HCL Informix 15.0.0

Informix DB-Access User's Guide

Contents

Chapter 1. DB-Access User's Guide.................................3

Getting started with DB-Access.....................................3

Requirements for the Informix® server DB-Access utility... 4

Requirements for the Informix® Client Software Development Kit DB-Access utility.......................... 5

Demonstration databases.......................................6

Start DB-Access...8

The full-screen menu interface.................................... 18

The Query-language option.................................. 19

The Database option...25

The Table option... 27

The Connection and Session options.................. 29

Appendixes... 30

How to read online help for SQL statements........31

Appendixes...

Index..33

Chapter 1. DB-Access User's Guide

The *HCL® Informix® DB-Access User's Guide* describes how to use the utility to access, modify, and retrieve information from HCL® Informix® database servers.

Important: Use with the current version of the Informix® database server. If you use 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.

This publication is written for the following users:

- Database users
- Database administrators
- Database-application programmers

This publication 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

Getting started with DB-Access

provides a menu-driven interface for entering, running, and debugging Structured Query Language (SQL) statements and Stored Procedure Language (SPL) routines. You can also run interactively from the command line.

You use SQL and SPL commands 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.

You can use for the following aspects of database processing:

- Running ad hoc queries that you run 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 of databases
- Practicing the SQL and SPL statements and examples that are provided in the *HCL® Informix® Guide to SQL: Tutorial* or the *IBM® Informix® Database Design and Implementation Guide*
- Testing applications that you intend to store for use in a production environment
- Creating demonstration databases

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

The utility is included with the Informix® server and with the Informix® Client Software Development Kit.

Requirements for the Informix® server DB-Access utility

Before you start DB-Access, prepare the Informix® server environment.

Do the following tasks before you start the utility that is included with the Informix® server:

- Set environment variables
- If you require globalization, set up the Global Language Support (GLS) locale
- Start the database server

Environment variables for DB-Access

As part of the installation and setup process, the system or database administrator sets certain environment variables that enable HCL® Informix® products to work within a particular operating-system environment.

You must have `$INFORMIXDIR/bin` in your path if you use on a UNIX™ operating system or `%INFORMIXDIR%\bin` in your path if you use on a Windows™ operating system. Your operating system uses the path to locate the initialization script and the dbaccess executable file.

In a UNIX™ environment, the database server must have the appropriate terminal that is set up from among the terminals that are listed by the **INFORMIXTERM** environment variable.

uses the terminal definitions in the `terminfo` directory unless the **INFORMIXTERM** environment variable is set to the `termcap` file. If fails to initialize the menus that are based on the **INFORMIXTERM** setting, tries to use the other setting. For example, if fails to initialize the menus using the `terminfo` directory, starts the menus using the `termcap` file.

You can set the following optional environment variables:

DBACCNOIGN

Rolls back an incomplete transaction if you run the LOAD command in menu mode.

DBCENTURY

Sets the appropriate expansion for DATE and DATETIME values that have only a two-digit year, such as 04/15/12.

DBDATE

Specifies the user formats of DATE values.

DBEDIT

Sets the default text editor without changing the default text editor that is associated with the operating-system shell.

For more information about how uses the text editor that you specify as default, see A system editor on page 20.

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 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.

If you use the **IFX_LONGID** environment variable to support SQL identifiers with up to 128 bytes, some error, warning, or other messages of might truncate database object names that include more than 18 bytes in their identifiers. You can avoid this truncation by not declaring names that have more than 18 bytes.

GL_DATETIME

Defines the end-user formats for data values in DATETIME columns. In databases where **GL_DATETIME** has a nondefault setting, you must also set the **USE_DTENV** environment variable to `1` for user formats to be applied correctly in some load and unload operations.

Important:

The `%F` directive implies no default separator between the SECOND and FRACTION fields of DATETIME values. Defining no separator before the `%F` directive concatenates SECOND and FRACTION values, as in the following example, where **GL_DATETIME** has this setting:

```
%Y-%m-%d %H:%M:%S%F
```

Below is the end-user display for a DATETIME YEAR TO FRACTION(2) value on August 23, 2013, at exactly 53 seconds after 1:15 PM:

```
2013-08-23 13:15:5300
```

Here `5300` represents 53 seconds, concatenated with the FRACTION precision of 2. To display a separator between the integer and fractional parts of the seconds value, your **GL_DATETIME** setting must include a literal separator character immediately before the `%F` formatting directive.

Requirements for the Informix® Client Software Development Kit DB-Access utility

Before you start on a client, set up the client environment.

The utility on a client can directly access Informix® databases with which Client SDK has a client/server connection.

