SHADOW MAINFRAME ADAPTER CLIENT FOR ADABAS

SHADOW MAINFRAME ADAPTER CLIENT INSTALLATION AND ADMINISTRATION

POWERED BY

SHADOW

Date: January, 2004

Address inquiries to:

NEON Systems, Inc.
14100 SW Freeway, Suite 500
Sugar Land, Texas 77478

World Wide Web: http://www.neonsys.com

Phone: 1-800-505-6366
(281) 491-4200 (Corporate Sales, Customer Support)
Fax: (281) 242-3880

NEON
SYSTEMS, INC.

Contents

Part I: Introduction

Chapter 1: Introduction 1-1

Overview 1-1

Shadow Mainframe Adapter Client for ADABAS 1-1

Part II: Planning and Installation

Chapter 2: Planning the Shadow Mainframe Adapter Client Installation 2-1

Prerequisites 2-1

Supported Operating Systems 2-2

Support for Multi-Threaded Applications 2-2

Chapter 3: Installing Shadow Mainframe Adapter Client 3-1

Installation Files 3-1

Installing Shadow Mainframe Adapter Client on Windows 3-2

Installation Steps 3-2

Additional Installation Options 3-9

Installing Shadow Mainframe Adapter Client on UNIX 3-12

Installation Steps 3-12

Additional Installation Options 3-22

Common UNIX Errors 3-25

Part III: Administration

Chapter 4: Shadow Mainframe Adapter Client: Configuration 4-1

Connection Configuration 4-1

Step 1: Configure the Data Source 4-1

Step 2: Configure the Java Connection 4-13

Additional Configuration and Administration 4-15

DBCS Support 4-15

Determining Shadow Mainframe Adapter Client Version Information 4-15

Chapter 5: jDemo 5-1

Overview 5-1

Starting jDemo 5-2

Supported Features 5-3

Establishing a Connection 5-3

Viewing Data Source Information 5-6

Executing SQL 5-10
Closing jDemo 5-11

Chapter 6: Shadow Mainframe Adapter Client Trace Facility 6-1

Overview 6-1
Shadow Mainframe Adapter Client Trace Facility for Windows 6-2
Shadow Mainframe Adapter Client Trace. 6-2
Shadow Mainframe Adapter Client Dynamic JDBC Trace 6-4
Shadow Mainframe Adapter Client Trace Facility for UNIX 6-6
Shadow Mainframe Adapter Client Trace. 6-7
Shadow Mainframe Adapter Client Dynamic JDBC Trace 6-8
Shadow Mainframe Adapter Client Trace Facility Options 6-11
Severity Level Trigger. 6-11
Tracing Options 6-12

Chapter 7: Accessing ADABAS Data 7-1

Identifying and Authenticating ADABAS Users. 7-1
Executing the Sample Program 7-1
Using a CALL Statement to Obtain Map Metadata. 7-5

Chapter 8: Programming. 8-1

SQL Supported Statements 8-1
ASSIGNMENTS 8-2
CLOSE cursor 8-2
COMMIT [WORK]. 8-2
CONNECT 8-3
DBCLOSE. 8-3
DELETE 8-4
FETCH cursor 8-4
[FIND] SELECT 8-5
HISTOGRAM 8-9
HOLD 8-10
INSERT. 8-11
OPEN cursor 8-12
READ 8-13
RELEASE 8-16
ROLLBACK 8-16
SET 8-17
SHOW 8-18
TRACE 8-18
UPDATE 8-19
Statement Criterion 8-20
Selection Criterion. 8-21
WHERE Criterion 8-26

Search Criterion . . . 8-27
Cursor Processing . . . 8-28
Using Table Joins to Obtain Data from Multiple ADABAS Files . . . 8-31
Statement Syntax . . . 8-31
Considerations for Using Table Joins . . . 8-34
Concatenation Notation . . . 8-34
Concatenating Multiple Fields with a Superdescriptor . . . 8-34
Concatenating Values in a Redefined Field . . . 8-35
HEX Notation . . . 8-35

Part IV: Appendices

Appendix A: Shadow Mainframe Adapter Client Keywords . . . A-1

Setting a Keyword . . . A-1
Keyword Descriptions . . . A-1
Setting Keywords to Benefit Performance . . . A-32

Appendix B: National Language Support . . . B-1

Languages Supported . . . B-1
Setting the Language . . . B-2

Appendix C: JDBC Compliance . . . C-1

JDBC Compliance . . . C-1

About this Publication

This book contains user documentation for Shadow Mainframe Adapter Client, the Mainframe Adapter Client component of the Shadow Connect product.

How this Publication Is Organized

This book contains the following chapters:

Part I: Introduction

- Chapter 1, “Introduction,” provides a brief overview of the Shadow Mainframe Adapter Client product.

Part II: Planning and Installation

- Chapter 2, “Planning the Shadow Mainframe Adapter Client Installation,” describes the prerequisites for installing Shadow Mainframe Adapter Client.
- Chapter 3, “Installing Shadow Mainframe Adapter Client,” details the installation of Shadow Mainframe Adapter Client on all of the supported operating platforms. This chapter includes step-by-step installation instructions as well as installation considerations.

Part III: Administration

- Chapter 4, “Shadow Mainframe Adapter Client: Configuration,” covers general administrative procedures for configuring Shadow Mainframe Adapter Client.
- Chapter 5, “jDemo,” desribes the functionality provided by the jDemo program, a graphical user interface (GUI) to access data structures using Shadow Mainframe Adapter Client.
- Chapter 6, “Shadow Mainframe Adapter Client Trace Facility,” provides information on using the Shadow Mainframe Adapter Client Trace Facility to capture traces to solve problems that can occur during installation and use of Shadow Mainframe Adapter Client.
- Chapter 7, “Accessing ADABAS Data,” covers the use of Shadow Mainframe Adapter Client for accessing ADABAS data, including details about identifying and authenticating ADABAS users, executing the sample program, and displaying map metadata information.
- Chapter 8, “Programming,” details the SQL syntax statements supported by Shadow Mainframe Adapter Client for ADABAS.

Part IV: Appendices

- Appendix A, “Shadow Mainframe Adapter Client Keywords,” provides details about each Shadow Mainframe Adapter Client keyword and shows you how to change a keyword setting.
- Appendix B, “National Language Support,” describes the National Language Support (NLS) of Shadow Mainframe Adapter Client.
- Appendix C, “JDBC Compliance,” provides information about JDBC compliance.

Reader’s Comments

Please e-mail any comments or questions you have about our documentation to `support@neonsys.com`.

Thank you!

Part I

Introduction

CHAPTER 1: Introduction

This chapter gives a general introduction to the Shadow Connect product, including details about the Shadow Mainframe Adapter Client for ADABAS component.

Topics include the following:

- Overview
  - Shadow Mainframe Adapter Client for ADABAS

Overview

Shadow Connect is an efficient, easy-to-use, and flexible solution for integrating mainframe data sources and transaction environments to Mainframe Adapter Client/Mainframe Adapter Server and n-tier environments. The unique Shadow Connect architecture provides maximum flexibility with minimal impact on CPU cycles.

Shadow Mainframe Adapter Client for ADABAS

The Shadow Connect product offers various connectivity options, including Shadow Mainframe Adapter Client for ADABAS. With Shadow Mainframe Adapter Client for ADABAS, any JDBC enabled application can use standard JDBC facilities to make SQL requests directly to ADABAS. The end result is a returned relational result set, with no host programming.

Shadow Mainframe Adapter Client for ADABAS consists of the following components:

- Shadow Mainframe Adapter Server
- Shadow Mainframe Adapter Client
- Shadow Interface™ for ADABAS

Shadow Mainframe Adapter Server

The Shadow Mainframe Adapter Server component, which resides on the mainframe, offers the following benefits:

- Provides native access to ADABAS, CICS, DB2, IMS/DB, IMS/TM, Natural, and VSAM from a single tool.
- Eliminates of the need for a mid-tier gateway.
- Installs in less than one day.
- Incorporates centralized online monitoring, control, and diagnostic capabilities.

Shadow Mainframe Adapter Client

The Mainframe Adapter Client component, which consists of Shadow Mainframe Adapter Client, is a connector that enables Java applications to integrate z/OS data and transactional sources through the JDBC API. The Shadow Mainframe Adapter Client provide applications with transparent access to z/OS data and transactions through standard interfaces, and performs data and SQL dialect conversations, data compression, and network optimization in conjunction with the Shadow Mainframe Adapter Server.

The Shadow Mainframe Adapter Client component is installed on the Mainframe Adapter Client side or the mid-tier Mainframe Adapter Server by using a standard installation process; it is quick and easy to install and to use.

Features

Features and benefits include the following:

- **Robust Java Support.** Provides robust supports of JVM 1.2 (J2EE) and Java servlets and is JDBC 2.0 compliant. This support includes taking advantage of Java capabilities including multi-threading, connection pooling, and batch updates.
- **Multiple Platforms.** Runs on a growing range of platforms including HP-UX, IBM AIX, Linux, Linux/390, Sun Solaris, and Windows.
- **Multiple Communcations Protocols.** Operates over multiple communications protocols such as TCP/IP, SNA, or MQ Series.
- **Security Features.** Integrates tightly with enterprise security packages such as RACF, ACF2, and Top Secret. In addition, security is further improved by directly authenticating userids. Finally, reliability and resilience are improved by smoothly handling error messages coming from Security Access Facility (SAF), the z/OS security interface.
- **Two-Phase Commit Support.** Enables robust transactions by supporting two-phase commit.
- **Customizations and Optimization.** Delivers high performance with numerous field-proven customizations and optimizations. For example, data and SQL dialect conversations, data compression, and network optimization is performed in conjunction with the Shadow Mainframe Adapter Server.
- **Diagnostics and Control Features.** Simplifies support of the solution with end-to-end diagnostics. In addition, administration is made easy with comprehensive monitoring and control.
- **DBCS Support.** Permits use of multiple language character sets via DBCS support.

Shadow Interface for ADABAS

The Shadow Interface for ADABAS provides seamless, real-time controlled access to ADABAS data. It gives JDBC Mainframe Adapter Clients the ability to access ADABAS data in a relational model through SQL-based queries. The end result is a returned relational result set, with no host programming.

Features

The Shadow Interface for ADABAS also offers the following features:

- **Simplified ADABAS Access.** Allows distributed applications to access ADABAS with no mainframe coding required. In addition, the Shadow Interface for ADABAS offers a rapid method to Web-enable ADABAS applications.
- **Read/Write Access.** Provides read/write capabilities.
- **SQL Support.** Provides direct SQL or native access to ADABAS, including support for native ADABAS transactional commands, with no mainframe coding required. The SQL support allows SQL-based users to take advantage of key ADABAS features by retaining key constructs such as HISTOGRAM, GET ISN, GET LOGICAL, GET PHYSICAL, etc. In addition, a subset of ANSI SQL is supported to include such commands as SELECT, INSERT, UPDATE, and DELETE that only make sense in a relational model.
- **Support for Transaction Commands.** Support for transaction commands ensures the same Natural look and feel when coding on the desktop as when coding Natural on the mainframe.
- **Advanced Support.** Supports multi-value fields (MU), periodic groups (PE), cursor processing, and multi-file joins.
- **Data Mapping.** Offers online functionality to build an internal “map” that contains metadata information—a row/column/data type representation of the ADABAS field definitions. The result is a complete representation, in column and row format, of the physical ADABAS file. The map can be edited using a panel-based interface to establish more meaningful column names. In addition, multiple maps can be generated from a single ADAREP report.
- **Security Features.** Allows application of RACF, ACF2, or Top Secret security to ADABAS database assets without utilizing proprietary ADABAS security offerings.

Accessing ADABAS Data

The Shadow Interface for ADABAS provides ADABAS access using the subset of ANSI SQL and the superset of Software AG ADABAS native SQL.

The Mainframe Adapter Client makes a SQL request which is transformed by the Shadow Interface for ADABAS into single or multiple ADABAS direct calls, using data maps to outline the result set.

When the Mainframe Adapter Client application executes the SQL request, the Shadow Interface for ADABAS converts the request into native ADABAS calls and then converts the results into relational format for return to the Mainframe Adapter Client application. The end result is a returned relational result set, in a row and column view, with no host programming. The proper ADABAS control block, format buffer, record buffer, search buffer, value buffer and ISN buffer are created based on the SQL statement processing requirements.

Since ADABAS does not maintain a relational catalog describing information for every table, the Shadow Mainframe Adapter Server maintains the information in the Shadow Data Mapping Facility (DMF).

Recovery After Losing Mainframe Adapter Client Connectivity

If Mainframe Adapter Client connectivity is lost, the Shadow Interface for ADABAS component will issue a ROLLBACK and DBCLOSE on behalf of the Mainframe Adapter Client. The ADABAS UQE is eliminated for this connection once the DBCLOSE is issued. Dangling UQEs for application failures are not permitted.

Part II

Planning and Installation

CHAPTER 2:

Planning the Shadow Mainframe Adapter Client Installation

This chapter discusses the planning considerations required for installing Shadow Mainframe Adapter Client, part of the Mainframe Adapter Client component of the Shadow Connect product.

Topics include the following:

- Prerequisites
- Supported Operating Systems
- Support for Multi-Threaded Applications

Prerequisites

Before you install Shadow Mainframe Adapter Client on your machine, you must meet the following prerequisites:

- **All Users:**
  - A color depth on the target system of at least 256 colors.
  - JVM 1.2 or higher.

> ***Note:***
> A simple way to verify the availability of JVM 1.2 or higher is to execute the following from within the path:
>
> ```
> $ java -version
> ```

- **Windows Users:**
  - Approximately 30 MB of available disk space for installation.
  - A minimum of 16 MB of RAM.
  - A 386 or higher processor.
- **UNIX Users:** Up to 50 MB of available disk space for installation.
- **Linux Users:** glibc v2.1.1 or higher.
- **LU 6.2 Users:** One of the following:
  - Wall Data’s Rumba for Office
  - Wall Data’s Rumba for Mainframe
  - Attachmate’s EXTRA! v4.1 (or the 4.0 APPC upgrade) for Windows
  - IBM Network Services/DOS version 1.0
  - Any other LU 6.2 stack that supports APPC

Supported Operating Systems

Shadow Mainframe Adapter Client currently support the following operating platforms:

- AIX
- HP-UX
- Linux
- Linux/390
- Solaris
- Windows

Support for Multi-Threaded Applications

JDBC users require a “thread-safe” environment. Shadow Mainframe Adapter Client is enhanced to work with multi-threaded applications, which include many popular Mainframe Adapter Server products that use concurrent threads or thread-based connection pooling models to invoke operations. Shadow Mainframe Adapter Client is fully thread-safe and does not require thread affinity. No additional parameter or code changes need to be made.

CHAPTER 3: *Installing Shadow Mainframe Adapter Client*

This chapter describes the procedure for installing Shadow Mainframe Adapter Client, part of the Mainframe Adapter Client component of the Shadow Connect product.

Topics include the following:

- Installation Files
- Installing Shadow Mainframe Adapter Client on Windows
- Installing Shadow Mainframe Adapter Client on UNIX

Note:
Before installing Shadow Mainframe Adapter Client, ensure that the prerequisites have been met. For more information, see Chapter 2, “Planning the Shadow Mainframe Adapter Client Installation,” of this guide.

Installation Files

You can install Shadow Mainframe Adapter Client with installation files from the Shadow Connect CD. This CD should have been included in the package you received, containing the Shadow Connect product. When executing the installation file, you will need to pick the appropriate Shadow Mainframe Adapter Client file for your operating system, based on Table 3–1.

Table 3–1. Shadow Mainframe Adapter Client Files

Operating System	Installation File*
AIX 4.3 or higher	`IBMShadowMainframe Adapter Client_3_9_nnn_aix.bin`
HP-UX 11.0 or higher	`IBMShadowMainframe Adapter Client_3_9_nnn_hp.bin`
Linux	`IBMShadowMainframe Adapter Client_3_9_nnn_linux.bin`
Linux/390	`IBMShadowMainframe Adapter Client_3_9_nnn_linux390.bin`
Solaris 2.6 or higher	`IBMShadowMainframe Adapter Client_3_9_nnn_solaris.bin`

Table 3–1. Shadow Mainframe Adapter Client Files (Continued)

Operating System	Installation File*
32-bit Windows (XP, 2000, NT, 98, or 95)	`IBMShadowMainframe Adapter Client_3_9_nnn_win32.exe`

* Note: The "`nnn`" represents the Shadow Mainframe Adapter Client build number.

Installing Shadow Mainframe Adapter Client on Windows

Installing the Shadow Mainframe Adapter Client on Windows involves the following:

- Installation steps.
- Additional installation options.

Installation Steps

To install Shadow Mainframe Adapter Client on Windows, do the following:

1. Install Shadow Mainframe Adapter Client.
2. Verify the search paths.
3. Verify the installation.

Step 1: Install Shadow Mainframe Adapter Client

Note:
The Shadow Mainframe Adapter Client installation can be customized by changing the default settings, as detailed in "Additional Installation Options" on page 3-9.

