel_stock.sql** on the **mystores** database:

```
dbaccess mystores sel_stock
```

The following sample command executes the SQL statements in the **sel_all.sql** file on the database that file specifies:

```
dbaccess - sel_all.sql
```

Viewing the Information Schema

Use DB-Access to execute the **xpg4_is.sql** file in the `$INFORMIXDIR/etc` directory. This SQL file creates the Information Schema and installs the views for a specified database. The following command creates the Information Schema for database **mystores**:

```
dbaccess mystores $INFORMIXDIR/etc/xpg4_is.sql
```

The Information Schema adds to the database four information-only views that conform to X/Open XPG4 with Informix extensions. After you run **xpg4_is.sql**, use DB-Access to retrieve information about the tables and columns that you have access to in the specified database. For more information on Information Schema views, see the [Informix Guide to SQL: Reference](#).



Tip: Informix does not recommend that you install these XPG4-compliant views on an ANSI database, because their format differs considerably from that of the ANSI-compliant Information Schema views that the SQL standards committee has defined.

Checking for ANSI Compliance

To check your SQL statements for compliance with ANSI standards, include the **-ansi** option or set the **DBANSIWARN** environment variable. Use the **-ansi** option with other **dbaccess** options such as **-dc** (to create a database), **-tc** or **-ta** (to create or alter a table), or **-qc filename** (to choose a command file). The following command checks for ANSI compliance while DB-Access creates the database **research**:

```
dbaccess -ansi -dc research
```

You do not need to specify the **-ansi** option on the command line if the **DBANSIWARN** environment variable is set.

DB-Access displays the SQLSTATE value with the warning under the following circumstances:

- You include the **-ansi** option or set the **DBANSIWARN** environment variable.
- You access or create an ANSI database.
- You run DB-Access in line mode or specify a **.sql** input file.
- Execution of a SQL statement generates a warning rather than an error.

For more information on ANSI-compliant databases, see the [Informix Guide to SQL: Reference](#) and the [Informix Guide to SQL: Tutorial](#). For more information on SQLSTATE values, see the GET DIAGNOSTICS statement in the [Informix Guide to SQL: Syntax](#). The [Informix Guide to SQL: Syntax](#) also provides information about ANSI compliance in Informix SQL syntax.

Activating the XLUF Feature for Nonprintable Characters

You can use the **-X** option to activate the hexadecimal load and unload format (XLUF) feature in DB-Access at the command line. When you invoke DB-Access with the **-X** option, the LOAD and UNLOAD SQL statements can format nonprintable ASCII signs in hexadecimal format. A **.unl** file that the UNLOAD statement produces contains the hexadecimal format changes.



Important: *The **.unl** files that contain data in a hexadecimal format are not compatible with Informix database servers prior to Version 6.0. However, **.unl** files generated without the XLUF functionality are fully compatible with Version 6.0 or later database servers.*

For more information, see the descriptions of the LOAD and UNLOAD statements in the [Informix Guide to SQL: Syntax](#). Also see the discussion of the various SQL utilities in the [Informix Migration Guide](#) and the data types information in the [Informix Guide to SQL: Reference](#).

Using DB-Access Interactively in Non-Menu Mode

If you do not want to use the menus and do not have a prepared SQL file, use your keyboard or standard input device to enter SQL statements.

Reading from the Keyboard or Standard Input Device

When you invoke DB-Access without a menu argument (such as **-q**) and with a hyphen as the final argument, DB-Access processes commands from the standard input device (on UNIX platforms) or the keyboard (on Windows NT platforms). DB-Access reads what you type until you indicate that the input is complete. Then DB-Access processes your input and writes the results to the standard output device (on UNIX platforms), or the command window (on Windows NT).

Interactive Input

DB-Access reads and executes SQL statements from the terminal keyboard interactively. While DB-Access runs interactively, the *greater than* (>) prompt marks the line where you type your next SQL statement.

When you type a semicolon (;) to end a single SQL statement, DB-Access processes that statement. When you press CTRL-D to end the interactive session, DB-Access stops running. The following example shows user input and results in an interactive session:

```
dbaccess - -
>database stores_demo;

Database selected.

>select count(*) from systables;

(count(*))

      21

1 row(s) retrieved.

>^D

dbaccess - -
>database stores_demo;

Database selected.

>select count(*) from systables;

(count(*))

      21

1 row(s) retrieved.

>^D
```

UNIX

Batch Command Input on UNIX Platforms

You can use an in-line shell script to supply one or more SQL statements. For example, you can use the UNIX C, Bourne, or Korn shell with in-line standard input files:

```
dbaccess mystores- <<EOT!  
select avg(customer_num) from customer  
where fname matches '[A-G]*';  
EOT!
```

You can use a pipe to supply SQL statements:

```
echo 'select count(*) from systables' | dbaccess mystores
```

DB-Access interprets any line that begins with an exclamation mark (!) as a shell command. You can mix shell escape lines with SQL statements and put them in SQL statements, as follows:

```
dbaccess mystores  
>select  
!echo hello  
>hello  
count(*) from systables;  
>  
(count(*))  
  
21  
  
1 row(s) retrieved.  
>
```

Connecting to a Database Environment in Non-Menu Mode

You can use the CONNECT ... USER syntax in SQL statements that you issue in interactive mode. However, DB-Access does not support the USER clause of the CONNECT statement when you connect to a default database server.

Connecting in Interactive Non-Menu Mode

When you include the USER 'user identifier' clause in a CONNECT statement in interactive mode, DB-Access prompts you to enter a password. You can either enter a user identifier or press the RETURN key. If you enter a user identifier, follow the syntax guidelines described in [Informix Guide to SQL: Syntax](#). If you enter a password, it does not appear on the screen.

The following two command examples show how to connect to a database server in interactive mode. The first example uses the CONNECT statement without specifying a user identifier.

```
dbaccess - -
> connect to '@starfish';
Connected.
```

If you include the USER clause in a CONNECT statement, as the second example shows, DB-Access uses echo suppression to prompt you for a password:

```
> connect to '@starfish' user 'marae';
ENTER PASSWORD:
Connected.
```



Important: For security reasons, do not enter the password on the screen where it can be seen. Also, do not include the USING password clause in a CONNECT statement when you use DB-Access interactively. If you are in interactive mode and attempt to enter a password before the prompt, an error message appears.

Connecting with a File or Shell File in Background Mode

You can execute the USER clause of a CONNECT statement in a DB-Access file that includes the USER clause. The following example uses a command file that contains a CONNECT statement with a USING clause to connect to a database server:

```
dbaccess - connfile.sql
```



Important: An SQL command file that contains the statement

```
CONNECT USER user_id USING password
```

should be protected from access by anyone other than the **user_id** that the USER clause identifies.

UNIX

The following example uses a shell file to connect to a database server. DB-Access prompts you for a password.

```
dbaccess - - <<\!
connect to '@starfish' user 'marae';
!

ENTER PASSWORD:
```



Related Manuals

For information on how to install the database server, set environment variables, and put the database server on-line, see the manuals for your database server listed in [Figure 1-4](#).

Figure 1-4
Related Manuals

Task or Feature	Topic	System	Manual
Installing the database server		All	Installation Guide
Setting environment variables	.cshrc or .login	UNIX	Installation Guide Informix Guide to SQL: Reference
Setting environment variables	Registry	Windows NT	Installation Guide Informix Guide to SQL: Reference
Setting UNIX permissions	chmod	UNIX	UNIX documentation
Putting database server on-line	oninit	All	Administrator's Guide
Creating database spaces	onspaces	All	Administrator's Guide
Granting permissions on database objects	GRANT	All	Informix Guide to SQL: Reference

Using the Full-Screen Menu Interface

In This Chapter	2-3
Using the DB-Access User Interface	2-3
Using the Keyboard with DB-Access	2-4
Using Menus	2-5
Choosing a Menu Option	2-6
Leaving a Menu Screen.	2-6
Using Response Screens	2-7
Reading the Screen Header	2-7
Selecting or Entering Values on the Screen	2-7
Leaving a Response Screen	2-8
Using the HELP Screen	2-8
Using an Editor Screen	2-9
Alternative Approaches	2-9
Database-Level Tasks.	2-9
Table-Level Tasks	2-12
Related Manuals.	2-14

In This Chapter

This chapter provides introductory information about the following topics:

- Using the DB-Access user interface
- Alternative approaches
- Related manuals

If you are familiar with DB-Access, you might prefer to skip this introductory chapter.

Using the DB-Access User Interface

The DB-Access user interface combines the following features:

- A hierarchy of menus
- Screens that prompt you for brief responses and choices from selection lists
- Contextual HELP screens
- The interactive Schema Editor, which helps you structure tables
- An SQL programmer environment, which includes the following features:
 - The built-in SQL editor where you enter and modify SQL and SPL statements
 - An option to use another editor of your choice
 - The database server syntax checker and runtime debugger
 - Storage, retrieval, and execution of SQL and SPL routines
- A choice of output for database queries and reports

Nonprogrammers build their databases with the interactive Schema Editor described in [Chapter 5, “The Table Option.”](#) Programmers use the SQL programmer environment described in [Chapter 3, “The Query-language Option.”](#)

The remainder of this chapter summarizes how to navigate through menus and screens.

Using the Keyboard with DB-Access

The terminal keyboard has special keys that you use to instruct DB-Access. Before you begin to use DB-Access, locate the following keys:

- | | |
|-----------|---|
| Arrows | <p>The arrow keys are usually found at the lower right of your keyboard.</p> <p>The up arrow [↑] key moves the cursor up one line. If your terminal does not have a [↑] key, type CTRL-K.</p> <p>The down arrow [↓] key moves the cursor down one line. If your terminal does not have a [↓] key, type CTRL-J.</p> <p>The left arrow [←] key moves the cursor back one position on the screen. If your terminal does not have a [←] key, type CTRL-H.</p> <p>The right arrow [→] key moves the cursor forward one position on the screen. If your terminal does not have a [→] key, use the CTRL-L key.</p> |
| BACKSPACE | <p>The BACKSPACE key is usually located at the top right of the keyboard and might be marked with a left arrow.</p> |
| CONTROL | <p>The CONTROL key is often labeled CTRL or CNTRL and is usually located on the left side of the keyboard. This manual refers to the CONTROL key as CTRL. On some systems, type CTRL-C to cancel or abort a program or to leave a current menu and return to the menu one level above. This manual refers to CTRL-C as the Interrupt key.</p> |

- DELETE The DELETE key is sometimes labeled RUBOUT, CANCEL, or DEL.

On most systems, use the DELETE key to cancel or abort a program or to leave a current menu and return to the menu one level above. This manual refers to the DELETE key as the DEL key or the Interrupt key.
- ESCAPE The ESCAPE key is usually located on the upper left corner of your keyboard and might be labeled ESC.
- RETURN The RETURN key is located on the right side of the keyboard. It might be labeled RETURN or NEWLINE, or it might be indicated with a bent left arrow.
- SPACEBAR The SPACEBAR is generally unlabeled.
- Interrupt Issue the command `stty -a` on the command line and check the “intr” setting to see if your terminal recognizes CTRL-C or DELETE (described earlier) or another key as the Interrupt key.

Using Menus

Menus appear at the top of each screen. Generally, the DB-Access main menu, as [Figure 2-1](#) shows, is the top level of the menu hierarchy. You can select a submenu as the top level if you invoke DB-Access with a submenu option, such as those described in [“Executing a Command File” on page 1-22](#).

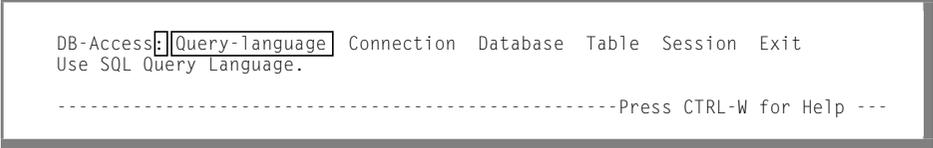


Figure 2-1
The DB-Access Main Menu

A message below the option menu briefly describes the function of the highlighted option. If you highlight another option, the description changes. To find out what an option does, simply highlight it and read the description.

The dashed line at the bottom of the header displays the name of the *current database*, if one is selected, and a reminder to press CTRL-W for Help.

Choosing a Menu Option

In [Figure 2-1](#), the box, or highlight, indicates that Query-language is the current option. Press RETURN to choose the highlighted option or choose another option as follows:

- Use the SPACEBAR or the left and right arrow keys to move the highlight. Options are arranged in a ring. If you move the highlight to the right, past the end of the list, the first option becomes current again.

Press RETURN to choose the highlighted option.

- Type the shortcut letter shown in *uppercase* in the name of the option that you want to select. (You can type the shortcut letter in either uppercase or lowercase.)

Because some menus have multiple options that start with the same letter, the shortcut key is not always the first letter of an option name. For example, in [Figure 2-2](#), the DATABASE menu, both **Create** and **cLose** start with the letter “c.” As indicated by the uppercase shortcut keys, press the C key to select the Create option and press the L key to select the cLose option..



Figure 2-2
A Sample Menu Screen

DB-Access displays the screen for the menu option that you select.

Leaving a Menu Screen

If the menu has an Exit option, press the E key to exit the menu. If no Exit option exists, use the Interrupt key (usually DEL or CTRL-C) to exit the menu. You then return to the menu, screen, or command line from which you selected the exit menu.

Using Response Screens

Some screens, such as the SELECT DATABASE screen in [Figure 2-3](#), prompt you for a name or value.

Figure 2-3
A Sample Response Screen

```
SELECT DATABASE >> 
Select a database with the Arrow keys, or enter a name, then press Return.

----- Press CTRL-W for Help -----

mystores@dbserver1
sysmaster@dbserver1
```

GLS

If Global Language Support (GLS) is enabled, the listed items are sorted according to the code-set collation order of the current locale. ♦

Reading the Screen Header

The top line of a response screen displays a prompt as the screen name followed by double angle brackets (>>) and the cursor. A message below the prompt gives brief instructions, such as `Press CTRL-W for Help`. The dashed line identifies the database that you select as current.

Selecting or Entering Values on the Screen

Where you enter your response depends on the operation, as the following two choices show:

- You might type a value in the header of an entry screen, after the double angle brackets (>>) at the top of the screen. For example, you could type a valid database name after `SELECT DATABASE>>`, as [Figure 2-3](#) shows.

Press RETURN when you finish typing, and DB-Access displays the next screen or takes other appropriate action.

- Some screens have a list on the lower part of the screen from which you can choose an item instead of typing your selection in the header. For example, the SELECT DATABASE screen in [Figure 2-3](#) lists the databases available on the current database server.

Use the arrow keys to highlight the item that you want and then press RETURN. DB-Access displays the next screen or takes other appropriate action.

Leaving a Response Screen

Without a menu, a screen cannot have an Exit option. Press the Interrupt key (usually DEL or CTRL-C) to return to the previous menu or screen.

Using the HELP Screen

Press CTRL-W to display a HELP screen that provides information about the current menu option or screen function. [Figure 2-4](#) shows some of the text that appears when you press CTRL-W for information about the Run option on the SQL menu.

```
HELP: [Screen] Resume
Displays the next page of Help text.
```

```
-----
The Run option runs the current SQL statements and displays the
output on your terminal.
```

```
If there are errors:
```

```
If there are errors, an error message will appear on the
bottom of the screen and the Modify option will be highlighted.
```

Figure 2-4
*A Partial HELP
Screen with Text*

If the Help text is longer than one page, the Screen option is highlighted. Press RETURN to view the next screen. To select the Resume option, highlight it with the SPACEBAR or the right arrow key and then press RETURN or the R key.

If the Help text is only one page, the Resume option is highlighted, and you need only press RETURN.

For an illustration of how to read the syntax diagrams that appear when you request on-line Help for creating, modifying, or editing an SQL statement, see [Appendix A](#).

Using an Editor Screen

You need to use a text editor to prepare and modify SQL statements and command files when you select the Query-language option on the main menu. DB-Access provides two ways to edit the text of SQL statements and command files before you run them: the *SQL editor* and a *system editor*. Because you only use these screens with the Query-language option, refer to [Chapter 3, “The Query-language Option,”](#) for detailed instructions.

Alternative Approaches

This section illustrates some common database tasks and lists alternative procedures to use them. This section can help you determine your preferred method of using the DB-Access interface.

For each task listed in this section, the number in the last column indicates which chapter to consult for detailed instructions. You also need the companion documents listed in [“Related Manuals” on page 2-13](#).

Database-Level Tasks

The database you use is called the *current* database. To select an existing database as current, you can:

- Use the following command line syntax:

```
dbaccess databasename
```

For more information about using DB-Access from the command line, see [“Invoking DB-Access” on page 1-13](#).

- Use any method from within DB-Access that is described in the following list.

Menu Screen	Option or SQL Statement	Action	Chapter
CONNECTION	Connect	Prompts for database server, username, password, and database name. Connects you according to the choices you make.	6
SQL	CONNECT	Connect to a database. If you use a fully-qualified database name, you simultaneously connect to the database server.	3
DATABASE	Select	Choose a current database from a list of databases available on the current database server.	4
SQL	DATABASE	Choose a database as current.	3

To create your own database, use one of the following methods.

Menu Screen	Option or SQL Statement	Action	Chapter
DATABASE	Create	Prompts for a database name, dbspace, and log options and then creates the new database.	4
SQL	CREATE DATABASE	Creates the database you name in the statement with the appropriate setup for the keywords you use.	3

To drop a database, use one of the following methods.

Menu Screen	Option or SQL Statement	Action	Chapter
DATABASE	Drop	Lists available databases and drops the database you choose from the list.	4
SQL	DROP DATABASE	Drops the database named in the command.	3

To close a database, use one of the following methods.

Menu Screen	Option or SQL Statement	Action	Chapter
DATABASE	cLose	Close the current database.	4
SQL	CLOSE DATABASE	Close the current database.	3
CONNECTION	Disconnect	Close the current database and disconnect from a database server.	6
SQL	DISCONNECT CURRENT	Close the current database and disconnect from the current database server.	3

To display information about a database, such as the dbspaces that contain it, choose the Info option on the DATABASE menu.

Table-Level Tasks

To create a table, allocate storage, or apply fragmentation, use one of the following methods.

Menu Screen	Option or SQL Statement	Action	Chapter
TABLE	Create	Use the Schema Editor.	5
SQL	CREATE TABLE	Use the SQL editor or system editor and SQL statements.	3

To make changes to the structure of a table, use one of the following methods.

Menu Screen	Option or SQL Statement	Action	Chapter
TABLE	Alter	Guides you, with menus, through the available choices for changing an existing table.	5
CREATE TABLE	Modify	Enables you to change a schema before you build the table.	5
SQL	ALTER TABLE	Changes an existing table according to the keywords you include with the statement.	3
SQL	ALTER FRAGMENT	<ul style="list-style-type: none"> ■ Changes an existing fragmentation strategy (for a table or index). ■ Creates the table fragments. 	3

To drop a table from a database, use one of the following methods.

Menu Screen	Option or SQL Statement	Action	Chapter
TABLE	Drop	Drops the current table.	4
SQL	DROP TABLE	Drops the table named in the command.	3

To display information about the structure and characteristics of a table, use one of the following methods.

Menu Screen	Option or SQL Statement	Action	Chapter
TABLE	Info	Provides a menu of available table information categories.	4
SQL	Info (option)	Prompts you to select from a list of available tables and then provides the same menu as the Info option on the TABLE screen.	3
SQL	INFO (SQL statement)	Provides the information specified in the keywords you include with the INFO statement.	3

To display the data stored in a table, run a SELECT statement from the SQL editor. Use the procedures described in [Chapter 3, “The Query-language Option.”](#)



Tip: Several command files are included with DB-Access that contain sample practice SELECT statements. [Appendix B](#) lists the demonstration files that are supplied with the **stores_demo** database.

Related Manuals

As you work with any of the following Informix companion documents, you might find reasons to use DB-Access:

- The *Informix Guide to Database Design and Implementation* introduces database-definition fundamentals for a variety of objectives, strategies, and Informix database servers.
- The *Informix Guide to SQL: Tutorial* introduces data-manipulation fundamentals and explains how to design SQL and other language applications to select, combine, report, and alter data.
- The Informix Client Software Developer's Kit contains programmer's guides for developers whose data resides on an Informix database server.

For more information on topics related to DB-Access see:

- The *Informix Guide to SQL: Syntax*, which describes the syntax for all SQL and SPL statements available through DB-Access.
- The *Informix Guide to SQL: Reference*, which contains information about:
 - the Informix system catalog and Information Schema views.
 - the structure and contents of the demonstration database.
 - environment variables.
- The *Informix Guide to GLS Functionality*, which provides information on GLS features.

The Query-language Option

In This Chapter	3-3
Overview of the SQL Menu	3-3
Using a System Editor	3-4
The Query-language Option	3-5
Entering a New SQL Statement	3-7
Using the Editing Keys	3-7
Editing Restrictions	3-8
Running an SQL Statement	3-10
Statements That the Run Option Supports	3-11
Viewing Successful Results.	3-15
What Happens When Errors Occur	3-16
Modifying an SQL Statement	3-17
Redirecting Query Results	3-19
Sending Output to a Printer	3-19
Sending Output to a File	3-19
The New-file Option	3-20
The Append-file Option	3-20
Sending Output to a Pipe	3-21

Choosing an Existing SQL Statement	3-21
Saving the Current SQL Statement	3-23
Displaying Table Information	3-24
Dropping an SQL Statement	3-26
Support for SPL Routines	3-28
Related Manuals	3-30

In This Chapter

This chapter describes how to use the Query-language option on the DB-Access main menu. When you select the Query-language option, DB-Access displays the SQL menu. Use the various SQL menu options to enter, modify, save, retrieve, and run SQL statements.

Use the Query-language option to:

- Learn SQL and SPL.
For example, use the Query-language option to practice the examples in the *Informix Guide to SQL: Tutorial*.
- Create and alter table structures as an alternative to the DB-Access Schema Editor.
- Select, display, add, update, and delete data.
DB-Access has no menu options for data manipulation.

This chapter also describes support for SPL routines.

Overview of the SQL Menu

As you use the various SQL menu options, DB-Access retains the statements, if any, in the editor. These statements are called the *current* statements.

Use the following steps to create and test SQL:

- To enter a new SQL statement or to enter multiple statements separated by semicolons, use the New option.
If you prefer to enter or modify a query with an editor of your choice, use the Use-editor option. This option provides an alternative to the New and Modify options.

- To execute your statements, use the Run option.
When you choose the Run option, a message appears or the data retrieved by a query appears with the number of rows retrieved.
To send the query results to an output file or printer instead of your terminal, use the Output option.
- If a syntax error occurs when you run the query, or if you want to change the query, use the Modify or Use-editor option to revise the text of the query.
To reexecute the query, use the Run option.
- To save the text of the query in a command file, use the Save option.
To run or modify a query that you saved in a command file, use the Choose option to select the file.
- To delete a command file in which you saved a query, use the Drop option.

Using a System Editor

When you want to enter or modify a long SQL statement or series of statements, you might prefer the flexibility and familiarity of a system editor to the SQL editor. Select the Use-editor option from the SQL menu to use the system editor.

If you have not set the **DBEDIT** environment variable, you must select a text editor to use for the session. If you select Use-editor, DB-Access prompts you to accept or override the default system editor once each session, as [Figure 3-1](#) shows.

```
USE-EDITOR >>vi
Enter editor name. (RETURN only for default editor)

----- mystores@dserver1 ----- Press CTRL-W for Help -----
```

Figure 3-1
Sample System Editor Screen for Entering and Modifying SQL Statements

UNIX

WIN NT

The default editor that DB-Access displays depends on the preference that you establish for your operating system:

- Common UNIX system editors are **vi** and **ex**. ♦
- If you use a text-formatting program, such as **Write** or **WordPad**, as the system default, you must save the **.sql** files as text. ♦

Press RETURN to select the default editor you named after the USE-EDITOR prompt. To use a different editor, type the name of that editor and press RETURN.

The Query-language Option

To select SQL, choose the Query-language option from the main menu. Press the Q key or highlight the Query-language option and press RETURN. The Query-language option displays the SQL menu, as [Figure 3-2](#) shows.

```
SQL: [New] Run Modify Use-editor Output Choose Save Info Drop Exit
Enter new SQL statements using the SQL editor.

-----Press CTRL-W for Help -----
```

Figure 3-2
The SQL Menu

If you select the Query-language option on the main menu and have not selected a database, the SELECT DATABASE screen appears. Specify a database at the prompt or press the Interrupt key to display the SQL menu. From the SQL menu, create or choose and run an SQL statement to specify the current database.



Important: In the *SELECT DATABASE* screen, the names of databases are limited to 18 characters. If a database name is longer than 18 characters, you will see the first 17 characters of the name followed by a '+' sign. Click on the '+' sign to display the complete long name in *vi*.

The SQL menu has the following options.

Option	Purpose	Instructions
New	Clears current statements and positions cursor in SQL editor.	“Entering a New SQL Statement” on page 3-7
Run	Executes current SQL statements.	“Running an SQL Statement” on page 3-10
Modify	Allows you to modify current SQL statements in SQL editor.	“Modifying an SQL Statement” on page 3-17
Use-editor	Invokes a system editor so that you can modify current statements or create new statements. Use-editor is interchangeable with New and Modify.	“Using a System Editor” on page 3-4
Output	Redirects Run-option output to a file, printer, or system pipe.	“Redirecting Query Results” on page 3-19
Choose	Lists SQL command files so that you can choose a file to execute or modify.	“Choosing an Existing SQL Statement” on page 3-21
Save	Saves current SQL statements in a file for later use.	“Saving the Current SQL Statement” on page 3-23
Info	Displays table information, such as columns, indexes, privileges, constraints, triggers, status, and fragmentation strategy.	“Displaying Table Information” on page 3-24
Drop	Deletes a specified SQL command file.	“Dropping an SQL Statement” on page 3-26
Exit	Returns to main menu.	none

Entering a New SQL Statement

To enter an SQL statement, perform one of the following actions:

- Select the New option on the SQL menu. The NEW screen appears and indicates that you are using the SQL editor, as [Figure 3-3](#) shows.

```

NEW:  ESC   = Done editing      CTRL-A = Typeover/Insert   CTRL-R = Redraw
      CTRL-X = Delete character  CTRL-D = Delete rest of line

----- mystores@dbsvrer1 ----- Press CTRL-W for Help -----
  
```

Figure 3-3
The NEW Screen for Entering New SQL Statements

- If you prefer to work with a different editor from the one that is installed on your system, select the Use-editor option. This option is described in [“Using a System Editor”](#) on page 3-4.