Do the following tasks before you start utility that is included with :

- Set the client/server connectivity information in the `sqlhosts` file.
  - If you renamed the file or moved it from the default location, set the INFORMIXSQLHOSTS environment variable to the full path name and file name of the file that contains the connectivity information.
  - You can use the syncsqlhosts utility to convert connection information from the Windows™ registry format to the `sqlhosts` file format.

- Set the INFORMIXDIR environment variable to the Client SDK installation directory.
- Set the INFORMIXSERVER environment variable for a default server name.

Demonstration databases

You can practice using with a demonstration database.

If you use HCL® Informix® demonstration databases, you can add, delete, or change the provided data and scripts. You can restore the database to its original condition.

You can configure the following demonstration databases:

- The **stores_demo** database illustrates a relational schema with tables about a fictitious wholesale sporting-goods distributor, as well as other tables that are used in examples. Tables containing electricity usage data illustrate time series information.Many examples in HCL® Informix® manuals are based on the **stores_demo** database.
- 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.

The scripts that you use to install the demonstration databases are in the `$INFORMIXDIR/bin` directory on UNIX™ and in the `%INFORMIXDIR%\bin` directory on Windows™.

Some operating systems require that you have execute permissions to run SQL command files, read permissions to open these files or their contents in , 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.

The demonstration scripts are designed for the default locale. If you use a non-default locale, such as **en_us.utf8**, some features, such as the SET COLLATION statement, might not function correctly.

Creating a demonstration database

You create demonstration databases by running the dbaccessdemo command.

About this task

When you create a demonstration database, the script confirms that you want to copy sample SQL command files. Command files that the demonstration database includes have a `.sql` extension and contain sample SQL statements that you can use.

To create a demonstration database:

1. Create a directory.

   You must have UNIX™ read and execute permissions for each directory in the path name that you create.
2. Change directories to the new directory and run the dbaccessdemo or dbaccessdemo_ud command.
3. The initialization script displays a series of messages on the screen as the database is created. Press **Y** to copy the command files into the directory that you created.

Result

The demonstration database is created. You are the owner and database administrator (DBA) of that database.

Results

If you want to discard changes that you made to your database or to the command files, rerun the dbaccessdemo or dbaccessdemo_ud command and press **Y** to replace the existing command files with the original versions.

dbaccessdemo command: Create demonstration databases

Use the dbaccessdemo or dbaccessdemo_ud command to create the demonstration databases.

Syntax for stores_demo

```
dbaccessdemo [ -log ] [ dbname ] [ -dbspacedbspace_name ] [ -nots ]
```

Syntax for superstores_demo

```
dbaccessdemo_ud [ -log ] [ dbname ] [ -dbspacedbspace_name ]
```

-log

Requests transaction logging for the demonstration database.

dbname

Substitutes for the default database name. Must follow Identifier naming guidelines.

-dbspace

Requests a particular dbspace location for the demonstration database.

dbspace_name

Houses the demonstration database. If you do not specify a dbspace name, by default, the data for the database is put in the root dbspace. To create a dbspace, use the onspaces utility.

-nots

Prevents the creation of the time series tables in the **stores_demo** database.

Examples

The following command creates a database that is 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
```

The following command creates a database that is named **superstores_demo**:

```
dbaccessdemo_ud
```

Start DB-Access

You start by running the dbaccess command from the command line. You can choose whether to use the menu interface or the command-line interface.

You can start and use in the following ways:

- Start at the main menu.
- Start from a specific menu or screen.
- Run a file that contains SQL statements without showing the menus.
- Start and run interactively at the command line, without the menu interface.

On Windows™, you can set up the program icon to run any of the dbaccess commands.

If the **TERM**, **TERMCAP**, or **TERMINFO** environment variables on UNIX™ do not enable to recognize the type of terminal you use, the main menu does not show. Instead, a message similar to the following text is displayed:

```
Unknown terminal type.
```

If you use a window interface on a UNIX™ terminal, issue the dbaccess command from a nonscrolling console window.

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

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

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

dbaccess command: Start DB-Access

Use the dbaccess command to start DB-Access. Include options to specify the database, command files, or to go to a specific menu screen.

Syntax

dbaccess [**-ansi**] [**-X**] { *database* [@*server*] [{ **-q** [<QUERY-LANGUAGE menu options> (explicit id)] [{ *filename* | *table* }] | **-t** [<TABLE menu options> (explicit id)] [*table*] }] | **-d** [<DATABASE menu options> (explicit id)] | **-c** [<CONNECTION menu options> (explicit id)] | **-s** | [{ **-e** | **-m** }] { *database* | **-** } *filename* } [**-a**] [{ (explicit id) | **-version** | **-V** | **-history** }]

The dbaccess command without options starts the main menu with no database selected and no options activated. You select submenus from the main menu.

-ansi

Causes to generate a warning whenever it encounters HCL® Informix® extension to ANSI-compliant syntax. For more information, see Example: Check for ANSI compliance on page 14.

-a

Stops a process directly after the first error is encountered. Stopping a process from continuing after the first error can ensure greater database consistency.

-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 *filename*.

-history

Can be abbreviated to -his. Displays the previous commands that you ran from the dbaccess command line during the current session. Commands that you ran from the menu interfaces are not shown. Each command is numbered. In the interactive mode without menus, you can rerun previous commands with the following command:

```
run number
```

The *number* is the number of the command to run. For an example, see Run DB-Access in interactive mode without menus on page 15.

-m

Displays all error messages generated by multiple levels of the server that pertain to an SQL statement in command files.

-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

Displays the version number and serial number for without launching the application. You cannot use any other options with -V.

-version

Displays the version number and build information for , including the GLS library version, without launching the application. You cannot use any other options with -version.

-X

Activates the hexadecimal format for LOAD and UNLOAD statements.

database

Name of the database that you want to connect to at the startup of your current session. A hyphen (-) indicates that the database is specified in a DATABASE statement in a command file.

filename

Names a command file to load with the SQL menu.

server

Name of the database server.

table

Specifies a table in the database.

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.

CONNECTION menu options

The CONNECTION menu options for the dbaccess command represent short cut keys for the CONNECTION menu.

-cc

Chooses the Connect option on the CONNECTION menu.

-cd

Chooses the Disconnect option on the CONNECTION menu.

DATABASE menu options

The DATABASE menu options for the dbaccess command represent short cut keys for the DATABASE menu.

-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

The dbspaces information for the current database

-din

The 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

The QUERY-LANGUAGE menu options for the dbaccess command represent short cut keys for the QUERY-LANGUAGE menu.

-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 about fragmentation strategy for the table

-qig

Information about 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 is then displayed and shows the text.

Restriction: 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

The TABLE menu options for the dbaccess command represent short cut keys for the TABLE menu.

-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 about fragmentation strategy for the table

-tig

Information about 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.

Example: Start DB-Access for a database

This example shows how to start DB-Access and specify a database to which to connect.

Assume that the database server that you have online contains a database named **mystores**. To make the **mystores** database the current database, start with the following command:

```
dbaccess mystores
```

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

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

When starts, the database and database server name that you specify are displayed on the dashed line, as the following figure shows.

Figure 1. The main menu with database and database server name

```
DB-Access:   Query-language  Connection  Database  Table  Session  Exit



---------------- newstores@xyz ---------------------Press CTRL-W for Help ---
```

Example: Run a command file

This example shows how to start DB-Access and run a command file that contains SQL statements.

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

```
dbaccess mystores sel_stock
```

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

```
dbaccess - sel_all.sql
```

Some operating systems require that you have execute permissions to run SQL command files, read permissions to open these files or their contents in , 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.

Example: View the Information Schema

This example shows how to start DB-Access and view the Information Schema for the specified database.

The `xpg4_is.sql` file in the `$INFORMIXDIR/etc` directory 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 HCL® Informix® extensions. After you run `xpg4_is.sql`, use to retrieve information about the tables and columns that you have access to in the specified database.

Tip: Do not install XPG4-compliant views on an ANSI database, because the format of XPG4-compliant views differs considerably from the format of the ANSI-compliant Information Schema views that are defined by the SQL standards committee.

Example: Check for ANSI compliance

This example shows how to start DB-Access and check whether a database is ANSI-compliant.

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 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.

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 in command-line mode or specify a `.sql` input file.
- Running an SQL statement generates a warning rather than an error.

Example: Show nonprintable characters in hexadecimal

This example starts DB-Access and activates the hexadecimal load and unload format (XLUF) so that the LOAD and UNLOAD SQL statements can format nonprintable ASCII signs in hexadecimal format.

The following command activates the XLUF format for the **mystores** database:

```
dbaccess -X mystores
```

A `.unl` file that the UNLOAD statement produces contains the hexadecimal format changes.

Run DB-Access in interactive mode without menus

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 from the command line.

When you start without a menu argument and with a hyphen as the final argument, processes commands from the standard input device (on UNIX™) or the keyboard (on Windows™). reads what you type until you indicate that the input is completed. Then processes your input and writes the results to the standard output device (on UNIX™), or the command window (on Windows™).

reads and runs SQL statements from the terminal keyboard interactively. While 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, processes that statement. When you press **CTRL-D** to end the interactive session, 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
```

Batch command input (UNIX™)

You can use an inline shell script to supply one or more SQL statements. For example, you can use the UNIX™ C, Bourne, or Korn shell with inline 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, as in this UNIX™ example:

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

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.
>
```

View and rerun commands

You can view and rerun commands that you ran from the command line during the current session.

To view the previous commands, run the dbaccess -history command. In the following example, the command history shows three previous commands:

```
dbaccess -history - -
1> create database sales_demo;

Database created.