To install Shadow Mainframe Adapter Client on any 32-bit Windows operating system, do the following:

1. Locate the Shadow Mainframe Adapter Client executable installation file, as listed in Table 3–1 on page 3-1.

2. Launch the Shadow Mainframe Adapter Client executable installation file. An introductory dialog box offering language choices is displayed, as shown in Figure 3–1.

Figure 3–1. Introductory Dialog Box -- Choosing a Language

3. From the drop-down menu, select the language.
4. Click **OK**. After the system prepares the files, the **Introduction** dialog box is displayed, as shown in Figure 3–2. This dialog box offers general information for proceeding with the installation.

Figure 3–2. Introduction Dialog Box

5. Click **Next**. The **License Agreement** dialog box is displayed, as shown in Figure 3–3.

Figure 3–3. License Agreement Dialog Box

6. After reviewing the license agreement, to accept the terms and continue with the installation, click the appropriate radio button.

   Note:
   If you do not accept the terms of the license agreement, you will not be able to continue with the installation.

7. Click **Next**. The **Choose Install Set** dialog box is displayed, as shown in Figure 3–4.

Figure 3–4. Choose Install Set Dialog Box

8. Choose the appropriate installation option. The installation options are as follows:

   - **Typical:** The typical Shadow Mainframe Adapter Client installation offers a full installation with the setting of user (local) environment variables.

   - **Administrative:** The administrative Shadow Mainframe Adapter Client installation offers a full installation with the setting of system (global) environment variables.

   Note:
   The selected option will be displayed in blue text.

9. Click **Next**. The **Choose Install Folder** dialog box is displayed, as shown in Figure 3–5.

Figure 3–5. Choose Install Folder Dialog Box

10. Choose the destination for the installed files. You may do either of the following:

    - Accept the default destination folder for the installation (`C:\Program Files\IBM\Shadow`).

    - Select a different location using the **Choose** button.

11. Click **Next**. The installation will search for previously installed versions of Shadow Mainframe Adapter Client.

    - If a previous Shadow Mainframe Adapter Client installation is not found, then the **Pre-Installation Summary** dialog box will be displayed, as shown in Figure 3–7 on page 3-7. See step 12 on page 3-7.

- If a previous Shadow JDBC installation is found, then the **Existing Installation Found** dialog box will be displayed, as shown in Figure 3–6.

Figure 3–6. Existing Installation Found Dialog Box

> ***Note:***
> Users are only allowed one installation location. For existing installations, users will not be given the option to select the destination for the installed files; the files will be installed to the same location as the existing installation by default.

a. Select one of the following options:

   – Completely overwrite the existing Shadow Mainframe Adapter Client installation, regardless of which installation is more current.

   – Upgrade the existing files only if they are older than the ones about to be installed.

b. Click **Next**. The **Pre-Installation Summary** dialog box will be displayed, as shown in Figure 3–7.

Figure 3–7. Pre-Installation Summary Dialog Box

12. Click **Install**. A screen is displayed that shows the progress of the installation for Shadow Mainframe Adapter Client. After the files have been installed, the **View Readme File** dialog box will be displayed, as shown in Figure 3–8.

Figure 3–8. View Readme File Dialog Box

13. Review the README file, if desired, and then click **Next**. The installation process is completed, as indicated by the **Install Complete** dialog box shown in Figure 3–9.

Figure 3–9. Install Complete Dialog Box

14. Click **Done** to exit.

> ***Note:***
> On Windows 98, the computer must be restarted after installing Shadow Mainframe Adapter Client.

Step 2: Verify the Search Paths

Verify that the CLASSPATH and PATH environment variables are set correctly for the Shadow Mainframe Adapter Client files. The Shadow Mainframe Adapter Client files can be located in any directory as long as the environment variables are set so that they can be found.

> ***Note:***
> If you chose a typical installation, user environment variables should be defined. If you chose an administrative installation, system level environment variables should be defined. The type of installation was selected in step 8 on page 3-5.

Verifying the CLASSPATH Environment Variable

Ensure that the CLASSPATH environment variable points to the jar file you wish to use. Assuming a default installation, the jar file will be copied to the following location:

```
C:\Program Files\IBM\Shadow\jdbc
```

The jar file is named as follows:

- **Debug Version:** `scjd12ts.jar`
- **Non-Debug Version:** `scjd12.jar`

Example

If the jar files are installed under `C:\Program Files\IBM\Shadow\jdbc`, to use the non-debug version of Shadow Mainframe Adapter Client, the CLASSPATH environment variable should include the following value:

```
C:\Program Files\IBM\Shadow\jdbc\scjd12.jar
```

Verifying the PATH Environment Variable

The PATH environment variable should point to the location of the `bin` directory.

Example

Assuming a default installation, the PATH environment variable should point to the following location:

```
C:\Program Files\IBM\Shadow\bin
```

Step 3: Verify the Installation

To verify the installation of Shadow Mainframe Adapter Client, you can use the jDemo program, a graphical interface demo program, that is installed with Shadow Mainframe Adapter Client.

Doc Reference:
The jDemo program is documented in Chapter 5, "jDemo," of this guide.

Additional Installation Options

The Shadow Mainframe Adapter Client installation can be configured and customized by specifying the installer properties. The installer properties are configured within a user-created text file called `installer.properties`. From this file, you can configure default installation properties, such as the installation directory location and the components to be installed, as well as specify the installation mode.

Configuring the Installation Properties

To configure the installation properties, simply do the following:

1. Within the directory where the actual Shadow Mainframe Adapter Client installation file is located, create a text file called `installer.properties`.

Note:
To use an `installer.properties` file that is located in a different directory, see "Command Line Installation Option" on page 3-11.

2. Specify the installation properties within the `installer.properties` file. The details of the properties that can be set are described in Table 3–2. An example `installer.properties` file is shown in Figure 3–10.
3. Run the installation program. With the `installer.properties` file located in the same directory, the installation program will automatically use the `installer.properties` file to configure the installation settings.

Table 3–2. Windows Installer Properties

Installer Property	Description
`USER_INSTALL_DIR`	This property specifies the destination for the installed files. You must provide a complete, absolute path to the location. ***Note:*** Use forward slashes for the path—even for Windows! See the example in Figure 3–10.
`INSTALLER_UI`	This property sets the installation mode. Possible values are as follows: • GUI: Executes the installation in GUI mode, as illustrated in the steps shown in "Step 1: Install Shadow Mainframe Adapter Client" on page 3-2. • SILENT: Executes the installation in silent mode. In silent mode, the installation GUI is not invoked; thus, no user interaction is required or accepted. If non-default installation settings are required, they must be configured manually in the `installer.properties` file. ***Note:*** When Shadow Mainframe Adapter Client is installed in silent mode, it will also be uninstalled in silent mode.
`CHOSEN_INSTALL_SET`	This property chooses the installation set option. Possible values are as follows: • TYPICAL • ADMINISTRATIVE For details about the various installation options, see step 8 on page 3-5.
`OVERWRITE_OR_UPGRADE`	This property determines how the installation should handle the existence of another Shadow Mainframe Adapter Client install, as follows: • OVERWRITE EXISTING INSTALL • UPGRADE OLDER FILES For more information about these options, see step a on page 3-6.

```
# This is a sample installer.properties file that can be used with the
# Shadow Mainframe Adapter Client installation.

############################################################################
# Use the USER_INSTALL_DIR setting to control where Shadow Mainframe Adapter
Client will
# be installed. Provide a complete, absolute path to the location.
# NOTE: Use forward slashes on Windows!
############################################################################
USER_INSTALL_DIR=C:/Program Files/IBM/Shadow

############################################################################
# Use the INSTALLER_UI setting to control whether the installer launches a
# GUI, a text interface, or runs silently. For a GUI, use "GUI" as the value.
# For the text interface, use "CONSOLE" as the value. For a silent install,
# use "SILENT" as the value. This sample is configured to run silently, as
# that is typical when running with a properties file.
############################################################################
INSTALLER_UI=SILENT

############################################################################
# The CHOSEN_INSTALL_SET setting allows you to choose between the high level
# install sets that are available in the install. They are "TYPICAL" and
# "ADMINISTRATIVE".
#
# TYPICAL: Includes Shadow Mainframe Adapter Client, samples, and user env
variables.
# ADMINISTRATIVE: Includes Shadow Mainframe Adapter Client, samples, and
system env
#                 variables.
############################################################################
CHOSEN_INSTALL_SET=TYPICAL

############################################################################
# If an existing installation is found on your system, the Windows installer
# will ask you if you would like to overwrite the existing install or upgrade
# the existing installation. Use the OVERWRITE_OR_UPGRADE setting to make
# that choice. Valid values are "OVERWRITE EXISTING INSTALL" or "UPGRADE
```

Figure 3–10. Example Installer Properties File -- Windows

Command Line Installation Option

By executing the installation program via the command line, you can force the installation to use an `installer.properties` file that is located in a different directory. Use the “`-f`” command line option to accomplish this, as follows:

```
C:\>install_path\install_file -f prop_path\\installer.properties
```

Note:
Note the "escaped" back-slashes used to specify the path for the `installer.properties` file.

Example

Assume the following:

- The installation program is located at `C:\Installs`.
- The `installer.properties` file is located at `C:\Properties`.

To run the installation with the `installer.properties` file, type the following on the command line:

```
C:\>C:\Installs\IBMShadowMainframe Adapter
Client_3_9_nnn_win32.exe -f C:\\Properties\\installer.properties
```

Installing Shadow Mainframe Adapter Client on UNIX

Installing Shadow Mainframe Adapter Client on UNIX involves the following:

- Installation steps.
- Additional installation options.
- Common UNIX errors.

Installation Steps

To install Shadow Mainframe Adapter Client on UNIX, do the following:

1. Set the file permissions.
2. Install Shadow Mainframe Adapter Client.
3. Verify the search paths.
4. (If using debugging) Set NEONTRACE log file permissions.
5. Verify the installation.

Step 1: Set the File Permissions

Before attempting to install Shadow Mainframe Adapter Client on UNIX, the file permissions for the installation file must be updated to allow execution of the file. The following command will set the appropriate permissions:

```
chmod 777 IBMShadowMainframe Adapter Client_3_9_nnn_xxx.bin
```

Where `nnn` is the build number and `xxx` is the suffix reflecting the UNIX platform (see Table 3–1 on page 3-1 for the list of file names).

Step 2: Install Shadow Mainframe Adapter Client

Considerations

The following should be considered before installing Shadow Mainframe Adapter Client:

- **Upgrade Considerations:** When performing an upgrade, ensure that all files were properly upgraded.

  Doc Reference:
  For details about resolving upgrade issues, see "Upgrade Issues" on page 3-19.

- **Installation Options:** Shadow Mainframe Adapter Client offers three installation methods:

  – **GUI Installation:** The GUI installation, which is the standard installation method, requires a UNIX system that has X Windows enabled. See "GUI Installation" on page 3-14.

  – **Console Installation:** The console installation, which executes via a text interface instead of the GUI, can be executed by users running the installation via a telnet connection or on any UNIX system that does not have X Windows enabled. See "Console Installation" on page 3-18.

  – **Silent Installation:** The silent installation requires no user input and does not invoke the GUI, so it can also be executed by users running the installation via a telnet connection or on any UNIX system that does not have X Windows enabled. See "Silent Installation" on page 3-19.

  Additional customizable options include changing the default installation settings.

  Note:
  When running the UNIX installation via Windows telnet, you need to install with either the console installation or the silent installation.

  Doc Reference:
  For details regarding the installation options, see "Additional Installation Options" on page 3-22.

GUI Installation

To install Shadow Mainframe Adapter Client using the GUI installation on a UNIX system with X Windows enabled, do the following:

1. Ensure that your DISPLAY environment variable is set appropriately.
2. Locate the appropriate binary file, as listed in Table 3–1 on page 3-1.
3. Launch the installation file. An introductory dialog box offering language choices is displayed, as shown in Figure 3–11.

Figure 3–11. Introductory Dialog Box -- Choosing a Language

4. From the drop-down menu, select the language.
5. Click **OK**. After the system prepares the files, the **Introduction** dialog box is displayed, as shown in Figure 3–12. This dialog box offers general information for proceeding with the installation.

Figure 3–12. Introduction Dialog Box

6. Click **Next**. The **License Agreement** dialog box is displayed, as shown in Figure 3–13.

Figure 3–13. License Agreement Dialog Box

7. After reviewing the license agreement, to accept the terms and continue with the installation, click the appropriate radio button.

    Note:
    If you do not accept the terms of the license agreement, you will not be able to continue with the installation.

8. Click **Next**. The **Choose Install Set** dialog box is displayed, as shown in Figure 3–14.

Figure 3–14. Choose Install Set Dialog Box

9. Choose the appropriate installation option. The installation options are as follows:

   - **Typical:** The typical Shadow Mainframe Adapter Client installation offers a full installation with the setting of user (local) environment variables.
   - **Administrative:** The administrative Shadow Mainframe Adapter Client installation offers a full installation with the setting of system (global) environment variables.

   Notes:
   - The selected option will be displayed in blue text.
   - Shadow Mainframe Adapter Client installations on Linux/390 ***do not*** set environment variables during the installation process.

10. Click **Next**. The **Choose Install Folder** dialog box is displayed, as shown in Figure 3–15.

Figure 3–15. Choose Install Folder Dialog Box

11. Choose the destination for the installed files. You may do either of the following:

   - Accept the default destination folder for the installation (`/home/IBM/Shadow`).
   - Select a different location using the **Choose** button.

12. Click **Next**. The **Pre-Installation Summary** dialog box will be displayed, as shown in Figure 3–16.

Figure 3–16. Pre-Installation Summary Dialog Box

13. Click **Install**. A screen is displayed that shows the progress of the installation for Shadow Mainframe Adapter Client. After the files have been installed, the **View Readme File** dialog box will be displayed, as shown in Figure 3–17.

Figure 3–17. View Readme File Dialog Box

14. Review the README file, if desired, and then click **Next**. The installation process is completed, as indicated by the **Install Complete** dialog box shown in Figure 3–18.

Figure 3–18. Install Complete Dialog Box

15. Click **Done** to exit.

Console Installation

The Shadow Mainframe Adapter Client installation can also be run in console mode instead of GUI mode. You will still have the same options and functionality as the GUI version, but everything will be displayed as text.

To install in console mode, do either of the following:

- Launch the installation from the command line using the "`-i console`" option, as follows:

  ```
  $ /home/file -i console
  ```

  Where `file` is the name of the Shadow Mainframe Adapter Client installation file (see Table 3–1 on page 3-1).

Or

- Add the following line to the `installer.properties` file:

  ```
  INSTALLER_UI=CONSOLE
  ```

Doc Reference:
For details regarding the `installer.properties` file, see "Additional Installation Options" on page 3-22.

Silent Installation

The Shadow Mainframe Adapter Client installation can also be run “silently.” In silent mode, the installation GUI is not invoked; thus, no user interaction is required or accepted. If non-default installation settings are required, they must be configured manually in the `installer.properties` file.

Note:
When Shadow Mainframe Adapter Client is installed in silent mode, it will also be uninstalled in silent mode.

To install in silent mode, do either of the following:

- Launch the installation from the command line using the “`-i silent`” option, as follows:

  ```
  $ /home/file -i silent
  ```

  Where `file` is the name of the Shadow Mainframe Adapter Client installation file (see Table 3–1 on page 3-1).

Or

- Add the following line to the `installer.properties` file:

  ```
  INSTALLER_UI=SILENT
  ```

Doc Reference:
For details regarding the `installer.properties` file, see “Additional Installation Options” on page 3-22.

Upgrade Issues

When upgrading Shadow Mainframe Adapter Client on UNIX, if your installation receives errors indicating that certain files could not be upgraded, check the installation log for errors such as the following:

```
Install File:  /path/file
               Status: ERROR
               Additional Notes: ERROR - ZeroGnd: /path/file
               (Cannot open or remove a file containing a running
               program.)
```

If such errors were noted, do the following:

1. Ensure that no programs are currently using any of the Shadow Mainframe Adapter Client components.

2. Manually remove any files that cannot be removed by the installation, such as `install.dir.xxx` (an NFS mount file).
3. Rerun the upgrade, checking the installation log to verify a successful installation.

Step 3: Verify the Search Paths

The search paths must be specified for both the jar file and the library file. In addition, if using Shadow Mainframe Adapter Client data sources, the search path for the `shadow.ini` configuration file must be specified. The Shadow Mainframe Adapter Client files can be located in any directory as long as the environment variables are set so that they can be found.

Note:

- If you chose a typical installation, user environment variables should be defined. If you chose an administrative installation, system level environment variables should be defined. The type of installation was selected in step 9 on page 3-16.
- Shadow Mainframe Adapter Client installations on Linux/ 390 ***do not*** set environment variables during the installation process; thus, any required environment variables must be set manually by the user.

To use Shadow Mainframe Adapter Client, verify the following search paths:

1. Verify the CLASSPATH environment variable.
2. Verify the search path for the library file.
3. Verify the SHADOW_INI environment variable.