As [Figure 3-3](#) shows, the NEW screen starts with the cursor positioned below the header, which indicates where you enter text on the screen. Use the editor to enter statements and edit them before you run them. To string several SQL statements together, separate them with a semicolon.

Using the Editing Keys

The editing keys listed at the top of the SQL editor screen perform the following special functions when you enter SQL statements:

- CTRL-A switches between *insert* mode and *typeover* mode. You are automatically in typeover mode when you begin to use the SQL editor.
- In insert mode, the text beneath the cursor shifts to the right as you enter new characters.
 - In typeover mode, characters you enter replace the text beneath the cursor.
- CTRL-D deletes characters from the current cursor position through the end of the line.

- CTRL-R redraws the current screen. Use it when you receive an electronic message or some other interference that makes it difficult to read the SQL statement text that you enter.
- CTRL-X deletes a character beneath the cursor.
- ESC returns you to the SQL menu when you finish entering or editing the SQL statement. You can then run or modify the statement or select the system editor for more extensive editing.

For more information on arrow and cursor-positioning keys, see [“Using the Keyboard with DB-Access” on page 2-4](#).

Editing Restrictions

The SQL editor does not display more than 80 characters on a line and does not wrap lines, as the following two such restrictions show:

- If you choose an existing command file in which the characters in a line extend beyond the 80th column, DB-Access displays a percent sign (%) in the 80th column to indicate an overflow. You cannot see all the characters beyond the percent sign, but the statement should run correctly.
- If you type characters in a new command file so that a line extends beyond the 80th column, DB-Access overwrites all the characters in the 80th column. You cannot see the overflow, and the statement does not run correctly.

To make the full text appear on the screen, press RETURN at a logical place in the first 80 characters of each line. If you need to type a quoted character string that exceeds 80 characters, such as an insert into a long CHAR column, use a system editor instead of the SQL editor.

If you want to include comments in the text:

- Use double dashes for ANSI-compliant databases.
- Preface each comment line with a double dash (--) comment indicator. The comment indicator spans the entire line.
- Use curly brackets ({ }) for databases that are not ANSI-compliant. Enclose the entire comment indicator between the curly brackets.

When you use the SQL editor, you can type as many lines of text as you need. You are not limited by the size of the screen, although you might be limited by the memory constraints of your system or the maximum SQL statement size of 64 kilobytes.

If you insert more lines than one screen can hold, the SQL editor scrolls down the page with the additional text. The beginning and ending line numbers of the current page are displayed on the fourth line of the text-entry screen, as [Figure 3-4](#) shows.

```
NEW:   ESC    = Done editing      CTRL-A = Typeover/Insert   CTRL-R = Redraw
        CTRL-X = Delete character  CTRL-D = Delete rest of line

-- 3 to 20 of 20 ----- mystores@dbserver1 ----- Press CTRL-W for Help -----
```

Figure 3-4
SQL Statement
Text-Entry
Screen with
Scrolling



When you finish entering a new SQL statement or statements, press ESC to return to the SQL menu.

Warning: *If you do not use the Save option to save your typed statements, they will be deleted the next time you select an option that clears the SQL editor (such as New or Choose).*

Running an SQL Statement

After you exit the editor screen, the SQL menu reappears with the Run option highlighted and the statement text displayed in the bottom of the screen, as [Figure 3-5](#) shows.

```
SQL: [ ] New [Run] Modify Use-editor Output Choose Save Info Drop Exit
Run the current SQL statements.

----- mystores@dbserver1 ----- Press CTRL-W for Help -----
-

CREATE TABLE mystock
(
  stock_num      SMALLINT,
  manu_code      CHAR(3),
  description    CHAR(15),
  unit_price     MONEY(6),
  unit           CHAR(4),
  unit_descr    CHAR(15),
  PRIMARY KEY (stock_num, manu_code) CONSTRAINT stock_man_primary,
  FOREIGN KEY (manu_code) REFERENCES manufact
)
```

Figure 3-5
The SQL Menu with SQL Statement Text Ready to Run

Press RETURN or the R key to select the Run option. DB-Access first checks each statement to ensure that it conforms to the SQL syntax and usage rules. If your statements contain no syntax mistakes, DB-Access processes them.



Tip: You can check your SQL statements for ANSI compatibility if you set the `DBANSIWARN` environment variable or invoke DB-Access with the `-ansi` option.

Statements That the Run Option Supports

Figure 3-6 lists the statements that you can execute with the Run option.

Figure 3-6
SQL Statements for Run Option

SQL Statement	Options	
	XPS	IDS
ALTER FRAGMENT	✓	✓
ALTER INDEX		✓
ALTER TABLE	✓	✓
BEGIN WORK	✓	✓
CLOSE DATABASE	✓	✓
COMMIT WORK	✓	✓
CONNECT	✓	✓
CREATE CAST		✓
CREATE DATABASE	✓	✓
CREATE DISTINCT TYPE		✓
CREATE DOMAIN	✓	✓
CREATE FRAGMENT	✓	✓
CREATE INDEX	✓	✓
CREATE OPAQUE TYPE		✓
CREATE OPCLASS		✓
CREATE FUNCTION		✓
CREATE PROCEDURE	✓	✓
CREATE ROLE		✓
CREATE ROW TYPE		✓

(1 of 4)

Statements That the Run Option Supports

SQL Statement	Options	
	XPS	IDS
CREATE SCHEMA	✓	✓
CREATE SYNONYM	✓	✓
CREATE TABLE	✓	✓
CREATE TRIGGER		✓
CREATE VIEW	✓	✓
DATABASE	✓	✓
DELETE	✓	✓
DISCONNECT	✓	✓
DROP CAST		✓
DROP DATABASE	✓	✓
DROP DISTINCT TYPE		✓
DROP DOMAIN	✓	✓
DROP INDEX	✓	✓
DROP OPAQUE TYPE		✓
DROP OPCLASS		✓
DROP FUNCTION		✓
DROP PROCEDURE	✓	✓
DROP ROLE		✓
DROP ROW TYPE		✓
DROP SYNONYM	✓	✓
DROP TABLE	✓	✓
DROP TRIGGER		✓

(2 of 4)

SQL Statement	Options	
	XPS	IDS
DROP VIEW	✓	✓
EXECUTE PROCEDURE	✓	✓
EXECUTE FUNCTION		✓
GRANT	✓	✓
GRANT FRAGMENT		✓
INFO	✓	✓
INSERT	✓	✓
LOAD	✓	✓
LOCK TABLE	✓	✓
OUTPUT	✓	✓
RENAME COLUMN	✓	✓
RENAME DATABASE	✓	✓
RENAME TABLE	✓	✓
REVOKE	✓	✓
REVOKE FRAGMENT		✓
ROLLBACK WORK	✓	✓
SELECT	✓	✓
SET	✓	✓
SET DATASKIP	✓	✓
SET DEBUG FILE TO	✓	✓
SET EXPLAIN	✓	✓
SET ISOLATION	✓	✓

(3 of 4)

Statements That the Run Option Supports

SQL Statement	Options	
	XPS	IDS
SET LOCK MODE	✓	✓
SET LOG	✓	✓
SET OPTIMIZATION	✓	✓
SET PDQPRIORITY	✓	✓
SET OPTIMIZATION	✓	✓
SET TRANSACTION	✓	✓
START VIOLATIONS TABLE	✓	✓
STOP VIOLATIONS TABLE	✓	✓
UNLOAD	✓	✓
UNLOCK TABLE	✓	✓
UPDATE	✓	✓
UPDATE STATISTICS	✓	✓

(4 of 4)

For information about additional statements for Optical Subsystem, see the [Guide to the Optical Subsystem](#).



Tip: To execute statements that are not listed, use the SQL menu options New (or Use-editor) and Save to enter and save them, and then run the saved file from the command line.

Viewing Successful Results

If a statement other than a query completes successfully, a message appears at the bottom of the screen. For the statement that [Figure 3-5](#) shows, the following message appears:

```
Table created.
```

If you use the Run option with a SELECT statement and that SELECT statement runs correctly, DB-Access displays the requested results below the header. If your query retrieves more rows than can fit on a single screen, the results screen has a menu at the top, as [Figure 3-7](#) shows.

Figure 3-7

The Menu for Displaying Additional Query Results

```
RUN: Next Restart Exit
Display the next page of query results.

----- mystores@dbserver1 ----- Press CTRL-W for Help -----

customer_num 106
call_dtime   1997-06-12 08:20
user_id      maryj
call_code    D
call_descr   Order was received, but two of the cans of ANZ tennis balls within
              the case were empty
res_dtime    1997-06-12 08:25
res_descr    Authorized credit for two cans to customer, issued apology. Called
              ANZ buyer to report the QA problem.

customer_num 110
call_dtime   1997-07-07 10:24
user_id      richc
call_code    L
call_descr   Order placed one month ago (6/7) not received.
res_dtime    1997-07-07 10:30
res_descr    Checked with shipping (Ed Smith). Order sent yesterday- we were
              waiting for goods from ANZ. Next time will call with delay if
              necessary.
```

Advance through the output as follows:

- With the Next option highlighted, press RETURN to advance the display through the selected rows. The last screen of data has a message at the bottom that indicates the number of rows that the query returned.

- Select Restart to return to the first screen of query output.
- Select Exit to exit the output display and return to the SQL menu. If you exit before you reach the last output screen, a message at the bottom of the screen indicates that you interrupted the query.

What Happens When Errors Occur

If you make any syntax or typing mistakes in an SQL statement, DB-Access does not process the statement. Instead, it continues to display the text of the statement with a message that describes the error. For example, if a syntax error occurs, the following message appears at the bottom of the screen:

```
201: A syntax error has occurred.
```

If an execution or runtime error occurs, DB-Access continues to process the statement and returns an error message. For example, if you try to create a table that was already created, the following message appears at the bottom of the screen:

```
310: Table (mavis.mystock) already exists in database.
```

If you try to execute a statement that contains more than one SQL statement, you might not see the error message immediately. If, for example, the first statement is a SELECT statement that runs correctly and the next statement contains a typing error, the data that the first statement retrieved appears on the screen before the error message appears for the second statement.

When DB-Access detects an error, it gives you an opportunity to edit the statement that caused the error. Processing stops when the Modify option on the SQL menu is highlighted. Select one of the following methods to correct the statement:

- Press RETURN to choose Modify, which returns you to the SQL editor.
- Select the Use-editor option to use the default editor of your choice.

Modifying an SQL Statement

When DB-Access finds an error in an SQL statement that you are trying to run, the **Modify** option is highlighted on the SQL menu, and the current statement text and error message are displayed, as [Figure 3-8](#) shows.

Figure 3-8

The SQL Menu with SQL Statement Text to Be Modified

```
SQL: [ ] New Run [Modify] Use-editor Output Choose Save Info Drop Exit
Modify the current SQL statements using the SQL editor.

----- mystores@dbserver1 ----- Press CTRL-W for Help -----

CREATE TABLE mystock
(
  stock_num      SMALLINT,
  manu_code      CHAR(3),
  description     CHAR(15),
  unit_price     MONEY(6),
  unit           CHAR(4),
  unit_descr     CHAR(15),
  PRIMARY KEY (stock_num, manu_code) CONSTRAINT stock_man_primary,
  FOREIGN KEY (manu_code) REFERENCES manufact
)
```

```
201: A syntax error has occurred.
```

If you press RETURN, DB-Access calls the SQL editor and positions the cursor on the line with the first error. You can correct the error with the SQL editor, or you can press ESC to exit to the SQL menu and select the Use-editor option to edit the statement with your system editor. To exit, perform one of the following steps:

- If you use the SQL editor to make changes, press ESC when you finish editing the statement.
- If you use the system editor to make changes, exit the file according to the convention for that editor.

You then return to the SQL menu, where you can press RETURN to run the statement again.

If the SQL statement is new, the screen is blank. If you want to change or call up this statement with the Choose option, the text of the SQL statement appears on the screen.

If an error occurs while you run an SQL statement, the edit screen contains the error message with a pointer to the likely cause. [Figure 3-9](#) shows how an editor screen might look after a syntax error. The editor used in this example displays the name of the temporary file assigned to the SQL statement.

```
CREATE TABLE mystock
(
  stock_num      SMALLINT,
  manu_code      CHAR(3),
  description     CHAR(15),
  unit_price     MONEY(6),
  ^
#
#201:A syntax error has occurred
#
  unit           CHAR(4),
  unit_descr     CHAR(15),
  PRIMARY KEY (stock_num, manu_code) CONSTRAINT stock_man_primary,
  FOREIGN KEY (manu_code) REFERENCES manufact
)
```

"tmp/rsqa02775.err" 11 lines, 132 characters

Figure 3-9
*A Temporary
Text-Editing File
with SQL Statement
Text to Be Corrected*

Make your corrections to the text. When you finish entering or modifying your SQL statement or statements, exit the editor as you normally do. The SQL menu reappears with the Run option highlighted. The statement text appears in the bottom half of the screen.

Press RETURN to run the statement or select another menu option to save the statement in a command file or direct its output.

Redirecting Query Results

The output from a SELECT statement normally appears on the screen. You can, instead, use the Output option on the SQL menu to route query results to the printer, store them in a system file, or pipe them to a program. This option has the same purpose as the SQL statement OUTPUT.

The SELECT statement must be on the screen as the current statement. Then you can select the Output option from the SQL menu, which displays the OUTPUT menu, as [Figure 3-10](#) shows.

Figure 3-10
The OUTPUT Menu
for Redirecting
Query Results

```

OUTPUT: [Printer] New-file Append-file To-pipe Exit
Send query results to a printer.

----- mystores@dbserver1 ----- Press CTRL-W for Help -----

SELECT * FROM customer
    
```

To return to the SQL menu without redirecting query results from an OUTPUT screen, press the Interrupt key.

Sending Output to a Printer

To send your query results directly to a printer, select the Printer option from the OUTPUT menu. DB-Access sends the results to your default printer and displays a message on the bottom of the screen that indicates how many rows were retrieved. The query results do not appear on the screen. You can set the DBPRINT environment variable to specify a default printer.

Sending Output to a File

You can write query results to a new file or append the results to an existing file. If you do not specify a path when DB-Access prompts you for a filename, the file will be stored in the directory that you were in when you invoked DB-Access.

The New-file Option

To create a new file to store query results, select the New-file option on the OUTPUT menu. The OUTPUT NEW-FILE screen appears, as [Figure 3-11](#) shows.

```
OUTPUT NEW-FILE >>
Enter the name you want to assign to the new file, then press Return.

----- mystores@dbserver1 ----- Press CTRL-W for Help -----

SELECT * FROM customer
```

Figure 3-11
The OUTPUT
NEW-FILE Screen

Type a name for the file and press RETURN. DB-Access forwards the results of the query to that file and displays a message that indicates how many rows were retrieved. The query results do not appear on the screen.



Warning: If you enter the name of an existing file, this procedure overwrites the existing file with the query results.

The Append-file Option

To add your query results to the end of an existing file without replacing its contents, select the Append-file option on the OUTPUT menu. The OUTPUT APPEND-FILE screen appears, as [Figure 3-12](#) shows.

```
OUTPUT APPEND-FILE >>
Enter the name of the file you want to append results to, then press Return.

----- mystores@dbserver1 ----- Press CTRL-W for Help -----

SELECT * FROM customer
```

Figure 3-12
The OUTPUT
APPEND-FILE
Screen

Type the name of an existing file where you want to append the query results and press RETURN. DB-Access appends the query results to the end of that file and displays a message that indicates how many rows were retrieved. The query results do not appear on the screen.

Sending Output to a Pipe

If you want to send query results to a pipe, select the To-pipe option on the OUTPUT menu. The OUTPUT TO-PIPE screen appears, as [Figure 3-13](#) shows.

```
OUTPUT TO-PIPE >>
Enter the name of the pipe you want to send results to, then press Return.

----- mystores@dbserver1 ----- Press CTRL-W for Help -----

SELECT * FROM customer
```

Figure 3-13
The OUTPUT
TO-PIPE Screen

UNIX

Specify a target program, such as **more**, through which to pipe output. DB-Access sends the results to that pipe.

On a UNIX operating system, you must have permission to run the target program. ♦

Choosing an Existing SQL Statement

When you save SQL statements in a command file, as described in “[Saving the Current SQL Statement](#)” on [page 3-23](#), you can retrieve the command file and run or edit the SQL statements at any time.

Select the Choose option on the SQL menu to display the CHOOSE screen with a list of the command files that you can access. These files have the extension **.sql**, although the extension is not shown. For example, [Figure 3-14 on page 3-22](#) lists the command files included in the demonstration database.

```
CHOOSE >>
Choose a command file with the Arrow Keys, or enter a name, then press Return.

----- mystores@dbserver1 ----- Press CTRL-W for Help -----
_
[ ]
alt_cat      c_state     d_trig      sel_ojoin1
c_calls      c_stock     d_view      sel_ojoin2
c_cat        c_stores    del_stock   sel_ojoin3
c_custom     c_table     ins_table   sel_ojoin4
c_index      c_trig      opt_disk    sel_order
c_items      c_type      sel_agg     sel_sub
c_manuf      c_view1     sel_all     sel_union
c_orders     c_view2     sel_group   upd_table
```

Figure 3-14
The CHOOSE Screen
Listing Current
.sql Files

If no current database exists, the list includes all the command files located in the current directory and in any directories that the **DBPATH** environment variable specifies.



Important: This list includes only those filenames that have the **.sql** extension. If you create a new SQL file outside of DB-Access and save it without the **.sql** extension, it will not appear in the list of files to choose. Add the **.sql** extension to the filename and then select **Choose** again.

DB-Access can only recognize files that are stored in the directory from which you started DB-Access. If the **Choose** command results in an empty list, and you know you have command files, exit DB-Access, change directories to the directory that contains your **.sql** files, and restart DB-Access.

To select a command file, use the arrow keys to highlight its name or enter the name of the file at the prompt.

When the SQL menu reappears, it displays the command file statements on the screen as the current statements. To modify, run, edit, or output these statements, choose the appropriate menu option.

To leave the CHOOSE screen without selecting a command file, press the Interrupt key, which returns you to the SQL menu.

Saving the Current SQL Statement

You can save SQL statements in a file for later use, such as to invoke the statements from the command line (see “[Executing a Command File](#)” on [page 1-22](#)) or retrieve the saved statements with the Choose option on the SQL menu.

To save the current SQL statement or statements in a file, select the Save option on the SQL menu. The SAVE screen appears and prompts you to enter a name for the command file, as [Figure 3-15](#) shows.

```
SAVE >>□
Enter the name you want to assign to the command file.

-----mystores@dbserver1 ----- Press CTRL-W for Help -----

select max (ship_charge), min (ship_charge)
      from orders;
```

Figure 3-15
The SAVE Screen for
Saving Statements
in a File

You assign the left portion of the filename. Use 1 to 10 characters. Start with a letter, then use any combination of letters, numbers, and underscores (_). Press RETURN to save the file.

UNIX

You can use uppercase and lowercase letters in the name. However, remember that UNIX operating systems are case sensitive. The file **orders** is not the same as **Orders** or **ORDERS**. ♦

DB-Access appends the extension **.sql** to the name that you assign when it stores the statements in a file. For example, if you name your file **cust1**, DB-Access stores the file with the name **cust1.sql**. The CHOOSE screen still lists **cust1**, but the operating system identifies the same file as **cust1.sql** if you list the directory files from the command line.

To leave the SAVE screen without assigning a name to a command file, press the Interrupt key, and you return to the SQL menu.

Displaying Table Information

Use the Info option on the SQL menu to display information about the columns, indexes, access privileges, reference privileges, constraints (referential, primary-key, check, unique, and defaults), triggers, status, and fragmentation strategy of a table. The Info option has the same purpose as the SQL statement INFO.

The INFO menu displays the following options.

Option	Purpose	Instructions
Columns	Lists the columns in the specified table, shows the data type, and shows whether null values are allowed	“Displaying Column Information” on page 5-47
Indexes	Lists the name, owner, and type (unique or duplicate) of each index for a specified table. (The display also shows if the index is clustered, the access method used, and the names of the columns that are indexed.)	“Displaying Index Information” on page 5-51
Privileges	Lists the users who have table-level Select, Update, Insert, Delete, Index, and Alter privileges	“Displaying Table-Level Privileges” on page 5-52
References	Lists the users who have the table-level References privilege for the specified table and which columns they can reference	“Displaying References Privileges” on page 5-53
Status	Lists the table name, the table owner, the size of the row (in number of bytes), the number of rows in the table (as of the last UPDATE STATISTICS statement), the number of columns in a row, and the date the table was created	none

(1 of 2)

Option	Purpose	Instructions
cOnstraints	Displays referential, primary-key, check, and unique constraints and default values for the columns in the specified table	“Displaying Column Constraints and Defaults” on page 5-54
triGgers	Displays header and body information for a specified trigger	“Displaying Triggers” on page 5-57
Table	Redisplays the INFO FOR TABLE menu so that you can select a new table to request information on the INFO menu	none
Fragments	Displays fragmentation strategy for the selected table	“Dropping a Table” on page 5-60
Exit	Returns to the SQL menu	none

(2 of 2)

The Table option on the main menu displays a TABLE menu, which in turn has an Info option. The Info option screens are the same for both the SQL menu and TABLE menu. For more information on the Info option, see [“Displaying Table Information” on page 5-43](#).

To leave the INFO FOR TABLE screen without requesting table information, press the Interrupt key.

Dropping an SQL Statement

Your current database directory stores files that contain SQL statements. You might have installed some of these files with the demonstration database. You might have created other files and put them in the directory with the Save option on the SQL menu. Additional files might have become available when you installed a language supplement with DB-Access.

To remove command files from the current database directory, select the Drop option on the SQL menu. The DROP COMMAND FILE screen appears with an alphabetical list of command files in the current database, as [Figure 3-16](#) shows.

```
 DROP COMMAND FILE >>
 Enter the name of the sql command file you wish to drop from the database.

 ----- stores_demo@dbserver1 ----- Press CTRL-W for Help -----

```

alt_cat	c_state	d_trig	sel_ojoin1
c_calls	c_stock	d_view	sel_ojoin2
c_cat	c_stores	del_stock	sel_ojoin3
c_custom	c_table	ins_table	sel_ojoin4
c_index	c_trig	opt_disk	sel_order
c_items	c_type	sel_agg	sel_sub
c_manuf	c_view1	sel_all	sel_union
c_orders	c_view2	sel_group	upd_table
c_proc	d_proc	sel_join	

Figure 3-16
The DROP
COMMAND FILE
Screen with
Sample Files

GLS

If GLS is enabled, the order in which DB-Access lists the names of command files might vary, depending on the locale you use. ♦

Type the name of the command file that you want to drop or highlight it with the arrow keys and press RETURN. A special menu appears that asks for confirmation before it drops the command file as [Figure 3-17](#) shows.

```

CONFIRM:  No  Yes
No, I do not want to drop it.

----- stores_demo@dbserver1 ----- Press CTRL-W for Help ---
-----
alt_cat      c_state      d_trig       sel_ojoin1
c_calls      c_stock      d_view       sel_ojoin2
c_cat        c_stores     del_stock    sel_ojoin3
c_custom     c_table      ins_table    sel_ojoin4
c_index      c_trig       opt_disk     sel_order
c_items      c_type       sel_agg      sel_sub
c_manuf      c_view1      sel_all      sel_union
c_orders     c_view2      sel_group    upd_table
c_proc       d_proc       sel_join
    
```

Figure 3-17
The DROP
COMMAND FILE
Confirmation Menu

The default is No to help prevent you from deleting a command file by mistake. To drop the highlighted command file, press the Y key or use the right arrow key to highlight Yes and press RETURN. DB-Access drops the command file and returns you to the SQL menu.

To leave the DROP COMMAND FILE screen without dropping a command file, press the N key, RETURN, or the Interrupt key. You then return to the SQL menu.

Support for SPL Routines

You can create and execute routines written in SPL from the SQL menu.

You can store the SPL routine in a separate command file and then call it from an application or execute it as a stand-alone program. After you create the SPL routine, you can execute it within DB-Access with the appropriate SQL statement. The following example details the steps.

To create and execute an SQL routine

1. To create the text of the routine, type directly in the NEW screen or the Use-editor screen. Enter the SPL and SQL statements for your routine in the statement block of a CREATE PROCEDURE statement.

If you use Dynamic Server, use the CREATE FUNCTION statement if the routine returns values.

For more information on the CREATE FUNCTION statement, see the [Informix Guide to SQL: Syntax](#). ♦
2. Use the Run option to create the routine and register it in the **sysprocedures** system catalog table.
3. Use the NEW screen to enter an EXECUTE PROCEDURE statement that names the routine that you want to run.

If you use Dynamic Server and created your routine with the CREATE FUNCTION statement, enter an EXECUTE FUNCTION statement to run the function. ♦
4. Use the Run option to execute the routine and display the results.

[Figure 3-18](#) shows the text of the routine in the **c_proc.sql** command file, which is supplied with the demonstration database. To try this routine, use the Choose option and then select **c_proc**.

If you use Dynamic Server, change the word *procedure* in **c_proc.sql** to *function* because the routine returns a value. ♦

IDS

IDS

IDS

To register the routine in the database, select the Run option, as [Figure 3-18](#) shown.