2> create table customer (
2>       customer_code integer,
2>       customer_name char(31),
2>       company_name char(20));

Table created.

3> insert into customer values (102, "Carole Sadler", "Sports Spot");

1 row(s) inserted.

4> history;

1   create database sales_demo
2   create table customer (
          customer_code integer,
          customer_name char(31),
          company_name char(20))
3   insert into customer values (102, "Carole Sadler", "Sports Spot")
```

You can rerun the third command by running the run 3 command:

```

4> run 3;

1 row(s) inserted.
```

Connect to a database environment in interactive mode

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

When you include the USER '*user identifier*' clause in a CONNECT statement in interactive mode, prompts you to enter a password.

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 -nohistory- -

> connect to '@starfish';

Connected.
```

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

```
> connect to '@starfish' user 'marae';

ENTER PASSWORD:

Connected.
```

Restriction: 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 interactively. If you are in interactive mode and attempt to enter a password before the prompt, an error message is displayed.

You can run the USER clause of a CONNECT statement in a 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 following statement is protected from access by anyone other than the **user_id** that the USER clause identifies:

```
CONNECT TO '@dbserver' USER 'user_id' USING password
```

For UNIX™, the following example uses a shell file to connect to a database server. prompts you for a password.

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

ENTER PASSWORD:
```

Here the delimiting quotation marks preserve letter case in the database server name and in the authorization identifier of the user.

The full-screen menu interface

The full-screen menu interface guides you through running SQL statements.

The 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 that 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

The Query-language option

Use the Query-language option to enter, modify, save, retrieve, and run SQL statements. retains the statements, if any, in the editor. These statements are called the *current* statements.

Use the Query-language option to:

- Learn SQL and SPL.

  For example, use the Query-language option to practice the examples in the *HCL® Informix® Guide to SQL: Tutorial*.
- Create and alter table structures as an alternative to the Schema Editor.
- Select, display, add, update, and delete data.

The SQL menu has the following options.

Option	Purpose
New	Clears current statements and positions cursor in SQL editor.
Run	Runs current SQL statements. A message is displayed or the data that is retrieved by a query is displayed with the number of rows retrieved.
Modify	Allows you to modify current SQL statements in SQL editor.
Use-editor	Starts a system editor so that you can modify current statements or create new statements. Use-editor is interchangeable with New and Modify.
Output	Redirects Run-option output to a file, printer, or system pipe.
Choose	Lists SQL command files so that you can choose a file to run or modify.
Save	Saves current SQL statements in a file for later use.
Info	Shows table information, such as columns, indexes, privileges, constraints, triggers, status, and fragmentation strategy.
Drop	Deletes a specified SQL command file.
Exit	Returns to main menu.

SQL editor

When you choose the New of Modify option, you see 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. If you do not use the Save option to save your typed statements, they are deleted when you select an option that clears the SQL editor (such as New or Choose).

The SQL editor does not display more than 80 characters on a line and does not wrap lines.

- If you choose an existing command file in which the characters in a line extend beyond the 80th column, 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 runs correctly.
- If you type characters in a new command file so that a line extends beyond the 80th column, 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 show on the screen, press **Enter** at a logical place in the first 80 characters of each line.

If you must 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 minus signs for ANSI-compliant databases.
- Preface each comment line with a double minus sign (--) comment indicator. The comment indicator spans the entire line.
- Use braces ({ }) for databases that are not ANSI-compliant. Enclose the entire comment indicator between the braces.

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, prompts you to accept or override the default system editor once each session.

The default editor that 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 editor 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**.

Statements that the Run option supports

After you exit the editor screen, the SQL menu reopens with the Run option highlighted and the statement text is displayed in the bottom of the screen. You can run most SQL statements with the Run option.