Verifying the CLASSPATH Environment Variable

Ensure that the CLASSPATH environment variable points to the jar file you wish to use. Assuming a default installation, the jar file will be copied to the following location:

```
/home/IBM/Shadow/jdbc
```

The jar file is named as follows:

- **Debug Version:** `scjd12ts.jar`
- **Non-Debug Version:** `scjd12.jar`

Example

If the jar files are installed under `/home/IBM/Shadow/jdbc`, to use the non-debug version of Shadow Mainframe Adapter Client, the CLASSPATH environment variable should be set as follows:

```
CLASSPATH=/home/IBM/Shadow/jdbc/scjd12.jar:$CLASSPATH
```

Verifying the Search Path for the Library File

The jar file pointed to by the CLASSPATH variable requires a certain library file. Assuming a default installation, the jar file will be copied to the following location:

```
/home/IBM/Shadow/lib
```

The library file is named as follows:

- **Debug Version:** `libscjd12ts.xx`
- **Non-Debug Version:** `libscjd12.xx`

Where `xx` represents the file extension, which depends on your UNIX operating system.

Ensure that the directory name where the library is located is specified with the appropriate environment variable, as follows:

- **Linux, Linux/390, and Solaris:** The LD_LIBRARY_PATH environment variable should specify the directory name for the required library file.

- **HP-UX:** The SHLIB_PATH environment variable should specify the directory name for the required library file.

- **AIX:** The LIBPATH environment variable should specify the directory name for the required library file.

Example (Solaris)

Assuming the appropriate library file is installed at `/home/IBM/Shadow/lib`, the library path variable would be specified as follows for Solaris:

```
LD_LIBRARY_PATH=/home/IBM/Shadow/lib:$LD_LIBRARY_PATH
```

Verifying the SHADOW_INI Environment Variable

Ensure that the SHADOW_INI environment variable points to the `shadow.ini` configuration file. Assuming a default installation, the configuration file file will be copied to the following location:

```
/home/IBM/Shadow/bin/shadow.ini
```

Example

If the configuration file is installed at `/home/IBM/Shadow/bin`, the SHADOW_INI environment variable should be set as follows:

```
SHADOW_INI=/home/IBM/Shadow/bin/shadow.ini:$SHADOW_INI
```

Step 4: Set NEONTRACE Log File Permissions

Note:
This step is only required if debugging will be used.

If debugging is used, the NEONTRACE log file used for debugging must have READ/WRITE permission by the current effective user.

Step 5: Verify the Installation

To verify the installation of Shadow Mainframe Adapter Client, you can use the jDemo program, a graphical interface demo program, that is installed with Shadow Mainframe Adapter Client.

Doc Reference:
The jDemo program is documented in Chapter 5, “jDemo,” of this guide.

Additional Installation Options

The Shadow Mainframe Adapter Client installation can be configured and customized by specifying the installer properties. The installer properties are configured within a user-created text file called `installer.properties`. From this file, you can configure default installation properties, such as the installation directory location and the components to be installed, as well as specify the installation mode.

Configuring the Installation Properties

To configure the installation properties, simply do the following:

1. Within the directory where the actual Shadow Mainframe Adapter Client installation file is located, create a text file called `installer.properties`.

   Note:
   To use an `installer.properties` file that is located in a different directory, see “Command Line Installation Option” on page 3-24.

2. Specify the installation properties within the `installer.properties` file. The various preference settings are listed in Table 3–3. An example `installer.properties` file is shown in Figure 3–19.

3. Run the installation program. With the `installer.properties` file located in the same directory, the installation program will automatically use the `installer.properties` file to configure the installation settings.

Table 3–3. UNIX Installer Properties

Installer Property	Description
`USER_INSTALL_DIR`	This property specifies the destination for the installed files. You must provide a complete, absolute path to the location.
`INSTALLER_UI`	This property sets the installation mode. Possible values are as follows: • GUI: Executes the installation in GUI mode, as illustrated in the steps shown in “GUI Installation” on page 3-14. • CONSOLE: Executes the installation via a text interface instead of the GUI, as described in “Console Installation” on page 3-18. • SILENT: Executes the installation in silent mode, requiring no user input, as described in “Silent Installation” on page 3-19. In silent mode, the installation GUI is not invoked; thus, no user interaction is required or accepted. If non-default installation settings are required, they must be configured manually in the `installer.properties` file. ***Note:*** When Shadow Mainframe Adapter Client is installed in either console or silent mode, it will also be uninstalled in that mode.
`CHOSEN_INSTALL_SET`	This property chooses the installation set option. Possible values are as follows: • TYPICAL • ADMINISTRATIVE For details about the various installation options, see step 9 on page 3-16.

```
# This is a sample installer.properties file that can be used with the
# Shadow Mainframe Adapter Client installation.

############################################################################
# Use the USER_INSTALL_DIR setting to control where Shadow Mainframe Adapter
Client will
# be installed. Provide a complete, absolute path to the location.
############################################################################
USER_INSTALL_DIR=/usr/local/IBM/Shadow

############################################################################
# Use the INSTALLER_UI setting to control whether the installer launches a
# GUI, a text interface, or runs silently. For a GUI, use "GUI" as the value.
# For the text interface, use "CONSOLE" as the value. For a silent install,
# use "SILENT" as the value. This sample is configured to run silently, as
# that is typical when running with a properties file.
############################################################################
INSTALLER_UI=SILENT

############################################################################
# The CHOSEN_INSTALL_SET setting allows you to choose between the high level
# install sets that are available in the install. They are "TYPICAL" and
# "ADMINISTRATIVE".
#
# TYPICAL: Includes Shadow Mainframe Adapter Client, samples, and user env
variables.
# ADMINISTRATIVE: Includes Shadow Mainframe Adapter Client, samples, and
system env
```

Figure 3–19. Example Installer Properties File -- UNIX

Command Line Installation Option

By manually specifying the installer properties file, you can force the installation to use an `installer.properties` file that is located in a different directory. Use the “`-f`” command line option to accomplish this, as follows:

```
$ /install_path/install_file -f //prop_path//installer.properties
```

Notes:

- Note the “escaped” slashes used to specify the path for the `installer.properties` file.
- Command line options can also be used to change the installation mode. For more information, see “Step 2: Install Shadow Mainframe Adapter Client” on page 3-13.

Example

Assume the following:

- The installation program is located at `/installs`.
- The `installer.properties` file is located at `/properties`.

To run the installation with the `installer.properties` file, type the following on the command line:

```
$ /installs/file.bin -f //properties//installer.properties
```

Where `file` is the name of the Shadow Mainframe Adapter Client installation file (see Table 3–1 on page 3-1).

Common UNIX Errors

GUI Installation Failure

Symptom: An attempt to launch the Shadow Mainframe Adapter Client installation in GUI mode fails with the following error:

```
$ ./IBMShadowMainframe Adapter Client_3_9_nnn_xxx.bin
Preparing to install...
Extracting the JRE from the installer archive...
Unpacking the JRE...
Extracting the installation resources from the installer
archive...
Configuring the installer for this system's environment...

Launching installer...

Invocation of this Java Application has caused an
InvocationTargetException. This application will now exit. (LAX)

Stack Trace:
java.lang.NoClassDefFoundError
        at java.lang.Class.forName0(Native Method)
        at java.lang.Class.forName(Class.java:115)
        at java.awt.GraphicsEnvironment.getLocalGraphics
        Environment(GraphicsEnvironment.java:53)
        at java.awt.Window.<init>(Window.java:183)
        at java.awt.Frame.<init>(Frame.java:310)
        at java.awt.Frame.<init>(Frame.java:257)
        at com.zerog.ia.installer.Main.c(Unknown Source)
        at com.zerog.ia.installer.Main.main(Unknown Source)
        at java.lang.reflect.Method.invoke(Native Method)
        at com.zerog.lax.LAX.launch(Unknown Source)
        at com.zerog.lax.LAX.main(Unknown Source)
GUI-
```

Description: This error indicates that the DISPLAY environment variable is not set appropriately.

Internal Error in scodbcco.c

Symptom: A message warning of an internal error detected in `scodbcco.c`, such as the following:

```
Wed May 14 16:41:25 2003 internal error detected: file scodbcco.c
line 3416 rc = 0 from yp_get_default_domain
Wed May 14 16:41:25 2003 internal error detected: file scodbcco.c
line 957 rc = -1 from scinnsck
```

Description: This message indicates that the Shadow Mainframe Adapter Client Trace Facility is not configured properly; tracing has not been set to a valid log file. For example, in the following configuration, an appropriate path name is not specified:

```
NEONTRACE="INFO JDBCLOG=jdbc.log log=odbc.log"
```

Part III

Administration

CHAPTER 4: Shadow Mainframe Adapter Client: Configuration

Shadow Mainframe Adapter Client is part of the Mainframe Adapter Client component of Shadow Connect. Shadow Mainframe Adapter Client enables Java applications to integrate z/OS data and transactional sources through the JDBC API.

This chapter will cover the following topics:

- Connection Configuration
  - Step 1: Configure the Data Source
  - Step 2: Configure the Java Connection
- Additional Configuration and Administration
  - DBCS Support
  - Determining Shadow Mainframe Adapter Client Version Information

Connection Configuration

To configure the connection, do the following:

1. Configure the data source.
2. Configure the Java connection.

Step 1: Configure the Data Source

To use Shadow Mainframe Adapter Client, the data source settings must be specified. This may be done using one of the following methods:

- Using the JDBConnection string.
- Using the jConfig tool to configure a JDBC data source.

Using the Mainframe Adapter Clientions String

Note:
Any Shadow Mainframe Adapter Client keywords set in the connection string override any keywords set within the JDBC data source settings.

You can make a connection without having to create a data source by appropriately configuring the Mainframe Adapter Clientions string.

Mainframe Adapter Clention String Syntax

Note:
Instead of formatting the Mainframe Adapter Clention string manually, you can also use the jConfig tool graphical user interface (GUI) to create a connection string. For further details, see "Configuring a Mainframe Adapter Clention String" on page 4-13.

You must specify the following in the Mainframe Adapter Clention string:

- The data source name.
- All the required keywords to complete the connection.
- Any optional keywords.

The general format for the Mainframe Adapter Clention string is as follows:

```
jdbc:neon:<data-source-name>;
<keyword1>=<value1>;...;<keywordN>=<valueN>
```

Data Source Name

The `jdbc:neon:<data-source-name>` statement must be specified as follows:

- If the connection string is being used ***instead of*** a JDBC data source, the data source name does not apply and can be set to any arbitrary value.
- If a predefined JDBC data source is being used (possibly in conjuction with a connection string), the data source name must be set appropriately to identify the data source.

Doc Reference:
For more information about configuring a JDBC data source, see "Configuring a JDBC Data Source" on page 4-4.

Required Keywords

Specify the following required keywords in the Mainframe Adapter Clention string:

- **UID:** Specifies the mainframe userid.
- **PWD:** Specifies the mainframe password.
- **HOST:** Specifies the host name or IP address. This field can contain a hostname string or a TCP/IP address in dot-notation format.

- **PORT:** Specifies the port number on which Shadow Mainframe Adapter Server is listening. You can find this number by examining Shadow Mainframe Adapter Server. It is often set to 1200.

  Note:
  Failover Shadow Mainframe Adapter Servers can be specified by listing host/port combinations as follows:

  ```
  HOST=Host1,Host2,Host3;PORT=1200,1210,1500
  ```

  Based on the host/port combinations specified, Shadow Mainframe Adapter Client would try to connect to Host1:1200, Host2:1210, and Host3:1500, in that order.

  Specified host names can be the same host. Note that this syntax should ***not*** be used to specify Shadow Mainframe Adapter Server group directors.

- **SUBSYS:** Specify NONE.
- **CPFX:** Specify SHADOW.

Optional Keywords

In addition, any other optional Shadow Mainframe Adapter Client keywords can be set, as appropriate.

Note:
To use direct SQL access (where you do not have to prefix SQL statements with the CALL SHADOW_ADABAS statement), you must also set the DBTY keyword to ADABAS.

Doc Reference:
For a list of the keywords that are valid for the Mainframe Adapter Cliention string, see Appendix A, "Shadow Mainframe Adapter Client Keywords," of this guide.

Using the jConfig Tool

The jConfig tool is installed as part of the Shadow Mainframe Adapter Client installation. The jConfig tool offers an easy-to-use graphical interface that can be used to create and configure JDBC data sources and connection strings.

Note:
JDBC data source settings are stored in the `SHADOW.INI`. configuration file, the location of which is as follows:

- **Windows:** The location is specified in the registry.
- **UNIX:** The file is located as described in "Verifying the SHADOW_INI Environment Variable" on page 3-21 within Chapter 3, "Installing Shadow Mainframe Adapter Client," of this guide.

You can use the jConfig tool to do the following:

- Configure a JDBC data source.
- Import data sources.
- Configure a Mainframe Adapter Clientiion string.

Configuring a JDBC Data Source

To use the jConfig tool to create and/or configure a JDBC data source, do the following:

1. Launch the jConfig tool as follows:
   - **Windows Users:** Go to **Start →Programs →IBM →Shadow Mainframe Adapter Client →jConfig**.
   - **UNIX Users:** Launch the `ShadowConfig.sh` file, which will be installed to the following location by default:

     ```
     /home/IBM/Shadow/bin/
     ```

     Note:
     When the jConfig tool is launched, the script will display the command that launches jConfig as follows:

     ```
     Running java -DSHADOW_INI=...
     ```

     The "`-DSHADOW_INI=...`" parameter defines the location of the `shadow.ini` file. This parameter is based on the SHADOW_INI environment variable. If it is incorrect, please verify the SHADOW_INI setting as described in "Verifying the SHADOW_INI Environment Variable" on page 3-21 within Chapter 3, "Installing Shadow Mainframe Adapter Client," of this guide.

   The system displays the jConfig tool, as shown in Figure 4–1.

Note:
The jConfig tool also accepts the following command line arguments:

- The **browse** command starts jConfig in browse mode to browse and select data sources for use with jDemo (see Chapter 5, “jDemo,” of this guide).
- The **import** command will copy and migrate existing data sources for use with Shadow Mainframe Adapter Client. The user interface ***will not*** be displayed.

  Note: For UNIX, an absolute path for the `odbc.ini` file is required following the **import** command.

Figure 4–1. Main Window -- jConfig Tool

2. Either create a new data source or edit an existing data source, as follows:
   - Click **Add** to add a new JDBC data source.
   - Click **Import from ODBC.INI** to select existing data sources to import and convert into JDBC data sources. For more information, see “Importing Data Sources” on page 4-11.
   - Click **Configure** to edit an existing JDBC data source.
   - Click **Clone** to make a copy of an existing JDBC data source.
   - Click **New from Connection String** to create a JDBC data source based on a connection string that you provide.

The system displays the **JDBC Data Source Configuration** dialog box in a new window, as shown in Figure 4–2.

Note:
When creating a JDBC data source with the jConfig tool, the Shadow Mainframe Adapter Client version (debug versus non-debug) will be determined by the CLASSPATH setting. For more information, see the following within Chapter 3, “Installing Shadow Mainframe Adapter Client,” of this guide:

- **Windows Users:** See “Step 2: Verify the Search Paths” on page 3-8.
- **UNIX Users:** See “Step 3: Verify the Search Paths” on page 3-20.

Figure 4–2. JDBC Data Source Configuration

3. In the **Data Source Information** section, specify the following:

   - **DSN Name:** Type a name for the data source. For existing data sources, the name will be entered by default; however, you can change the data source name by typing over the existing name.
   - **DSN Description:** (Optional) Type a description for the data source, if desired.

> ***Note:***
> If you enter a description, spaces are allowed.

- **DBMS Type:** Select one of the following values, which is the means by which you will be accessing data:
  - **ADABAS:** Select **ADABAS** as the DBMS type for direct SQL access. If you do this, you do not have to prefix SQL statements with the CALL SHADOW_ADABAS statement.
  - **DB2:** If you select **DB2** as the DBMS type, your SQL statement must include the CALL SHADOW_ADABAS statement.

4. In the **TCP/IP Connection Information** section, specify the following:
   - **Host Name: (Required)** The host name or TCP/IP address for the connection. This field can contain a hostname string or a TCP/IP address in dot-notation format.
   - **Port Number: (Required)** The port number on which Shadow Mainframe Adapter Server is listening. You can find this number by examining Shadow Mainframe Adapter Server. It is often set to 1200.

     > ***Note:***
     > Failover Shadow Mainframe Adapter Servers can be specified by listing multiple host/port values, separated by commas. Shadow Mainframe Adapter Client will try to connect to each listed host/port pair in order.
     >
     > Specified host names can be the same host. Note that this syntax should ***not*** be used to specify Shadow Mainframe Adapter Server group directors.

   - **User ID:** Mainframe userid.
   - **Password:** Mainframe password. This is automatically encrypted (both on the screen and in the data source settings).
5. In the **Optional Information** section, click **Advanced**. The system displays the **Advanced Information** dialog box, as shown in Figure 4–3.