Figure 3-18
Displaying the Text of an SPL Routine on the SQL Menu

```
SQL:  New Run  Modify  Use-editor  Output  Choose  Save  Info  Drop  Exit
Run the current SQL statements.
----- mydata@mynewdb ----- Press CTRL-W for Help -----
create procedure read_address (lastname char(15))
    returning char(15), char(15), char(20), char(15),char(2), char(5);
define p_fname, p_city char(15);
define p_add char(20);
define p_state char(2);
define p_zip char(5);
select fname, address1, city, state, zipcode
    into p_fname, p_add, p_city, p_state, p_zip
    from customer
    where lname = lastname;

return p_fname, lastname, p_add, p_city, p_state, p_zip;
end procedure;
```

DB-Access displays a message to indicate that the database server created the routine. To execute the routine, select New from the SQL menu and then enter the appropriate EXECUTE statement. In the following example, the user requests the address of a customer whose last name is Pauli:

```
EXECUTE PROCEDURE read_address ("Pauli")
```

If you use Dynamic Server, enter:

```
EXECUTE FUNCTION read_address("Pauli")
```

◆

IDS

After you enter the EXECUTE PROCEDURE or EXECUTE FUNCTION statement on the NEW screen, press ESC to return to the SQL menu. Select Run from the SQL menu to execute the routine. [Figure 3-19](#) shows the result of executing the routine.

Figure 3-19
Result of Executing an SPL Routine on the SQL Menu

```
SQL:  New  Run  Modify  Use-editor  Output  Choose  Save  Info  Drop  Exit
Run the current SQL statements.
----- mydata@mynewdb ----- Press CTRL-W for Help -----
Ludwig
Pauli
213 Erstwild Court
Sunnyvale
CA
94086
```



Tip: SPL routines are stored in the system catalog tables in executable format. Use the Routines option on the DATABASE INFO menu to display a list of the routines in the current database or to display the text of a specified routine.

Related Manuals

The following companion manuals provide details for Informix implementation of SQL statements and SPL programming logic:

- For tutorial information on SQL statements and SPL routines, see the [Informix Guide to SQL: Tutorial](#).
- For reference information on the syntax and usage of SQL and SPL statements, see the [Informix Guide to SQL: Syntax](#).
- For reference information on SQL system catalog tables, data types, and environment variables, see the [Informix Guide to SQL: Reference](#).
- For additional information on extended data types and user-defined routines, see [Extending Informix Dynamic Server 2000](#). ♦

The Database Option

In This Chapter	4-3
Selecting a Database Menu Option	4-4
Selecting a Database	4-5
List of Available Databases	4-5
Specifying a Database	4-6
Creating a Database	4-7
Specifying a Dbspace	4-8
Specifying Logging	4-8
Exiting the CREATE DATABASE Menu	4-9
Displaying Database Information	4-9
Retrieving Information on Dbspaces	4-11
Retrieving Nondefault Locale Information	4-11
Retrieving Information on Routines	4-12
Selecting a Different Database	4-14
Deleting a Database	4-15
The DROP DATABASE Screen	4-15
Confirming Your Decision to Delete a Database	4-16
Closing a Database	4-16
Related Manuals	4-18

In This Chapter

This chapter describes how to use the Database option. To perform any of the following actions, select the Database option from the main menu.

- Create a database or select a database.
The database you work with is called the *current* database.
- Retrieve and display information about a database, such as available dbspaces and the text of routines.
- Delete an existing database or close the current database.
- Commit or rollback transactions.

You can only access databases that reside on the current database server. To select a database server as current, you can specify a database server when you invoke DB-Access, you can use the Connection menu, or you can run a CONNECT statement from the SQL menu. If you do not explicitly select a database server, DB-Access uses the default database server that the **SINFORMIXSERVER** environment variable specifies as the current database.

Selecting a Database Menu Option

Figure 4-1 shows the DATABASE menu. To reach the DATABASE menu from the main menu, press the D key or highlight the Database option and press RETURN.

Figure 4-1
*The
DATABASE
Menu*

```
DATABASE: [ Select ] Create Info Drop cLose Exit
Select a database to work with.

----- Press CTRL-W for Help -----
```

The DATABASE menu displays the following options.

Option	Purpose	Instructions
Select	Makes a database the current database.	Figure 4-5 on page 4-8
Create	Builds a new database and makes that database the current database.	“Creating a Database” on page 4-7
Info	Displays information about the current database.	“Displaying Database Information” on page 4-9
Drop	Removes a database from the system.	“Deleting a Database” on page 4-15
cLose	Closes the current database.	“Closing a Database” on page 4-16
Exit	Exits the DATABASE menu and returns you to the main menu.	none

If you select or create a database when another database is already open, DB-Access closes that database before it makes your selection the current or new database. [Figure 4-17 on page 4-17](#) shows the TRANSACTION menu that appears if you attempt to open a new database without first terminating a transaction.

If you enter the name of a nonexistent database or a database that DB-Access cannot locate, DB-Access displays an error message.

Selecting a Database

To work with an existing database, choose the Select option from the DATABASE menu. The SELECT DATABASE screen appears, as [Figure 4-2](#) shows.

```
SELECT DATABASE >>
Select a database with the Arrow Keys, or enter a name, then press Return.

----- Press CTRL-W for Help -----
demodb@dbserver1
mydata@dbserver1
productn@factory
```

Figure 4-2
The
SELECT DATABASE
Screen



The SELECT DATABASE screen also appears whenever you need to specify a database, such as when you choose the Table or Query-language option without specifying a database on the DB-Access command line.

Important: In the SELECT DATABASE screen, the names of databases are limited to 18 characters. If a database name is longer than 18 characters, you will see the first 17 characters of the name followed by a '+' sign. Click on the '+' sign to display the complete long name in vi.

To leave the SELECT DATABASE screen and return to the DATABASE menu without selecting a database, press the Interrupt key.

List of Available Databases

When the SELECT DATABASE screen appears, the first database in the list of available databases is highlighted, accompanied by the names of database servers. The list is organized alphabetically by database server and then by database for each database server. You can display a maximum of 512 database names on the SELECT DATABASE screen.

The list of available databases that is displayed depends on the settings of certain environment variables. For example:

- DB-Access displays the names of all databases on the current database server and in your **DBPATH** setting.
- If you use multiple database servers, the **ONCONFIG** environment variable determines the current database server.

The list of available databases displayed also depends on the current connection. For example:

- If no current connection exists or the current connection is an implicit default connection, all databases in the **DBPATH** appear.
- If a current explicit connection exists, all databases in the **DBPATH** setting that pertain to the current database server appear.

Specifying a Database

You can select a database name on the **SELECT DATABASE** screen in any of the following ways:

- To select the first database on the list, which is already highlighted, press **RETURN**.
- Use the arrow keys to highlight the name of the database and press **RETURN**.
- Type the database name and press **RETURN**.

For example, to select the **demodb** database on the current database server, type **demodb** or highlight **demodb@dbserver1** and press **RETURN**.

To specify a database on another database server, include the database server with the database name. For example, to select the **productn** database on the **factory** database server, type the name or highlight **productn@factory** and press **RETURN**.

The name of the database that you select appears on the dashed line below the screen header.

Creating a Database

To create a new database instead of selecting an existing one, select the Create option from the DATABASE menu. The CREATE DATABASE screen appears, as [Figure 4-3](#) shows.

```
CREATE DATABASE >>
Enter the name you want to assign to the new database, then press Return.

----- Press CTRL-W for Help -----
```

Figure 4-3
The
CREATE DATABASE
Screen

Enter a name for the database that you want to create and press RETURN.

You can assign any name to your database, as long as you follow the syntax guidelines described in the [Informix Guide to SQL: Syntax](#).

To create a database on another database server, specify the server name with the database name. Follow the syntax guidelines described in the [Informix Guide to SQL: Syntax](#).

After you name the new database, the CREATE DATABASE menu appears as [Figure 4-4](#) shows.

```
CREATE DATABASE : [dbspace] Log Exit
Select a dbspace for storing the database's data.

----- Press CTRL-W for Help -----
```

Figure 4-4
The
CREATE DATABASE
Menu

You can specify that a database be stored in a nonroot dbspace or create a database with or without buffered transaction logging or create an ANSI-compliant database.

Specifying a Dbspace

Select the Dbspace option and the SELECT DBSPACE screen appears, as [Figure 4-5](#) shows.

Figure 4-5
The SELECT DBSPACE Screen

```
SELECT DBSPACE >>
Select a dbspace with the Arrow Keys, or enter a name, then press Return.

----- Press CTRL-W for Help -----

rootdbs
pers_dbs
empl_dbs
```

From the list of dbspaces, select an available dbspace in which to store database data. The default is **rootdbs** or a dbspace that you create as the default. You then return to the CREATE DATABASE menu.

Specifying Logging

To specify the type of transaction logging, select the Log option. The LOG menu appears, as [Figure 4-6](#) shows.

Figure 4-6
The LOG Menu

```
LOG : [None] Log Buffered_log log_Mode_ansi Exit
Do not activate transaction logging.

----- Press CTRL-W for Help -----
```

The LOG menu displays the following options.

Option	Purpose
None	Default, does not activate transaction logging.
Log	Specifies unbuffered transaction logging.
Buffered_log	Specifies buffered transaction logging.
log_Mode_ansi	Creates an ANSI-compliant database with unbuffered transaction logging.
Exit	Exits the LOG menu and returns you to the CREATE DATABASE menu.

Exiting the CREATE DATABASE Menu

When you exit the CREATE DATABASE menu, you must confirm or discard the new database, as [Figure 4-7](#) shows.

```
EXIT :  Create-new-database Discard-new-database
Create new database

----- Press CTRL-W for Help -----
```

Figure 4-7
The CREATE DATABASE Confirmation Screen

The default is Create-new-database. Press RETURN to create a database with the specified parameters, make it the current database, and return to the DATABASE menu. If you do not want to create the new database, press the D key or use the right arrow key to move the cursor to the Discard-new-database option and press RETURN. DB-Access does not create a new database.

If you exit without specifying a value for dbspace or logging, the defaults apply to the database.

Displaying Database Information

Use the Info option on the DATABASE menu to display information about dbspaces and nondefault locale settings and to read the body of routines in the current database.



Tip: This menu option provides information about the database that the SQL statement INFO does not display.

When you select the Info option on the DATABASE menu, the SELECT DATABASE screen appears, as [Figure 4-2](#) shows.

After you select a database, the DATABASE INFO menu appears, with the database you selected identified in the dashed line.

Figure 4-8 shows the DATABASE INFO menu for Extended Parallel Server and Informix Dynamic Server 2000.

```

DATABASE INFO: [dbSpace] Nls Procedures Databases Exit
Display DBSPACE information for a database.

----- demodb@dbserver1 ----- Press CTRL-W for Help -----
    
```

Figure 4-8
The
DATABASE INFO
Menu

◆
The DATABASE INFO menu displays the following options.

Option	Purpose	Instructions
dbSpace	Retrieves information about the dbspaces in the current database.	“Retrieving Information on Dbspaces” on page 4-11
NLS	Displays Native Language Support (NLS) settings for a database that supports NLS. This option is provided for compatibility with legacy database versions that support NLS.	“Retrieving Nondefault Locale Information” on page 4-11
Procedures	Lists the procedures in the current database. Select a procedure name to see the statements in the body of that procedure.	“Retrieving Information on Routines” on page 4-12
Routines	In Dynamic Server, the routines option lists the procedures and functions in the current database. Select a routine name to see the body of that routine as text.	“Retrieving Information on Routines” on page 4-12
Databases	Selects a different database about which to retrieve information.	“Selecting a Different Database” on page 4-14
Exit	Leaves the DATABASE INFO menu and returns you to the DATABASE menu.	none

To leave the DATABASE INFO menu without displaying information about the current database, press the Interrupt key to return to the DATABASE menu.

Retrieving Information on Dbspaces

To retrieve information about the dbspaces in the current database, select the dBspace option from the DATABASE INFO menu, as [Figure 4-9](#) shows.

```

DATABASE INFO: [dBspace] Nls Routines Databases Exit
Display DBSPACE information for a database.

----- demodb@dbserver1 ----- Press CTRL-W for Help -----

  Id      Name          Number ofWhen
  -----
  3      dspace2        104/28/94 N

```

Figure 4-9
The
DATABASE INFO
Menu
with Dbspaces
Information
Displayed

Retrieving Nondefault Locale Information

Global Language Support (GLS) and Native Language Support (NLS) affect the order in which lists appear in DB-Access. GLS enables the display and appropriate ordering of non-English data. Legacy database server versions used NLS for this purpose.

NLS

If the current database supports NLS, you can select the Nls option on the DATABASE INFO menu to display information about collating sequence and C CType (character classification type), as [Figure 4-10](#) shows.

```

DATABASE INFO - stores_d[no]: dBsp[ce] Nls Routines Databases Exit
Display NLS information for a database.

----- demodb@dbserver1 ----- Press CTRL-W for Help -----

fr_FR.88591 Collating Sequence
C CType

```

Figure 4-10
The
DATABASE INFO
Menu with
NLS Information
Displayed

An error message displays if the database does not support NLS or the environment variables for NLS are not properly set. ♦

GLS

DB-Access does not provide an option on the DATABASE INFO menu for displaying the GLS collating sequence and character classification type. To obtain information about the GLS locale enabled for your database server, enter the following query with the SQL editor that is described in [Chapter 3](#):

```
SELECT tabname, site FROM systables
WHERE tabid = 90 OR tabid = 91
```

The row with tabid 90 stores the COLLATION category of the database locale. The row with tabid 91 stores the CTYPE category of the database locale. [Figure 4-11](#) shows the result of the preceding query for the default U.S. English locale.

Figure 4-11
Retrieving GLS Information

```
SQL:  New  Modify Use-editor Output Choose Save Info Drop Exit
Run the current SQL statements.
----- mydata@mynewdb ----- Press CTRL-W for Help -----
tabname          site
GL_COLLATE       en_US.819
GL_CTYPE         en_US.819
2 row(s) retrieved
```

For further information on the COLLATION and CTYPE categories in a GLS locale file, see the [Informix Guide to GLS Functionality](#). ♦

Retrieving Information on Routines

Depending on the database server product that you use, select either the Procedures or Routines option on the DATABASE INFO menu to display the text from a selected routine. The SELECT PROCEDURE or SELECT ROUTINE screen appears and shows a list of SPL routines in the current database.

IDS

Figure 4-12 shows the SELECT ROUTINE screen that appears if you use Dynamic Server. The SELECT PROCEDURE screen looks the same although the prompt uses different wording for an SPL routine.

```
SELECT ROUTINE >>
Select a routine with the Arrow Keys, or enter a name, then press Return.

----- demodb@dbserver1 ----- Press CTRL-W for Help -----

read_address
```

Figure 4-12
The
SELECT ROUTINE
Screen

◆
If the routine exists in the system catalog and it fits on the DATABASE INFO menu, the text appears on the screen, as Figure 4-13 shows.

```
DATABASE INFO: read_address: dSpace Nls Routines Databases Exit
Display routine text for a selected routine.

----- demodb@dbserver1 ----- Press CTRL-W for Help -----

create procedure read_address (lastname char(15))
  returning char(15), char(15), char(20), char(15), char(2), char(5);
define p_fname, p_city char(15);
define p_add char(20);
define p_state char(2);
define p_zip char(5);
select fname, address1, city, state, zipcode
into p_fname, p_add, p_city, p_state, p_zip
from customer
where lname = lastname;

return p_fname, lastname, p_add, p_city, p_state, p_zip;

end procedure
```

Figure 4-13
The
DATABASE INFO
Menu with Text of
Selected Routine
Displayed

If the routine text does not fit on one screen, the DISPLAY menu appears with partial text, as [Figure 4-14](#) shows.

```
DISPLAY: [Next] Restart Exit
Display the next page of results.

----- demodb@dbserver1 ----- Press CTRL-W for Help -----

create procedure read_address (lastname char(15))
  returning char(15), char(15), char(20), char(15),char(2), char(5);
define p_fname, p_city char(15);
define p_add char(20);
define p_state char(2);
define p_zip char(5);
```

Figure 4-14
*The DISPLAY Menu
with Partial Routine
Text Displayed*

To display the next page of text, select the Next option. To display text from the beginning, select the Restart option.

Selecting a Different Database

To display information about a different database, select the Database option on the DATABASE INFO menu. The SELECT DATABASE screen appears and you can select a database, as described in [“Selecting a Database” on page 4-5](#). You can then use the other options of the DATABASE INFO menu, as this chapter describes.

Deleting a Database

To delete an existing database on a specified database server, select the Drop option from the DATABASE menu. The DROP DATABASE screen appears, as [Figure 4-15](#) shows.

Figure 4-15
The
DROP DATABASE
Screen

```

DROP DATABASE >> 
Enter the name of the database you wish to drop.

----- demodb@dbserver1----- Press CTRL-W for Help -----

mydata@dbserver1
demodb@dbserver1
personnel@mynewdb

```

You cannot delete the current database. The current database is the database whose name appears in the dashed line below the header of the display.

The DROP DATABASE Screen

To delete a database, use the DROP DATABASE screen in either of the following ways:

- Type the database name and press RETURN.
- Use the arrow keys to highlight the name of the database that you want to delete and press RETURN.

For example, to delete the **mydata** database, type **mydata** or highlight **mydata@dbserver1** with an arrow key and press RETURN.

To leave the DROP DATABASE screen without deleting a database, press the Interrupt key. You return to the DATABASE menu.

Confirming Your Decision to Delete a Database

When you delete a database, DB-Access displays a special menu that asks for confirmation before it deletes the database, as [Figure 4-16](#) shows.

```
CONFIRM:  No  Yes
No, I do not want to drop it.

----- demodb@dbserver1 ----- Press CTRL-W for Help -----

mydata@dbserver1
demodb@dbserver1
personnel@mynewdb
```

Figure 4-16
The
DROP DATABASE
Confirmation Menu

The default is No, which helps prevent deleting a database by mistake. If you want to delete the highlighted database, press the Y key or use the right arrow key to highlight Yes, and press RETURN. DB-Access deletes the database and all data that it contains. Be absolutely sure that you choose the correct database to delete.

Closing a Database

To close the current database, choose the cClose option from the DATABASE menu and press RETURN. The message `Database Closed` appears at the bottom of the screen, which indicates that the current database is closed. The database name disappears from the Help line, but the database server name remains.

If you select the cClose option when no database name is on the Help line, DB-Access displays an error message.

If you begin a transaction but do not commit it or roll it back, and then try to close a database with transactions, the TRANSACTION menu appears, as [Figure 4-17](#) shows.



Figure 4-17
The TRANSACTION Menu for Databases with Transactions

The TRANSACTION menu ensures that you either commit or roll back an active transaction before you close the current database. The following list shows the two menu options:

- The default option is Commit.
Press the Y key or RETURN and DB-Access commits the transactions and closes the database.
- If you want to roll back the transactions, use an arrow key to move the highlight to the Rollback option.
Press RETURN, and DB-Access rolls back the transactions and closes the database.



Warning: Select an option carefully. You might commit transactions you do not want if you select Commit, and you will lose any new transactions if you select Rollback.

If you press the Interrupt key, DB-Access displays the DATABASE menu without committing or rolling back the transactions.

The TRANSACTION menu also appears whenever you attempt to open a new database or try to leave the DB-Access menu system without first terminating a transaction. These instances are the only times when you can access the TRANSACTION menu from DB-Access.



Important: If you begin a transaction in an ANSI-compliant database but do not issue a COMMIT statement or ROLLBACK statement, then try to close the database using a non-menu mode, DB-Access will commit the transaction for you. If you do not want to commit the transaction, issue both a ROLLBACK statement and a CLOSE DATABASE statement from the command line.

Related Manuals

The following manuals contain information pertinent to database connection and creation:

- For syntax information, see the [Informix Guide to SQL: Syntax](#).
- For information about environment variables, see the [Informix Guide to SQL: Reference](#).
- For information about GLS considerations that apply to opening or connecting to a database on a different database server, see the [Informix Guide to GLS Functionality](#).

The database server examines the client locale information that a client (such as DB-Access) passes, verifies the database locale, and determines the server-processing locale for transferring data between the client and the database. ♦

The Table Option

In This Chapter	5-3
The TABLE Menu	5-3
Creating or Altering a Table	5-4
The CREATE TABLE Screen	5-4
The ALTER TABLE Screen	5-5
Using the Schema Editor	5-7
Adding Columns to a Table (Add Option)	5-8
Column Name	5-9
Column Data Type	5-9
Data Length or Range	5-11
Column Index	5-12
Column Index Fill Factor	5-13
Null Value Permission	5-13
Building the Table (Exit Option)	5-14
Modifying Columns (Modify Option)	5-15
Deleting Columns (Drop Option)	5-16
Arranging Storage and Locking (Table_options)	5-17
Selecting Dbspaces	5-18
Fragmenting a New Table	5-18
Altering Fragmentation for an Existing Table	5-23
Attaching a Dbspace	5-25
Detaching a Dbspace	5-27
Fragmenting an Existing Table	5-28
Setting the Extent Size	5-29
Determining the Lock Mode	5-30
Adding or Dropping Rowids.	5-31
Defining Constraints	5-32
Defining Primary-Key Constraints	5-32
Defining Foreign-Key Constraints	5-34

Defining Check Constraints	5-36
Defining Unique Constraints	5-38
Defining Default Values	5-39
Displaying Table Information	5-43
Displaying Column Information	5-47
DB-Access Data Types	5-47
Extended Data Types in Dynamic Server	5-49
Displaying Index Information	5-51
Displaying Table-Level Privileges	5-52
Displaying References Privileges	5-53
Displaying Column Constraints and Defaults	5-54
Displaying Referential Constraints	5-55
Sample Primary and Check Constraint Displays	5-56
Displaying Triggers	5-57
Displaying Fragmentation Information	5-59
Dropping a Table	5-60
Related Manuals	5-62

In This Chapter

This chapter describes how to use the features of the Table option on the main menu. Select this option if you want to perform any of the following table-management tasks without SQL programming:

- Create a new table
- Define fragmentation strategy for a new or existing table
- Alter, delete, or display information about an existing table

The TABLE Menu

When you press the T key or select the Table option from the main menu, the TABLE menu appears, as [Figure 5-1](#) shows.

```
TABLE: [Create] Alter Info Drop Exit
Create a new table.
-----mydata@mydbserv-----Press CTRL-W for Help -----
```

Figure 5-1
The
TABLE Menu



If no current database exists when you select the Table option, the SELECT DATABASE screen appears. Select from a list of databases defined for the current database server or press the Interrupt key to display the main menu.

Important: In the SELECT DATABASE screen, the names of databases are limited to 18 characters. If a database name is longer than 18 characters, you will see the first 17 characters of the name followed by a + sign. Click on the + sign to display the complete long name in vi.

Use the TABLE menu options as the following table shows.

Option	Purpose	Instructions
Create	Enables you to define the structure of a new table	“Creating or Altering a Table” on page 5-4
Alter	Enables you to alter the structure of an existing table	“Creating or Altering a Table” on page 5-4
Info	Displays information about the structure of a table	“Displaying Table Information” on page 5-43
Drop	Deletes a table from the database	“Dropping a Table” on page 5-60
Exit	Returns to the DB-Access main menu	none

Creating or Altering a Table

The Create option on the TABLE menu provides a menu that guides you through the functions of a CREATE TABLE statement. The Alter option provides menus that guide you through the functions of an ALTER TABLE or ALTER FRAGMENT statement.

The CREATE TABLE Screen

When you select the Create option on the TABLE menu, the CREATE TABLE screen appears, as [Figure 5-2](#) shows.

```
CREATE TABLE >>   
Enter the table name you wish to create with the schema editor.  
  
----- mydata@mydbserv ----- Press CTRL-W for Help -----
```

Figure 5-2
The
CREATE TABLE
Screen

At the prompt, type the name of the new table and press RETURN. You can assign any name to the table, as long as you follow the syntax guidelines for naming database objects described in the [Informix Guide to SQL: Syntax](#).

After you enter the new table name, the CREATE TABLE menu appears, as [Figure 5-3](#) shows.

Figure 5-3
The CREATE TABLE Menu

```
CREATE TABLE clients:  Modify Drop Screen Table_options Constraints Exit
Adds columns to the table above the line with the highlight.

---- Page 1 of 1 ---- mydata@mydbserv ----- Press CTRL-W for Help -----

Column Name          Type          Length  Index Nulls

```

The ALTER TABLE Screen

When you select the Alter option on the TABLE menu, the ALTER TABLE prompt and a list of the tables defined in the current database appear, as [Figure 5-4](#) shows.

Figure 5-4
The
ALTER TABLE
Prompt

```
ALTER TABLE >> 
Enter the table name you wish to alter with the schema editor.

----- mydata@mydbserv ----- Press CTRL-W for Help -----


items
orders
```

Enter the name of the table that you want to alter after the prompt or use the arrow keys to highlight the table name in the list. After you press RETURN, the ALTER TABLE menu and the table schema appear, as [Figure 5-5](#) shows.

```
ALTER TABLE clients:  Modify Drop Screen Table_options Constraints Exit
Adds columns to the table above the line with the highlight.

---- Page 1 of 1 ---- mydata@mybserv ----- Press CTRL-W for Help -----

Column Name          Type          Length  Index  Nulls
-----
customer_num         Serial        101     Unique No
fname                Char          15      Yes
lname                Char          15      Yes
company              Char          20      Yes
```

Figure 5-5
The
ALTER TABLE
Menu



Important: You must have the Alter privilege to successfully alter a table. Without the privilege, you can use the menus from the ALTER TABLE screen, but an error results when you attempt to select Build-new-table from the EXIT screen. For references explaining the Alter privilege and other table-level privileges, see [page 5-3](#).

To use the LOAD statement to insert data into a table, you must have both Insert and Select privileges for the table. You need the Select privilege because DB-Access must read SELECT information about the columns before inserting data into the table. If you do not have the Select privilege, the LOAD command fails and you get error message -272, “No SELECT permission.” If you have the Select but not the Insert privilege, you get error -275, “No INSERT permission.”

Using the Schema Editor

Both the CREATE TABLE and ALTER TABLE menus have the same options, which are described in the following list.