To run 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.

The following is a list of SQL statements that you can run with the Run option.

- ALLOCATE COLLECTION
- ALLOCATE DESCRIPTOR

- ALLOCATE ROW
- ALTER ACCESS_METHOD
- ALTER FRAGMENT
- ALTER FUNCTION
- ALTER INDEX
- ALTER PROCEDURE
- ALTER ROUTINE
- ALTER SECURITY LABEL COMPONENT
- ALTER SEQUENCE
- ALTER TABLE
- BEGIN WORK
- CLOSE
- CLOSE DATABASE
- COMMIT WORK
- CONNECT
- CREATE ACCESS_METHOD
- CREATE AGGREGATE
- CREATE CAST
- CREATE DATABASE
- CREATE DISTINCT TYPE
- CREATE EXTERNAL TABLE
- CREATE FUNCTION
- CREATE FUNCTION FROM
- CREATE INDEX
- CREATE OPAQUE TYPE
- CREATE OPCLASS
- CREATE PROCEDURE
- CREATE ROLE
- CREATE ROUTINE FROM
- CREATE ROW TYPE
- CREATE SCHEMA
- CREATE SECURITY LABEL COMPONENT
- CREATE SECURITY LABEL
- CREATE SECURITY POLICY
- CREATE SEQUENCE
- CREATE SYNONYM
- CREATE TABLE
- CREATE TRIGGER
- CREATE VIEW
- CREATE XADATASOURCE
- CREATE XADATASOURCE TYPE
- DATABASE

- DEALLOCATE COLLECTION
- DEALLOCATE DESCRIPTOR
- DEALLOCATE ROW
- DECLARE
- DELETE
- DESCRIBE
- DESCRIBE INPUT
- DISCONNECT
- DROP ACCESS METHOD
- DROP AGGREGATE
- DROP CAST
- DROP DATABASE
- DROP FUNCTION
- DROP INDEX
- DROP OPAQUE TYPE
- DROP OPCLASS
- DROP PROCEDURE
- DROP ROLE
- DROP ROW TYPE
- DROP SECURITY LABEL COMPONENT/POLICY/LABEL
- DROP SEQUENCE
- DROP SYNONYM
- DROP TABLE
- DROP TRIGGER
- DROP TYPE
- DROP VIEW
- DROP XADATASOURCE
- DROP XADATASOURCE TYPE
- EXECUTE
- EXECUTE FUNCTION
- EXECUTE IMMEDIATE
- EXECUTE PROCEDURE
- FETCH
- FLUSH
- FREE
- GET DESCRIPTOR
- GET DIAGNOSTICS
- GRANT
- GRANT DBSECADM
- GRANT DEFAULT ROLE
- GRANT EXEMPTION
- GRANT FRAGMENT

- GRANT SECURITY LABEL
- INFO
- INSERT
- LOAD
- LOCK TABLE
- MERGE
- OPEN
- OUTPUT
- PREPARE
- PUT
- RENAME COLUMN
- RENAME DATABASE
- RENAME INDEX
- RENAME SEQUENCE
- RENAME TABLE
- REVOKE
- REVOKE DBSECADM
- REVOKE DEFAULT ROLE
- REVOKE EXEMPTION
- REVOKE FRAGMENT
- REVOKE SECURITY LABEL
- ROLLBACK WORK
- SAVE EXTERNAL DIRECTIVES
- SELECT
- SET AUTOFREE
- SET COLLATION
- SET CONNECTION
- SET CONSTRAINTS
- SET DATASKIP
- SET DEBUG FILE TO
- SET DEFERRED PREPARE
- SET DESCRIPTOR
- SET ENCRYPTION PASSWORD
- SET ENVIRONMENT
- SET EXPLAIN
- SET ISOLATION
- SET LOCK MODE
- SET LOG
- SET OPTIMIZATION
- SET PDQPRIORITY
- SET ROLE
- SET SESSION AUTHORIZATION

- SET STATEMENT CACHE
- SET TRANSACTION
- START VIOLATIONS TABLE
- STOP VIOLATIONS TABLE
- TRUNCATE
- UNLOAD
- UNLOCK TABLE
- UPDATE
- UPDATE STATISTICS
- WHENEVER

Redirect query results

The output from a SELECT statement is normally displayed on the screen. You can 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. The Output option has the same result as the OUTPUT statement of SQL.

The SELECT statement must be on the screen as the current statement. Select the Output option from the SQL menu, which displays the OUTPUT menu.

You have the following output options:

- Send your query results directly to a printer. 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 show on the screen. You can set the **DBPRINT** environment variable to specify a default printer.
- Write query results to a new file or append the results to an existing file. If you do not specify a path when prompts you for a file name, the file is stored in the directory that you were in when you started .
- Send query results to a pipe. Specify a target program, such as **more**, through which to pipe output. sends the results to that pipe.

  On UNIX™ systems, you must have permission to run the target program.

  On Windows™ systems, the cat utility can serve as a target program through which to pipe output.

Save the current SQL statement

You can save SQL statements in a file for later use, such as to run the statements from the command line 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. Enter a name for the command file:

- Use 1 - 10 characters. Start with a letter, then use any combination of letters, numbers, and underscores (_). Press **Enter** to save the file.
- UNIX™: File names are case-sensitive. The file `orders` is not the same as `Orders` or `ORDERS`.

appends the extension `.sql` to the file name. For example, if you name your file `cust1`, 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.

What happens when errors occur

If you make any syntax or typing mistakes in an SQL statement, does not process the statement. Instead, it continues to display the text of the statement with a message that describes the error.

If an execution or runtime error occurs, 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 is displayed at the bottom of the screen:

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

If you try to run a statement that contains more than one SQL statement, you might not see an 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 shows on the screen before the error message is displayed for the second statement.

When detects an error, processing stops and the Modify option on the SQL menu is highlighted. Select one of the following methods to correct the statement:

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

The Database option

Use the Database option to work with databases and transactions.

Use the Database option to:

- Create a database or select a database.

  The database that 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 access only databases that are on the current database server. To select a database server as current, you can specify a database server when you start , 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, uses the default database server that the **$INFORMIXSERVER** environment variable specifies as the current database.

If you select or create a database when another database is already open, closes that database before it makes your selection the current or new database.

The DATABASE menu displays the following options.

Option	Purpose
Select	Makes a database the current database
Create	Builds a new database and makes that database the current database
Info	Displays information about the current database
Drop	Removes a database from the system. You cannot delete the current database.
cLose	Closes the current database
Exit	Exits the DATABASE menu and returns you to the main menu

List of available databases

When you choose the Select option, the SELECT DATABASE screen opens. 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.

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 see the first 17 characters of the name followed by a '+' sign. Enter a '+' sign to display the complete long name in **vi**. To exit from **vi**, press `ESC ZZ`.

The list of available databases that is displayed depends on two factors:

- The settings of certain environment variables.
  - If you use one database server, 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 current connection. For example:
  - If no explicit connection exists, displays the databases in the **DBPATH** setting.
  - If a current explicit connection exists, all databases in the **DBPATH** setting that pertain to the current database server are displayed.

Close a database

To close the current database, use the cLose option from the DATABASE menu.

If you begin a transaction but do not commit it or roll it back, and then you try to close a database with transactions, the TRANSACTION menu opens. The TRANSACTION menu ensures that you either commit or roll back an active transaction before you close the current database.

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

The TRANSACTION menu also opens whenever you attempt to open a new database or try to leave the menu system without first terminating a transaction.

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, commits 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.

The Table option

Use the Table option to work with tables.

Use the Table option 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

Use the TABLE menu options as the following table shows.

Option	**Purpose**
Create	Enables you to define the structure of a new table. The CREATE TABLE menu provides data type 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. DB-Access can construct only a nonclustered, ascending B-tree column index. If you want hash or hybrid fragmentation, use the SQL menu to enter and run the CREATE TABLE or ALTER TABLE statement.
Alter	Enables you to alter the structure of an existing table, including columns, fragmentation, and constraints. You must have the Alter privilege to successfully alter a table. **Note:** If you use the UI to build the SQL statements to Alter a table without the Alter privilege, the table will be locked and the operation will fail. Further, the lock will be held until you exit the alter menu. To use the LOAD statement to insert data into a table, you must have both Insert and Select privileges for the table.
Info	Displays information about the structure of a table
Drop	Deletes a table from the database

Option	Purpose
Move	Moves a table from the current database to another database.
Exit	Returns to the DB-Access main menu

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

Option	Purpose
Add	Displays the Schema Editor, from which you can add a new column to the table
Modify	Displays the columns that you defined with the Add option so that you can modify the column structure before building the table
Drop	Drops an existing column from the table
Screen	Displays the next screen of column definitions in the Schema Editor
Table_options	Enables you to display and select storage spaces for a new table. Displays choices from which to 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.
Constraints	Enables you to define primary-key, foreign-key, check, and unique constraints, and to set default column values
Exit	Builds, rebuilds, or discards the schema and structure that you specified with the other options, and then returns to the TABLE menu

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

Display table information

Use the Info option on the TABLE menu to display information about the structure of a table.

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 are displayed.
- If globalization 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 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 about the **customer** table that **dba** created in the **accounts** database on the database server **topend**:

```
INFO FOR TABLE >> accounts@topend:dba.customer
```

The INFO menu has the following options.

Option	Purpose
Columns	Lists data type by column name and indicates which columns can contain a null value
Indexes	Describes each index that is defined for a specified table
Privileges	Lists the users who have Select, Update, Insert, Delete, Index, or Alter privileges for the specified table
References	Lists the users who have the table-level References privilege for the specified table and the names of the columns they can reference
Status	Lists the table name, owner, row size, number of rows and columns, and creation date of the current table
cOnstraints	Displays the referential, primary, unique, and check constraints, and the default values for the columns in the specified table
triGgers	Displays header and body information for a specified trigger
Table	Redisplays the INFO FOR TABLE menu so that you can select a different table for examination
Fragments	Lists fragmented dbspaces assigned to the table and, for expression-based fragmentation, displays the expression that is assigned to each dbspace
Exit	Returns to the TABLE menu

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.

The Connection and Session options

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 session.

For the globalization considerations that apply to establishing a connection between a client application, such as , and a database, see the *HCL® Informix® GLS User's Guide*. 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.

On Windows™, if you specify a user identifier but no domain name for a connection to a machine that expects both a domain name and a user name (`domain\user`), checks only the local machine and the primary domain for the user account. If you explicitly specify a domain name, that domain is used to search for the user account. The attempted connection fails with error -951 if no matching `domain\user` account is found on the local machine.

The CONNECTION menu displays the following options.

Option	Purpose
Connect	Connects to a database environment. To access a specific database, you must have permission.
Disconnect	Disconnects from the current database environment
Exit	Returns to the main menu

When you use the Connect option, 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 that are 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.

Implicit closures

closes any open connections or databases when you connect to a new environment.

closes any open connections or databases in the following situations:

- When you connect to a new database environment without explicitly disconnecting from the current one, 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, 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 must close a database that still has outstanding transactions, it prompts you to commit or roll back those transactions.

Appendixes

How to read online help for SQL statements

Specific conventions are used to represent the syntax of SQL statements in online help screens.

You can request online 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 shows when you request online Help for SQL statements in is different from the syntax diagrams in the *HCL® Informix® Guide to SQL: Syntax*.

The conventions and rules governing SQL statement syntax in online help screens are described in the following list.

ABC

Any term in an SQL statement that is 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 is displayed in lowercase letters. In the previous example, 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.

|

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.

The *HCL® Informix® Guide to SQL: Syntax* contains more detailed syntax diagrams and instructions for interpreting the diagram format that is used in the publication.

Index

Special Characters

- %INFORMIXDIR%\bin 4
- $INFORMIXDIR/bin 4

A

- ansi command-line option to dbaccess 8
- ANSI compliance
  - checking SQL statements for 14
- ANSI database 26
- ANSI-compliant database
  - and SQLSTATE value 14
- ANSI, checking SQL statements for compliance 14

C

- c command-line option to dbaccess 8
- Client SDK and DB-Access 5
- CLOSE DATABASE statement 26
- Closing a database
  - from a menu 26
- Command files
  - executing from the command line 14
  - rules for naming 24
  - saving 24
- Command line
  - additional features 15
  - interactive input through standard input 15
  - reading from standard input 15
- Command options. 8
- Command-line options, displaying the main menu 8
- COMMIT 26
- COMMIT statement 26
- Committing transactions with the TRANSACTION menu 26
- CONNECT statement 17
- Connecting to database environment in background mode 17
- CONNECTION menu
  - PASSWORD prompt screen 29
  - USER NAME prompt screen 29
- Conventions
  - online Help 31
- Create option, TABLE menu 27
- CTRL-D 15
- CTRL-W 31
- Current statement, definition of 19

D

- d command-line option to dbaccess 8
- DATABASE menu
  - cLose option 26
  - options 10
  - selecting options from the command line 10, 10
- Database servers
  - connecting to 29
  - disconnecting implicitly 30
  - SELECT DATABASE SERVER screen 29
  - selecting from a menu 29
- Databases
  - cLose option 26
  - closing 26
  - closing implicitly 30
  - current 25
- DB-Access
  - environment variables affecting 4
  - USER NAME prompt screen 29
  - what it is 3
- DB-Access utility 6
- dbaccess
  - CSDK utility 3
  - database server utility 3
- dbaccess command options
  - -a 8
  - -ansi 8
  - -c 8
  - -cc 10
  - -cd 10
  - -d 8
  - -dc 10
  - -dd 10
  - -di 10
  - -dl 10
  - -ds 10
  - -e 8
  - -m 8
  - -q 8
  - -qc 11
  - -qd 11
  - -qi 11
  - -qm 11
  - -qn 11
  - -qs 11
  - -qu 11
  - -s 8
  - -t 8