Figure 4–3. Advanced Information

6. Specify the following information:
   - **Subsystem:** Specify NONE.
   - **Plan Name:** Not applicable.
   - **Catalog Prefix:** Specify SHADOW.
   - **Default DB:** Not applicable.
   - **Table Filter:** Not applicable.
   - **Maximum Rows:** This field contains an integer that limits the number of rows returned from a single query. If no value is entered, no restriction is placed on the number of rows returned.
   - **Optimized Fetch:** Not applicable.
   - **Accessible Tables:** Not applicable.
   - **Retain Cursor:** Not applicable.
   - **Static SQL:** Not applicable.
7. Click **OK** to return to to the **JDBC Data Source Configuration** dialog box, as shown in Figure 4–2 on page 4-6.
8. If you wish to enable base tracing, click **Debug**. The system will display the **Debug Information** dialog box, as shown in Figure 4–4. You can use this dialog box to configure the Shadow Mainframe Adapter Client Trace Facility options.

Notes:

- To enable tracing, you must be using a debug version of Shadow Mainframe Adapter Client (as specified via the CLASSPATH setting).
- The **Debug Information** dialog box will ***only*** configure base tracing. To configure JDBC traces, you must set the NEONTRACE environment variable appropriately.

Doc Reference:

For more information about the Shadow Mainframe Adapter Client Trace Facility, see Chapter 6, “Shadow Mainframe Adapter Client Trace Facility,” of this guide.

Figure 4–4. Debug Information

9. Click **OK** to return to the **JDBC Data Source Configuration** dialog box, as shown in Figure 4–2 on page 4-6.
10. In the **Optional Information** section, click **Parameters**. The system will display the **Optional Parameter Information** dialog box, as shown in Figure 4–5. You can use this dialog box to set additional Shadow Mainframe Adapter Client keyword values.

Note:
Some of these keyword values may need to be set to non-default values. The keywords and their values are detailed in Appendix A, “Shadow Mainframe Adapter Client Keywords,” of this guide.

Figure 4–5. Optional Parameter Information

Note:
You can sort the list either by **Name** (the long name) or **Keyword** by simply clicking the appropriate column heading.

11. Click **OK** to return to to the **JDBC Data Source Configuration** dialog box, as shown in Figure 4–2 on page 4-6.
12. Click **OK** to return to the main window of the jConfig tool (see Figure 4–1 on page 4-5). Your entered values will be written into the `SHADOW.INI` settings for that JDBC data source.

Testing a JDBC Data Source Connection

Once you have created a JDBC data source, you can verify the configuration by testing the JDBC data source connection as follows:

1. Within the main window of the jConfig tool (see Figure 4–1 on page 4-5), from the **JDBC Data Sources** list, select the data source.

2. Click **Test Connection**. The system displays the **Test Connection** dialog box, indicating the results of the test, as shown in Figure 4–6.

Figure 4–6. Test Connection -- Successful Connection

If the connection test is not successful, the resulting error message will be displayed to offer assistance in problem diagnosis.

Importing Data Sources

When Shadow Mainframe Adapter Client is installed, any existing data sources that have been determined to be Shadow Mainframe Adapter Client data sources[1] will be automatically copied and migrated for use with the updated version of Shadow Mainframe Adapter Client.

Note:
After the data sources are imported, existing data sources will ***not*** be removed; it will be left to the user to delete unused data sources.

In addition, the jConfig tool also offers a feature to allow users to manually select existing data sources to be imported for use with Shadow Mainframe Adapter Client. To import data sources, do the following:

1. From the main window of the jConfig tool (see Figure 4–1 on page 4-5), click **Import from ODBC.INI**. The system will locate existing Shadow Mainframe Adapter Client data sources as follows:

   - **Windows:** The jConfig tool uses the registry to find Shadow Mainframe Adapter Client data sources defined in the `ODBC.INI` settings.

1 "Shadow Mainframe Adapter Client data sources" are defined as ODBC/TCPIP data sources with the APPL, DBTY, HOST, and PLAN keywords configured (empty string values are acceptable). For details about Shadow Mainframe Adapter Client keywords, see Appendix A, "Shadow Mainframe Adapter Client Keywords," of this guide.

- **UNIX:** The jConfig tool will display the **Select ODBC.INI File to Import Data Sources** dialog box, allowing users to navigate to and select the appropriate file from which to import existing data sources.

Upon finding the existing Shadow Mainframe Adapter Client data sources, the system will display the **Import Data Sources** dialog box, as shown in Figure 4–7.

Figure 4–7. Import Data Sources

Note:
By default, only the data sources that have been determined to be Shadow Mainframe Adapter Client data sources will be displayed. If you wish to display all existing data sources, select the **Show all data sources in ODBC.INI** check box.

2. From the list of data sources, select the ones to be imported.
3. Click **OK**.
4. The data source will be imported and added to the list of data sources shown in the main window of the jConfig tool.

Note:
If an imported ODBC data source has the same name as an existing JDBC data source, the imported data source name will be suffixed by a “2”.

Configuring a Mainframe Adapter Clientïon String

To use the jConfig tool to create and/or configure a Mainframe Adapter Clientïon string, do the following:

1. Use the jConfig tool to configure a JDBC data source as described in "Configuring a JDBC Data Source" on page 4-4.

   > ***Note:***
   > In order to create a connection string, you do ***not*** need to save the JDBC data source.

2. From the **JDBC Data Source Configuration** dialog box (see Figure 4–2 on page 4-6), click **Show Connection String**. The system will display the **Connection String** dialog box, as shown in Figure 4–8.

   From the **Connection String** dialog box, you can copy the connection string to paste into Java source codes or configuration dialog boxes for application Mainframe Adapter Servers.

Figure 4–8. Connection String

3. Close the dialog box as follows:

   - Click **Return to Configuration** to return to the **JDBC Data Source Configuration** dialog box (see Figure 4–2 on page 4-6).
   - Click **Exit Configuration** to return to the main window of the jConfig tool (see Figure 4–1 on page 4-5).

Step 2: Configure the Java Connection

There are two methods to make a Java connection with Shadow Mainframe Adapter Client:

- (Recommended) Using `javax.sql.DataSource` to connect.
- Using `java.sql.Driver` to connect.

Using javax.sql.DataSource to Connect

The recommended way of getting the correct `DataSource` object is via the Java Naming and Directory Interface (JNDI) service, which involves using `javax.sql.DataSource` to connect, as follows:

1. The `DataSource` object will depend on the environment, as follows:

   - **Users in a J2EE compatible environment:** Register the `com.neon.jdbc.DataSource` object with JNDI as follows:

     ```
     com.neon.jdbc.DataSource ds = new
     com.neon.jdbc.DataSource();
     ds.setDataSourceName ("XYZ");
     Context ctx = new InitialContext ();
     ctx.bind ("jdbc/MyDB", ds);
     ```

     Where `XYZ` specifies the name of the data source created in "Step 1: Configure the Data Source" on page 4-1.

   - **Users in a non-J2EE compatible environment:** Use the `com.neon.jdbc.SimpleDataSource` object after including the `jdbc2_0-stdext.jar` file in the CLASSPATH environment variable as follows (assuming default installations):

     – **Windows:** `SET CLASSPATH=C:\Program Files \IBM\Shadow\jdbc\jdbc2_0-stdext.jar;%CLASSPATH%`

     – **UNIX:** `CLASSPATH=/home/IBM/Shadow/jdbc /jdbc2_0-stdext.jar:$CLASSPATH`

2. Retrieve the `DataSource` object as follows:

   ```
   Context ctx = new InitialContext ();
   DataSource ds = (DataSource) ctx.lookup ("jdbc/MyDB");
   ```

3. Connect to the data source, as follows:

   ```
   Connection conn = ds.getConnection ("username", "password");
   ```

   Where `username` specifies the userid and `password` specifies the password.

Using java.sql.Driver to Connect

To use `javax.sql.DataSource` to connect, do the following:

1. Create the following driver class:

   ```
   Class cls = Class.forName("com.neon.jdbc.Driver");
   ```

2. Create a driver instance as follows:

   ```
   java.sql.Driver driver = (java.sql.Driver) cls.newInstance();
   ```

3. Connect to the data source using the Mainframe Adapter Clientíon string as follows:

```
java.sql.Connection conn =
driver.connect ("jdbc:neon:XYZ;UID=xxx;PWD=yyy", null);
```

Where `XYZ` specifies the name of the data source created in "Step 1: Configure the Data Source" on page 4-1, `xxx` specifies the userid, and `yyy` specifies the password.

Additional Configuration and Administration

Other configuration options and administrative features include the following:

- DBCS support.
- Determining Shadow Mainframe Adapter Client version information.

DBCS Support

Shadow Mainframe Adapter Client supports DBCS on all of its supported platforms ***except*** Windows 95/98.

Note:
In order to support DBCS on Linux, you must have glibc2.1.1 on your Linux machine.

Doc Reference:
For a list of the platforms supported by Shadow Mainframe Adapter Client, see Chapter 2, "Planning the Shadow Mainframe Adapter Client Installation," of this guide.

Determining Shadow Mainframe Adapter Client Version Information

Shadow Mainframe Adapter Client offers a feature to enable users to quickly and easily determine information about their version of Shadow Mainframe Adapter Client.

The `DriverInfo` class will return information about Shadow Mainframe Adapter Client.

Assuming a default location, the `DriverInfo` class will be located as follows:

- **Windows:** `C:\Program Files\IBM\Shadow\jdbc`
- **UNIX:** `/home/IBM/Shadow/jdbc`

Execute the `DriverInfo` class to determine Shadow Mainframe Adapter Client information as follows:

```
java DriverInfo -jdbc
```

The Shadow Mainframe Adapter Client name, version, build date, and other relevant information will be returned.

Example

Figure 4–9 shows an example of executing the `DriverInfo` class on Windows to return information about Shadow Mainframe Adapter Client.

Figure 4–9. Execute DriverInfo for Shadow Mainframe Adapter Client

CHAPTER 5: jDemo

The jDemo program is a graphical user interface (GUI) provided with Shadow Mainframe Adapter Client, part of the Mainframe Adapter Client component of the Shadow Connect product. The jDemo interface is used to access data structures using Shadow Mainframe Adapter Client.

Topics include the following:

- Overview
- Starting jDemo
- Supported Features
  - Establishing a Connection
  - Viewing Data Source Information
  - Executing SQL
- Closing jDemo

Overview

jDemo was installed onto your computer as part of the standard Shadow Mainframe Adapter Client installation. jDemo can be used to demonstrate some basic database functionality through Shadow Mainframe Adapter Client. It provides a GUI to allow the user to do the following:

- Make a database connection.
- Retrieve general data source information.
- Execute SQL statements to access ADABAS, CICS, DB2, IMS, and VSAM data, as well as Natural/ACI programs.
- Collect query results.

Prerequisites

jDemo can be run on any Mainframe Adapter Client machine meeting the following prerequisites:

- Shadow Mainframe Adapter Client must be installed.
- Java Runtime Environment v1.2 or greater must be available.

Starting jDemo

To start jDemo, do the following:

- **Windows Users:** From the Mainframe Adapter Client PC Windows **Start** menu, select **Programs →IBM →Shadow Mainframe Adapter Client → jDemo**.
- **UNIX Users:** Launch the `NeonJavaDemo.sh` file, which will be installed to the following location by default:

  ```
  /home/IBM/Shadow/bin/
  ```

  Note:
  When the jDemo tool is launched, the script will display the command that launches jDemo as follows:

  ```
  Running java -DSHADOW_INI=...
  ```

  The "`-DSHADOW_INI=...`" parameter defines the location of the `shadow.ini` file. This parameter is based on the SHADOW_INI environment variable. If it is incorrect, please verify the SHADOW_INI setting as described in "Verifying the SHADOW_INI Environment Variable" on page 3-21 within Chapter 3, "Installing Shadow Mainframe Adapter Client," of this guide.

The main jDemo window opens, as shown in Figure 5–1.

Figure 5–1. Main Window -- jDemo

Supported Features

jDemo supports the following features:

- **Establishing a Connection:** Users can establish single or multiple connections to data sources through various means:
  - Establish connections using a connection string.
  - Establish connections using a defined JDBC data source.
- **Viewing Data Source Information:** Users can view the following data related to a data source once a connection is established:
  - Display Shadow Mainframe Adapter Client name and version information for an established connection.
  - Display database name and version information for an established connection.
  - Display tables for the data source.
- **Executing SQL:** Users can execute SQL statements against and display query results.

Establishing a Connection

To establish a connection using jDemo, do one of the following:

- In the jDemo window, go to the **Connections** menu and select **Add New Connection**.

Or

- In the jDemo window, click **New**.

The system displays the **New Database Connection** dialog box, as shown in Figure 5–2.

Note:
You ***can*** have more than one connection open at the same time.

Figure 5–2. New Database Connection

From the **New Database Connection** dialog box, you can do either of the following:

- Establish a connection using a connection string.
- Establish a connection using a defined JDBC data source.

Establishing a Connection Using a Connection String

To establish a connection using a connection string, you can do either of the following:

- Type a valid connection string into the text box as follows:

  ```
  jdbc:neon:DataSourceName;keyword1=value1;keyword2=value2;...
  ```

  Where `DataSourceName` can be set to any arbitrary value because it does not apply and ***must not*** specify an existing JDBC data source. The keyword and value pairs should specify the desired configuration.

Doc Reference:
For more information about the connection string syntax, see “Using the Mainframe Adapter Clientíon String” on page 4-1 within Chapter 4, “Shadow Mainframe Adapter Client: Configuration,” of this guide.

Or

- Click **Browse** to use the jConfig tool to create a connection string.

Doc Reference:
For more information about using the jConfig tool to create a connection string, see "Configuring a Mainframe Adapter Clientlon String" on page 4-13 within Chapter 4, "Shadow Mainframe Adapter Client: Configuration," of this guide.

Note:
The **Userid** and **Password** fields can be used to specify the userid and password, respectively. This will prevent the password from being displayed in clear text in the connection string. Any values specified in those fields will override any values specified in the connection string.

Establishing a Connection Using a JDBC Data Source

To establish a connection using a JDBC data source, click **Browse**. The system opens the jConfig tool, which allows users to select an existing, defined JDBC data source or create a new JDBC data source.

Doc Reference:
For more information about creating and configuring a JDBC data source, see "Configuring a JDBC Data Source" on page 4-4 within Chapter 4, "Shadow Mainframe Adapter Client: Configuration," of this guide.

Note:
The **Userid** and **Password** fields can be used to specify the userid and password, respectively, overriding any values specified in the JDBC data source configuration.

Established Connections

For each established connection, the system issues a message to indicate the connection was established, as shown in Figure 5–3.

Figure 5–3. Connection Established

In addition, after the connection is made, the connection is displayed in the **Current Connection** list as “data-source-name -- com.neon.jdbc.Driver”.

Viewing Data Source Information

Once a connection is established, the following data related to a data source can be viewed:

- Shadow Mainframe Adapter Client name and version information.
- Database name and version information.
- Tables, table contents, and/or table columns for the data source.

Displaying Shadow Mainframe Adapter Client Information

To display the Shadow Mainframe Adapter Client name and version information, select the **Driver** tab, as shown in Figure 5–4.

Figure 5–4. jDemo -- Driver Tab

Displaying Database Information

To display the database name and version information, select the **Database** tab, as shown in Figure 5–5.

Figure 5–5. jDemo -- Database Tab

Displaying Tables and Related Information

To display the tables, table contents, and/or table columns for the data source, select the **Tables** tab. When the tab is first selected, no tables are listed.

Displaying Tables

To display the tables for the data source, do the following:

1. From the **Tables** tab, click **Load Tables**. The system displays the **Load Tables -- Parameters** dialog box, as shown in Figure 5–6.

Figure 5–6. Load Tables -- Parameters

2. If desired, specify criteria in the available fields to filter the tables that will be returned.
3. Click **OK**. jDemo will load and display the tables that meet the specified criteria, as shown in Figure 5–7.

Figure 5–7. jDemo -- Tables Tab

Displaying Table Contents

To display the table contents for a table, do the following:

1. Do one of the following to select a table:
   - If the table information for the desired table has already been loaded and displayed (as shown in Figure 5–7), then from the **Tables** tab, within the TABLE_NAME column, simply select the desired table name. The row will be highlighed, as shown in Figure 5–8.

Or

- Within the **Table** field, type the name of the table.

Figure 5–8. jDemo -- Tables Tab with Selected Table

2. Click **Show Table Contents**. The system displays the table contents in the **Results** pane, as shown in Figure 5–9.

Figure 5–9. Table Contents

Displaying Table Columns

To display the columns for a table, do the following:

1. If the table information for the desired table has already been loaded and displayed (as shown in Figure 5–7 on page 5-7), then from the **Tables** tab,

within the TABLE_NAME column, simply select the desired table name. The row will be highlighed (see Figure 5–8).

2. Click **Show Table Columns**. The system displays the **Load Table Columns -- Parameters** dialog box, as shown in Figure 5–10.

Figure 5–10. Load Table Columns -- Parameters

3. Specify criteria in the available fields to filter the tables that will be returned.

Note:
If a table was selected in step 1, then the **Schema Pattern** and **Table Name Pattern** fields will be specified according to the selected table; otherwise, you must specify the criteria appropriately.

4. Click **OK**. The system displays the table columns in the **Results** pane, as shown in Figure 5–11 on page 5-9.

Figure 5–11. Table Columns

Executing SQL

jDemo is a flexible tool for executing SQL queries.