Option	Purpose	Instructions
Add	Displays the Schema Editor, from which you can add a new column to the table	“Adding Columns to a Table (Add Option)” on page 5-8
Modify	Displays the columns that you defined with the Add option so that you can modify the column structure before building the table	“Modifying Columns (Modify Option)” on page 5-15
Drop	Drops an existing column from the table	“Deleting Columns (Drop Option)” on page 5-16
Screen	Displays the next screen of column definitions in the Schema Editor	none
Table_ options	Displays and enables you to select storage spaces for a new table Displays choices from which you set a fragmentation strategy for a new table Enables you to set extent sizes and lock mode for a new table Adds or deletes rowids for an existing fragmented table	“Arranging Storage and Locking (Table_options)” on page 5-17
Constraints	Enables you to define primary-key, foreign-key, check, and unique constraints Enables you to set default column values	“Defining Constraints” on page 5-32 “Defining Default Values” on page 5-39
Exit	Builds, rebuilds, or discards the schema and structure that you described with the other options and then returns to the TABLE menu	“Building the Table (Exit Option)” on page 5-14



Important: You must use the SPACEBAR to move between menu options because the arrow keys control cursor movement in the Schema Editor.

To leave the CREATE TABLE menu or ALTER TABLE menu and return to the TABLE menu without building or altering a table, press the Interrupt key.

Adding Columns to a Table (Add Option)

To add a new column to a table, you define a new line in the Schema Editor that appears below the dashed line. When you create or alter the columns of a table, the Schema Editor issues prompts to assist you.



Important: Before you use the Add option from the ALTER TABLE menu, you must position the highlight in the Schema Editor to indicate where you want to insert the new column or columns. To move the highlight within the displayed columns, use the up and down arrow keys. To scroll more of the column list onto the screen, use the Screen option on the menu. When you select the Add option, the highlighted line moves down to make an empty line for the new column.

The Schema Editor progresses from left to right, completing one horizontal line of description for each column, with the name of the column at the left. Use the right arrow key to move the highlight to each field. To accept the default entry for each field, press RETURN or an arrow key.

As you finish one column, the cursor moves to the next line down, so that you can type another column name. Thus, the columns that make up the table are listed vertically.

You can change or bypass any field entry in a line before you move to the next line in either of the following ways:

- Use the left arrow key to move back to a field that you passed on the current line.
- Press the Interrupt key to cancel a prompt without inserting a value into the current (highlighted) field.

After you move the cursor to another line, you must use the Modify option on the CREATE TABLE menu to change your entry, as “[Modifying Columns \(Modify Option\)](#)” on page 5-15 describes.

Column Name

The Add option on the CREATE TABLE (or ALTER TABLE) menu places the cursor on an empty line and displays the ADD COLUMN NAME prompt. Type the name of the column after the ADD COLUMN NAME prompt and press RETURN. You can assign any name, as long as you follow the identifier syntax guidelines described in the [Informix Guide to SQL: Syntax](#).

DB-Access enters the specified name under Column Name, as [Figure 5-6](#) shows.

```

ADD COLUMN NAME >>
Enter column name. RETURN adds it. INTERRUPT returns to CREATE/ALTER menu.

---- Page 1 of 1 ---- mydata@mydbserv ----- Press CTRL-W for Help ----

Column Name                Type                Length  Index Nulls
customer_num
    
```

Figure 5-6
The ADD COLUMN NAME Screen with Column Name Entered

Column Data Type

The ADD TYPE menu displays various data types, as [Figure 5-7](#) shows.

```

ADD TYPE clients : Char Numeric Serial Date Money date-Time ...
Permits any combination of letters, numbers, and punctuation.

---- Page 1 of 1 ---- mydata@mydbserv ----- Press CTRL-W for Help ----

Column Name                Type                Length  Index Nulls
customer_num                
    
```

Figure 5-7
The ADD TYPE Menu for Defining Column Data Types

To select the data type for the column, type the first capitalized letter of the data type, using either uppercase or lowercase letters or the SPACEBAR to highlight it and then press RETURN.

Important: Use the spacebar to move to your choice. Use the arrow keys to control cursor movement in the lower part of the screen.



IDS

The CREATE TABLE menu provides options for built-in data types. To define a column with one of the extended data types, such as smart large objects, user-defined (opaque) data types, or a collection data type, use the SQL menu to enter and run a CREATE TABLE statement. ♦

If you select one of the following data type categories from the ADD TYPE menu, DB-Access displays one or two submenus for that category.

ADD TYPE Category	Data Type Submenu	Additional Submenu
Numeric	Integer	
	Smallint	
	Decimal	
	Float	Smallfloat or Float
Char	Char (press C to select)	
	Nchar (press N to select)	
Variable-length	Varchar	Varchar (press V to select) Nvarchar (press N to select)
	Text or Byte	Table
		Blobspace



Tip: Although some data types described above are not included in the menu mode, you can use any data types in interactive, non-menu mode.

Locale Character Data

If you use character data in a default locale, select Char for fixed-length data or Varchar if the table will have varying-length entries in that column.

If you use a nondefault locale, select Nchar for fixed length or Nvarchar for varying length. ♦

GLS

XPS

Large Object Storage Location

If you select VARIABLE-LENGTH TEXT or BYTE data type, perform one of the following actions to indicate where that large-object data should reside:

- Select Table to store the full data directly in the column.
Always select Table for databases that reside on Extended Parallel Server. ♦
- Select BlobSpace to store the actual TEXT or BYTE data in a large-object space (blobSpace). The table column then holds the blobSpace location.

DB-Access displays the SELECT BLOBSPACE screen, as [Figure 5-8](#) shows. Use the arrow keys to choose a blobSpace from the alphabetical list or type the blobSpace name at the top of the screen.

```
SELECT BLOBSPACE >>
Select a blobSpace with the Arrow Keys, or enter a name, then press Return

----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----

cust_b1ob
```

Figure 5-8
The SELECT BLOBSPACE Screen for Storing Variable-Length Data

Data Length or Range

If you select any of the following data types for the column, a new ADD screen appears. Enter the appropriate information in the Length field.

Data Type	Length or Range
Char	Enter length (the default is 20).
Nchar	Enter length (the default is 20).
Numeric	For the fixed-point form of the DECIMAL type, enter the precision and scale (the default is 16, 2). For the floating-point form of the DECIMAL type, enter the precision only.
Serial	Enter the starting number (the default is 1).
Money	Specify a length (the default is 16, 2).

(1 of 2)

Adding Columns to a Table (Add Option)

Data Type	Length or Range
Datetime	Specify first to last datetime qualifiers.
Interval	Specify first to last interval qualifiers.
Varchar	Specify a maximum length (from 1 to 255 bytes) and a minimum space (from 0 to 255 bytes).
Nvarchar	Specify a maximum length (from 1 to 255 bytes) and a minimum space (from 0 to 255 bytes).

(2 of 2)

Column Index

DB-Access can construct only a nonclustered, ascending B-tree column index. Select the Yes option to create this type of index with the ADD INDEX menu, as [Figure 5-9](#) shows.

```
ADD INDEX      clients :  Yes  No
Specifies that this column will NOT have an index.

----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----

Column Name          Type          Length  Index Nulls
customer_num         Serial        101     
```

Figure 5-9
The
ADD INDEX
Menu

DB-Access displays an ADD DUPLICATES screen. Press RETURN or the Y key to allow duplicate values, and the word Dups appears in the Index field. Press the N key to prevent duplicate values. The word Unique appears in the Index field.

DB-Access displays the ADD FILL FACTOR PERCENTAGE screen, as [Figure 5-10](#) shows.

If you do not want to index the values in this column or if you want any other type of index, such as an R-tree index, select the No option. You must create an R-tree index directly with SQL.

Column Index Fill Factor

Use the ADD FILL FACTOR PERCENTAGE screen, as [Figure 5-10](#) shows, to set the fill-factor percentage when you create an index on a single column. The index column has fill factor and Unique or Dups abbreviated to U or D.

Figure 5-10
The
ADD FILL FACTOR
PERCENTAGE
Screen

```

ADD FILL FACTOR PERCENTAGE >>
Enter the fill factor percentage. RETURN adds it.

----- Page 1 of 1 -----personnel ----- Press CTRL-W for Help -----

Column Name                Type                Length  Index  Nulls
empl_num                   Integer              20      U 70%  No
last_name                  Char                 20      D 90%  No
insurance                  Integer              20      Dups   Yes
ss_num                     Integer              20      Unique No
    
```



Important: You can only set a fill-factor value when you create a new index. You can modify the fill factor through the Modify option on the CREATE TABLE menu. However, you cannot alter it through the ALTER TABLE menu after the table for the index is created.

Enter any positive value to a maximum of 100. A value less than 1 or greater than 100 results in an error.

If you press RETURN without entering a value, the index will have the fill-factor percentage set in the database server ONCONFIG file. If ONCONFIG has no fill-factor setting, the index will have the default fill-factor value of 90 percent.

Null Value Permission

Specify whether the column allows null values on the ADD NULLS menu, as [Figure 5-11](#) shows.

Figure 5-11
The ADD NULLS
Menu

```

ADD NULLS    clients :  Yes  No
Permits null values in this column.

----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----

Column Name                Type                Length  Index Nulls
customer_num              Serial              101     Unique 
    
```

Select **Yes** to allow null values in the column or **No** to force the column to always have a non-null value.

To add another column definition to the table or return to the CREATE TABLE menu, press **Interrupt**, an arrow key, or **RETURN**.

Building the Table (Exit Option)

When you complete the schema for the new (or modified) table, select the **Exit** option on the CREATE TABLE (or ALTER TABLE) menu. DB-Access displays the menu, as [Figure 5-12](#) shows.

```
CREATE TABLE clients: [Build-new-table] Discard-new-table
Builds a new table and returns to the Table Menu.

----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----

Column Name          Type          Length  Index  Nulls
customer_num         Serial        101     Unique No
fname                Char          15      Yes
lname                Char          15      Yes
company              Char          20      Yes
address1              Char          20      Yes
address2              Char          20      Yes
city                  Char          15      Yes
state                 Char           2      Yes
zipcode               Char           5      Dups  Yes
phone                 Char          18      Yes
```

Figure 5-12
The
CREATE TABLE
Menu

To create the table that contains the displayed columns and return to the TABLE menu, select **Build-new-table**. To return to the TABLE menu without saving the new or modified table definition, select **Discard-new-table**.

Modifying Columns (Modify Option)

To modify an existing column, perform the following steps:

1. Use the arrow keys to highlight the column definition that you want to modify.
2. If necessary, select Screen from the CREATE TABLE menu to display the next screen of column definitions in the Schema Editor.
3. Select the Modify option on the CREATE TABLE (or ALTER TABLE) menu and press RETURN.
4. Move the highlight to the field you want to modify.

DB-Access prompts appropriately for the field where the highlight is located. Each of these prompt screens works like the corresponding ADD screen.

For instructions on specific prompts, see the following table.

Prompt	Instructions
MODIFY COLUMN NAME	“Column Name” on page 5-9
MODIFY TYPE	“Column Data Type” on page 5-9
MODIFY LENGTH	“Data Length or Range” on page 5-11
MODIFY INDEX	“Column Index” on page 5-12
MODIFY NULLS	“Null Value Permission” on page 5-13

5. Move the highlight to the next field and repeat the process.
6. Select Exit to leave the screen after you have elected to build the table or discard the schema, as [Figure 5-12 on page 5-14](#) shows.

To leave a Modify screen or menu without making any changes, press the Interrupt key at any time.

Deleting Columns (Drop Option)

To delete a column from a table schema, perform the following steps:

1. Position the highlight anywhere on the column that you want to delete.
2. Select the Drop option on the CREATE TABLE (or ALTER TABLE) menu.

The column line is then partially or completely highlighted on the screen.

DB-Access displays the DROP menu that prompts you to verify your decision, as [Figure 5-13](#) shows.

Figure 5-13

The DROP Menu

```
DROP clients :  Yes No
Deletes the highlighted column from the table.

---- Page 1 of 1 ---- mydata@mydbserv ----- Press CONTROL-W for Help ----

Column Name          Type          Length  Index Nulls
-----
customer_num         Serial        101     Unique No
fname                Char          15      Yes
lname                Char          15      Yes
company              Char          20      Yes
```

Select Yes from the DROP menu to delete the line currently highlighted in the Schema Editor; select No to keep the line.

Arranging Storage and Locking (Table_options)

To display the TABLE_OPTIONS menu, as [Figure 5-14](#) shows, select Table_options from the CREATE TABLE menu (or ALTER TABLE menu). You can then specify storage-management parameters, such as location and distribution of data on the storage media.

Figure 5-14
The
TABLE_OPTIONS
Menu

```
TABLE_OPTIONS clients: [Storage]eXtent_size Next_size Lock_mode Exit
Define dbspace or fragmentation strategy for table storage.

----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----

Column Name          Type          Length  Index Nulls
-----
```

The TABLE_OPTIONS menu contains the following options.

Option	Purpose	Instructions
Storage	Displays dbspaces and enables you to assign a dbspace to the current table	“Selecting Dbspaces” on page 5-18
	Enables you to define a fragmentation strategy for the current table	“Fragmenting a New Table” on page 5-18
eXtent_size	Enables you to specify the initial extent size of the table	“Setting the Extent Size” on page 5-29
Next_size	Enables you to specify the next extent size	“Setting the Extent Size” on page 5-29
Lock_mode	Enables you to select either Page or Row as the lock mode	“Determining the Lock Mode” on page 5-30
Exit	Returns to the previous menu	none

For assistance in setting Table_options values such as dbspaces, fragmentation strategy, extent sizes, and lock mode, see your [Administrator’s Guide](#) and [Performance Guide](#).

Selecting Dbspaces

To display the STORAGE menu, as [Figure 5-15](#) shows, select the Storage option from the TABLE_OPTIONS menu.

```
STORAGE new_acct:  Fragment Exit
Select a dbspace in which to store the table.

----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----
```

Figure 5-15
The STORAGE Menu for Storing Table Data and Defining Fragmentation Strategy

To display the SELECT DBSPACE screen, as [Figure 5-16](#) shows, select Dbspace from the STORAGE menu. Use the arrow keys to highlight a dbspace from the list of dbspaces in the current database and then press RETURN.

```
SELECT DBSPACE >>
Select a dbspace with the Arrow Keys, or enter a name, then press Return.

----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----


pers_dbs
empl_dbs
```

Figure 5-16
The SELECT DBSPACE Screen for Specifying Table Storage

Select Fragment to set up fragmentation strategy in a series of additional menus. For instructions, see [“Fragmenting a New Table” on page 5-18](#) or [“Altering Fragmentation for an Existing Table” on page 5-23](#).

Fragmenting a New Table

To arrange fragmentation for a new table, display the FRAGMENT menu, as [Figure 5-17](#) shows. You reach this menu through the following steps:

1. On the main menu, select Table.
2. On the TABLE menu, select Create.
3. On the CREATE TABLE, select Table_options.
4. On the TABLE_OPTIONS menu, select Storage.
5. On the STORAGE menu, select Fragment.

```
FRAGMENT new_acct:  eXpression rOwids Exit
Select and define a round robin fragmentation strategy.

----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----
```

Figure 5-17
The FRAGMENT Menu for Defining Fragmentation Strategy

Select the strategy that you want from the FRAGMENT menu, as the following table shows.

Option	Purpose	Instructions
Round_robin	Selects a round-robin strategy for fragmentation	“Round-Robin Setup”
eXpression	Selects an expression strategy for fragmentation	“Expression Strategy Setup” on page 5-21
rOwids	Adds a column that contains rowids to a fragmented table (The database server does not automatically assign rowids when you insert rows in a fragmented table.)	Press the O key to explicitly enable access by rowid
Exit	Exits the FRAGMENT menu and returns to the STORAGE menu	Figure 5-15 on page 5-18

DB-Access has no FRAGMENT menu option for hash or hybrid fragmentation. If you want this type of strategy, use the SQL menu to enter and run the CREATE TABLE or ALTER TABLE statement.

Extended Parallel Server does not support rowids for fragmented tables. ♦

XPS

Round-Robin Setup

To display the ROUND_ROBIN menu, as [Figure 5-18](#) shows, select the Round_robin option on the FRAGMENT menu.

```
ROUND_ROBIN new_acct: Add Modify Drop Screen Exit
Add a dbspace to the fragmentation strategy above the line with the highlight.
----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----
Dbspace Name
dbspace1
dbspace2
```

Figure 5-18
The ROUND_ROBIN Menu for Selecting Fragment Storage Spaces

The ROUND_ROBIN menu has the following options.

Option	Purpose
Add	Displays dbspaces so that you can add a new dbspace to the round-robin fragment space assigned to the current table
Modify	Enables you to redefine the fragmentation strategy for the highlighted dbspace
Drop	Deletes the highlighted dbspace from the existing strategy, but does <i>not</i> delete the dbspace from the database server
Screen	Scrolls the screen to display more of the available dbspaces
Exit	Returns to the FRAGMENT menu

Use the arrow keys to highlight a dbspace from the list on the SELECT DBSPACE screen, as [Figure 5-16 on page 5-18](#) shows, and press RETURN. If you try to add a dbspace that is already part of another strategy, an error message appears.

When you return to the ROUND_ROBIN menu, the screen displays all dbspaces currently chosen for the strategy.

Expression Strategy Setup

To display the EXPRESSION menu, as [Figure 5-19](#) shows, select the eXpression option on the FRAGMENT menu.

Figure 5-19
The EXPRESSION Menu for Defining Expression Fragmentation Strategy

```

EXPRESSION new_acct:    Modify  Drop  Screen  Exit
Add a strategy definition.

----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----

Dbospace Name  Expression
  field1 <100
dbospace2      field1 >=100 and field1 <200
dbospace3      remainder
    
```

The EXPRESSION menu has the following options.

Option	Purpose
Add	Adds a new dbospace to those that will contain fragments of this table, according to the expression fragmentation strategy
Modify	Modifies the dbospace or expression associated with that dbospace
Drop	Deletes the highlighted dbospace and expression from the existing strategy, but does <i>not</i> delete the dbospace from the database server
Screen	Scrolls the screen to display more of the available dbospaces
Exit	Returns to the FRAGMENT menu



Important: DB-Access does not perform data validation on the expression of the strategy.

The Add option on the EXPRESSION menu displays the SELECT DBSPACE screen, as [Figure 5-16 on page 5-18](#) shows. Use the arrow keys to highlight a dbospace from the list and press RETURN. If you try to add a dbospace that is already part of another strategy, an error message appears.

After you select a dbspace, the EDIT EXPRESSION menu appears, as [Figure 5-20](#) shows. From this menu you can enter an expression that determines whether a particular record belongs in the highlighted dbspace.

```

EDIT EXPRESSION new_acct: [New] Modify Use-editor Exit
Enter a new expression which will determine where a record will be stored.

----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----

Dbspace Name      Expression
-----
dbspace1          field1<100
dbspace2          field1>=100 and field1<200
dbspace3          remainder
    
```

Figure 5-20
The
EDIT EXPRESSION
Menu for Defining
and Editing
Expressions

The EDIT EXPRESSION menu has the following options.

Option	Purpose
New	Displays the blank SQL editor screen so that you can enter a new expression
Modify	Displays the current expression on the SQL editor screen so that you can modify the expression
Use-editor	Displays the current expression in the system editor so that you can modify the expression
Exit	Returns to the EXPRESSION menu

After you exit the editor, DB-Access displays the CONFIRM CHANGES menu, as [Figure 5-21](#) shows.

```

CONFIRM CHANGES dbspace1: [ ] [SAVE] DISCARD
Save changes.

-----mydata@mydbserv----- Press CTRL-W for Help -----

field1<100
    
```

Figure 5-21
The CONFIRM
CHANGES Menu

To save the edits to the expression, press RETURN. To discard the edits to the expression, select DISCARD. You return to the EXPRESSION menu.

Altering Fragmentation for an Existing Table

If you are altering a table, you arrange or remove fragmentation with the ALTER FRAGMENT menu. To reach the ALTER FRAGMENT menu:

- On the main menu, select Table.
- On the TABLE menu, select Alter.
- Select the Table_options option from the ALTER TABLE menu.
- Select the Storage option from the TABLE_OPTIONS menu.
- Select the Fragment option from the STORAGE menu.

Figure 5-22 shows how the ALTER FRAGMENT menu displays a table with a round-robin fragmentation strategy. Figure 5-23 shows how the ALTER FRAGMENT menu displays a table with an expression-based fragmentation strategy.

```
ALTER FRAGMENT - new_acct: Add Drop Screen Init aTtach detaCh Exit
Add one new dbspace to the end of the list.

----- Page 1 of 1 ----- newstores@mydbserv ----- Press CTRL-W for Help ----

Dbspace Name

dbspace1
dbspace2
dbspace3

```

Figure 5-22
The
ALTER FRAGMENT
Menu for
Round-Robin
Fragmentation
Strategy

```
ALTER FRAGMENT - new_acct: Add Modify Drop Screen Init aTtach detaCh Exit
Add an expression to the fragmentation strategy above the line with the highlight.

----- Page 1 of 1 ----- newstores@mydbserv ----- Press CTRL-W for Help ----

Dbspace Name Expression

dbspace1 field1 <100
dbspace2 field1 >=100 and field1 <200
dbspace3 remainder

```

Figure 5-23
The
ALTER FRAGMENT
Menu for
Expression
Fragmentation
Strategy

The ALTER FRAGMENT menu has the following options.

Option	Purpose	Instructions
Add	Adds a dbspace to the round-robin scheme	“Round-Robin Setup” on page 5-20 or “Expression Strategy Setup” on page 5-21 , depending on the fragmentation type of the current table
Drop	Drops one dbspace from those used for fragments of the current table	
Screen	Scrolls more dbspaces onto the screen	none
Init	Describes the fragmentation scheme (if any) of an existing table and enables you to change the strategy, attach fragments, detach fragments, or remove fragmentation	“Fragmenting an Existing Table” on page 5-28
aTtach	Attaches one or more tables to the current table, modifies or deletes a selected attachment	“Attaching a Dbspace” on page 5-25
detaCh	Detaches records from the current table from a specific dbspace and creates a new table with those records	“Detaching a Dbspace” on page 5-27
Exit	Returns to the TABLE_OPTIONS menu	none



Important: You can execute only one menu option in an ALTER FRAGMENT menu, and it can be applied to the current strategy only once. For example, you can add only one dbspace to a round-robin strategy, and you cannot delete a dbspace during the same ALTER TABLE session.

Attaching a Dbspace

Select the **aTt**ach option on the ALTER FRAGMENT expression strategy menu, and the ATTACH TABLES menu appears. [Figure 5-24](#) shows the ATTACH TABLES menu for a table that has expression fragmentation. The round-robin version of this screen shows only the table name for each fragment.

```
ATTACH TABLES new_acct:  Modify Drop Screen Exit
Add one new table to the fragmentation strategy.

----- Page 1 of 1 ----- newstores@mydbserv ----- Press CTRL-W for Help ----

Table Name      Expression      Position      Dbspace
-----
acct            field1 <100    BEFORE       dbspace1
cur_acct       field1 >=100
```

Figure 5-24
The
ATTACH TABLES
Menu for
Expression
Fragmentation
Strategy

Select **Add** on the ATTACH TABLES menu to begin attaching a fragment. On the SELECT ATTACHING TABLE screen, highlight or type in the table name you want and press RETURN, as [Figure 5-25](#) shows.

```
SELECT ATTACHING TABLE >> 
Select a table with the Arrow Keys, or enter the name, then press Return.

----- Page 1 of 1 ----- newstores@mydbserv ----- Press CTRL-W for Help ----


cur_acct
myacct
```

Figure 5-25
The SELECT
ATTACHING TABLE
Screen

If the table uses round-robin strategy, the resulting new fragment or fragments will be positioned at the end of the fragmentation strategy and the ALTER FRAGMENT menu returns.

If the table uses expression strategy, complete the following steps:

1. After you select a table, the EDIT EXPRESSION menu appears. See [Figure 5-20 on page 5-22](#).

2. When you exit the EDIT EXPRESSION menu, the ADD DEFINE ATTACH POSITION menu appears, as [Figure 5-26](#) shows.

```
ADD DEFINE ATTACH POSITION acct: [Before] After None
Define a position for an attaching fragment before a dbspace.

----- Page 1 of 1 ----- newstores@mydbserv ----- Press CTRL-W for Help -----
```

Figure 5-26
*The ADD DEFINE
ATTACH POSITION
Menu*

- ❑ Select **Before** to attach the new fragment *before* a dbspace that you select in the next step.
 - ❑ Select **After** to attach the new fragment *after* the dbspace that you select in the next step.
 - ❑ Select **None** to attach the new fragment in the default position.
3. If you select **Before** or **After** as the attach position, the SELECT DBSPACE screen appears, as [Figure 5-27](#) shows, listing the dbspaces that the strategy encompasses prior to attaching the new one.

```
SELECT DBSPACE >> [ ]
Select a dbspace with the Arrow Keys, or enter a name, then press Return.

----- Page 1 of 1 ----- newstores@mydbserv ----- Press CTRL-W for Help -----

[dbspace1]
dbspace2
dbspace3
```

Figure 5-27
*The
SELECT DBSPACE
Screen Listing
Fragmented
Dbspaces*

Select the dbspace before or after which you want to attach the added fragment.

The ATTACH TABLES menu reappears, as [Figure 5-24 on page 5-25](#) shows, and shows values for the Position and Dbspace fields.

Detaching a Dbspace

The detaCh option from the ALTER FRAGMENT menu displays the DETACH DBSPACE screen, as [Figure 5-28](#) shows.

```
DETACH DBSPACE >>
Select a dbspace with the Arrow Keys, or enter a name, then press Return.
----- Page 1 of 1 ----- newstores@mydbserv ----- Press CTRL-W for Help -----
dbspace1
dbspace2
dbspace3
```

Figure 5-28
*The
DETACH DBSPACE
Screen for
Removing
Fragmentation*

Select the dbspace from which you want the records copied into a new, unfragmented table. You can select a dbspace from the list or type in a dbspace name. If you enter an invalid dbspace name, an error message appears.