  - -ta 12
  - -tc 12
  - -td 12
  - -ti 12
  - -V 8
  - -version 8
  - -X 8
  - connect_menu_option 8
  - database 8
  - database_menu_option 8
  - filename 8
  - query_menu_option 8
  - table 8
  - table_menu_option 8
- dbaccess, starting 8
- dbaccessdemo command 7
- DBACCNOIGN environment variable 4
- DBEDIT environment variable 4, 4
- DBFLTMASK environment variable 4, 4
- Defaults
  - database server, selecting 29
  - operating system editor 20
  - printer, sending output to 24
- DELIMIDENT environment variable 4, 4, 4
- Demonstration databases 6
  - installing 6
  - models 6
  - reinitializing 6
  - stores_demo setup 7
  - superstores_demo 6
  - superstores_demo setup 7
  - working directory required for 6
- Distributed databases, requesting table information on another server 28
- Domain name 29

E

- e command-line option to dbaccess 8
- Editor
  - restrictions 19
- Environment variables
  - DBACCNOIGN 4
  - DBANSIWARN 14
  - DBCENTURY 4
  - DBDATE 4
  - DBEDIT 4
  - DBFLTMASK 4
  - DBPATH 26
  - DELIMIDENT 4
  - GL_DATETIME 4
  - IFX_LONGID 4
  - INFORMIXTERM environment variable 4
  - ONCONFIG 26
  - setting for default editor 20
  - USE_DTENV 4
- Error messages
  - terminal setup 8
- Errors
  - connecting to a server after opening a database 30
  - running SQL statements 25
  - using command-line options 11
- Exit option
  - SQL menu 19
  - TABLE menu 27

F

- Files
  - .sql extension for command files 24
  - reading from standard input 15
  - saving current SQL statement in 24
  - storing query results in 24

G

- GL_DATETIME environment variable 4
- GLS library version 8

H

- Help
  - calling with CTRL-W 31
  - how to read syntax diagrams 31
  - online syntax information for SQL statements 31
- help command-line option to dbaccess 8
- history command-line option to dbaccess 8

I

- IFX_LONGID 4
- IFX_LONGID environment variable 4
- INFO FOR TABLE screen 28
- INFO menu
  - available options 28
  - displaying column information 28
  - exiting 28
  - listing tables 28
  - option on the TABLE menu 28
  - with SQL 28
- Info option
  - Exit option 28
  - TABLE menu 27
  - Table option 28
- Information
  - displaying for tables 28
- Information Schema 14
- INFORMIXDIR/bin directory 6
- Input
  - interactive 15
  - reading from standard 15

Interactive input, through standard input 15
Invoking DB-Access
checking for ANSI compliance 14
DATABASE menu options 10, 10
displaying the main menu 8
executing a command file 14
SQL menu options 11
TABLE menu options 12

K

Keys
CTRL-D 15
CTRL-W 31

M

Main menu
displaying from the command line 8
Query-language option 19
selecting the Table option 27
Menu options
SQL menu 19
TABLE menu 27
Menus
TRANSACTION 26

N

nohistory command-line option to dbaccess 8

O

Options
for SQL menu 19
for TABLE menu 27
Options. 8
Output option
Printer option 24
SQL menu 24
To-pipe option 24

P

PASSWORD prompt screen 29
Passwords, prompt in DB-Access interactive mode 17
PATH
DB-Access requirements 4
demonstration database and 6
Permissions, UNIX 6
Pipe
reading from 15
redirecting query results to a program 24
sending query results to 24
Printing the results of a query 24

Q

q command-line option to dbaccess 8
Query
sending results to a file 24
sending results to a pipe 24
sending results to a printer 24
QUERY-LANGUAGE menu, options 11
Query-language option
how to use 19
on the main menu 19

R

Reading from standard input 15
Restrictions, for SQL editor 19
ROLLBACK 26
Rolling back transactions 26
Running SQL statements
when there are errors 25

S

s command-line option to dbaccess 8
Save option
rules for naming saved files 24
SQL menu 24
Saving command files 24
Screens
for DATABASE menu 25
for SQL menu 19
for TABLE menu 27
INFO FOR TABLE 28
SELECT DATABASE SERVER 29
SELECT DATABASE SERVER screen 29, 29
Selecting a database server 29
Shell
Bourne 15
C 15
Korn 15
SQL
how to read syntax in online Help screens 31
using from a menu 20
SQL command files
must be in current directory 6, 6
SQL editor
editing restrictions 19
SQL menu
available options 19
OUTPUT menu 24
Output option 24
Save option 24
SAVE screen 24
selecting options from the command line 11
SQL statements
current, defined 19
editing 20
editing with the system editor 20
executing from standard input 15
interactive input on terminal 15
reading from standard input 15
redirecting query results 24
saving to a command file 24
selecting the SAVE screen 24
sending output to a file 24
sending output to a printer 24
sending query results to a pipe 24
syntax conventions in online Help 31
what happens when there are errors 25
sqlhosts, display connectivity information in 29
SQLSTATE value displayed 14
Starting DB-Access
command-line options 8
stdin, for interactive input 15
stores_demo 6, 6
superstores_demo 6, 6

T

t command-line option to dbaccess 8
Table
displaying from another server 28
displaying information on the screen 28
TABLE menu
available options 27
guidelines for using 27
Info option 28
options 12
selecting options from the command line 12
Terminal
as standard input 15
Text
editing with the system editor 20
Text editor
internal editor 19
SQL editor 19
TRANSACTION menu 26
Transactions, committing or rolling back 26
Troubleshooting, terminal type unknown 8

U

UNIX
case sensitivity and filenames 24
permissions 6
system editors 20
USE_DTENV environment variable 4
USER clause of CONNECT statement
in DB-Access interactive mode 17
User name
CONNECT statement with 17
specifying when connecting in background mode 17
USER NAME prompt screen 29
Using DB-Access with the Client SDK 5

V

V command-line option to dbaccess 8
version command-line option to dbaccess 8

W

Working directory 6

X

X command-line option to dbaccess 8

Y

Year values, two and four digit 4