Query Line Options

jDemo offers two query line options:

- **Single Line Queries:** (Default) By going to the **Options** menu and selecting **Single Line Query**, users can only enter single-lined SQL statements in the SQL text box.
- **Multi-Line Queries:** By going to the **Options** menu and selecting **Multi-Line Query**, users can only enter SQL statements spanning multiple lines in the SQL text box.

Executing a SQL Query

To execute a SQL query using jDemo, do the following:

1. In the **SQL** text box, type a SQL statement such as the following:

   ```
   SELECT FIRST_NAME, LAST_NAME FROM EMPLOYEE
   ```

   Doc Reference:
   For details about the SQL syntax for Shadow Mainframe Adapter Client for ADABAS, see “SQL Supported Statements” on page 8-1 within Chapter 8, “Programming,” of this guide.

2. Click **Query**. The query is executed and the results are displayed in the **Results** pane, as shown in Figure 5–12.

Figure 5–12. jDemo -- Executing a Query

Closing jDemo

To close jDemo, perform the following steps:

1. To disconnect from the data source, do one of the following:

   - In the jDemo window, go to the **Connections** menu and select **Remove Current Connection**.

   Or

   - In the jDemo window, click **Disconnect**.

2. Go to the **File** menu and select **Exit**.

CHAPTER 6: Shadow Mainframe Adapter Client Trace Facility

Debugging versions of Shadow Mainframe Adapter Client, part of the Mainframe Adapter Client component of the Shadow Connect product, include the Shadow Mainframe Adapter Client Trace Facility, which can produce trace log data. The output from this trace facility can be very useful in solving problems that can occur during installation and use.

This chapter will cover the following topics:

- Overview
- Shadow Mainframe Adapter Client Trace Facility for Windows
- Shadow Mainframe Adapter Client Trace Facility for UNIX
- Shadow Mainframe Adapter Client Trace Facility Options

Overview

The Shadow Mainframe Adapter Client Trace Facility traces selected Shadow events. It creates a trace file in a selected directory and records the traced events in that file. If a trace file by that name is already present, the trace facility adds any newly recorded events to it.

The format of the trace output data always includes a date, time, and pertinent information about each event. An example of trace output data is shown in Figure 6–1.

```
Fri Oct 01 22:19:30 1993   pcbColName       = 0x0a5f:759a
Fri Oct 01 22:19:30 1993   pfSqlType        = 0x0a5f:759e
Fri Oct 01 22:19:30 1993   pcbColDef        = 0x0a5f:75a0
Fri Oct 01 22:19:30 1993   pibScale         = 0x0a5f:7594
Fri Oct 01 22:19:30 1993   pfNullable       = 0x0a5f:759c
Fri Oct 01 22:19:30 1993 SQLDescribeCol exiting - return = SQL_SUCCESS(0)
Fri Oct 01 22:19:30 1993   szColName        = 'REMARKS'
Fri Oct 01 22:19:30 1993   *pcbColName      = 7
Fri Oct 01 22:19:30 1993   *pfSqlType       = SQL_CHAR(1)
Fri Oct 01 22:19:30 1993   *pcbColDef       = 64
Fri Oct 01 22:19:30 1993   *pibScale        = 9999
Fri Oct 01 22:19:30 1993   *pfNullable      = SQL_NULLABLE_UNKNOWN(2)
Fri Oct 01 22:19:30 1993 SQLFetch entered
Fri Oct 01 22:19:30 1993   lpstmt           = 0x095f:0000
Fri Oct 01 22:19:30 1993 internal error detected: file scodbcre.c line 1228
rc = 0 from scclxlat
Fri Oct 01 22:19:30 1993 SQLFetch exiting - return = SQL_SUCCESS(0)
Fri Oct 01 22:19:30 1993 SQLGetData entered
Fri Oct 01 22:19:30 1993   lpstmt           = 0x095f:0000
```

Figure 6–1. Example Shadow Mainframe Adapter Client Trace Output Data

Shadow Mainframe Adapter Client Trace Facility for Windows

Note:
In order to enable tracing, you must have specified the debug version of Shadow Mainframe Adapter Client (`scjd12ts.jar`) via the CLASSPATH environment variable. See “Step 2: Verify the Search Paths” on page 3-8 within Chapter 3, “Installing Shadow Mainframe Adapter Client,” of this guide.

The Shadow Mainframe Adapter Client Trace Facility offers the following features:

- **Shadow Mainframe Adapter Client Trace:** A tracing facility to create base traces and JDBC traces.
- **Shadow Mainframe Adapter Client Dynamic JDBC Trace:** Dynamic control of JDBC traces.

Shadow Mainframe Adapter Client Trace

You can create a base trace and/or JDBC trace with the Shadow Mainframe Adapter Client Trace Facility. You can enable and control the Shadow Mainframe Adapter Client Trace Facility by by appropriately setting the NEONTRACE environment variable.

When you use the NEONTRACE system environment variable to set the tracing options for the Shadow Mainframe Adapter Client Trace Facility, the trace starts at connection time. To set this variable, use the following form:

```
NEONTRACE=options JDBCLOG=jdbcpath LOG=path
```

Where:

options
Specifies the tracing options, delimited with spaces. Possible options are listed in “Shadow Mainframe Adapter Client Trace Facility Options” on page 6-11.

JDBCLOG
Creates a JDBC trace file.

jdbcpath
(If creating a JDBC trace) Specifies the full path name and file name of the JDBC trace file.

LOG
Creates a base trace file.

Notes:

- If the LOG keyword is present in the NEONTRACE environment variable, any other keywords must ***precede*** the LOG keyword.
- If you do not want to create a base trace file, set the LOG keyword to NULL as follows:

  ```
  LOG=NULL
  ```

path
(If creating a base trace) Specifies the full pathname and file name of the base trace file.

Considerations

In addition, users creating traces with the Shadow Mainframe Adapter Client Trace Facility should do the following:

- If your base tracing options are not specified in the NEONTRACE environment variable setting, then any tracing options specified via the data source settings (either in a connection string or in a data source defined with the jConfig tool) will set the base tracing options.

  Note:
  Tracing options set via the data source settings will ***only*** affect the base tracing, not the JDBC tracing.

- Ensure that the CLASSPATH environment variable points to the debug jar file, which is named `scjd12ts.jar`. Also, ensure that it appears before any other Shadow Mainframe Adapter Client jar file on the CLASSPATH.

Doc Reference:
For more information about setting the CLASSPATH environment variable, see "Step 2: Verify the Search Paths" on page 3-8 within Chapter 3, "Installing Shadow Mainframe Adapter Client," of this guide.

Examples

For example, to generate an INFO level JDBC trace to `C:\JDBCLOG.TXT` and a driver trace to `C:\NEONLOG.TXT`:

- **For Windows 95 or higher:** Add the following line to your `AUTOEXEC.BAT`:

```
NEONTRACE=INFO JDBCLOG=C:\jdbclog.txt LOG=C:\neonlog.txt
```

- **For Windows NT:** Open the **Control Panel**, select the **System** icon, and add the following line to the environment variable set:

```
NEONTRACE=INFO JDBCLOG=C:\jdbclog.txt LOG=C:\neonlog.txt
```

Shadow Mainframe Adapter Client Dynamic JDBC Trace

Dynamic JDBC tracing is supported with the scjdlog utility, which receives the tracing messages via TCP/IP and writes them to the log file on disk.

The scjdlog utility allows users to dynamically control the trace by supporting the commands shown in Table 6–2.

Table 6–1. scjdlog Utility Commands

Command	Description
`-p port-number`	Change the listening port of the scjdlog utility. The default value for the port number is 1201.
`-l NEONTRACE-options`	Change the tracing options for the JDBC dynamic trace. A trace option of NONE effectively "turns off" the JDBC dynamic trace.
`-f filename`	Change the file name of the JDBC trace log file. The default value for the file name is `neonj.txt`.
`-q`	Quit the scjdlog utility.

Enabling Shadow Mainframe Adapter Client Dynamic JDBC Tracing

You can perform dynamic JDBC tracing for Windows as follows:

1. Ensure that the CLASSPATH environment variable points to the debug jar file, which is named `scjd12ts.jar`. Also, ensure that it appears before any other Shadow Mainframe Adapter Client jar file on the CLASSPATH.

   Doc Reference:
   For more information about setting the CLASSPATH environment variable, see "Step 2: Verify the Search Paths" on page 3-8 within Chapter 3, "Installing Shadow Mainframe Adapter Client," of this guide.

2. Set an environment variable called NEONTRACE as follows:

   ```
   NEONTRACE=options JDBCLOG=:
   ```

   Where `options` consists of the options you initially select (see "Shadow Mainframe Adapter Client Trace Facility Options" on page 6-11), delimited with spaces, and the colon ("`:`") in the JDBCLOG keyword specifies the use of dynamic tracing.

   Optionally, you can also specify an IP address and a listening port, as follows:

   ```
   NEONTRACE=options JDBCLOG=ip-address:port-number
   ```

   Note:
   If you do not want to capture a base trace, set the LOG keyword of the NEONTRACE environment variable to NULL as follows:

   ```
   NEONTRACE=options JDBCLOG=: LOG=NULL
   ```

3. From a **Command Prompt** window, run `scjdlog.exe`. In a default installation, this file is located as follows:

   ```
   C:\Program Files\IBM\Shadow\bin
   ```

   The scjdlog utility is now waiting for tracing information to be written to it. Leave it running while you continue the required steps.

   Note:
   This window will become unusable because it is running the dynamic trace.

4. Open a second **Command Prompt** window and do the following to verify that the trace information is being captured:

   a. Assuming a default installation, go to the following directory:

      ```
      C:\Program Files\IBM\Shadow\bin
      ```

   b. Verify that the size of the `neonj.txt` file is increasing, indicating that the trace information is being written to this file.

5. From the second **Command Prompt** window, you may issue commands to control the trace dynamically with the scjdlog utility. For a description of the commands supported by the scjdlog utility, see Table 6–1.

   Examples:

   - To generate a dynamic INFO level JDBC trace:

     ```
     scjdlog -l INFO
     ```

   - To stop writing the trace log:

     ```
     scjdlog -l NONE
     ```

   - To quit the scjdlog utility:

     ```
     scjdlog -q
     ```

Shadow Mainframe Adapter Client Trace Facility for UNIX

Note:
In order to enable tracing, you must have specified the debug version of Shadow Mainframe Adapter Client (`scjd12ts.jar`) via the CLASSPATH environment variable. See “Step 3: Verify the Search Paths” on page 3-20 within Chapter 3, “Installing Shadow Mainframe Adapter Client,” of this guide.

The Shadow Mainframe Adapter Client Trace Facility offers the following features:

- **Shadow Mainframe Adapter Client Trace:** A tracing facility to create base traces and/or JDBC traces.
- **Shadow Mainframe Adapter Client Dynamic JDBC Trace:** Dynamic control of JDBC traces.

Shadow Mainframe Adapter Client Trace

You can create a base trace and/or JDBC trace with the Shadow Mainframe Adapter Client Trace Facility. You can enable and control the Shadow Mainframe Adapter Client Trace Facility by by appropriately setting the NEONTRACE environment variable.

When you use the NEONTRACE environment variable to set the tracing options for the Shadow Mainframe Adapter Client Trace Facility, the trace starts at connection time. To set this variable, use the following form:

```
NEONTRACE=options JDBCLOG=/jdbcpath LOG=/odbcpath
```

Note:
When setting the NEONTRACE environment variable via the command line, you must enclose the settings in quotes.

Where:

options
Specifies the tracing options, delimited with spaces. Possible options are listed in “Shadow Mainframe Adapter Client Trace Facility Options” on page 6-11.

JDBCLOG
Creates a JDBC trace file.

jdbcpath
(If creating a JDBC trace) Specifies the full path name and file name for the JDBC trace file.

LOG
Creates a base trace file.

Notes:
- If the LOG keyword is present in the NEONTRACE environment variable, any other keywords must ***precede*** the LOG keyword.
- If you do not want to create a base trace file, set the LOG keyword to NULL as follows:

```
LOG=NULL
```

odbcpath
(If creating a base trace) Specifies the full path name and file name for the base tracing file.

Considerations

In addition, users creating traces with the Shadow Mainframe Adapter Client Trace Facility should do the following:

- If your base tracing options are not specified in the NEONTRACE environment variable setting, then any tracing options specified via the data source settings (either in a connection string or in a data source defined with the jConfig tool) will set the base tracing options.

  Note:
  Tracing options set via the data source settings will ***only*** affect the base tracing, not the JDBC tracing.

- Ensure that the CLASSPATH environment variable points to the debug jar file, which is named `scjd12ts.jar`. Also, ensure that it appears before any other Shadow Mainframe Adapter Client jar file on the CLASSPATH.

  Doc Reference:
  For more information about setting the CLASSPATH environment variable, see "Step 3: Verify the Search Paths" on page 3-20 within Chapter 3, "Installing Shadow Mainframe Adapter Client," of this guide.

Example

For example, to generate an INFO level JDBC trace to `/tmp/jdbclog.txt` and a driver trace to `/tmp/neonlog.txt`, set the following environment variable:

```
NEONTRACE=INFO JDBCLOG=/tmp/jdbclog.txt LOG=/tmp/neonlog.txt
```

Shadow Mainframe Adapter Client Dynamic JDBC Trace

Dynamic JDBC tracing is supported with the scjdlog utility, which receives the tracing messages via TCP/IP and writes them to the log file on disk.

The scjdlog utility allows users to dynamically control the trace by supporting the commands shown in Table 6–2.

Table 6–2. scjdlog Utility Commands

Command	Description
`-p port-number`	Change the listening port of the scjdlog utility. The default value for the port number is 1201.
`-l NEONTRACE-options`	Change the tracing options for the JDBC dynamic trace. A trace option of NONE effectively “turns off” the JDBC dynamic trace.
`-f filename`	Change the file name of the JDBC trace log file. The default value for the file name is neonj.txt.
`-q`	Quit the scjdlog utility.

Enabling Shadow Mainframe Adapter Client Dynamic JDBC Tracing

You can perform dynamic JDBC tracing for UNIX as follows:

1. Ensure that the CLASSPATH environment variable points to the debug jar file, which is named `scjd12ts.jar`. Also, ensure that it appears before any other Shadow Mainframe Adapter Client jar file on the CLASSPATH.

    Doc Reference:
    For more information about setting the CLASSPATH environment variable, see “Step 3: Verify the Search Paths” on page 3-20 within Chapter 3, “Installing Shadow Mainframe Adapter Client,” of this guide.

2. Set an environment variable called NEONTRACE as follows:

    ```
    NEONTRACE=options JDBCLOG=:
    ```

    Note:
    When setting the NEONTRACE environment variable via the command line, you must enclose the settings in quotes.

    Where `options` consists of the options you initially select (see “Shadow Mainframe Adapter Client Trace Facility Options” on page 6-11), delimited with spaces, and the colon (“`:`”) in the JDBCLOG keyword specifies the use of dynamic tracing.

    Optionally, you can also specify an IP address and a listening port, as follows:

    ```
    NEONTRACE=options JDBCLOG=ip-address:port-number
    ```

> ***Note:***
> If you do not want to capture a base trace, set the LOG keyword of the NEONTRACE environment variable to NULL as follows:
>
> ```
> NEONTRACE=options JDBCLOG=: LOG=NULL
> ```

3. From a telnet session, do the following to prepare for the dynamic trace:

   a. Log on to your UNIX system.

   b. Run scjdlog. In a default installation, this file is located as follows:

      ```
      /home/IBM/Shadow/bin/
      ```

      The scjdlog utility is now waiting for tracing information to be written to it. Leave it running while you continue the required steps.

   > ***Note:***
   > This telnet session will become unusable because it is running the dynamic trace.

4. Open a second telnet session and do the following to verify that the trace information is being captured:

   a. Log on to your UNIX system.

   b. Go to the `/home/IBM/Shadow/bin` directory.

   c. Verify that the size of the `neonj.txt` file is increasing, indicating that the trace information is being written to this file.

5. From the second telnet session, you may issue commands to control the trace dynamically with the scjdlog utility. For a description of the commands supported by the scjdlog utility, see Table 6–2.

   Examples:

   - To generate a dynamic INFO level JDBC trace:

     ```
     scjdlog -l INFO
     ```

   - To stop writing the trace log:

     ```
     scjdlog -l NONE
     ```

   - To quit the scjdlog utility:

     ```
     scjdlog -q
     ```

Shadow Mainframe Adapter Client Trace Facility Options

The Shadow Mainframe Adapter Client Tracing Facility offers the following options to control the trace:

- Severity level trigger.
- Tracing options.

Severity Level Trigger

You can specify the severity of the Mainframe Adapter Client messages to be traced. When a severity level is specified, all events at and above that severity level will be traced and recorded. The severity options are listed below from the highest to the lowest level:

- **NONE:** With this option, no messages are traced or recorded. No trace file is created or opened. This option effectively turns off the Shadow Mainframe Adapter Client Trace Facility.

- **FATAL:** This option, the highest severity level, causes only program termination messages to be traced and recorded.