After you correctly enter a dbspace on the DETACH DBSPACE screen, the NEW TABLE screen appears, as [Figure 5-29](#) shows.

```
NEW TABLE >>
Enter the name you want assigned to the new table, then press Return.

----- Page 1 of 1 ----- newstores@mydbserv ----- Press CTRL-W for Help -----
```

Figure 5-29
The NEW TABLE Screen for Naming a Detached dbspace

Enter the name you want to assign to the new, unfragmented table. This table stores the records from the dbspace you previously selected through the DETACH DBSPACE screen. The display returns to the ALTER FRAGMENT menu.

Fragmenting an Existing Table

If a table has no fragmentation strategy when you select the Fragment option on the STORAGE menu, the ALTER FRAGMENT menu appears, as [Figure 5-30](#) shows.

```
ALTER FRAGMENT - new_acct:  Attach Exit
Define a fragmentation strategy.

----- Page 1 of 1 ----- newstores@mydbserv ----- Press CTRL-W for Help -----
```

Figure 5-30
The ALTER FRAGMENT Menu for No Fragmentation Strategy

The ALTER FRAGMENT menu has the following options.

Option	Purpose	Instructions
Init	Provides the following options: <ul style="list-style-type: none"> ■ fragments a previously unfragmented table ■ removes fragmentation from a table ■ changes the fragmentation strategy for a table 	The ALTER FRAGMENT-INIT menu has the same options as the FRAGMENT menu under CREATE TABLE. For instructions, see “Fragmenting a New Table” on page 5-18 .
Attach	Enables you to define a fragmentation strategy and select the tables to fragment with the new strategy. This option has the same effect as the following statement: <pre>ALTER FRAGMENT ON TABLE table1... ATTACH table1, table2</pre>	The ALTER FRAGMENT-ATTACH menu offers round-robin and expression fragmentation. For instructions on both types of strategy, see “Fragmenting a New Table” on page 5-18 .
Exit	Returns to the TABLE_OPTIONS menu	none



Important: You can perform only one operation during an ALTER FRAGMENT session.

Setting the Extent Size

When you create a table, you can specify how much initial disk space, or *initial extent size*, to reserve for the table. You can also specify the size of additional extents, or *next extent spaces*, the database server adds if the initial extent becomes full.

To specify an initial extent size, select the eXtent_size option on the TABLE_OPTIONS menu. DB-Access displays the Extent Size screen, as [Figure 5-31](#) shows. After you set initial extent size, select the Next_size option from the TABLE_OPTIONS menu to display the Next Size screen, as [Figure 5-32](#) shows.

```
Extent Size >> 
Specify initial extent size in kilobytes.

----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----

Column Name          Type          Length  Index Nulls
-----

```

Figure 5-31
The (Initial) Extent Size Screen

```
Next Size >> 
Specify next extent size in kilobytes.

----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----

Column Name          Type          Length  Index Nulls
-----

```

Figure 5-32
The Next (Extent) Size Screen

To select extent size on either screen, perform one of the following actions:

- Press RETURN to accept the default size of 8 kilobytes.
- Type a number (representing kilobyte units) and press RETURN. The minimum extent size is 4 kilobytes.

Determining the Lock Mode

When you select the Lock_mode option on the TABLE_OPTIONS menu, DB-Access displays the LOCK_MODE menu, as [Figure 5-33](#) shows.

```
LOCK_MODE clients:  Page  Row  Exit
Locking is at page level. This is the default.

----- Page 1 of 1 ----- mydata@mydbserv ----- Press CTRL-W for Help -----

Column Name          Type          Length  Index Nulls
-----

```

Figure 5-33
The LOCK_MODE Menu

The LOCK_MODE menu lets you choose the mode to use when the database locks the rows in a table. The LOCK_MODE menu has the following options.

Option	Purpose
Page	Locks the entire page on which a row resides
Row	Locks a selected row individually
Exit	Exits to the TABLE_OPTIONS menu

One row of a table is the smallest object that you can lock. A disk page contains one or more rows of a table. To determine if you will enhance performance by locking a disk page rather than individual rows on the page, see your *Performance Guide*. Unless you specify row-level locking before you exit, DB-Access uses the default (Page).

Adding or Dropping Rowids

You can add or delete rowids only when you alter an existing table. To reach the ALTER ROWID menu:

- Select Table_options from the ALTER TABLE menu.
- Select Rowids from the TABLE OPTIONS menu.

The ALTER ROWID menu has the following options.

Option	Purpose
Add	Adds a column with rowids to the fragmented table
Drop	Discards the rowid column previously added
None	Cancels the selection you made on this screen so that you can exit without altering the table
Exit	Exits to the TABLE_OPTIONS menu

If you select Add or Drop, another menu prompts you to verify your selection. Select Yes to execute the Add or Drop; select No to cancel the Add or Drop.

Defining Constraints

You can use the DB-Access Schema Editor to define constraints for columns in a specified table. You can define primary-key, foreign-key, column-level and table-level check, and unique constraints, as well as add and modify column default values.

If you select the Constraints option from the CREATE TABLE menu, the CONSTRAINTS menu appears, as [Figure 5-34](#) shows.



```
CONSTRAINTS - mytab:  Primary Foreign Check Unique Defaults Exit
Define a primary key constraint.
-----mydata@mydbserv----- Press CTRL-W for Help -----
```

Figure 5-34
The
CONSTRAINTS
Menu

The CONSTRAINTS menu has the following options.

Option	Purpose
Primary	Lists the columns that make up the table so that you can choose the column or columns that make up the primary key
Foreign	Asserts a foreign-key relationship for a column
Check	Enables you to specify valid values for a column and forces the validation of data entry in that column
Unique	Declares that a column must contain a unique value
Defaults	Enables you to set a default value for a column
Exit	Returns you to the CREATE TABLE menu

Defining Primary-Key Constraints

To add, modify, or delete primary-key constraints for the current table, select the Primary option on the CONSTRAINTS menu. The PRIMARY KEY menu appears, as [Figure 5-35 on page 5-33](#) shows.

```

PRIMARY KEY mytab: █ Add Modify Drop Screen Exit
Add a constraint name or column name.

-----Page 1 of 1 -----mydata@mydbserv----- Press CTRL-W for Help -----

Constraint Name      Column Name
constraint1          column1

```

Figure 5-35
The
PRIMARY KEY
Menu

When you enter a primary constraint, DB-Access validates your entry by verifying the following information:

- The column name is not repeated.
- No more than 16 column names appear in one constraint.
- The column is not assigned the BYTE or TEXT data type.

Use the menu options as follows:

- Use the Screen option to scroll to any additional constraint descriptions so that you can select one to modify or delete.
- If you select the Add option to add a primary-key referential constraint, the Schema Editor inserts and highlights a new line at the top of the constraint list. If you move your cursor to an existing constraint line and select the Modify option, you can change the contents of the line.
 1. At the ADD (or MODIFY) CONSTRAINT NAME prompt, enter a constraint name.

If you press RETURN in this field without typing a constraint name, the database server assigns a temporary constraint name, such as **unassigned1**, **unassigned2**, and so on. This temporary constraint name exists until you modify it or the table is built or discarded. The database server assigns a permanent constraint name at the time that you create the table.
 2. At the ADD (or MODIFY) COLUMN NAME prompt, enter a column name.
- Select Drop to eliminate the constraint that the highlight indicates. If the highlighted field is Constraint Name, then all columns associated with that constraint are deleted. The lines are removed and any gaps are closed in the display.

Defining Foreign-Key Constraints

To create, modify, or delete foreign-key constraints for the current table, select the Foreign option on the CONSTRAINTS menu. The FOREIGN KEY menu appears, as [Figure 5-36](#) shows.

```
FOREIGN KEY mytab:  Add Modify Drop Screen Exit
Add a constraint name or referencing/referenced column pair.

-----Page 1 of 1 -----mydata@mydbserv----- Press CTRL-W for Help -----

Constraint   Referencing Column   Referenced Table   Referenced Column   CD
unassigned0  column1              table2              column1              Y
column2      column2
column3      column3
```

Figure 5-36
The
FOREIGN KEY
Menu

Use the menu options as follows:

- Use the Screen option to scroll any additional constraint descriptions onto the screen so that you can select one to modify or delete.
- Select the Add option to specify the constraint name, referenced table, and cascading deletes, or to add one or more referencing and referenced column pairs to the constraint.
 1. At the ADD CONSTRAINT NAME prompt, enter a constraint name. If you press RETURN in this field without typing a constraint name, DB-Access assigns a temporary constraint name. The database server assigns a permanent name to the constraint when the constraint is created.
 2. At the ADD COLUMN NAME prompt, enter the name of the *referencing* column.
 3. At the ADD REFERENCED TABLE prompt, enter the name of the *referenced table*.
 4. At the second ADD COLUMN NAME prompt, enter the name of the *referenced column*.
 5. When the CD (Cascading Deletes) field is the current field, the ADD ENABLE CASCADING DELETES menu appears, as [Figure 5-37 on page 5-35](#) shows.

```

ADD ENABLE CASCADING DELETES mytab3:  No  Yes
No, I do not want cascading deletes.

----- Page 1 of 1 -----mydata@mydbserv----- Press CTRL-W for Help -----

Constraint Referencing Column Referenced Table Referenced Column CD
cons1      col1                yourtab          col6                N

```

Figure 5-37
The
ADD ENABLE
CASCADING
DELETES
Menu

Select **Yes** to enable cascading deletes. When you delete a referenced (parent) record, you also delete all corresponding referencing (child) records. (Option **Yes** is equivalent to the **ON DELETE CASCADE** option of the **REFERENCES** clause in the **CREATE TABLE** statement.)

Select **No** to prevent cascading deletes. A referenced (parent) column cannot be deleted if referencing (child) records exist.

For a detailed description of referential integrity and cascading deletes, see the **CREATE TABLE** statement in the [Informix Guide to SQL: Syntax](#).

When you complete the Cascading Deletes entry, the cursor returns to the Constraint field. Enter another constraint or press the Interrupt key to return to the top line of the FOREIGN KEY menu.

- Move your cursor to an existing constraint line and select the **Modify** option to change the contents of the line. You can modify Constraint, Referencing Column, and Referenced Table information.

Change the entry for the field and press **RETURN** to modify it, or press the Interrupt key if you do not want to modify a foreign constraint.

- The **Drop** option displays a list of foreign-key constraints and prompts you to select one to delete.

If the current (highlighted) field is **Constraint**, then the *entire* constraint is deleted. If any other field is highlighted, then only that referenced and referencing pair is deleted.

The default is **Yes**. Press **RETURN** to delete the highlighted constraint. Move the cursor to highlight **No** if you do not want to delete that constraint. You return to the FOREIGN KEY menu.

Defining Check Constraints

The CHECK CONSTRAINTS menu lets you add, modify, or delete a check constraint for the current table. Select the Check option on the CONSTRAINTS menu to access the CHECK CONSTRAINTS menu, as [Figure 5-38](#) shows.

```
CHECK CONSTRAINTS mytab: Add Modify Drop Screen Exit
Add a check constraint.

-----mydata@mydbserv----- Press CTRL-W for Help -----

Constraint Name      Value
cons2                 (column1 > (c ...
cons3                 column2 < col ...
cons4                 column3 > 100
```

Figure 5-38
The CHECK
CONSTRAINTS
Menu

The CHECK CONSTRAINTS menu displays any previously added check constraints. The first 36 characters of the check value appear on the CHECK CONSTRAINTS menu. Use the CHECK CONSTRAINTS menu options as follows:

- The Drop option lets you delete the current check constraint. The default is Yes. Press RETURN to delete the highlighted constraint. Move the cursor to highlight No and then press RETURN if you do not want to delete that constraint. You return to the CHECK CONSTRAINTS menu.
- Select the Add option to add a check constraint for the table and enter the name and value of the constraint.
 1. At the ADD (or MODIFY) CONSTRAINT NAME prompt, enter a name. If you press RETURN in this field without specifying a value, a temporary constraint name is assigned, and it exists until the table is built or discarded.
 2. From THE ADD (or MODIFY) CHECK VALUE menu, as [Figure 5-39 on page 5-37](#) shows, select the SQL editor or specify another editor to enter the check-constraint value.

```

ADD CHECK VALUE cons99: [New] Modify Use-editor Exit
Enter a new check value using the SQL editor.

-----mydata@mydbserv----- Press CTRL-W for Help -----
    
```

Figure 5-39
The ADD CHECK VALUE Menu

The ADD CHECK VALUE menu has the following options.

Option	Purpose
New	Displays the blank SQL editor screen so that you can enter a new check value
Modify	Displays the current check value on the SQL editor screen so that you can modify the value
Use-editor	Displays the current check value in the system editor so that you can modify the value
Exit	Returns to the CHECK CONSTRAINTS menu

If you break from the ADD CHECK VALUE menu or exit without defining the check value, you return to the CHECK CONSTRAINTS menu. If you defined the check value, you remain in add mode, a new line is inserted, the Constraint Name is the current field, and the ADD CONSTRAINT NAME prompt appears.

Defining Unique Constraints

The UNIQUE CONSTRAINTS menu lets you add, modify, or delete a unique constraint for the current table. To access the UNIQUE CONSTRAINTS menu, select the Unique option on the CONSTRAINTS menu, as [Figure 5-40](#) shows.

```

UNIQUE CONSTRAINTS mytab:  Add  Modify  Drop  Screen  Exit
Add a unique constraint.

-----mydata@mydbserv----- Press CTRL-W for Help -----

Constraint Name  Column Name
cons2           column1
               column2
               column3
cons3           column4
    
```

Figure 5-40
The UNIQUE
CONSTRAINTS
Menu

Use the UNIQUE CONSTRAINTS menu options as follows:

- Select Screen to display the next screen of unique constraints.
- If you select the Add option, the Schema Editor adds a new line.
 1. At the ADD CONSTRAINT NAME prompt, enter a constraint name. If you press RETURN in this field without typing a name, a temporary constraint name is assigned, which exists until the table is built or discarded.
 2. At the ADD COLUMN NAME prompt, enter the name of the column that should have a unique value. A new line is added, and the Column Name is still the current field.
- The Modify option lets you modify either the Constraint Name or Column Name field. If you modify the Constraint Name field, the MODIFY CONSTRAINT NAME prompt appears. If you modify the Column Name field, the MODIFY COLUMN NAME prompt appears.
- Select the Drop option to delete the constraint or column name where the highlight is located.

If you delete a constraint name, all column names associated with that constraint name are also deleted.



Important: You cannot modify unique constraints after you create them. To identify the unique constraints listed on the UNIQUE CONSTRAINTS menu, use an asterisk (*) before the constraint name. If you try to modify a unique constraint using the Modify option in the UNIQUE CONSTRAINTS menu, an error message appears.

Data Validation

When you enter a unique constraint, DB-Access validates your entry by verifying the following information:

- The column name exists.
- The column name is not repeated.
- No more than 16 column names are present in one constraint.
- The column is not assigned the BYTE or TEXT data type.

Defining Default Values

Use the DEFAULTS menu to define default values for columns in a table, as [Figure 5-41](#) shows. To access the DEFAULTS menu, select the Defaults option on the CONSTRAINTS menu.

```

DEFAULTS mytab:  Add  Modify Drop Screen Exit
Add a column default.

-----Page 1 of 1 -----mydata@mydbserv----- Press CTRL-W for Help -----

Column Name  Type   Value
column1      User
column3      Null
column5      Today
column6      Current (Fraction to Fraction (5))
column7      Literal
column8      Literal 1200
  
```

Figure 5-41
The
DEFAULTS
Menu

The DEFAULTS menu has the following options.

Option	Purpose
Add	Adds a column default value of the appropriate data type
Modify	Lists an existing default name, data type, and value so that you can change the default attributes
Drop	Deletes a column default
Screen	Displays the next screen of defaults
Exit	Returns to the CONSTRAINTS menu

You can see the first 28 characters of the value.

To add a column default

1. Select the Add option.
The Schema Editor inserts a new line at the top of the list and makes the Column Name the current field.
2. At the ADD COLUMN NAME prompt, enter a value for the column name.
3. From the ADD DEFAULT TYPE menu that [Figure 5-42](#) shows, define the default value of a column in the current table.

Figure 5-42
The ADD DEFAULT TYPE Menu

```
ADD DEFAULT TYPE mytab: [Literal] User Current Null Today Db-server-name Site-name
Assign a literal value using either the SQL editor or a system editor.

-----mydata@mydbserv----- Press CTRL-W for Help -----
```

The ADD DEFAULT TYPE menu has the following options that let you assign default values to the column.

Option	Default Value Assigned
Literal	A literal default value entered either in the SQL editor or a user-specified system editor
User	The login name of the current user
Current	The current system clock time of day
Null	Null
Today	The current system date
Db-server-name	The current database server name
Site-name	The current site name

- If you select User, Null, Today, Db-server-name, or Site-name, the system assigns that value to the default type and returns you to the DEFAULTS menu in add mode.
- If you select Current as the default value, the qualifier is taken from the column definition.
- If you enter a default value and the type is DATETIME or INTERVAL, enter only the value. The qualifier comes from the column definition.
- If you select Literal as the default type, the ADD DEFAULT VALUE menu appears, which lets you assign a literal as the default value.

The ADD DEFAULT VALUE menu lets you add or modify the default value for a column in the current table with either the SQL editor or a system editor, as Figure 5-43 shows.

```
ADD DEFAULT VALUE column7: [New] Modify Use-editor Exit
Enter a new default value using the SQL editor.
-----mydata@mydbserv----- Press CTRL-W for Help -----
```

Figure 5-43
The ADD DEFAULT VALUE Menu

The ADD DEFAULT VALUE menu displays the following options.

Option	Purpose
New	Displays the blank SQL editor screen so that you can enter a new value
Modify	Displays the current default on the SQL editor screen so that you can modify the default value
Use-editor	Displays the current default in the system editor so that you can modify the current value
Exit	Returns to the ADD DEFAULT VALUE menu

To modify the column name, type, or value field

1. Select the Modify option to modify the Column Name, Type, or Value field where the highlight is located.
2. If you highlight a value, the MODIFY DEFAULT VALUE menu prompts you to invoke the SQL editor or the system editor so that you can type over or modify the existing default value.

Data Validation

When you enter a default value, DB-Access validates your entry. The database server validates the literal value and checks the following information:

- The column name must exist.
- The column type cannot be SERIAL.
- If the column does not allow nulls, you cannot specify the default type as Null.
- You can use the default type Current only with a DATETIME column type.
- You can use the default type Db-server-name only with a column type of CHAR, NCHAR, VARCHAR, or NVARCHAR, which has a minimum length of 18 characters.
- You can use the default type Site-name only with a column type of CHAR, NCHAR, VARCHAR, or NVARCHAR, which has a minimum length of 18 characters.
- You can use the default type Today only with a column type of DATE.
- You can use the default type User only with a column type of CHAR, NCHAR, VARCHAR, or NVARCHAR, which has a minimum length of 18 characters.

Displaying Table Information

Use the Info option on the TABLE menu to display information about columns, indexes, access privileges, reference privileges, constraints (referential, check, or unique), column default values, triggers, status, and fragmentation strategy of a table. No options exist to display table owners or information on views.

When you select the Info option on the TABLE menu, the INFO FOR TABLE screen appears, as [Figure 5-44](#) shows.

Figure 5-44

The
INFO FOR TABLE
Screen

```
INFO FOR TABLE >>
Choose a table with the Arrow Keys, or enter a name, then press Return.
----- mydata@mydbserv ----- Press CTRL-W for Help -----
clients
customer
orders
```

This screen lists the names of tables that exist in the current database. Note the following items:

- If you are not the table owner, the table name is prefixed by the owner name, as in **june.clients**.
- If the list of tables does not fit on one screen, the last entry is an ellipsis (...). Use the arrow keys to highlight the ellipsis, and the next page of table names appears.
- If Global Language Support is enabled, the list of table names is sorted according to the database collation rules defined when the database was created. Thus, different users using different collating sequences for DB-Access see the table names in the database listed in the same order. ♦

To request information about tables on a different database server, use the format *database@server:table* or *database@server:owner.table* at the prompt. The following example requests information on the **customer** table that **dba** created in the **accounts** database on the database server **topend**:

```
INFO FOR TABLE >> accounts@topend:dba.customer
```

To leave the INFO FOR TABLE screen without requesting table information, press the Interrupt key. You return to the TABLE menu.

GLS

You can select a table in one of the following ways:

- Type the table name and press RETURN.
You must use this method and include the full pathname if you want information on a table that is not in the current database.
- Use the arrow keys to highlight the table name that you want and press RETURN.

For example, for the **customer** table, type **customer** or use the arrow keys to highlight it and press RETURN. The INFO menu appears, with **customer** in the top line, as [Figure 5-45](#) and [Figure 5-46 on page 5-45](#) show.

```
INFO - customer: [Columns] Indexes Privileges References Status ...
Display column names and data types for a table.

----- mydata@mydbserv ----- Press CTRL-W for Help -----
```

Figure 5-45
*The INFO Menu for
Displaying Table
Information
(First Screen)*

```
INFO - customer: [...] ... [cOnstraints] triGgers Table Fragments Exit
Reference menu and primary, unique, check and defaults options.

----- mydata@mydbserv ----- Press CTRL-W for Help -----
```

Figure 5-46
*The INFO Menu for
Displaying Table
Information
(Second Screen)*

The INFO menu has the following options.

Option	Purpose	Instructions
Columns	Lists data type by column name and indicates which columns can contain a null value	“Displaying Column Information” on page 5-47
Indexes	Describes each index defined for a specified table	“Displaying Index Information” on page 5-51
Privileges	Lists the users who have Select, Update, Insert, Delete, Index, or Alter privileges for the specified table	“Displaying Table-Level Privileges” on page 5-52
References	Lists the users who have the table-level References privilege for the specified table and the names of the columns they can reference	“Displaying References Privileges” on page 5-53
Status	Lists the table name, owner, row size, number of rows and columns, and creation date of the current table	none
cOnstraints	Displays the referential, primary, unique, and check constraints, as well as the default values for the columns in the specified table	“Displaying Column Constraints and Defaults” on page 5-54
triGgers	Displays header and body information for a specified trigger	“Displaying Triggers” on page 5-57
Table	Redisplays the INFO FOR TABLE menu so that you can select a different table for examination	none
Fragments	Lists fragmented dbspaces assigned to the table and, for expression-based fragmentation, displays the expression assigned to each dbspace	“Displaying Fragmentation Information” on page 5-59
Exit	Returns to the TABLE menu	none



Tip: From the *CREATE TABLE* menu, use *Table-options* to view extent and lock mode information, or issue a *SELECT* statement to list the table description in the *systables* system catalog table.

Displaying Column Information

Use the *Columns* option on the *INFO* menu to display the following information for each column of the specified table: the name of the column, the data type of the column, and whether null values are allowed in the column.

[Figure 5-47](#) shows the kind of information that you see when you select the *Columns* option for the *cust_calls* table.

Column name	Type	Nulls
customer_num	INTEGER	no
call_dtime	DATETIME YEAR TO MINUTE	yes
user_id	CHAR(18)	yes
call_code	CHAR(1)	yes
call_descr	CHAR(240)	yes
res_dtime	DATETIME YEAR TO MINUTE	yes
res_descr	CHAR(240)	yes

Figure 5-47
Displaying Column Information for the cust_calls Table

DB-Access Data Types

The columns in [Figure 5-47](#) show that the *cust_calls* table consists of all built-in (standard) data types. The data types available through the *CREATE TABLE* menu hierarchy are built-in types.

The built-in types that the *Columns* option can display are BOOLEAN, BYTE, CHAR, CHARACTER VARYING, DATE, DATETIME, DECIMAL, FLOAT, INT8, INTEGER, INTERVAL, MONEY, NCHAR, NVARCHAR, SERIAL, SERIAL8, SMALL-FLOAT, SMALLINT, TEXT, VARCHAR, and user-defined types.

For descriptions of these data types, see the [Informix Guide to SQL: Reference](#).

Figure 5-48 shows the display of column information for a table that has BOOLEAN, INT8, and SERIAL8 columns as well as other built-in data types.

Column name	Type	Nulls
id_num	integer	yes
yes_or_no	boolean	yes
int8col	int8	yes
serial8col	serial8	yes
text_descr	char(20)	yes

Figure 5-48
Displaying Column Information for a Table with Several Built-in Types

Large Objects

Large objects are built-in data types that store a large amount of data in a single column. Within a table, large-object data type columns actually contain pointers to the *physical* storage spaces where the database server places the large data objects.

DB-Access can display any of the following large-object data types:

- TEXT, BYTE
- CLOB, BLOB ♦

IDS

The Columns option displays the specific data type for any column that contains pointers to large objects. [Figure 5-49](#) shows the display of column information for a table that has a BYTE column.

Column name	Type	Nulls
id	integer	yes
binary_col	byte	yes

Figure 5-49
Displaying Column Information for a BLOB Column

IDS

Extended Data Types in Dynamic Server

This section shows how DB-Access displays user-defined and complex data types for Dynamic Server.

Opaque Data Types

An opaque data type characterizes data that cannot be represented by any of the built-in types that belong to the database server. DB-Access can identify and display opaque data types.

For example, suppose you assign an opaque data type called **circle_t** to a column named **circle_col**. The Columns option displays the opaque data type name in the **Type** column, as [Figure 5-50](#) shows.

Column name	Type	Nulls
id	integer	yes
circle_col	circle_t	yes

Figure 5-50
Displaying Information for a Column with an Opaque Data Type

Collection Types

A collection type contains zero or more elements and is more specifically defined with one of the following data type names.

Data Type	Characteristics
SET	An unordered collection of elements in which duplicates are not allowed
MULTISET	An unordered collection of elements in which duplicates are allowed
LIST	An ordered collection of elements in which duplicates are allowed

DB-Access displays the specific kind of collection type in the **Type** column. For example, [Figure 5-51](#) shows the display of a SET data type column named **siblings**.