- **SEVERE:** (Default) This option causes only events of a severe or higher level to be traced and recorded.

- **ERROR:** This option causes all error messages to be traced and recorded. These messages usually include a description of what went wrong, a record of where the error was detected, any relevant return code, and a detailed error text message.

- **WARNING:** This option allows any warning messages issued by Shadow Mainframe Adapter Client to be traced and recorded.

- **INFO:** This option allows all informational events to be traced, including SQL. Note that values passed to and returned from Shadow Mainframe Adapter Client are included here. This option will cause a large quantity of information to be traced and recorded and is not recommended for general use; however, this option can be useful for application debugging.

- **DETAIL:** This option, the lowest severity level, causes all events, including function call events, to be traced. This value will cause huge amounts of information to be traced and recorded, and it is not recommended for general use.

Tracing Options

Tracing options are available for both base traces and JDBC traces.

Base Tracing Options

You can specify additional tracing options for the base trace as follows:

- **BUFFER:** This option traces network buffers transmitted between the Mainframe Adapter Client and Mainframe Adapter Server.
- **COMMTRACE:** This option traces detail information on all TCP/IP and LU 6.2 calls.
- **DATA:** This option traces the EBCDIC to ASCII translation of data. This option causes all conversion operations to and from DB2, ODBC C, and ODBC SQL data types to be traced. In each case, the input and output data is displayed in hexadecimal and character format. In addition, information is provided that designates which row and column of the table is currently being processed.
- **ENVLIST:** This option will trace out all environment variable settings into the Mainframe Adapter Client trace after a connection request, regardless of whether the connection request was successful or unsuccessful.
- **FLUSH:** Use this option with STAY. It helps to ensure that messages are not lost if your machine fails.
- **LOCKTRACE:** This option enables tracing for locks (mutex) that Shadow Mainframe Adapter Client obtains and releases.
- **SQL:** This option causes all SQL statements passed to the host to be traced, even if the trace severity level is higher than INFO. It also causes the column information for SELECT statements to be displayed, including the column number, name, and data type. For SQL_NUMERIC and SQL_DECIMAL data types, the precision and scale values are also displayed. This option is automatically set when INFO or DETAIL severity levels are selected.
- **STAY:** This option keeps the trace file open while conducting tracing. This can improve performance if you are using a disk file to store trace messages. This is generally a good option when doing heavy tracing.
- **STORAGE:** This option causes all storage GET and FREE operations to be traced. The trace record shows the name of the file requesting or freeing the storage area and the line number within the file. The record also indicates the size of the data area being obtained or freed, the address of the data area, and other important information. The final storage report is generated in the Windows environment when the DLL containing the storage manager functions is unloaded.

- **THREADID:** (Only available as an environment variable setting) This option traces a task or thread ID associated with every trace row.

  > ***Note:***
  > By default, the THREADID will always be included at the beginning of trace lines for traces in a multi-threaded environment.

- **UPDATE:** This option is for dynamic, live debugging.

JDBC Tracing Options

You can specify additional tracing options for the JDBC trace via the NEONTRACE environment variable as follows:

- **FLUSH:** This option helps to ensure that messages are not lost if your machine fails.
- **THREADID:** This option traces a task or thread ID associated with every trace row.

  > ***Note:***
  > By default, the THREADID will always be included at the beginning of trace lines for traces in a multi-threaded environment.

CHAPTER 7: Accessing ADABAS Data

This chapter contains instructions for accessing ADABAS data using Shadow Mainframe Adapter Client for ADABAS, part of the Shadow Mainframe Adapter Client component of the Shadow Connect product.

Topics include the following:

- Identifying and Authenticating ADABAS Users
- Executing the Sample Program
- Using a CALL Statement to Obtain Map Metadata

Identifying and Authenticating ADABAS Users

Shadow Mainframe Adapter Client for ADABAS uses the sign on userid passed by the Shadow Mainframe Adapter Client during logon for Mainframe Adapter Client authentication. The Shadow Mainframe Adapter Client and the Shadow Mainframe Adapter Server generate a unique identifier for each application instance. Shadow Mainframe Adapter Client for ADABAS uses this identifier to establish a connection with ADABAS that allows each application to be viewed as a distinct ADABAS user. This virtual connection identifier allows for the same Mainframe Adapter Client to have distinct UQEs for each application running on the Mainframe Adapter Client platform. Commands from one Mainframe Adapter Client application do not affect commands issued from another instance of the same application running on the Mainframe Adapter Client. The virtual connection ID is a 4-byte binary number.

Executing the Sample Program

The installation of Shadow Mainframe Adapter Client includes a sample Java program for accessing ADABAS data called `shadowadabas.java`. The sample program is located at the following location, assuming a default installation:

- **Windows:** `C:\Program Files\IBM\Shadow\samples\JDBC`
- **Unix:** `/home/IBM/Shadow/samples/jdbc`

Accessing ADABAS data using the Shadow Mainframe Adapter Client sample program involves the following procedures:

1. Verify that the CLASSPATH points to the appropriate jar file.

   Note:
   You should select a debug version of Shadow Mainframe Adapter Client if you are developing your own application or if you have problems that need to be diagnosed.

Doc Reference:
For details about setting the appropriate environment variables, see Chapter 3, “Installing Shadow Mainframe Adapter Client,” of this guide.

2. Edit the `shadowadabas.java` sample Java program as follows:

   a. Replace the existing placeholder for the connection string with a valid connection string.

      Doc Reference:
      For details about configuring the connection string, see “Step 1: Configure the Data Source” on page 4-1 within Chapter 4, “Shadow Mainframe Adapter Client: Configuration,” of this guide.

      Example:

```
String url   = "jdbc:neon:DSN;UID=******;PWD=******;
DBTY=ADABAS;HOST=MKT.NEONSYS.COM;PORT=1200;SUBSYS=NONE;
CPFX=SHADOW";
```

      Where:

      DSN
      Specifies the JDBC data source and must be as follows:

      - If the connection string is being used ***instead of*** a JDBC data source, the data source name does not apply and can be set to any arbitrary value.
      - If a predefined JDBC data source is being used (possibly in conjuction with a connection string), the data source name must be set appropriately to identify the data source.

      UID
      Specifies the mainframe userid.

      PWD
      Specifies the mainframe password.

DBTY
Specify the DBMS type, as follows:

- **To Use Direct SQL Access:** Specify ADABAS. If you do this, you do not have to prefix SQL statements with the CALL SHADOW_ADABAS statement; thus, you should change the string query accordingly.
- **To Use the CALL SHADOW_ADABAS Statement:** Specify DB2. If you do this, your SQL statement must include the CALL SHADOW_ADABAS statement.

HOST
Specify the host name or IP address. This field can contain a hostname string or a TCP/IP address in dot-notation format.

PORT
Specify the port number on which Shadow Mainframe Adapter Server is listening. You can find this number by contacting your Shadow Mainframe Adapter Server administrator. It is often set to 1200.

SUBSYS
Specify NONE.

CPFX
Specify SHADOW.

b. If necessary, replace the existing placeholder for the string query with a valid SQL statement.

Note:
If you set DBTY=ADABAS, you do not need to include the CALL SHADOW_ADABAS wrapper; otherwise, the SQL statement must be included in the CALL SHADOW_ADABAS wrapper.

Doc Reference:
For details about valid SQL statements, see Chapter 8, “Programming,” of this guide.

3. Compile the `shadowadabas.java` sample Java program by issuing the following command from the directory where the sample program is located:

```
javac shadowadabas.java
```

Note:
Depending on your JVM version, you may receive a notice that the sample program contains a deprecated API. In this case, you can do either of the following:

- Recompile the program with the `-deprecation` option, as follows:

  ```
  javac shadowadabas.java -deprecation
  ```

- Delete the following line from the sample program and recompile as instructed in step 3:

  ```
  DriverManager.setLogStream(System.out);
  ```

By compiling the `shadowadabas.java` sample Java program, the `shadowadabas.class` should be created in the same directory.

4. Execute the `shadowadabas.class` that was created by issuing the following command:

   ```
   java shadowadabas
   ```

   Based on the query provided in the sample program, the results should be as shown in Figure 7–1.

Figure 7–1. Sample Java Program Results

Using a CALL Statement to Obtain Map Metadata

For convenience, Shadow Mainframe Adapter Client offers users the ability to view the metadata information for ADABAS data maps on the Mainframe Adapter Client side with a simple CALL statement.

The format of the call is as follows:

```
CALL SHADOW_MAP(‘DESCRIBE’,‘mapname’)
```

Where `mapname` is the name of the map.

This call will return a result set with a single column called FORMAT that will contain detailed information on the fields of the map. The possible FORMAT column values and their SQL format equivalents are shown in Table 7–1.

Table 7–1. Converting FORMAT Column to SQL Format Equivalent

FORMAT Column	SQL Format Equivalent
CHARACTER	SQL_CHARACTER
NUMERIC	SQL_NUMERIC
DECIMAL	SQL_DECIMAL
INTEGER	SQL_INTEGER
SMALLINT	SQL_SMALLINT
FLOAT	SQL_FLOAT
DOUBLE	SQL_DOUBLE
DATE	SQL_DATE
MIME	SQL_MIME
TIMESTAMP	SQL_TIMESTAMP
VARCHAR	SQL_VARCHAR
LONGVARCHAR	SQL_LONGVARCHAR
BINARY	SQL_BINARY
VARBINARY	SQL_VARBINARY
LONGVARBINARY	SQL_LONGVARBINARY
UNICODE	SQL_UNICODE
UNICODE_VARCHAR	SQL_UNICODE_VARCHAR
UNICODE_LONGVARCHAR	SQL_UNICODE_LONGVARCHAR

CHAPTER 8: Programming

This chapter contains the supported SQL syntax statements for Shadow Mainframe Adapter Client for ADABAS, part of the Mainframe Adapter Client component of the Shadow Connect product.

Topics include the following:

- SQL Supported Statements
  - Syntax
- Statement Criterion
  - Selection Criterion
  - WHERE Criterion
  - Search Criterion
- Cursor Processing
  - Using the FOR Keyword
  - Omitting the FOR Keyword
- Using Table Joins to Obtain Data from Multiple ADABAS Files
- Concatenation Notation
  - Concatenating Multiple Fields with a Superdescriptor
  - Concatenating Values in a Redefined Field
- HEX Notation

SQL Supported Statements

Table 8–1 shows all of Software AG's Native SQL and ANSI Standard SQL syntax supported by Shadow Mainframe Adapter Client for ADABAS:

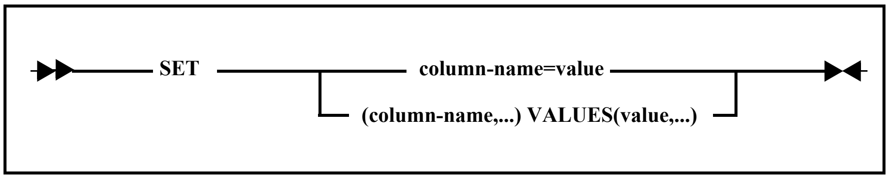
Table 8–1. Supported SQL Statements

Statements	ADABAS Native SQL	ANSI SQL
CLOSE cursor	Y	Y
COMMIT [WORK]	Y	Y
CONNECT	Y	N
DBCLOSE	Y	N
DELETE	Y	Y
FETCH cursor	Y	Y
[FIND] SELECT	Y	Y
HISTOGRAM	N	N
HOLD	Y	N

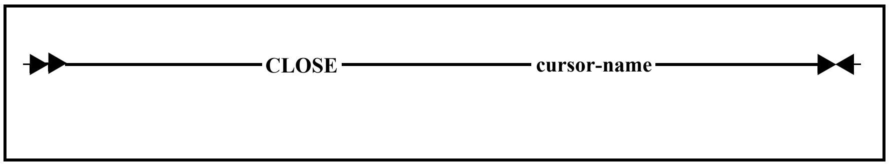
Table 8–1. Supported SQL Statements (Continued)

Statements	ADABAS Native SQL	ANSI SQL
INSERT	Y	Y
OPEN cursor	Y	Y
READ ISN/LOGICAL/PHYSICAL	Y	N
RELEASE	Y	N
ROLLBACK [WORK]	Y	Y
SET	N	N
SHOW	N	N
TRACE	Y	N
UPDATE	Y	Y

ASSIGNMENTS

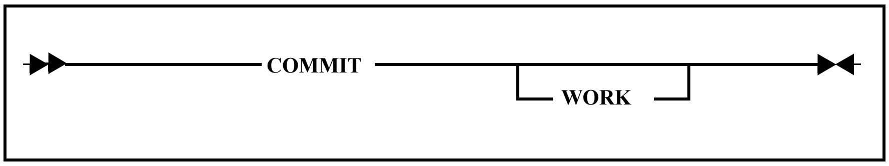


CLOSE cursor



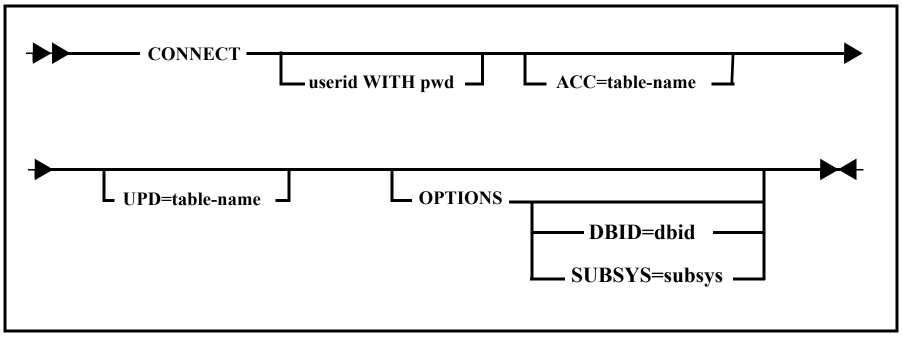
The CLOSE statement allows the termination of a previously opened cursor. Upon termination of the cursor, an ADABAS RC command is issued that uses the 4-character cursor name as the ADABAS command ID.

COMMIT [WORK]



The COMMIT statement, which indicates that the application unit of work is to be hardened on ADABAS, issues an ADABAS ET command. The COMMIT must be issued by the application program after any UPDATE, DELETE, or INSERT requests. The opposite of the COMMIT statement is the ROLLBACK statement (see “ROLLBACK” on page 8-16)

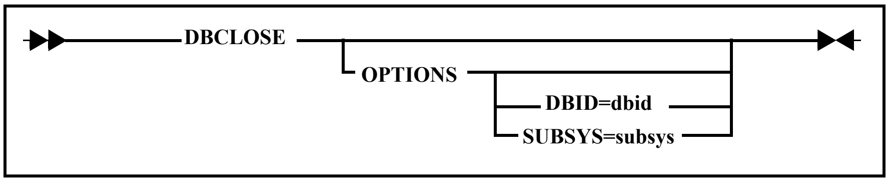
CONNECT



The CONNECT statement allows the application to explicitly establish a connection to ADABAS by issuing an OP command, thereby establishing the ADABAS UQE. If the CONNECT statement is not issued, the ADABAS connection is implicitly established on the first access or update statement issued from Shadow Mainframe Adapter Client for ADABAS.

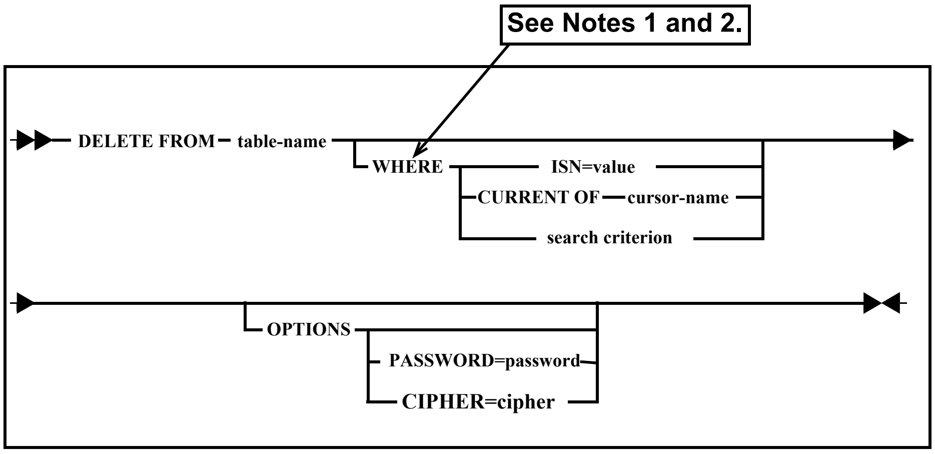
If the ADABAS startup parameter is OPENRQ=YES, then the connect statement should be the first CALL SHADOW_ADABAS statement issued from the Mainframe Adapter Client application.

DBCLOSE



The DBCLOSE statement allows the application to explicitly disconnect an ADABAS session without affecting the Shadow connection. The DBCLOSE statement causes an ADABAS CL command to be issued, which also implies an ADABAS COMMIT (ET).

DELETE



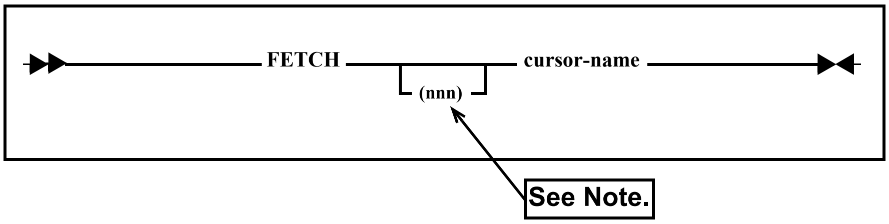
The DELETE statement allows for the deleting of one or more rows of ADABAS data. Search criteria may be specified as a given ISN value or the current record pointed to by an open and active cursor. This statement causes an ADABAS E1 command to be issued. It should subsequently be followed by a COMMIT or ROLLBACK command to have the delete committed or rolled back in ADABAS.

Notes:

1. Without the WHERE clause, every row in the requested table is deleted.
2. The WHERE criteria requires special consideration for use with MU and PE fields. For more information, see “Special Considerations for Updating or Inserting into MU Fields Defined as NU” on page 8-26.

Programming Tip:
The number of rows affected by the DELETE statement can be returned by the `stmt.executeUpdate()` method; however, the maximum number that can be returned is 65,535.

FETCH cursor



The FETCH statement allows the retrieval of one or more rows of data from an ADABAS table by issuing the command necessary to continue retrieving rows based on the original statement. This command can only be executed after a statement containing a DECLARE CURSOR FOR clause has been executed followed by an OPEN cursor statement. The FETCH command must be followed by a CLOSE cursor statement.

A syntax error in DECLARE CURSOR will not be checked until FETCH is issued.

Note:

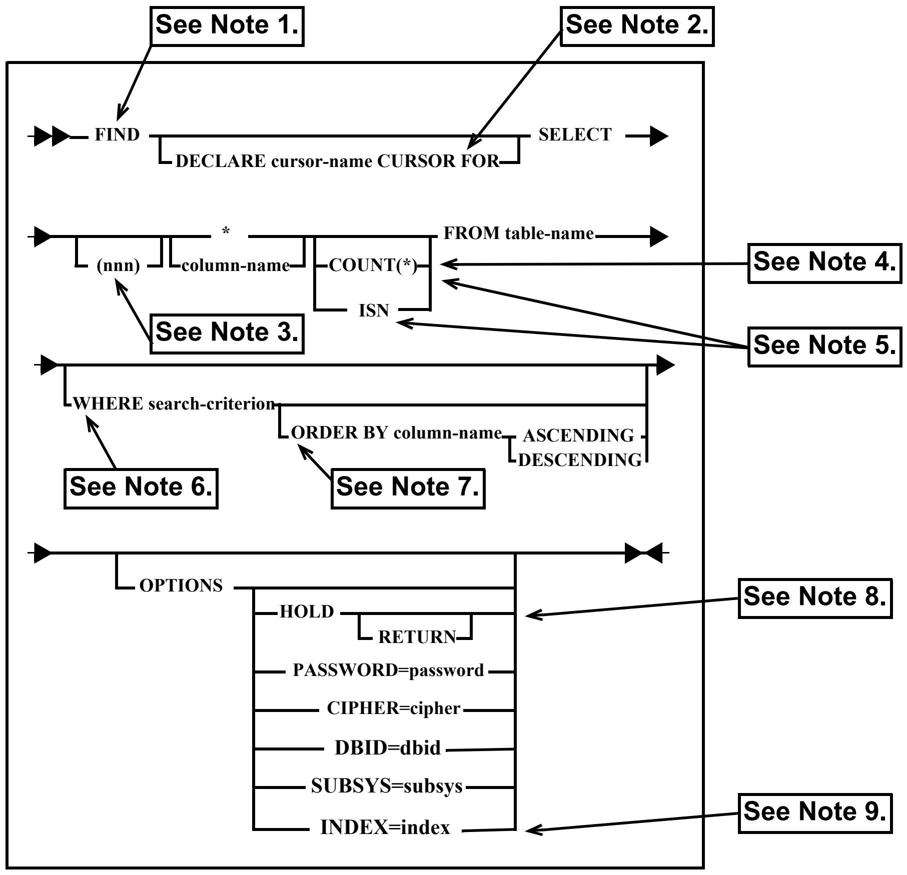
If the row delimiter specification (nnn) is not specified, the number of rows returned is 1.

Example:

The following pseudo logic describes the use of the FETCH statement:

```
DECLARE C001 CURSOR FOR SELECT * FROM EMPLOYEES
OPEN C001
dowhile (SQL_RC = 0)
    FETCH C001
loop
CLOSE C001
```

[FIND] SELECT



The SELECT statement allows the collection of data from ADABAS tables (files). The SELECT criteria determines which ADABAS columns (fields) are returned, while the WHERE criteria determines the set of rows that are returned to the application. In ADABAS direct call terminology, the SELECT criteria determines what is generated in the ADABAS format buffer and the WHERE criteria determines what is created in the ADABAS search and value buffers.

Note:
Shadow Mainframe Adapter Client for ADABAS also supports the SELECT statement for joining multiple tables. For more information, see "Using Table Joins to Obtain Data from Multiple ADABAS Files" on page 8-31.

Notes:

1. If the FIND keyword is omitted in the statement, Shadow Mainframe Adapter Client for ADABAS will determine the best command type for the request based on the contents of the request.

2. For more information about the use of the FOR keyword in cursor processing, see "Cursor Processing" on page 8-28.

3. The row delimiter specification (nnn) is not a valid option for the DECLARE CURSOR. The syntax error will be detected during FETCH.

4. The COUNT(*) keyword is used to return the number of rows for the SELECT statement. This is similar to the Natural statement FIND NUMBER. If no other column name is specified, then one row with a column name COUNT will be returned. If other column names are also specified, only columns from the first row that meet the WHERE criteria and the COUNT column are returned.

   Note:
   If you use the COUNT(*) keyword, then you must specify WHERE criteria ***unless*** the data map from which you SELECT contains a field with level 11 or that is redefined as COUNT.

5. The keywords COUNT(*) and ISN must be coded in the SELECT statement following any column name that are to be returned.

6. The WHERE criteria requires special consideration for use with MU and PE fields. For more information, see "Special Considerations for Searching for Particular Occurrences MU and PE Fields" on page 8-26. The WHERE clause can also contain an ISN specification as in ISN=nnn which specifies the ADABAS Internal Sequence Number of the record desired. ISN can be specified in the normal SELECT/FIND SELECT as well as the JOIN statement (for more information see "Considerations for Using Table Joins"

on page 8-34). However, there is a restriction. If ISN is specified in the WHERE criteria of a SELECT/ FIND SELECT statement, it can be the only criteria in the WHERE clause.

The following example is valid:

```
SELECT PERSONNEL_ID FROM EMPLOYEES WHERE ISN = 177
```

The following example is not valid because there is more than one criteria in the WHERE clause:

```
SELECT PERSONNEL_ID FROM EMPLOYEES WHERE ISN = 177 AND
PERSONNEL_ID = "50005800"
```

7. Up to 3 fields can be specified in the ORDER BY clause. In addition, the ORDER BY clause can also specify column position. The ORDER BY clause executes the appropriate commands for an ADABAS sort.

   Note:
   The column specified in the ORDER BY clause must be a descriptor; if a non-descriptor is specified, an ADABAS response code 28 will occur.

   If the ORDER BY clause directional option is not explicitly issued to specify an ascending or descending order, the default value will be an ascending order. If a directional option ***is*** explicitly specified, the following rules apply:

   - The valid syntax for issuing the ORDER BY clause directional option is as follows:
     - **For ascending order:** Either ASC or ASCENDING.
     - **For descending order:** Either DESC or DESCENDING.
   - ADABAS only supports global ASCENDING/DESCENDING requests; therefore, conflicting directional keywords are not allowed. For example, if more than one column is specified in the ORDER BY clause and the user wants to specify ASC or DESC, all column need to have either ASC or DESC—but not a combination of both. Alternatively, the user can specify the directional keyword in just one of the column and it will apply to all of the columns (i.e., ORDER BY COL1, COL2 DESC, COL3 will result in all columns being sorted in descending order). If the user tries to use mixed directional keywords, a syntax error message will be returned to the Mainframe Adapter Client.

8. The HOLD option in the OPTIONS clause allows you to lock the selected records. If any one of the records is locked by another user, the SELECT statement will wait until the record is released. To avoid the wait, use the RETURN option, and instead of waiting until the record is released, the SELECT will get a response code of -145.

9. OPTIONS INDEX is only valid for SELECT statements using dynamic mapping.

Examples:

The following results in an L2/RC command sequence to ADABAS:

```
SELECT * FROM EMPLOYEES
```

The following results in an S1/L1/RC command sequence:

```
SELECT * FROM EMPLOYEES WHERE LAST_NAME >= “JONES”
```

The following results in an S1/L1/RC command sequence:

```
SELECT * FROM EMPLOYEES WHERE LAST_NAME = JONES AND SEX = “F”
```

The following specifies a specific record to be displayed:

```
SELECT * FROM EMPLOYEES WHERE ISN=1892
```

The following results in a single S1/RC command sequence. The ISN quantity field of the ADABAS control block is returned to the Mainframe Adapter Client.

```
SELECT COUNT (*) FROM EMPLOYEES WHERE LAST_NAME = “JONES”
```

Note:
For the SELECT statement, either an L2 or S1 will be used. L2 is be used when there is no WHERE criteria specified. In all other cases, S1 will be used.

The following will return all the MU/PE columns with values:

```
SELECT NAME, ADDRESS(*), TELEPHONE(*) FROM EMPLOYESS WHERE
DEPARTMENT="ACCT"
```

For MU fields, the following will return all rows where any occurrence of that MU field matches the value specified:

```
SELECT AZ001, AZ002, AZ003, AZ004, AZ005 FROM EMPLOYEES WHERE
AZ001 = "FRE"
```

For PE fields without SEARCH_BY_PE_INDEX, the following will search all occurrences and will return all rows where any occurrence of that PE field matches the value specified, ignoring the index:

```
SELECT AS001, AS002, AS003, AS004, AS005 FROM EMPLOYEES WHERE
AS001 = 25000
```

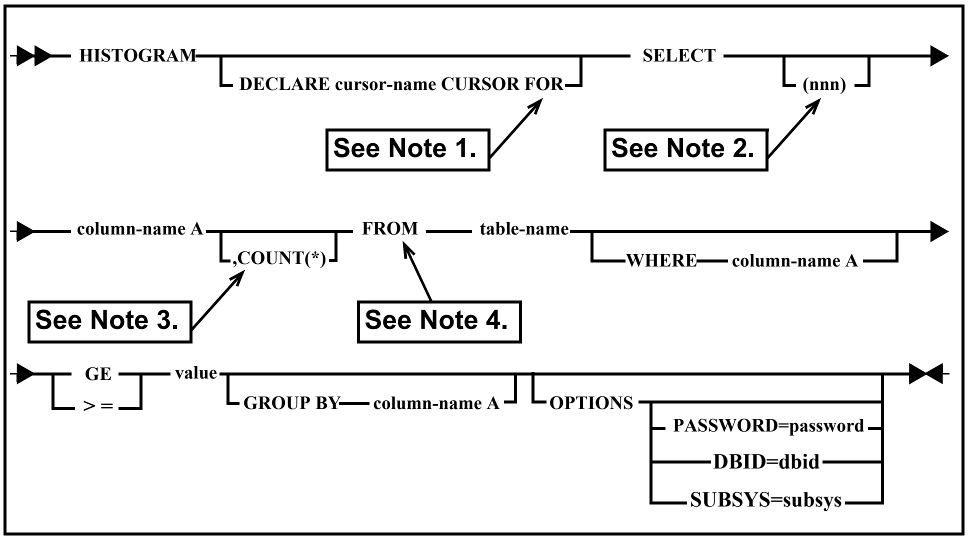
For PE fields with SEARCH_BY_PE_INDEX, the following will target the rows that match the particular occurrence of the PE field specified:

```
SELECT AS001, AS002, AS003, AS004, AS005 FROM EMPLOYEES WHERE
AS001 = 25000
```

For a field, AA, that was redefined into fields AA_PART_1 and AA_PART_2, the following statements are valid:

```
SELECT AA_PART_1, AA_PART_2 FROM EMPLOYEE
SELECT * FROM EMPLOYEE WHERE AA = "ABCDE|12345"
```

HISTOGRAM



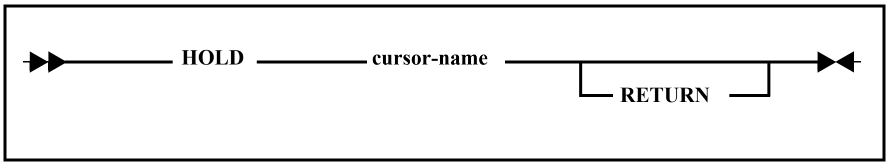
The HISTOGRAM statement allows the display of key (like descriptor, superdescriptor, or hyperdescriptor) values contained in an ADABAS table. The statement generates an ADABAS L9 command sequence.

All column names must be the same in this statement and the column name must be an ADABAS descriptor. The same column name must be specified in all locations designated by column-name A.

Notes:

1. For more information about the use of the FOR keyword in cursor processing, see "Cursor Processing" on page 8-28.
2. The the row delimiter specification (nnn) is not a valid option for the DECLARE CURSOR. The syntax error will be detected during FETCH.
3. If the COUNT(*) keyword is present, a count of each value present in the ADABAS table is returned.
4. After the FROM clause, you must include either WHERE or GROUP BY, or both.

HOLD



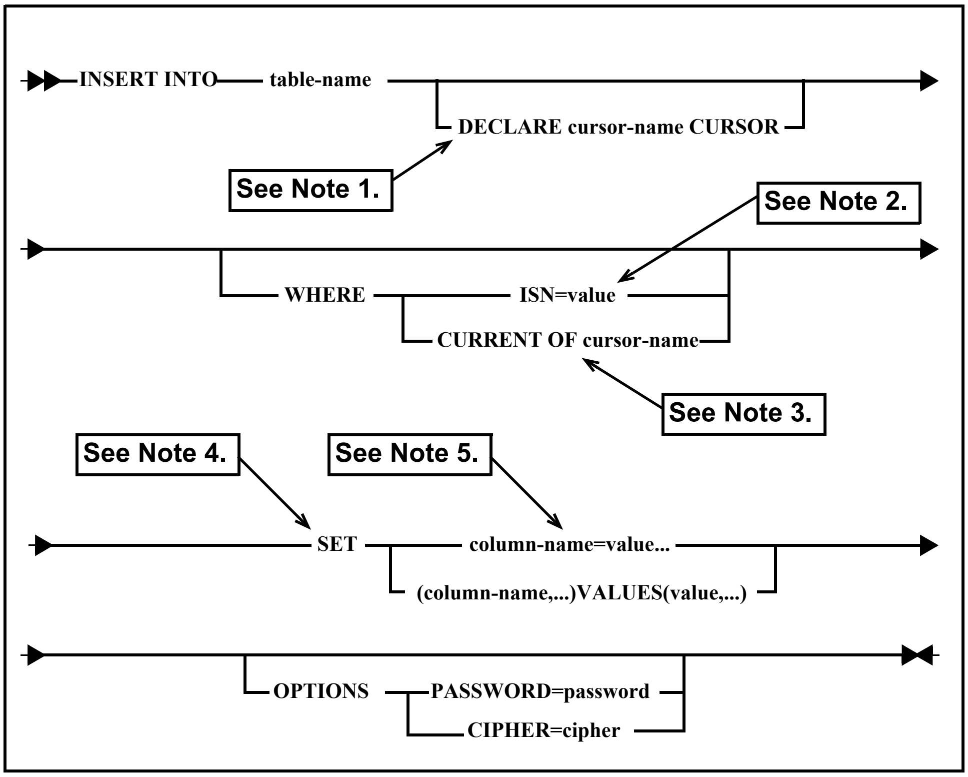
The HOLD statement allows for the locking of a single row during cursor processing. This eliminates the need to lock an entire set of records because only the row that the cursor presently represents is locked. An ADABAS HI command is issued for the current ISN in the previous FETCH statement.

Example:

The following pseudo logic describes the use of the HOLD statement:

```
DECLARE C001 CURSOR FOR SELECT * FROM EMPLOYEES
OPEN C001
dowhile (SQL_RC = 0)
   FETCH C001
  if  (row meets condition) then
     HOLD C001
     UPDATE Ö
     COMMIT
  ifend
doend
```

INSERT



The INSERT statement allows rows of data to be added to an ADABAS table. The ADABAS command issued is N1; however, if the WHERE criteria is coded with the ISN=value, an ADABAS N2 command is issued using the ISN number provided. The INSERT statement is generally followed by a COMMIT or ROLLBACK statement.