Column name	Type	Nulls
id	integer	yes
siblings	set	yes

Figure 5-51
Displaying Information for a Column with a Collection Data Type

Row Types

The Columns option for a table that includes a column with a row type displays the string **Row** in the **Type** column. DB-Access displays this string whether the column has a named or unnamed row type. Assume you define row type **rectangle_t** and assign it to column **rect**. [Figure 5-52](#) shows that the Columns display for **rect** returns `row` rather than the specific row-type name **rectangle_t** as the data type.

Column name	Type	Nulls
id	integer	yes
rect	row	yes

Figure 5-52
Column Information for a Row Data Type

Displaying Index Information

Figure 5-53 shows the kind of information that you see when you select the **Indexes** option for the **cust_calls** table. The **c_num_dt_ix** index is a B-tree index defined on two columns of the **cust_calls** table. The **c_num_cus_ix** index is a B-tree index defined on a single column of the **cust_calls** table.

Figure 5-53
Display of Index Information

Index name	Owner	Type/Clstr	Access_Method	Columns
c_num_dt_ix	velma	unique/No	B-Tree	customer_num call_dtime
c_num_cus_ix	velma	dupls/No	B-Tree	customer_num

The following table shows the meaning of each column in the display.

Display Column	Description
Index Name	The name of the index
Owner	The owner of the index
Type	The index type (unique or duplicate)
Clstr	Indicates whether the index is clustered. (A clustered index causes the table to be physically reordered in the same sequence as the index.)
Access Method	The index access method (such as B-tree or functional)
Columns	The column or columns on which the index is defined

For further information about the types of indexes available on your database server, see your [Performance Guide](#).

IDS

You can display information for non-B-tree indexes, including indexes based on user-defined secondary access methods that Dynamic Server permits. For example, the index shown in [Figure 5-54](#) is based on a Fulltext access method that a DataBlade module provides.

Figure 5-54
Display of Information for a DataBlade Module Index

Index name	Owner	Type/Clsr	Access_Method	Columns
text_idx	wilma	dupls/No	Fulltext	zone_descr

For information about user-defined access methods, see your DataBlade documentation or the documentation for the access method. ♦

Displaying Table-Level Privileges

[Figure 5-55](#) shows the kind of table-level access-privileges information that you see when you select the Privileges option for the **cust_calls** table.

User	Select	Update	Insert	Delete	Index	Alter
public	All	All	Yes	Yes	Yes	No

Figure 5-55
Display of Privileges Information

Unless your login is listed separately, you have the privileges given for **public** (a general category for all users).

If you want information about database-level privileges, use a SELECT statement to access the **sysusers** system catalog table.

Displaying References Privileges

Figure 5-56 shows the kind of information that you see when you select the References option for a table that has referential integrity.

User	Column References
betty	col1 col2 col3
wilma	All
public	None

Figure 5-56
*Display of
References
Information*

This display indicates the following table-level references privileges:

- The user **betty** can reference columns 1, 2, and 3 of the specified table.
- The user **wilma** can reference all the columns in the table.
- Users who only have privileges granted to **public** cannot access any columns in the table.

Displaying Column Constraints and Defaults

When you select the cOnstraints option from the INFO menu, the CONSTRAINTS menu appears, as [Figure 5-57](#) shows.

```
CONSTRAINTS - mytab: [ ] Reference Primary Check Unique Defaults Exit
Referenced and referencing options.

-----mydata@mydbserv----- Press CTRL-W for Help -----
```

Figure 5-57
The
CONSTRAINTS
Menu

You can select from the following options on the CONSTRAINTS menu.

Option	Purpose	Instructions
Reference	Shows referential (foreign-key) constraints associated with the specified table	“Displaying Referential Constraints” on page 5-55
Primary	Shows primary-key columns in the specified table	Figure 5-61
Check	Shows check constraints on columns in the specified table	Figure 5-62
Unique	Lists columns that must contain unique data	none
Defaults	Lists columns for which default values are defined and the default value	none
Exit	Returns to the INFO menu	none

Displaying Referential Constraints

When you select the Reference option from the CONSTRAINTS menu, the REFERENCE menu appears, as [Figure 5-58](#) shows.

```
REFERENCE - mytab: [Referencing] referenceD Exit
Display foreign key constraints.

-----mydata@mydbserv----- Press CTRL-W for Help -----
```

Figure 5-58
The REFERENCE Menu

The REFERENCE menu lets you display the following information:

- Foreign-key (referencing) constraints
- Columns of the current table
- Cascading deletes enabling
- Referenced columns (columns referenced as foreign keys by columns from another table)

Select the Referencing option on the REFERENCE menu to display the foreign-key constraints in the current table, as [Figure 5-59](#) shows.

```
REFERENCE - myacct: [Referenc] referenceD Exit
Display referential constraints.

-----sub_accounts -----Press CTRL-W for Help -----

Constraint Name   Referencing Column   Referenced Table   Referenced Column   CD
r107_13           ref_num              sub_accounts       acc_num              Y
                  ref_type                                acc_type
```

Figure 5-59
The REFERENCE Menu with Referencing Information

Select the referenceD option on the REFERENCE menu to display other tables and columns that reference your current columns as foreign keys, as [Figure 5-60](#) shows.

Figure 5-60
The REFERENCE Menu with Referenced Information

```
REFERENCE - myacct: [ ] Referenc [referenceD] Exit
Display columns which have foreign keys which reference this table.
-----accounts -----Press CTRL-W for Help -----
Constraint Name      Referenced Column  Referencing Table  Referencing Column  CD
r107_13              acc_num           sub_accounts      ref_acc             Y
                    acc_type          ref_type
```

Select Exit to return to the CONSTRAINTS menu.

Sample Primary and Check Constraint Displays

[Figure 5-61](#) shows the columns that form a primary key. This display results from selecting Primary from the CONSTRAINTS menu.

Figure 5-61
The CONSTRAINTS Menu with Primary-Key Constraint Information

```
CONSTRAINTS - mytab: [ ] References [Primary] Check Unique Defaults Exit
Primary key constraints
-----mydata@mydbserv----- Press CTRL-W for Help -----
Constraint Name      Column Name
constraint1          assembly
                    partnum
```

Select the Check option on the CONSTRAINTS menu to display the check constraints placed on columns of the current table, as [Figure 5-62](#) shows.

```

CONSTRAINTS - mytab:  Reference Primary  Check  Unique Defaults Exit
Display check constraints.

-----mydata@mydbserv----- Press CTRL-W for Help -----

Constraint name      Value
-----
cons2                (column1 > (column2 * 100 - 1000 + column1 / 2 -
                    column2/20 + 40 * 3 - 55 * column2 + 77 * column1))
cons3                column2 > column3
cons4                column3 > 100
    
```

Figure 5-62
The
CONSTRAINTS
Menu with
Check-Constraint
Information

Displaying Triggers

When you select the triGgers option from the INFO menu, the INFO FOR TRIGGER screen appears, as [Figure 5-63](#) shows.

```

INFO FOR TRIGGER>>
Choose a trigger with the Arrow Keys, or enter a name, then press Return.

-----mydata@mydbserv----- Press CTRL-W for Help -----


    
```

Figure 5-63
The
INFO FOR TRIGGER
Screen

Select a trigger from the list of trigger names in the current table. If you do not want to select a trigger, press the Interrupt key, and you return to the TABLE INFO menu.

If the header and body information for the selected trigger fit on one screen, the INFO menu reappears, displaying the trigger information, as [Figure 5-64](#) shows.

```
INFO - updrec_t: [ ]... [triggers] Status Table Exit
Display header and body information for a trigger.

----- mystores@dbserver1 ----- Press CTRL-W for Help -----

create trigger updrec_t
      unit_price on stock
      referencing old as pre_upd
      new as post_upd

(insert into log_record values (stock_num, CURRENT,
pre_upd.unit_price, post_upd.unit_price)) for each row;
```

Figure 5-64
Display of Trigger Information on the INFO Menu

[Figure 5-64](#) shows the header information for a trigger, which consists of the CREATE TRIGGER statement and trigger name, the SQL statement that triggers an event, and the referencing clause. The body of a trigger is the triggered action. In [Figure 5-64](#), the triggered action is the INSERT statement.

If the trigger does not fit on a single INFO menu screen, use the menu at the top of the screen as follows:

- Press N, or if the Next option is highlighted, press RETURN to advance to the next screen of trigger information. Continue to press N or RETURN as needed to page through the information.
- Select Restart at any time to display the trigger information from the beginning.
- Select Exit to return to the TABLE menu.

XPS

Extended Parallel Server does not support triggers. ♦

Displaying Fragmentation Information

Figure 5-65 shows the kind of information that you see when you select the Fragments option for an indexed table created with a round-robin fragmentation strategy.

Figure 5-65
Fragmentation Information where Round-Robin Strategy Applies to Both Table and Index

Idx/Tbl Name	Dbpace	Type	Expression
cust	dbspace1	T	
cust	dbspace2	T	
cust	rootdbs	T	
cust	dbspace1	I	
cust	dbspace2	I	
cust	rootdbs	I	

IDS

Idx/Tbl Name shows the object that was explicitly fragmented.

If you use Dynamic Server, the display includes a **Type** column to indicate whether the fragment on the line is part of an index or the table data. In Figure 5-65, the **cust** table was created with round-robin strategy, but the index was created without specifying a strategy. In this case, the indexes are located in default dbspaces. ♦

Suppose, when creating an index, you use the following statement to apply a fragmentation strategy:

```
create index idx on cust(custnum) fragment by expression
    custnum < 200 in dbspace1,
    custnum > 200 in dbspace2,
    remainder in rootdbs;
```

In this case, the fragmentation display looks like [Figure 5-66](#).

Idx/Tbl Name	DbSPACE	Type	Expression
cust	dbSPACE1	T	
cust	dbSPACE2	T	
cust	rootdbs	T	
idx	dbSPACE1	I	(custnum < 200)
idx	dbSPACE2	I	(custnum > 200)
idx	rootdbs	I	remainder

Figure 5-66
Fragmentation Information Where Table and Index Have Different Strategies

Idx/Tbl Name shows the index name because the fragmentation was explicitly applied to the index.

Dropping a Table

Use the Drop option on the TABLE menu to delete an existing table schema from the database. Press the D key, or highlight Drop and press RETURN. The DROP TABLE screen appears, as [Figure 5-67](#) shows.

```
DROP TABLE >> 
Enter the table name you wish to drop from the database.
----- mydata@mybserv ----- Press CTRL-W for Help -----

customer
orders
```

Figure 5-67
The DROP TABLE Screen

This screen lists the names of tables that exist in the current database. You can delete a table in one of the following ways:

- Type the table name and press RETURN. You must use this method and include the full pathname if you want to delete a table that is not in the current database.
- Use the arrow keys to highlight the name of the table that you want to delete from the database and press RETURN.

To leave the DROP TABLE screen without deleting a table, press the Interrupt key. You return to the TABLE menu.

Warning: When you delete a table, you delete both the table and all the data it contains.

After you select a table to delete, DB-Access displays the CONFIRM menu, which asks for confirmation before it deletes the table, as [Figure 5-68](#) shows.



```
CONFIRM:  No  Yes
No, I do not want to drop it.

-----mydata@mydbserv----- Press CTRL-W for Help -----

clients
customer
orders
```

Figure 5-68
The
CONFIRM
Menu

The default is No to prevent you from deleting a table. You must explicitly delete a table. Thus, if you want to delete the highlighted table, press the Y key or use the right arrow key to highlight Yes and press RETURN. DB-Access deletes the table.

Related Manuals

Have the following manuals available while you create or alter a table schema or structure:

- For information on how to name a table or a column, see the [Informix Guide to SQL: Syntax](#).
- For information about data types, see the [Informix Guide to SQL: Reference](#).
- For information about fragmentation and storage space allocation, see your [Administrator's Guide](#).
- For recommendations concerning fragmentation, indexes, and extent size, see your [Performance Guide](#).
- For information about nondefault locale names and character data types (CHAR, VARCHAR, NCHAR, NVARCHAR, and TEXT), see the [Informix Guide to GLS Functionality](#).
- For information on how to design, implement, and manage your database, see the [Informix Guide to Database Design and Implementation](#).

The Connection and Session Options

In This Chapter	6-3
Choosing the Connection Option	6-3
Connecting to a Database Environment	6-4
Permissions Needed.	6-6
Implicit Closures	6-7
Disconnecting from a Database Environment	6-8
Transaction Processing	6-8
Choosing the Session Option	6-9

In This Chapter

This chapter describes the Connection and Session options on the main menu.

Use the Connection option if you want to connect to a specific database server and database or explicitly disconnect from the current database environment. Use the Session option to display information about the current DB-Access session.

For the GLS considerations that apply to establishing a connection between a client application, such as DB-Access, and a database, see the [Informix Guide to GLS Functionality](#). The database server examines the client locale information passed by the client, verifies the database locale, and determines the server-processing locale for transferring data between the client and the database. ♦

Choosing the Connection Option

From the main menu, press the C key (or highlight the Connection option and press RETURN) to display the CONNECTION menu, as [Figure 6-1](#) shows.

```
CONNECTION: [C]onnect Disconnect Exit
Connect to a database environment.
```

```
----- Press CTRL-W for Help -----
```

Figure 6-1
The
CONNECTION
Menu

The CONNECTION menu displays the following options.

Option	Purpose	Instructions
Connect	Connects to a database environment	“Connecting to a Database Environment” on page 6-4
Disconnect	Disconnects from the current database environment	“Disconnecting from a Database Environment” on page 6-8
Exit	Returns to the DB-Access main menu	none

Connecting to a Database Environment

To connect to an existing database server, choose the Connect option from the CONNECTION menu. DB-Access displays a list of available database servers and prompts you to make a selection. Select a database server and DB-Access prompts you to enter a user name, as [Figure 6-2](#) shows.

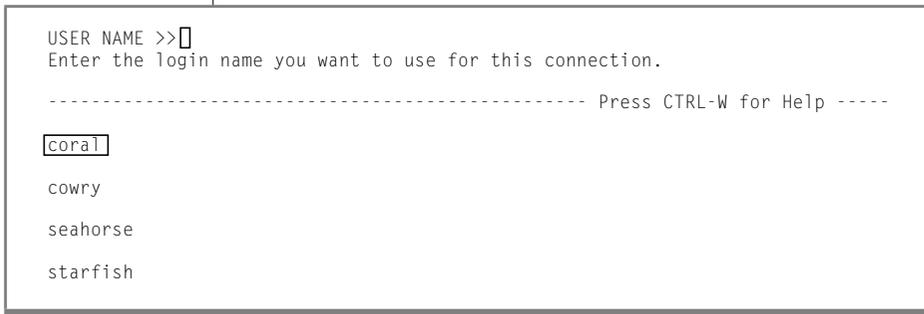


Figure 6-2
The
USER NAME
Prompt Screen

If you do not specify a user identifier on the USER NAME screen and press RETURN, you see the standard SELECT DATABASE screen listing databases on the chosen database server.

If you enter the login name that you want DB-Access to use when connecting to the target database server, DB-Access displays the PASSWORD screen, as [Figure 6-3](#) shows.

```
PASSWORD >>
Enter the password associated with the user identifier.

----- Press CTRL-W for Help -----

coral
cowry
seahorse
starfish
```

Figure 6-3
The
PASSWORD
Screen



From the PASSWORD screen, enter a password associated with the user identifier or press RETURN if you do not want to enter a password. For security reasons, the password that you enter on the screen is not displayed.

Tip: If you press CTRL-C on the USER NAME screen, DB-Access might try to connect to the specified database server rather than interrupt the session. This situation occurs because pressing CTRL-C on this screen is the equivalent of using the current user name.

If the user identifier and password combination are valid, you connect to the target database server. You can then select a database from that database server. The SELECT DATABASE SERVER screen appears, as [Figure 6-4](#) shows.

```
SELECT DATABASE SERVER>>
Select a server with the Arrow Keys, or enter a name, then press Return.

----- Press CTRL-W for Help -----

coral
cowry
seahorse
starfish
```

Figure 6-4
The SELECT
DATABASE SERVER
Screen

The `SELECT DATABASE SERVER` screen shows an alphabetical list of database servers from the `$INFORMIXDIR/etc/sqlhosts` file on UNIX or the `sqlhosts` entry in the registry on Windows NT. The first entry is highlighted. Press `RETURN` to select that database server or select a different database server. If entries span more than one page, use the arrow keys to move the cursor to highlight the ellipses (...), and the next page of database servers appears.

Permissions Needed

To access a specific database, you must have permission. If you do not have permission to connect to the specified database server, an error message appears. Select a different database server for which you have permission or ask your database server administrator for permission to connect to the database server.

If you have the correct permissions for the specified database server, you are prompted to specify a database to use on that database server.

When the `SELECT DATABASE` screen appears, the name of the specified database server is highlighted, as [Figure 6-5](#) shows.

```
SELECT DATABASE >>
Select a database with the Arrow Keys, or enter a name, then press Return.
----- @coral ----- Press CTRL-W for Help -----
borabora@coral
huahine@coral
moorea@coral
```

Figure 6-5
The
`SELECT DATABASE`
Screen

The `SELECT DATABASE` screen alphabetically lists all available databases on the specified database server. The database list on the `SELECT DATABASE` screen depends on the current connection. For example:

- If no current connection exists or the current connection is an implicit default connection, all the databases listed in the `DBPATH` environment variable setting are displayed.
- If a current explicit connection exists, all the databases in the `DBPATH` that pertain to the current server are displayed.

For information on setting the **DBPATH** environment variable, see the [Informix Guide to SQL: Reference](#).

To select a database, type the database name or use the arrow keys to highlight the name of a database, then press RETURN.

If you enter the CONNECT menu with a current connection, and the new connection succeeds, DB-Access disconnects from the previous environment and closes any databases that belong to that environment. For more information, see [“Implicit Closures” on page 6-7](#).

If you enter the name of a nonexistent database or a database that DB-Access cannot locate, an error message appears.

To leave the SELECT DATABASE screen without selecting a database, press the Interrupt key. You return to the CONNECTION menu, but your database server connection is not severed.

Implicit Closures

DB-Access closes any open connections or databases when you connect to a new environment, in the following situations:

- When you connect to a new database environment without explicitly disconnecting from the current one, DB-Access performs an implicit disconnect and the database closes.
- When you connect to a database@server and then close the database, the database server remains connected.
- When you connect to a database server, open a database, and then close the database, the database server remains connected.
- If you open a database and then try to connect to a database server, DB-Access performs an implicit disconnect and closes the database.

Only one connection is allowed. You must disconnect from the database server associated with the open database or close the database before you can connect to another database server.

If DB-Access needs to close a database that still has outstanding transactions, it prompts you to commit or rollback those transactions, as described in [“Transaction Processing” on page 6-8](#).

Disconnecting from a Database Environment

To disconnect from the current database server and close the current database, choose the Disconnect option from the CONNECTION menu. The DISCONNECT confirmation menu appears, as [Figure 6-6](#) shows.

```
DISCONNECT:  Yes  No
Disconnect from the current database environment.

----- moorea@cora] ----- Press CTRL-W for Help -----
```

Figure 6-6
The
DISCONNECT
Confirmation Menu

When you select the Disconnect option from the CONNECTION menu, you must confirm your decision on the DISCONNECT confirmation screen. The following two options are available:

- To confirm that you want to disconnect, press RETURN with the default **Yes** option highlighted. DB-Access disconnects from the database server and closes the database.
- If you do not want to disconnect, press the **N** key or use the right arrow key to highlight **No**, and press RETURN. DB-Access returns to the CONNECTION menu.

Transaction Processing

A database that has transaction logging prompts you to confirm or roll back any transactions when you explicitly disconnect from the current database environment or when you connect to another environment, which forces DB-Access to close an open database. The TRANSACTION menu appears, as [Figure 6-7](#) shows.

```
TRANSACTION:  Commit  Rollback
Commit the current transaction.

----- moorea@cora] ----- Press CTRL-W for Help -----
```

Figure 6-7
The *TRANSACTION*
Menu for Databases
with Transactions

The TRANSACTION menu ensures that you either commit or roll back an active transaction before you close the current database. You have the following menu options:

- The default is Commit. Press RETURN, and DB-Access commits the transaction and closes the database.
- If you want to roll back the transaction, use an arrow key to move the highlight to the Rollback option. Press RETURN and DB-Access rolls back the transaction and closes the database.



Warning: Select an option carefully. You might commit transactions that you do not want if you select Commit. You will lose any new transactions if you select Rollback.

If you press the Interrupt key, DB-Access displays the DATABASE menu without committing or rolling back the transaction.

Choosing the Session Option

From the main menu, press the S key or highlight the Session option and press RETURN. The DB-Access main menu remains on the screen and information about the current DB-Access session appears. For example, [Figure 6-8](#) shows the following information:

- The name of the database server for this session is **coral**.
- The database server type is Dynamic Server. This server type refers to all Informix database servers that are based on dynamic scalable architecture. For example, you would see Dynamic Server if you use Dynamic Server.
- The server is connected to a host named **carrots**.
- The server is a multithreaded server.

Choosing the Session Option

```
DB-Access: [Query-language Connection Database Table] [Session] Exit
Retrieve information about the current DB-Access session.

----- @coral ----- Press CTRL-W for Help -----

Server
coral
OnLine
Connected to host carrots
Multi-threaded
```

Figure 6-8
*Main Menu with
Sample Session
Information for a
Dynamic Server
Instance*

NLS

If you are running a legacy database that supports Native Language Support, the Session option shows the collating sequence and character type. ♦

GLS

The Session option does not display Global Language Support attributes, but you can use the method shown in [“Retrieving Nondefault Locale Information” on page 4-11](#) to obtain these settings. ♦

To exit from the Session information screen, select another option on the main menu.

How to Read On-Line Help for SQL Statements

This appendix shows the conventions that are used to represent the syntax of SQL statements in DB-Access on-line Help screens. You can request on-line help for SQL statements in either of the following ways:

- Highlight the New, Modify, or Use-editor options on the SQL menu and press CTRL-W.
- Press CTRL-W while you are on the NEW or MODIFY screens of the SQL menu.

The form of the syntax diagrams that appears when you request on-line Help for SQL statements in DB-Access is different from the syntax diagrams in the *Informix Guide to SQL: Syntax*.

The conventions and rules governing SQL statement syntax in DB-Access on-line Help screens are described in the following list.

ABC Any term in an SQL statement displayed in uppercase letters is a keyword. Type keywords exactly, disregarding case, as shown in the following example:

```
CREATE SYNONYM synonym-name
```

This syntax indicates you must type the keywords `CREATE SYNONYM` or `create synonym` without adding or deleting spaces or letters.

abc Substitute a value for any term that appears in lowercase letters. In the previous example, you should substitute a value for *synonym-name*.

() Type any parentheses as shown. They are part of the syntax of an SQL statement and are not special symbols.

[] Do not type brackets as part of a statement. They surround any part of a statement that is optional. For example:

```
CREATE [TEMP] TABLE
```

This syntax indicates that you can type either `CREATE TABLE` or `CREATE TEMP TABLE`.

(1 of 2)

| The vertical bar indicates a choice among several options. For example:

```
[VANILLA | CHOCOLATE [MINT] | STRAWBERRY]
```

This syntax indicates that you can enter either VANILLA, CHOCOLATE, or STRAWBERRY and that, if you enter CHOCOLATE, you can also enter MINT.

{ } When you must choose only one of several options, the options are enclosed in braces and are separated by vertical bars. For example:

```
{GUAVA | MANGO | PASSIONFRUIT}
```

This syntax indicates that you must enter either GUAVA, MANGO, or PASSIONFRUIT, but you cannot enter more than one choice.

... An ellipsis indicates that you can enter an indefinite number of additional items, such as the one immediately preceding the ellipsis. For example:

```
old-column-name  
...
```

This syntax indicates that you can enter a series of existing column names after the first one.

(2 of 2)

The *Informix Guide to SQL: Syntax* contains more detailed syntax diagrams, as well as instructions for interpreting the diagram format used in that book.

For a general explanation of how to use on-line Help in DB-Access, see “Using the HELP Screen” on page 2-8.

Demonstration SQL

This appendix shows the contents of the various command files that are available with DB-Access. These command files all have the extension **.sql** when displayed from the command line but appear without the extension on the SQL CHOOSE menu.

Keywords in these command files are shown in uppercase letters to make the SQL statements easier to read. Keywords in the actual command files are lowercase.



Important: *Although the command files are listed alphabetically in this appendix, you cannot execute the command files that create tables in that order without causing errors. The order in which the tables are created is very important because of the referential constraints that link those tables.*

When you select the Choose option on the SQL menu, the CHOOSE screen appears. It displays a list of the command files that you can access, similar to the display that [Figure B-1 on page B-2](#) shows. These files are included with the **stores_demo** database. Other **.sql** files are discussed later in this appendix.

Figure B-1
Command Files Listed on the CHOOSE Screen

```
CHOOSE >>
Choose a command file with the Arrow Keys, or enter a name, then press Return.

----- stores_demo @dbserver1 ----- Press CTRL-W for Help -----

alt_cat      c_state      d_trig       sel_ojoin1
c_calls      c_stock      d_view       sel_ojoin2
c_cat        c_stores_demo del_stock     sel_ojoin3
c_custom     c_table      ins_table    sel_ojoin4
c_index      c_trig       opt_disk     sel_order
c_items      c_type       sel_agg      sel_sub
c_manuf      c_view1      sel_all      sel_union
c_orders     c_view2      sel_group    upd_table
c_proc       d_proc       sel_join
```

If you do not see the command files included with your demonstration database, check the following:

1. Did you copy the demonstration SQL command files to your current directory when you ran the demonstration database initialization script? If not, you can rerun the initialization script to copy them.
2. Did you start DB-Access from the directory in which you installed the demonstration SQL command files? If not, exit DB-Access, change to the appropriate directory, and start DB-Access again.

For instructions on running the initialization script, copying command files, and starting DB-Access, refer to [Chapter 1, “Getting Started with DB-Access.”](#)

Use these command files with DB-Access for practice with SQL and the demonstration database. You can rerun the demonstration database initialization script whenever you want to refresh the database tables and SQL files.

SQL Files for the Relational Database Model

The SQL files described in this section are included with the **stores_demo** demonstration database and conform to the **stores_demo** schema. For more information about the **stores_demo** schema, refer to the [Informix Guide to SQL: Reference](#).

alt_cat.sql

The following command file alters the **catalog** table. It drops the existing constraint **aa** on the **catalog** table and adds a new constraint, **ab**, which specifies cascading deletes. You can use this command file and then the **del_stock.sql** command file for practice with cascading deletes on a database with logging.

```
ALTER TABLE catalog DROP CONSTRAINT aa;

ALTER TABLE catalog ADD CONSTRAINT
    (FOREIGN KEY (stock_num, manu_code) REFERENCES stock
     ON DELETE CASCADE CONSTRAINT ab);
```

c_calls.sql

The following command file creates the **cust_calls** table:

```
CREATE TABLE cust_calls
(
    customer_num          INTEGER,
    call_dtime            DATETIME YEAR TO MINUTE,
    user_id               CHAR(18) DEFAULT USER,
    call_code             CHAR(1),
    call_descr            CHAR(240),
    res_dtime             DATETIME YEAR TO MINUTE,
    res_descr            CHAR(240),
    PRIMARY KEY(customer_num, call_dtime),
    FOREIGN KEY(customer_num) REFERENCES customer(customer_num),
    FOREIGN KEY(call_code) REFERENCES call_type (call_code)
);
```

c_cat.sql

The following command file creates the **catalog** table. It contains a constraint, **aa**, which allows you to practice with cascading deletes by running the SQL statements in the **alt_cat.sql** and **del_stock.sql** command files on a database with logging.

```
CREATE TABLE catalog
(
  catalog_num      SERIAL(10001),
  stock_num        SMALLINT NOT NULL,
  manu_code        CHAR(3) NOT NULL,
  cat_descr        TEXT,
  cat_picture      BYTE,
  cat_advert       VARCHAR(255, 65),
  PRIMARY KEY (catalog_num),
  FOREIGN KEY (stock_num, manu_code) REFERENCES stock
  CONSTRAINT aa
);
```

c_custom.sql

The following command file creates the **customer** table:

```
CREATE TABLE customer
(
  customer_num     SERIAL(101),
  fname            CHAR(15),
  lname            CHAR(15),
  company          CHAR(20),
  address1         CHAR(20),
  address2        CHAR(20),
  city             CHAR(15),
  state            CHAR(2),
  zipcode          CHAR(5),
  phone            CHAR(18),
  PRIMARY KEY (customer_num)
);
```

c_index.sql

The following command file creates an index on the **zipcode** column of the **customer** table:

```
CREATE INDEX zip_ix ON customer (zipcode);
```

c_items.sql

The following command file creates the **items** table:

```
CREATE TABLE items
(
  item_num          SMALLINT,
  order_num         INTEGER,
  stock_num         SMALLINT NOT NULL,
  manu_code         CHAR(3) NOT NULL,
  quantity          SMALLINT CHECK (quantity >= 1),
  total_price       MONEY(8),
  PRIMARY KEY (item_num, order_num),
  FOREIGN KEY (order_num) REFERENCES orders (order_num),
  FOREIGN KEY (stock_num, manu_code) REFERENCES stock
    (stock_num, manu_code)
);
```

c_manuf.sql

The following command file creates the **manufact** table:

```
CREATE TABLE manufact
(
  manu_code         CHAR(3),
  manu_name         CHAR(15),
  lead_time         INTERVAL DAY(3) TO DAY,
  PRIMARY KEY (manu_code)
);
```

c_orders.sql

The following command file creates the **orders** table:

```
CREATE TABLE orders
(
  order_num         SERIAL(1001),
  order_date        DATE,
  customer_num      INTEGER NOT NULL,
  ship_instruct     CHAR(40),
  backlog           CHAR(1),
  po_num            CHAR(10),
  ship_date         DATE,
  ship_weight       DECIMAL(8,2),
  ship_charge       MONEY(6),
  paid_date         DATE,
  PRIMARY KEY (order_num),
  FOREIGN KEY (customer_num) REFERENCES customer (customer_num)
);
```

c_proc.sql

The following command file creates an SPL routine. It reads the full name and address of a customer and takes a last name as its only argument.

This routine shows the legacy use of CREATE PROCEDURE.

To conform with the SQL standard preferred with Dynamic Server, define a *function* if you want to return values from a routine. ♦

```
CREATE PROCEDURE read_address (lastname CHAR(15))
RETURNING CHAR(15),CHAR(15),CHAR(20),CHAR(15),CHAR(2),CHAR(5);
DEFINE p_fname, p_city CHAR(15);
DEFINE p_add CHAR(20);
DEFINE p_state CHAR(2);
DEFINE p_zip CHAR(5);
SELECT fname, address1, city, state, zipcode
      INTO p_fname, p_add, p_city, p_state, p_zip
      FROM customer
      WHERE lname = lastname;

      RETURN p_fname, lastname, p_add, p_city, p_state, p_zip;

END PROCEDURE;
```

c_state

The following command file creates the **state** table:

```
CREATE TABLE state
(
  code           CHAR(2),
  sname         CHAR(15),
  PRIMARY KEY (code)
);
```

c_stock.sql

The following command file creates the **stock** table:

```
CREATE TABLE stock
(
  stock_num          SMALLINT,
  manu_code          CHAR(3),
  description        CHAR(15),
  unit_price         MONEY(6),
  unit              CHAR(4),
  unit_descr        CHAR(15),
  PRIMARY KEY (stock_num, manu_code),
  FOREIGN KEY (manu_code) REFERENCES manufact
);
```

c_stores.sql

The following command file creates the **stores_demo** database:

```
CREATE DATABASE stores_demo;
```

c_table.sql

The following command file creates a database named **restock** and then creates a custom table named **sports** in that database:

```
CREATE DATABASE restock;

CREATE TABLE sports
(
  catalog_no        SERIAL UNIQUE,
  stock_no          SMALLINT,
  mfg_code          CHAR(5),
  mfg_name          CHAR(20),
  phone             CHAR(18),
  descript          VARCHAR(255)
);
```

c_trig.sql

The following command file creates a table named **log_record** and then creates a trigger named **upqty_i**, which updates it:

```
CREATE TABLE log_record
  (item_num      SMALLINT,
   ord_num       INTEGER,
   username      CHARACTER(8),
   update_time   DATETIME YEAR TO MINUTE,
   old_qty       SMALLINT,
   new_qty       SMALLINT);

CREATE TRIGGER upqty_i
  UPDATE OF quantity ON items
  REFERENCING OLD AS pre_upd
              NEW AS post_upd
  FOR EACH ROW(INSERT INTO log_record
    VALUES (pre_upd.item_num, pre_upd.order_num, USER,
    CURRENT,
            pre_upd.quantity, post_upd.quantity));
```

XPS

Extended Parallel Server does not support triggers. ♦

c_type.sql

The following command file creates the **call_type** table:

```
CREATE TABLE call_type
  (
   call_code      CHAR(1),
   code_descr     CHAR(30),
   PRIMARY KEY (call_code)
  );
```

c_view1.sql

The following command file creates a view called **custview** on a single table and grants privileges on the view to **public**. It includes the **WITH CHECK OPTION** keywords to verify that any changes made to underlying tables through the view do not violate the definition of the view.

```
CREATE VIEW custview (firstname, lastname, company, city) AS
  SELECT fname, lname, company, city
     FROM customer
     WHERE city = 'Redwood City'
     WITH CHECK OPTION;

GRANT DELETE, INSERT, SELECT, UPDATE
  ON custview
  TO public;
```

c_view2.sql

The following command file creates a view on the **orders** and **items** tables:

```
CREATE VIEW someorders (custnum, ocustnum, newprice) AS
  SELECT orders.order_num, items.order_num,
         items.total_price*1.5
     FROM orders, items
     WHERE orders.order_num = items.order_num
     AND items.total_price > 100.00;
```

d_proc.sql

The following command file drops the SPL routine that the **c_proc.sql** command file created:

```
DROP PROCEDURE read_address;
```

d_trig.sql

The following command file drops the trigger that the **c_trig.sql** command file created:

```
DROP TRIGGER upqty_i;
```

d_view.sql

The following command file drops the view named **custview** that the **c_view1.sql** command file created:

```
DROP VIEW custview;
```

del_stock.sql

The following command file deletes rows from the **stock** table where the stock number is 102. This delete will cascade to the **catalog** table (although the related manufacturer codes will remain in the **manufact** table). The **del_stock.sql** command file can be used following the **alt_cat.sql** command file for practice with cascading deletes on a database with logging.

```
DELETE FROM stock WHERE stock_num = 102;
```

After running the SQL statements in the **alt_cat.sql** and **del_stock.sql** command files, issue the following query on the **catalog** table to verify that the rows were deleted:

```
SELECT * FROM catalog WHERE stock_num = 102;
```

The **stores_demo** database has been changed. You might want to rerun the **dbaccessdemo** script to rebuild the original database.

ins_table.sql

The following command file inserts one row into the **sports** table that the **c_table.sql** command file created:

```
INSERT INTO sports
VALUES (0,18,'PARKR', 'Parker Products', '503-555-1212',
'Heavy-weight cotton canvas gi, designed for aikido or
judo but suitable for karate. Quilted top with side ties,
drawstring waist on pants. White with white belt.
Pre-washed for minimum shrinkage. Sizes 3-6.');
```

opt_disk.sql

The following command file provides an example of a SELECT statement on an optical-disc subsystem. It includes the read-only **family()** and **volume()** operators that support optical storage. (This is available only with the Optical Subsystem.)

The query generates a list of the volumes that contain pictures of bicycle helmets. One row of output (family, volume) is generated for each data row that contains a picture of a bicycle helmet. The **family()** operator returns the name of the optical family where an optical blob column is stored, and **volume()** returns the number of the volume where an optical blob column is stored. These functions are valid only for data stored on optical media.

```
SELECT family(cat_picture), volume(cat_picture)
   FROM catalog
  WHERE stock_num = 110;
```

sel_agg.sql

The SELECT statement in the following command file queries on table data using aggregate functions. It combines the aggregate functions MAX and MIN in a single statement.

```
SELECT MAX (ship_charge), MIN (ship_charge)
   FROM orders;
```

sel_all.sql

The following example command file contains all seven SELECT statement clauses that you can use in the Informix implementation of interactive SQL. This SELECT statement joins the **orders** and **items** tables. It also uses display labels, table aliases, and integers as column indicators; groups and orders the data; and puts the results into a temporary table.

```
SELECT o.order_num, SUM (i.total_price) price,  
       paid_date - order_date span  
FROM orders o, items i  
WHERE o.order_date > '01/01/90'  
       AND o.customer_num > 110  
       AND o.order_num = i.order_num  
GROUP BY 1, 3  
HAVING COUNT (*) < 5  
ORDER BY 3  
INTO TEMP temptabl;
```

sel_group.sql

The following example command file includes the GROUP BY and HAVING clauses. The HAVING clause usually complements a GROUP BY clause by applying one or more qualifying conditions to groups after they are formed, which is similar to the way the WHERE clause qualifies individual rows. (One advantage to using a HAVING clause is that you can include aggregates in the search condition; you cannot include aggregates in the search condition of a WHERE clause.)

Each HAVING clause compares one column or aggregate expression of the group with another aggregate expression of the group or with a constant. You can use the HAVING clause to place conditions on both column values and aggregate values in the group list.

```
SELECT order_num, COUNT(*) number, AVG (total_price) average  
FROM items  
GROUP BY order_num  
HAVING COUNT(*) > 2;
```

sel_join.sql

The following example command file uses a simple join on the **customer** and **cust_calls** tables. This query returns only those rows that show the customer has made a call to customer service.

```
SELECT c.customer_num, c.lname, c.company,
       c.phone, u.call_dtime, u.call_descr
FROM customer c, cust_calls u
WHERE c.customer_num = u.customer_num;
```

sel_ojoin1.sql

The following example command file uses a simple outer join on two tables. The use of the keyword **OUTER** in front of the **cust_calls** table makes it the subservient table. An outer join causes the query to return information on *all* customers, even if they do not make calls to customer service. All rows from the dominant **customer** table are retrieved, and null values are assigned to corresponding rows from the subservient **cust_calls** table.

```
SELECT c.customer_num, c.lname, c.company,
       c.phone, u.call_dtime, u.call_descr
FROM customer c, OUTER cust_calls u
WHERE c.customer_num = u.customer_num;
```

sel_ojoin2.sql

The following example command file creates an outer join, which is the result of a simple join to a third table. This second type of outer join is called a *nested simple join*.

This query first performs a simple join on the **orders** and **items** tables, retrieving information on all orders for items with a **manu_code** of **KAR** or **SHM**. It then performs an outer join, which combines this information with data from the dominant **customer** table. An optional **ORDER BY** clause reorganizes the data.

```
SELECT c.customer_num, c.lname, o.order_num,
       i.stock_num, i.manu_code, i.quantity
FROM customer c, OUTER (orders o, items i)
WHERE c.customer_num = o.customer_num
      AND o.order_num = i.order_num
      AND manu_code IN ('KAR', 'SHM')
ORDER BY lname;
```

sel_ojoin3.sql

The following example SELECT statement is the third type of outer join, known as a *nested outer join*. It queries on table data by creating an outer join, which is the result of an outer join to a third table.

This query first performs an outer join on the **orders** and **items** tables, retrieving information on all orders for items with a **manu_code** of KAR or SHM. It then performs an outer join, which combines this information with data from the dominant **customer** table. This query preserves order numbers that the previous example eliminated, returning rows for orders that do not contain items with either manufacturer code. An optional ORDER BY clause reorganizes the data.

```
SELECT c.customer_num, lname, o.order_num,
       stock_num, manu_code, quantity
FROM customer c, OUTER (orders o, OUTER items i)
WHERE c.customer_num = o.customer_num
      AND o.order_num = i.order_num
      AND manu_code IN ('KAR', 'SHM')
ORDER BY lname;
```

sel_ojoin4.sql

The following example queries on table data using the fourth type of outer join. This query shows an outer join, which is the result of an outer join of each of two tables to a third table. In this type of outer join, join relationships are possible *only* between the dominant table and subservient tables.

This query individually joins the subservient tables **orders** and **cust_calls** to the dominant **customer** table but does not join the two subservient tables. (An INTO TEMP clause selects the results into a temporary table.)

```
SELECT c.customer_num, lname, o.order_num,
       order_date, call_dtime
FROM customer c, OUTER orders o, OUTER cust_calls x
WHERE c.customer_num = o.customer_num
      AND c.customer_num = x.customer_num
INTO temp service;
```

sel_order.sql

The following example uses the ORDER BY and WHERE clauses to query. In this SELECT statement, the comparison 'bicycle%' (LIKE condition, or 'bicycle*' for a MATCHES condition) specifies the letters bicycle followed by any sequence of zero or more characters. It narrows the search further by adding another comparison condition that excludes a **manu_code** of PRC.

```
SELECT * FROM stock
      WHERE description LIKE 'bicycle%'
             AND manu_code NOT LIKE 'PRC'
      ORDER BY description, manu_code;
```

sel_sub.sql

The following example uses a subquery to query. This self-join uses a correlated subquery to retrieve and list the 10 highest-priced items ordered.

```
SELECT order_num, total_price
      FROM items a
      WHERE 10 >
             (SELECT COUNT (*)
              FROM items b
              WHERE b.total_price < a.total_price)
      ORDER BY total_price;
```

sel_union.sql

The following example uses the UNION clause to query on data in two tables. The compound query performs a union on the **stock_num** and **manu_code** columns in the **stock** and **items** tables. The statement selects items that have a unit price of less than \$25.00 or that have been ordered in quantities greater than three, and it lists their **stock_num** and **manu_code**.

```
SELECT DISTINCT stock_num, manu_code
      FROM stock
      WHERE unit_price < 25.00

UNION

SELECT stock_num, manu_code
      FROM items
      WHERE quantity > 3;
```

upd_table.sql

The following example updates the **sports** table that the **c_table.sql** command file created:

```
UPDATE sports
  SET phone = '808-555-1212'
  WHERE mfg_code = 'PARKR';
```

SQL Files for the Dimensional Database Model

The **sales_demo** database illustrates a dimensional schema for data-warehousing applications. This database model alters the **stores_demo** schema and data. The success of the files in this section requires two prerequisites:

- You must first create a **stores_demo** database with the following command:

```
dbaccessdemo -log
```
- The **createdw.sql** and **loaddw.sql** files must be in the same directory as the files with extension **.unl** that the **loaddw.sql** uses. ♦

createdw.sql

This file creates the new **sales_demo** database with logging and then creates tables within that database. It contains the following statements:

```
create database sales_demo with log;

create table product (
    product_code integer,
    product_name char(31),
    vendor_code char(3),
    vendor_name char(15),
    product_line_code smallint,
    product_line_name char(15));

create table customer (
    customer_code integer,
    customer_name char(31),
    company_name char(20));

create table sales (
    customer_code integer,
    district_code smallint,
    time_code integer,
    product_code integer,
    units_sold smallint,
    revenue money (8,2),
    cost money (8,2),
    net_profit money(8,2));

create table time
(
    time_code int,
    order_date date,
    month_code smallint,
    month_name char(10),
    quarter_code smallint,
    quarter_name char(10),
    year integer
);

create table geography (
    district_code serial,
    district_name char(15),
    state_code char(2),
    state_name char(18),
    region smallint);
```

loaddw.sql

This file contains the commands necessary to load data from two sources:

- The files with the extension **.unl** in your demonstration directory
- Data selected from the **stores_demo** database

These SQL statements in **loaddw.sql** accomplish these actions:

```
connect to "stores_demo ";
load from "add_orders.unl"
    insert into stores_demo :orders;
load from 'add_items.unl'
    insert into stores_demo :items;

connect to "sales_demo";
load from 'costs.unl'
    insert into cost;
load from 'time.unl'
    insert into time;

insert into geography(district_name, state_code, state_name)
    select distinct c.city, s.code, s.sname
from stores_demo :customer c, stores_demo :state s
    where c.state = s.code;
update geography-- converts state_code values to region
values
    set region = 1
    where state_code = "CA";
update geography
    set region = 2
    where state_code <> "CA";

insert into customer (customer_code, customer_name,
company_name)
    select c.customer_num, trim(c.fname) || " " || c.lname,
c.company
    from stores_demo :customer c;

insert into product (product_code, product_name, vendor_code,
vendor_name, product_line_code, product_line_name)
    select a.catalog_num,
        trim(m.manu_name) || " "|| s.description,
        m.manu_code, m.manu_name, s.stock_num, s.description
    from stores_demo :catalog a, stores_demo :manufact m,
        stores_demo :stock s
    where a.stock_num = s.stock_num and
        a.manu_code = s.manu_code and
        s.manu_code = m.manu_code;
```

```

insert into sales (customer_code, district_code,
                 time_code, product_code,
                 units_sold, revenue, cost, net_profit)
select c.customer_num, g.district_code, t.time_code,
p.product_code,
SUM(i.quantity), SUM(i.total_price),
SUM(i.quantity * x.cost),
SUM(i.total_price) - SUM(i.quantity * x.cost)
from stores_demo :customer c, geography g, time t,
     product p,
     stores_demo :items i, stores_demo :orders o, cost x
where c.customer_num = o.customer_num and
     o.order_num = i.order_num and
     p.product_line_code = i.stock_num and
     p.vendor_code = i.manu_code and
     t.order_date = o.order_date and
     p.product_code = x.product_code and
     c.city = g.district_name
GROUP BY 1,2,3,4;

connect to "stores_demo ";
load from 'add_orders.unl'
     insert into stores_demo :orders;
load from 'add_items.unl'
     insert into stores_demo :items;

connect to "sales_demo";
load from 'costs.unl'
     insert into cost;
load from 'time.unl'
     insert into time;

insert into geography(district_name, state_code, state_name)
select distinct c.city, s.code, s.sname
from stores_demo :customer c, stores_demo :state s
where c.state = s.code;
update geography-- converts state_code values to region
values
set region = 1
where state_code = "CA";
update geography
set region = 2
where state_code <> "CA";

insert into customer (customer_code, customer_name,
                    company_name)
select c.customer_num, trim(c.fname) || " " || c.lname,
c.company
from stores_demo :customer c;

```

```
insert into product (product_code, product_name, vendor_code,
    vendor_name, product_line_code, product_line_name)
select a.catalog_num,
    trim(m.manu_name) || " " || s.description,
    m.manu_code, m.manu_name, s.stock_num, s.description
from stores_demo :catalog a, stores_demo :manufact m,
    stores_demo :stock s
where a.stock_num = s.stock_num and
    a.manu_code = s.manu_code and
    s.manu_code = m.manu_code;

insert into sales (customer_code, district_code,
    time_code, product_code,
    units_sold, revenue, cost, net_profit)
select c.customer_num, g.district_code, t.time_code,
p.product_code,
    SUM(i.quantity), SUM(i.total_price),
    SUM(i.quantity * x.cost),
    SUM(i.total_price) - SUM(i.quantity * x.cost)
from stores_demo :customer c, geography g, time t, product
p,
    stores_demo :items i, stores_demo :orders o, cost x
where c.customer_num = o.customer_num and
    o.order_num = i.order_num and
    p.product_line_code = i.stock_num and
    p.vendor_code = i.manu_code and
    t.order_date = o.order_date and
    p.product_code = x.product_code and
    c.city = g.district_name
GROUP BY 1,2,3,4;
```

User-Defined Routines for the Object-Relational Database Model

The **superstores_demo** database does not replace the **stores_demo** database. Both databases are available. The **superstores_demo** database schema is not backward compatible with **stores_demo**. In many cases, you cannot use test queries developed for **stores_demo** against the tables of **superstores_demo** because the tables differ.

No SQL command files are associated specifically with **superstores_demo**. However, there are user-defined routines that you can use with the screens described in [Chapter 3, “The Query-language Option.”](#)

The **superstores_demo** database includes examples of the following new features:

- Collection types: SET, LIST
- Named row types: location_t, loc_us_t, loc_non_us_t
- Unnamed row types
- Type and table inheritance
- Built-in data types: BOOLEAN, SERIAL8, INT8
- Distinct data type: percent
- Smart large objects: BLOB and CLOB

The **superstores_demo** database has row types and tables to support the following table-inheritance hierarchies:

- customer/retail_customer
- customer/whlsale_customer
- location/location_us
- location/location_non_us

For a complete description of the **superstores_demo** tables and inheritance hierarchies, see the [Informix Guide to SQL: Reference](#). For more information on user-defined routines, see [Extending Informix Dynamic Server 2000](#).

Index

A

Access method, index 5-51
 ADD CHECK VALUE menu 5-37
 ADD DEFAULT TYPE menu 5-40
 ADD DEFAULT VALUE
 menu 5-42
 ADD NULLS menu 5-13
 ADD TYPE menu 5-9
 ALTER FRAGMENT menu 5-23
 no strategy 5-28
 round-robin 5-23
 ALTER ROWID menu 5-31
 ALTER TABLE screen 5-5
 ANSI compliance
 checking SQL statements for 1-23
 level Intro-14
 ANSI database 4-17
 ANSI-compliant database
 and SQLSTATE value 1-23
 specifying 4-7
 transaction logging 4-8
 ANSI, checking SQL statements for
 compliance 1-23
 Arrow keys 2-4

B

BACKSPACE key 2-4
 BLOB data type 5-47, 5-48
 Blobspace
 adding column for 5-11
 data types in Columns
 display 5-48
 specifying for TEXT or BYTE
 data 5-11

Boldface type Intro-6
 BOOLEAN data type 5-47
 Buffered logging 4-8
 Built-in data type 5-47
 BYTE data type 5-47, 5-48

C

Cascading deletes, enabling 5-35
 CHAR data type 5-47
 CHARACTER VARYING data
 type 5-47
 CHECK CONSTRAINTS
 menu 5-36
 chmod command 1-13
 Choose option (SQL menu) 3-21
 CHOOSE screen 3-22
 CLOB data type 5-47, 5-48
 CLOSE DATABASE statement 4-17
 Closing a database
 from a menu 4-16
 with active transactions 6-8
 with the Disconnect option 6-8
 Code set, ISO 8859-1 Intro-4
 Code, sample, conventions
 for Intro-10
 Collection data type 5-50
 LIST 5-50
 MULTISET 5-50
 SET 5-50
 Column
 allowing null values with the
 schema editor 5-13
 constraints for 5-32

- data types
 - adding 5-9
 - description 4-18, 5-62
 - selection 5-9
- data validation of default values 5-43
- default literal value 5-41
- default type, null 5-41
- defining 5-8
- displaying data type and nulls with the Info option 5-47
- displaying information with the Columns option 5-47
- dropping from a table with the schema editor 5-16
- indexing 5-12
- modifying 5-15
- null default value 5-41
- removing from the schema 5-16
- Column Name 5-9
- Command files
 - choosing (CHOOSE screen) 3-21
 - executing from the command line 1-22
 - modifying with an editor 2-9
 - rules for naming 3-23
 - saving 3-23
 - supplied SQL command files B-1
- Command line
 - additional features 1-24
 - interactive input through standard input 1-25
 - reading from standard input 1-24
 - syntax for invoking
 - DB-Access 1-14
 - using the -ansi flag 3-10
- Command options. *See* dbaccess command options.
- Command window should not have scroll bars 1-14
- Command-line conventions, elements of Intro-9
- Command-line options, displaying the main menu 1-15
- Comment icons Intro-7
- COMMIT 4-17
- COMMIT statement 4-17

- Committing transactions with the TRANSACTION menu 4-17, 6-9
- Compliance with industry standards Intro-14
- Confirmation screen
 - creating a database 4-9
 - dropping a command file 3-27
 - dropping a database 4-16
- CONNECT statement 1-26
- Connecting to a database environment 6-3
- Connecting to database environment in background mode 1-27
- CONNECTION menu 6-3, 6-4
 - options 1-18
 - PASSWORD prompt screen 6-5
 - USER NAME prompt screen 6-4
- Constraints
 - adding with CREATE TABLE 5-32
 - check, defining 5-36
 - default values, defining 5-39
 - defining 5-32
 - displaying 5-54
 - enabling cascading deletes 5-35
 - foreign key, defining 5-34
 - foreign key, information on 5-55
 - primary key, defining 5-32
 - primary key, information on 5-56
 - reference, information on 5-55
 - unique, defining 5-38
- CONSTRAINTS menu
 - options on 5-54
 - Primary option 5-56
 - Reference option 5-55
 - shown 5-32, 5-54
- Contact information Intro-14
- CONTROL key 2-4
- Conventions
 - documentation Intro-5
 - on-line Help A-1
- CREATE DATABASE menu 4-7
- Create option, TABLE menu 5-4

- CREATE TABLE menu 5-7
 - building schema on exit 5-14
 - exiting 5-14
 - Modify option, changing a column 5-15
- Creating a database from a menu 4-7
 - restrictions on naming 4-7
- Creating a table
 - fragmented (EXPRESSION menu) 5-21
 - fragmented (ROUND_ROBIN menu) 5-20
 - with the schema editor 5-4
- CTRL-A 3-7
- CTRL-D 1-25, 3-7
- CTRL-R 3-8
- CTRL-W 2-8, A-1
- CTRL-X 3-8
- Current statement, definition of 3-3
- Cursor
 - moving on the terminal screen 2-4
 - position when using SQL editor 3-7
 - where displayed on text-entry screens 2-7

D

- Data types
 - ADD TYPE menu options for 5-9
 - adding column 5-9
 - BLOB 5-47, 5-48, B-21
 - BOOLEAN 5-47, B-21
 - built-in 5-47
 - BYTE 5-47, 5-48
 - CHAR 5-47
 - CHARACTER VARYING 5-47
 - CLOB 5-47, 5-48, B-21
 - collection 5-50, B-21
 - DATE 5-47
 - DATETIME 5-47
 - DECIMAL 5-47
 - defining for a column with the schema editor 5-9

- described 4-18, 5-62
- distinct B-21
- FLOAT 5-47
- INT8 5-47
- INTEGER 5-47
- INTERVAL 5-47
- LIST 5-50, B-21
- MONEY 5-47
- MULTISET 5-50
- NCHAR 5-47
- NVARCHAR 5-47
- opaque 5-49
- row 5-50, B-21
- SERIAL 5-47
- SERIAL8 5-47, B-21
- SET 5-50, B-21
- SMALLFLOAT 5-47
- SMALLINT 5-47
- TEXT 5-47, 5-48
- VARCHAR 5-47
- Data validation
 - default column values 5-43
 - unique constraints 5-39
- Data warehousing model. *See*
 - Demonstration database,
 - sales_demo.
- Database
 - cLose option 4-16
 - closing 4-16
 - DISCONNECT menu 6-8
 - with active transactions 6-8
 - closing implicitly 6-7
 - confirming your decision to
 - drop 4-16
 - creating 4-7
 - creating ANSI-compliant 4-8
 - current 4-3
 - displaying information for 4-9
 - dropping 4-15
 - naming 4-7
 - selecting 2-9
 - selecting from a menu 4-5
 - storing in a dbspace 4-7
- Database administrator 1-12
 - See* Administrative tasks.
- DATABASE INFO menu
 - dBspaces option 4-11
 - how to exit 4-11
 - options 4-10
 - Routines option 4-12
- Database information
 - dbspaces 4-11
 - routines 4-12
- DATABASE menu
 - available options 4-4
 - cLose option 4-16
 - CREATE DATABASE screen 4-7
 - DROP DATABASE screen 4-15
 - Drop option 4-15
 - Info option 4-9
 - options 1-18
 - SELECT DATABASE screen 4-5
 - Select option 4-5
 - selecting options from the
 - command line 1-18
 - the SELECT DATABASE
 - screen 4-5
- Database server
 - connecting to 6-4
 - disconnecting from 6-8
 - disconnecting implicitly 6-7
 - name, saving in column 5-41
 - SELECT DATABASE SERVER
 - screen 6-4
 - selecting from a menu 6-4
- Data, viewing the next page on the
 - screen 3-15
- DATE data type 5-47
- DATETIME data type 5-47
- DB-Access
 - environment variables
 - affecting 1-7
 - menu hierarchy 1-5
 - USER NAME prompt screen 6-4
 - what it is 1-3
- dbaccess command options
 - ansi 1-17
 - c 1-17
 - cc 1-18
 - cd 1-18
 - connect_menu_option 1-18
 - d 1-17
 - database 1-17
 - database_menu_option 1-18
 - dc 1-18
 - dd 1-18
 - di 1-18
 - dl 1-19
 - ds 1-19
 - e 1-17
 - filename 1-17
 - m 1-17
 - q 1-17
 - qc 1-19
 - qd 1-19
 - qi 1-19
 - qm 1-19
 - qn 1-19
 - qs 1-20
 - qu 1-20
 - query_menu_option 1-18
 - s 1-17
 - t 1-17
 - ta 1-20
 - table 1-18
 - table_menu_option 1-18
 - tc 1-20
 - td 1-20
 - ti 1-20
 - v or -V 1-17
 - X 1-17
 - See also* dbaccess, invoking.
- DB-Access utility 1-8
- dbaccess, invoking 1-13
- DBACCNOIGN environment
 - variable 1-7
- DBANSIWARN 1-23
- DBEDIT environment variable 1-7
- DBFLTMASK environment
 - variable 1-7
- DBPATH environment variable 6-6
- dbspace
 - ALTER FRAGMENT menu 5-23
 - defining expression for 5-22
 - deleting from existing
 - strategy 5-20
 - modifying current 5-20
 - Round-robin fragmentation
 - with 5-20
 - SELECT DBSPACE screen 4-8
 - specifying for table storage 5-18
 - storing a database 4-7
- DECIMAL data type 5-47

Default

- ADD DEFAULT TYPE menu 5-40
- ADD DEFAULT VALUE menu 5-42
- column type, null 5-41
- column values 5-41
- column values, data validation 5-43
- column values, defining 5-39
- column values, displaying 3-24
- column, adding 5-40
- column, data validation 5-43
- database server, selecting 6-6
- dbspace for database data 4-8
- defining values for columns 5-39
- dropping a check constraint 5-36
- dropping a foreign key 5-35
- for ADD DUPLICATES screen 5-12
- for ADD NAME screen 5-8
- for CREATE DATABASE confirmation screen 4-9
- for DISCONNECT confirmation screen 6-8
- for DROP COMMAND FILE confirmation screen 3-27
- for DROP TABLE confirmation screen 5-61
- for exiting CREATE DATABASE menu 4-9
- for LOCK MODE menu 5-31
- for LOG menu 4-8
- initial extent size 5-30
- length and scale, for number type 5-11
- length, for CHAR data type 5-11
- length, for MONEY data type 5-11
- length, for NCHAR data type 5-11
- number, for SERIAL data type 5-11
- operating system editor 3-5
- printer, sending output to 3-19
- table storage location 5-18
- Default locale Intro-4
- DEFAULTS menu 5-39
- DELETE key 2-5
- Deletes, enabling cascading 5-35

DELIMITED environment variable 1-7

- Demonstration database installing 1-9
- models 1-9
- reinitializing 1-9
- sales_demo 1-11
- SQL command files B-1
- stores_demo setup 1-10
- stores_demo, SQL command files B-3
- superstores_demo 1-9
- superstores_demo setup 1-11
- working directory required for 1-10
- Demonstration databases 1-8
- Dependencies, software Intro-4
- Dimensional database. *See* Demonstration database, sales_demo.
- Disconnecting from a database environment 6-8
- Disk space, managing with Extent Size screen 5-30
- Distinct data type B-21
- Distributed databases, requesting table information on another server 5-44
- Documentation notes Intro-13
- Documentation, types of documentation notes Intro-13
- error message files Intro-12
- machine notes Intro-13
- on-line help Intro-12
- on-line manuals Intro-11
- printed manuals Intro-11
- related reading Intro-14
- release notes Intro-13
- DROP DATABASE screen selecting from the DATABASE menu 4-15
- two ways to drop a database 4-15
- Drop option confirmation screen for dropping command files 3-27
- SQL menu 3-26
- Dropping a command file confirming your decision 3-27
- from a menu 3-26

- Dropping a database, from a menu 4-15
- Dropping a table confirming your decision 5-61
- from a menu 5-60
- Dropping an SQL statement confirming your decision 3-27
- from a menu 3-26

E

- Editor creating new SQL statements 3-7
- entering literal values 5-41
- modifying SQL statements 3-17
- restrictions 3-8
- text 2-9
- ENABLE CASCADING DELETES menu 5-35
- Environment variables DBACCNOIGN 1-7
- DBANSIWARN 1-23, 3-10
- DBEDIT 1-7
- DBFLTMASK 1-7
- DBPATH 3-22, 4-6
- DELIMITED 1-7
- LC_COLLATE 3-22
- ONCONFIG 4-6
- setting for default editor 3-4
- en_us.8859-1 locale Intro-4
- Error message terminal setup 1-14
- Error message files Intro-12
- Errors connecting to a server after opening a database 6-7
- correcting with the SQL editor 3-17
- editing in SQL statements 3-18
- entering a fill factor 5-13
- executing command files B-1
- modifying a constraint 5-39
- no CONNECT permission 6-6
- running SQL statements 3-16
- using command-line options 1-20
- ESCAPE key 2-5, 3-8

Exit option

- CREATE DATABASE menu 4-9
- CREATE TABLE menu 5-14
- DATABASE INFO menu 4-10
- DATABASE menu 4-4
- LOCK MODE menu 5-31
- LOG menu 4-8
- RUN screen 3-16, 5-58
- SQL menu 3-6
- TABLE menu 5-4, 5-7

Expression strategy

- ATTACH TABLES menu 5-25
- DEFINE ATTACH POSITION menu 5-26
- EDIT EXPRESSION menu 5-22
- EXPRESSION menu 5-21
- FRAGMENT menu 5-19
- NEW TABLE screen 5-28
- SELECT ATTACHING TABLE screen 5-25

extended data types 5-10

Extent size 5-29

F

Feature icons Intro-7

Features of this product, new Intro-5

File

- appending query results to 3-20
- command, selecting 3-21
- command, shown B-2
- creating to store query results in 3-20
- reading from standard input 1-26
- saving current SQL statement in 3-23
- saving SQL statements in 3-21
- storing query results in 3-19
- .sql extension for command files 3-23, B-1

Fill factor, specifying 5-13

Find Error utility Intro-12

finderr utility Intro-12

FLOAT data type 5-47

Foreign-key constraints

- defining 5-34
- enabling cascading deletes 5-35
- information on 5-55

Fragment

- ALTER FRAGMENT menu in DB-Access 5-23
 - EXPRESSION menu in DB-Access 5-21
 - FRAGMENT menu in DB-Access 5-19
 - option on DB-Access STORAGE menu 5-19
 - ROUND_ROBIN menu in DB-Access 5-20
 - FRAGMENT menu, reaching 5-19
 - Fragmentation strategy no strategy menu in DB-Access 5-28
 - round-robin in DB-Access 5-23
- Function. *See* Routines.

G

Global Language Support (GLS) Intro-4

- displaying information on 4-11
- GRANT statement 1-12

H

Help

- calling with CTRL-W A-1
- exiting on-line help screen 2-8
- how to read syntax diagrams A-1
- on-line syntax information for SQL statements A-1
- paging through on-line information screens 2-8
- HELP menu 2-8
- How to use your terminal 2-4

I

Icons

- feature Intro-7
- Important Intro-7
- platform Intro-7
- product Intro-7
- Tip Intro-7
- Warning Intro-7

IFX 1-7

Important paragraphs, icon for Intro-7

Index

- access method 5-51
- allowing duplicate values 5-12
- creating with the ADD INDEX screen 5-12
- from DataBlade 5-52
- specifying a fill factor 5-13
- user-defined 5-52

Indexing a column, the ADD INDEX screen 5-12

Industry standards, compliance with Intro-14

INFO FOR TABLE screen 5-44

INFO menu

- available options 5-46
- Columns option 5-47
- cOnstraints option 5-54
- displaying column information 5-46
- displaying constraint information 5-54
- displaying table index information 5-51
- displaying table privileges information 5-52
- exiting 3-24, 5-46
- Fragmentation option 5-59
- listing tables 5-46
- option on the TABLE menu 5-43
- Privileges option 5-52
- References option 5-53
- triGgers option 5-57
- with SQL 5-45

Info option

- Columns option 5-47
- cOnstraints option 5-54
- DATABASE menu 4-9
- Exit option 5-46
- Indexes option 5-51
- Privileges option 5-52
- References option 5-53
- SQL menu 3-24
- TABLE menu 5-4
- Table option 5-46
- triGgers option 5-57, 5-59

Information

- displaying for current database 4-9
- displaying for current session 6-9
- displaying for tables 3-24, 5-43
- Information Schema 1-22
- INFORMIXDIR/bin directory 1-8
- Input
 - interactive 1-25
 - reading from standard 1-24
- INT8 data type 5-47
- INTEGER data type 5-47
- Interactive input, through standard input 1-25
- Interactive Schema Editor. *See* Schema editor.
- Interrupt key 2-5
- INTERVAL data type 5-47
- Invoking DB-Access
 - DATABASE menu options 1-18
 - SQL menu options 1-19
 - TABLE menu options 1-20
- Invoking DB-Access
 - checking for ANSI compliance 1-23
 - command-line options 1-13
 - displaying the main menu 1-15
 - executing a command file 1-22
- ISED. *See* Schema editor.
- ISO 8859-1 code set Intro-4

K

- Keys
- arrow 2-4
 - BACKSPACE 2-4
 - CONTROL 2-4
 - CTRL-A 3-7
 - CTRL-D 1-25, 3-7
 - CTRL-R 3-8
 - CTRL-W 2-8, A-1
 - CTRL-X 3-8
 - DELETE 2-5
 - ESCAPE 2-5, 3-8
 - Interrupt 2-5
 - RETURN 2-5
 - SPACEBAR 2-5
 - used with SQL editor 3-7

L

- Large objects
 - defined 5-48
 - simple 5-48
 - smart 5-48
- LIST data type 5-50
- Literal, assigning default value 5-41
- Locale Intro-4
 - default Intro-4
 - en_us.8859-1 Intro-4
- LOCK MODE menu 5-30
- Lock mode, specifying page or row 5-30
- LOG menu 4-8
- Logging
 - creating a database with 4-8
 - specifying buffered or unbuffered 4-8

M

- Machine notes Intro-13
- Main menu
 - displaying from the command line 1-15
 - Query-language option 3-5
 - selecting the Connection option 6-3
 - selecting the Session option 6-9
 - selecting the Table option 5-3
- Menu options
 - CONNECTION menu 6-4
 - DATABASE menu 4-4
 - SQL menu 3-6
 - TABLE menu 5-4
- Menus
 - ADD CHECK VALUE 5-37
 - ADD DEFAULT TYPE 5-40
 - ADD DEFAULT VALUE 5-42
 - ADD NULLS 5-13
 - ADD TYPE 5-9
 - chart of 1-5
 - CHECK CONSTRAINTS 5-36
 - CONNECTION 6-3
 - CONSTRAINTS 5-32, 5-54
 - DEFAULTS 5-39

- ENABLE CASCADING DELETES 5-35
- example 2-6
- HELP 2-8
- how to exit 2-6
- how to select an option 2-6
- INFO 5-45
- LOCK MODE 5-30
- REFERENCE 5-55
- TABLE OPTIONS 5-17
- TRANSACTION 4-17, 6-8
- UNIQUE CONSTRAINTS 5-38
- Message file for error messages Intro-12
- Mode
 - insert 3-7
 - locking 5-30
 - typeover 3-7
- Modify option (SQL menu) 3-17
- MONEY data type 5-47
- MULTISET data type 5-50

N

- Native Language Support
 - command files list order 3-22
 - displaying information on 4-11
- NCHAR data type 5-47
- New features of this product Intro-5
- Null values
 - default for column 5-41
 - specifying with the ADD NULLS menu 5-13
- NVARCHAR data type 5-47

O

- Object-relational model. *See* Demonstration database, superstores_demo.
- OID. *See* Object identifier.
- On-line help Intro-12
- On-line help. *See* Help.
- On-line manuals Intro-11
- Opaque data type 5-49

Options

- for CONNECTION menu 6-4
- for DATABASE menu 4-4
- for SQL menu 3-6
- for TABLE menu 5-4
- how to get Help for 2-8
- how to select on a menu 2-6
- See dbaccess command options.

Output option

- Append-file option 3-20
- New-file option 3-20
- Printer option 3-19
- SQL menu 3-19
- To-pipe option 3-21

P

- Page option (LOCK MODE menu) 5-31
- PASSWORD prompt screen 6-5
- Password, prompt in DB-Access interactive mode 1-27
- PATH
 - DB-Access requirements 1-7
 - demonstration database and 1-10
- Permissions, UNIX 1-10
- Pipe
 - reading from 1-26
 - redirecting query results to a program 3-19
 - sending query results to 3-21
- Platform icons Intro-7
- Primary-key constraints
 - defining 5-32
 - information on 5-56
- Printed manuals Intro-11
- Printing the results of a query 3-19
- Procedure. See Routines.
- Product icons Intro-7
- Program group
 - Documentation notes Intro-13
 - Release notes Intro-13

Q

- Query
 - appending results to an existing file 3-20
 - sending results to a file 3-19
 - sending results to a pipe 3-21
 - sending results to a printer 3-19
 - storing results in a new file 3-20
- QUERY-LANGUAGE menu, options 1-19
- Query-language option
 - how to use 3-6
 - on the main menu 3-5

R

- Reading from standard input 1-24
- REFERENCE menu
 - options 5-55
 - with referenced information 5-56
 - with referencing information 5-55
- Related reading Intro-14
- Relational model. See Demonstration database, stores_demo.
- Release notes Intro-13
- Restrictions, for SQL editor 3-8
- RETURN key 2-5
- rofferr utility Intro-12
- ROLLBACK 4-17
- Rolling back transactions 4-17, 6-9
- Rolling back transactions, with the TRANSACTIONS menu 6-9
- Round-robin strategy
- FRAGMENT menu 5-19
- ROUND_ROBIN menu 5-20
- Routines
 - creating and running 3-28
 - demonstration command file 3-28
 - displaying list of 4-12
 - displaying text of 4-12
 - SELECT ROUTINE screen 4-13
 - stored 3-30
 - sysprocedures system catalog table 3-28

- Row data type, defined 5-50
- Row locking 5-30
- Row option (LOCK MODE menu) 5-31
- Rowids, adding or dropping 5-31
- Run option
 - Exit option 3-16, 5-58
 - Next option for viewing data 3-15
 - Restart option to redisplay results 3-16
 - SQL menu 3-10
- Running SQL statements
 - when there are errors 3-16
 - when there are no errors 3-10

S

- sales_demo 1-8
- sales_demo database. See Demonstration database.
- Sample-code conventions Intro-10
- Save option
 - rules for naming saved files 3-23
 - SQL menu 3-23
- Saving command files 3-23
- Schema editor
 - creating a table 5-8
 - defining a column 5-8
 - dropping a column from a table 5-16
 - EXIT menu 5-14
- Schema, building table with 5-14
- Screens
 - ADD FILL FACTOR PERCENTAGE 5-13
 - CHOOSE 3-22
 - CREATE DATABASE confirmation 4-9
 - DISCONNECT confirmation 6-8
 - example 2-7
 - for DATABASE menu 4-3 to 4-15
 - for SQL menu 3-5 to 3-27
 - for TABLE menu 5-3
 - for text entry 2-7
 - how to enter text 2-7
 - how to exit 2-8
 - INFO FOR TABLE 5-44
 - SELECT BLOSPACE 5-11

- SELECT DATABASE 6-6
 - SELECT DATABASE SERVER 6-5
 - SELECT DBSPACE 5-18
 - SELECT PROCEDURE 4-13
 - SELECT ROUTINE 4-13
 - structure of header 2-7
 - SELECT BLOBSPACE screen 5-11
 - SELECT DATABASE screen 6-6
 - exit without selecting a database 6-7
 - selecting from the DATABASE menu 4-5
 - two ways to select a database 4-6
 - SELECT DATABASE SERVER screen 6-4, 6-5
 - SELECT DBSPACE screen 5-18
 - SELECT ROUTINE screen 4-12
 - Selecting a database server 6-4
 - Selecting a database, from a menu 4-5
 - SERIAL data type 5-47
 - SERIAL8 data type 5-47
 - Server. *See* Database server.
 - Session information 6-9
 - SET data type 5-50
 - Shell
 - Bourne 1-26
 - C 1-26
 - Korn 1-26
 - Sitename, saving in a column 5-41
 - Size
 - setting initial extent 5-29
 - setting next extent 5-29
 - SMALLFLOAT data type 5-47
 - SMALLINT data type 5-47
 - Software dependencies Intro-4
 - SPACEBAR 2-5
 - SPACEBAR key 2-5
 - SPL Routines 3-28
 - SQL
 - entering statements 3-7
 - how to read syntax in on-line Help screens A-1
 - INFO statement equivalent 3-24
 - using from a menu 3-14
 - SQL code Intro-10
 - SQL command files
 - must be in current directory 1-9
 - requirements for listing with Choose command 3-22
 - sales_demo B-16
 - SQL editor
 - assigning default data type for column 5-41
 - CTRL-A editing key 3-7
 - CTRL-D editing key 3-7
 - CTRL-R editing key 3-8
 - CTRL-X editing key 3-8
 - editing keys 3-7
 - editing restrictions 3-8
 - ESC key 3-8
 - how to use 3-7
 - modifying SQL statements 3-17
 - new SQL statements 3-7
 - SQL menu
 - available options 3-6
 - Choose option 3-21
 - CHOOSE screen 3-21
 - Drop option 3-26
 - Info option 3-24
 - Modify option 3-17
 - OUTPUT menu 3-19
 - Output option 3-19
 - Run option 3-10
 - running new SQL statements 3-10
 - Save option 3-23
 - SAVE screen 3-23
 - selecting options from the command line 1-19
 - SQL statements
 - appending query results to an existing file 3-20
 - choosing a command file 3-21
 - current, defined 3-3
 - dropping 3-26
 - editing 3-4
 - editing with the system editor 3-4
 - entering new ones with the SQL editor 3-7
 - executing from standard input 1-26
 - interactive input on terminal 1-25
 - modifying 3-17
 - reading from standard input 1-24
 - redirecting query results 3-19
 - saving to a command file 3-23
 - selecting the SAVE screen 3-23
 - sending output to a file 3-19
 - sending output to a printer 3-19
 - sending query results to a pipe 3-21
 - syntax conventions in on-line Help A-1
 - what happens when there are errors 3-16
 - sqlhosts, display connectivity information in 6-6
 - SQLSTATE value displayed 1-23
 - stdin, for interactive input 1-25
 - Stored procedures
 - SELECT PROCEDURE screen 4-13
 - stores_demo 1-8, 1-9
 - stores_demo database. *See* Demonstration database.
 - superstores_demo 1-8, 1-9
 - superstores_demo database. *See* Demonstration database.
 - System date, saving in a column 5-41
 - System requirements
 - database Intro-4
 - software Intro-4
-
- T**
- Table
 - allowing null values in a column 5-13
 - building schema 5-14
 - column constraints display 5-54
 - column information 5-47
 - constraint information 5-54
 - dbspace for storing 5-18
 - displaying from another server 5-44
 - displaying information on the screen 3-24, 5-43
 - displaying information with the triGgers option 5-57

dropping 5-60
dropping a column (schema editor) 5-16
extent sizes for 5-29
foreign-key constraint information 5-55
index information 5-51
indexing a column (schema editor) 5-12
inheritance B-21
lock mode for 5-30
modifying a column (schema editor) 5-15
modifying before building 5-15
primary-key constraint information 5-56
privileges display 5-52
privileges information 5-52
references display 5-53
removing a column from the schema 5-16
SELECT ATTACHING TABLE screen (expression fragmentation) 5-25
setting the extent sizes 5-17
setting the lock mode 5-17
storage location for 5-18
trigger information 5-57, 5-59
triggers display 5-59
TABLE menu
available options 5-4
guidelines for using 5-3
Info option 5-44
options 1-20
selecting options from the command line 1-20
selecting the CREATE TABLE screen 5-4
selecting the Drop option 5-60
TABLE OPTIONS menu 5-17
Table_options option (CREATE TABLE menu) 5-7
Terminal
as standard input 1-25
how to use the keys 2-4
Text
editing with the system editor 3-4
how to enter on a screen 2-7

TEXT data type 5-47, 5-48
Text editor
how to use 2-9
internal editor 3-9
SQL editor 3-9
Tip icons Intro-7
TRANSACTION menu 4-17, 6-8
Transactions, committing or rolling back 4-17
Trouble shooting
Choose command does not list your SQL command files 3-22
dbaccess window must not have scroll bars 1-14
Trouble shooting, terminal type unknown 1-14

U

Unbuffered logging
how to specify 4-8
Unique constraints
data validation 5-39
defining 5-38
UNIQUE CONSTRAINTS menu 5-38
UNIX
case sensitivity and filenames 3-23
chmod command 1-13
permissions 1-10
system editors 3-5
UNIX operating system
default locale for Intro-4
USER clause of CONNECT statement
in DB-Access interactive mode 1-27
User id, saving in a column 5-41
User name
CONNECT statement with 1-27
specifying when connecting in background mode 1-27
USER NAME prompt screen 6-4
user-defined data types 5-10
Users, types of Intro-3

V

VARCHAR data type 5-47

W

Warning icons Intro-7
Windows NT
default locale for Intro-4
Working directory 1-10

X

X/Open compliance level Intro-14

Y

Year values, two and four digit 1-7

Symbols

- 1-17
\$INFORMIXDIR/bin 1-7
%INFORMIXDIR%\bin 1-7