Notes:

1. For the INSERT statement, the FOR keyword is not allowed. For more information about cursor processing, see “Cursor Processing” on page 8-28.
2. The ISN clause allows users to assign an ISN value for the row to be inserted.
3. If the CURRENT OF cursor-name clause is specified in the WHERE criteria, the SET clause is optional. If the SET clause is specified in that case, it will override the CURRENT OF cursor-name clause.
4. The SET criteria describes the columns and data values that will be placed into the inserted row.
5. When inserting into an occurrence of an MU field defined as NU (null suppressed), there are special considerations for use. For more information, see “Special Considerations for Updating or Inserting into MU Fields Defined as NU” on page 8-26.

Example:

To add a row to the EMPLOYEES table with a last name of “Jones” and first name of “Sarah”:

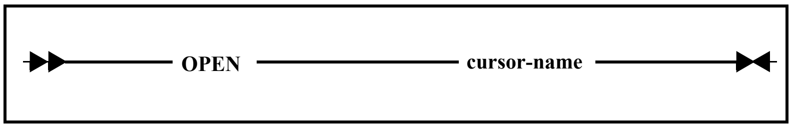
```
INSERT INTO EMPLOYEES SET LAST_NAME = “JONES” FIRST_NAME = “SARAH”
```

For redefined fields in INSERT statements, use the concatenation notation:

```
INSERT INTO EMPLOYEES SET AA = “x|y”
```

Where x and y represent each of the redefined fields for AA.

OPEN cursor



The OPEN statement is used to indicate the start of cursor processing for a prior given statement. No ADABAS command is issued for this statement. Shadow Mainframe Adapter Client for ADABAS creates the internal control structures to maintain the state of the cursor. See also “FETCH cursor” on page 8-4 and “CLOSE cursor” on page 8-2.

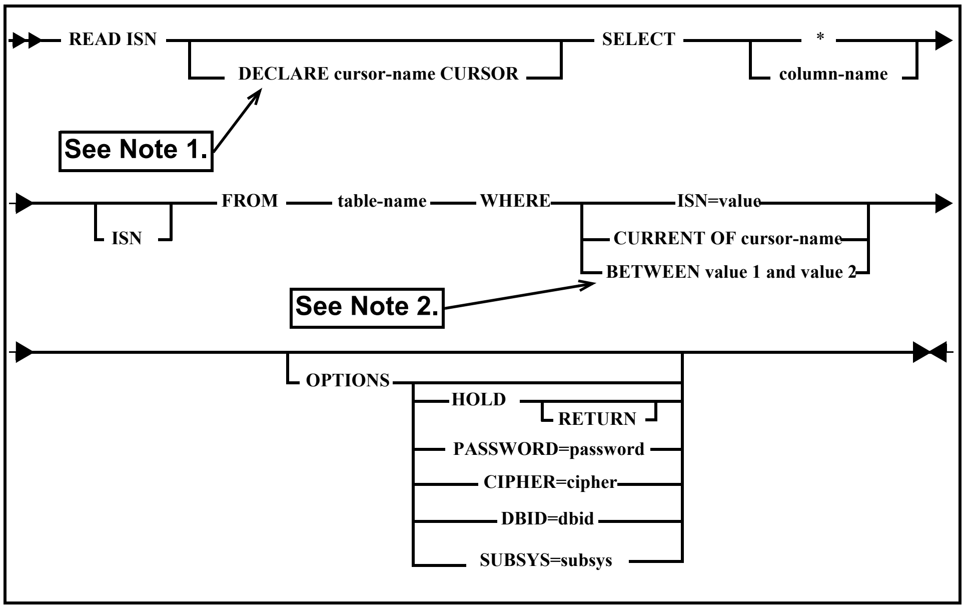
Example:

The following pseudo logic describes the use of the OPEN cursor statement:

```
DECLARE C001 CURSOR FOR SELECT * FROM EMPLOYEES
OPEN C001
dowhile (SQL_RC = 0)
    FETCH C001
loop
CLOSE C001
```

READ

READ ISN



The READ ISN statement allows the application to target a specific row within an ADABAS table using the ADABAS ISN as the search criteria. ADABAS L1 command(s) will be issued for this statement.

The READ ISN statement does not allow cursor processing (note there is no FOR keyword allowed); this means the row is returned as soon as you issue the READ ISN statement. OPEN cursor and FETCH cursor logic does not apply when you use READ ISN.

Notes:

6. The DECLARE CURSOR syntax provided in this statement is only for establishing cursor position for the returned row. If you use the BETWEEN operator, the cursor will be positioned on the last row returned. For more information about cursor processing, see “Cursor Processing” on page 8-28.

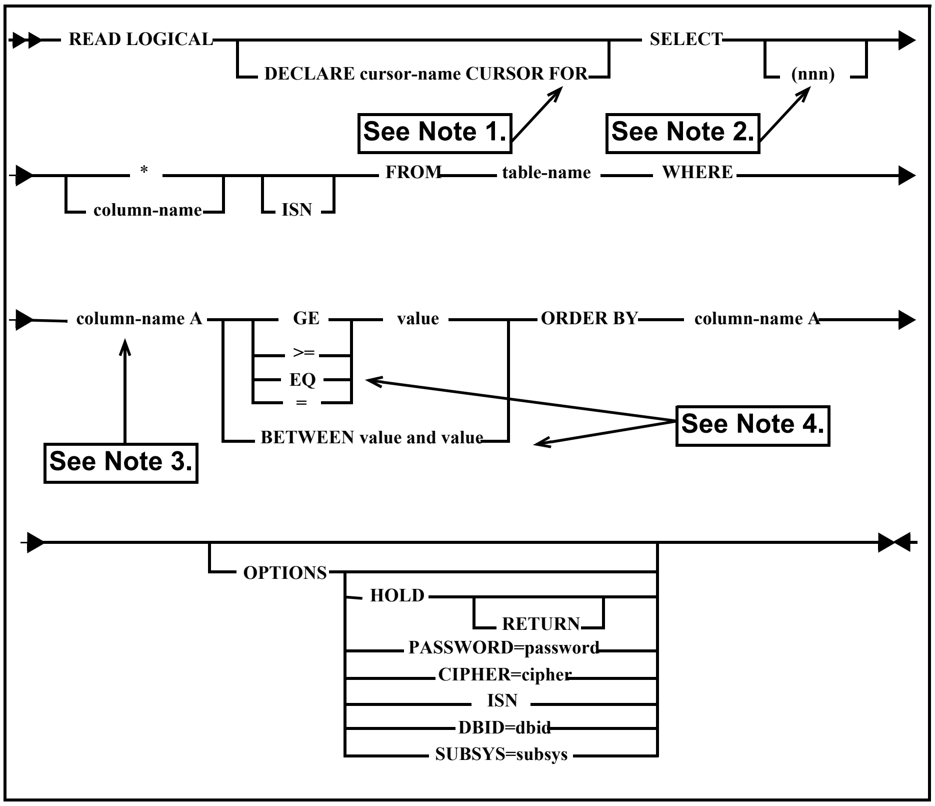
7. If the BETWEEN clause is used, each and every ISN in the range is requested. The ADABAS response 113 (missing ISN) response code is suppressed in this case.

Example:

To obtain all columns from ISN 100 in ADABAS table EMPLOYEES:

```
READ ISN SELECT * FROM EMPLOYEES WHERE ISN = 100
```

READ LOGICAL



The READ LOGICAL statement allows the application to read ADABAS data logically in the order of a given key (like descriptor or superdescriptor) value. We recommend that cursor processing be used, especially if the result set to be returned is large. This allows you to control the number of rows returned in a single request and the application to control when to terminate the READ LOGICAL sequence. The READ LOGICAL generates ADABAS L3 command sequences.

Notes:

1. For more information about the use of the FOR keyword in cursor processing, see “Cursor Processing” on page 8-28.

2. The row delimiter is not a valid option for the DECLARE CURSOR. The syntax error will be detected during FETCH.

3. The same column name must be specified in all locations designated by column-name A.

4. The EQ and BETWEEN clauses are available only in Shadow Connect v4.5 and above.

Examples:

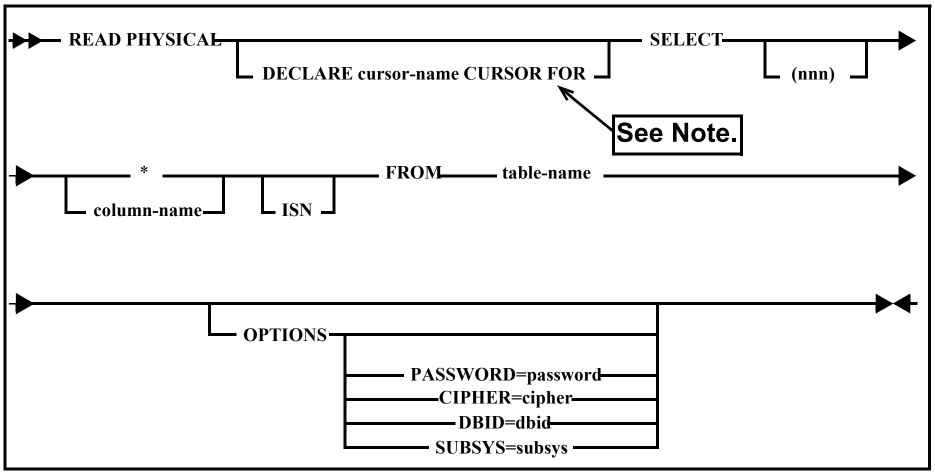
To read all columns from the EMPLOYEES table starting from the last name of "J":

```
READ LOGICAL SELECT * FROM EMPLOYEES WHERE LAST_NAME >= "J"
ORDER BY LAST_NAME
```

To read all columns from the EMPLOYEES table starting from the last name equal to "J":

```
READ LOGICAL SELECT * FROM EMPLOYEES WHERE LAST_NAME = "J"
ORDER BY LAST_NAME
```

READ PHYSICAL



The READ PHYSICAL statement allows the reading of ADABAS tables as they are physically stored in the ADABAS table. No key values are used and all data is read. An ADABAS L2 command sequence is generated.

Care must be taken for large ADABAS tables. If the table is large, the response time to the application will be long. We recommend that cursor processing be used whenever wait times need to be minimized.

Note:

For more information about the use of the FOR keyword in cursor processing, see "Cursor Processing" on page 8-28.

Examples:

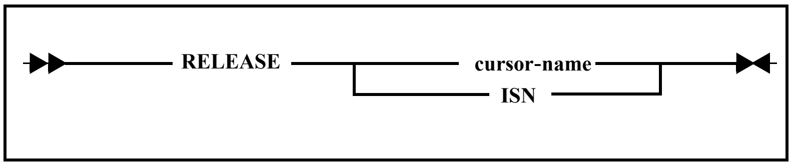
To read all rows, all columns in the EMPLOYEES table:

```
SELECT * FROM EMPLOYEES
```

To minimize the application wait time, use cursor processing:

```
READ PHYSICAL DECLARE C001 CURSOR FOR SELECT * FROM EMPLOYEES
OPEN C001
dowhile (SQL_rc = 0)
   FETCH (50) C001
doend
CLOSE C001
```

RELEASE

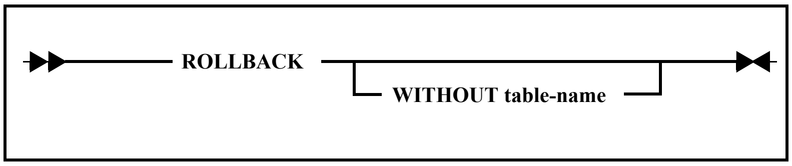


The RELEASE cursor allows for the release of an ADABAS resource. This generates an ADABAS RI command.

Example:

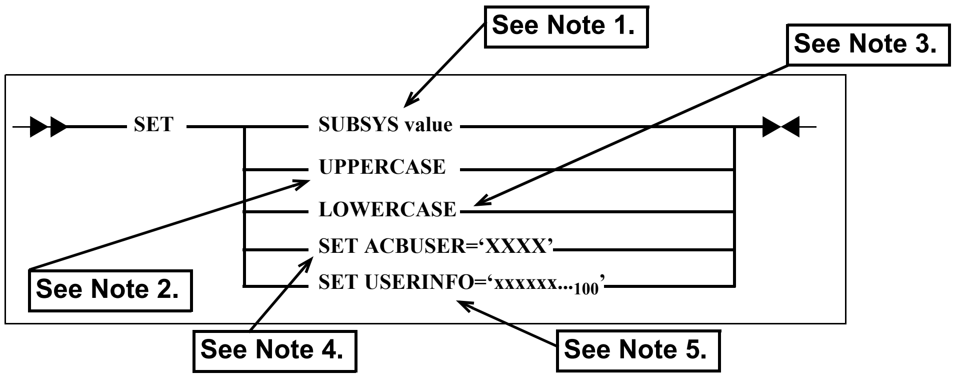
```
RELEASE C001
```

ROLLBACK



The ROLLBACK statement causes ADABAS to back out UPDATEs, INSERTs, and/or DELETEs executed after the previous COMMIT or ROLLBACK statement. An ADABAS BT command is issued. The opposite of ROLLBACK is COMMIT (see “COMMIT [WORK]” on page 8-2).

SET



The SET statement sets data to be processed. Uppercase and lowercase parameters of the SET statement allow the Mainframe Adapter Client to tell Shadow Mainframe Adapter Client for ADABAS how to translate the data portion of the statements sent to be processed.

Notes:

1. If the SUBSYS keyword is used, value will be an ADABAS subsystem name that corresponds to an ADABAS router (SVC) number, which was assigned during ADABAS installation. This allows the subsystem name defined via the Shadow Data Mapping Facility to be overridden during execution time.

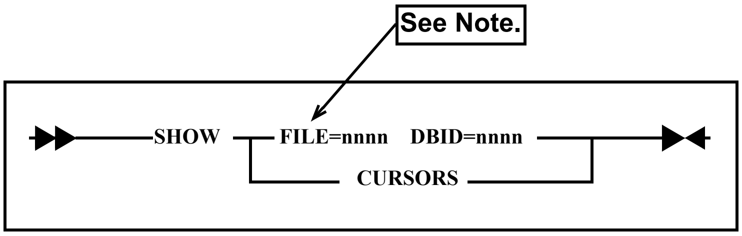
2. UPPERCASE indicates to accept input in either uppercase or lowercase, and then change all characters (including literal strings in the SQL statement) to uppercase values.

3. LOWERCASE indicates that the data should be left in the mode in which the Mainframe Adapter Client has keyed it. The default is LOWERCASE; however, the Shadow Mainframe Adapter Server parameter ADABASUPPERCASE allows an administrator to set the default as either SET UPPERCASE or SET LOWERCASE.

4. If SET ACBUSER is used, XXXX will be user-supplied, 4-character data that will be placed into the ADABAS control block user (ACBUSER) field.

5. If SET USERINFO is used, xxxxxx will be from 1 to 100 bytes of user-supplied character data. This parameter also checks to see if the ADABAS ADALINK routine has been assembled with at least 100 bytes of userinfo area set. The LNUINFO equate within ADALNK must be set to at least 100 bytes and ADALNK reassembled. This data is passed to the ADABAS user exits as described in the *ADABAS DBA Reference Manual.*

SHOW



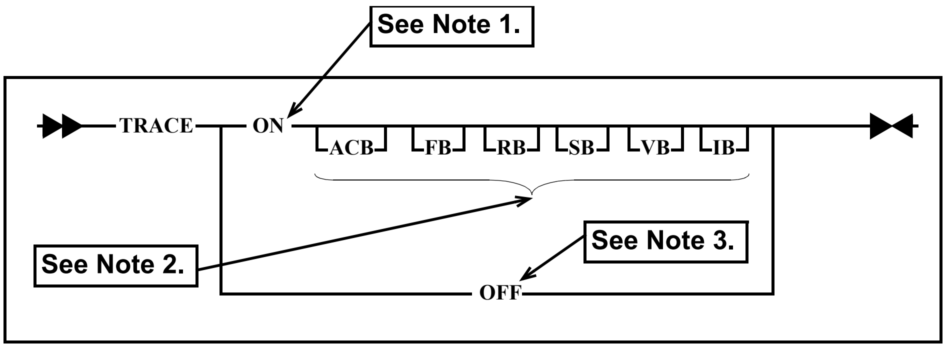
If the keywords FILE and DBID are used, this statement returns the ADABAS FDT for the given ADABAS FILE and DBID number using an ADABAS LF command.

If CURSORS is requested, the cursor control blocks are dumped to the Mainframe Adapter Client. This information is useful for debugging only in conjunction with NEON Systems, Inc. Customer Support.

Note:

Unless the ADABAS subsystem that you want to access is already assembled into your ADALNK routine ***or*** unless you have previously issued a database call, you must issue a SET SUBSYS statement (see "SET" on page 8-17). Otherwise, you will get a -213 response code when issuing SHOW FILE.

TRACE



The TRACE statement allows the display of the ADABAS control blocks generated and executed for a given statement. A before and after image of the ADABAS buffers are displayed to the application via a column named TRACE.

If you don't add any of the keywords, all of the ADABAS buffers will be traced. Adding any of the following keywords to the TRACE statement will trace particular buffers, as listed:

- **ACB:** ADABAS control block
- **FB:** Format buffer
- **RB:** Record buffer
- **SB:** Search buffer
- **VB:** Value buffer
- **IB:** ISN buffer

Notes:

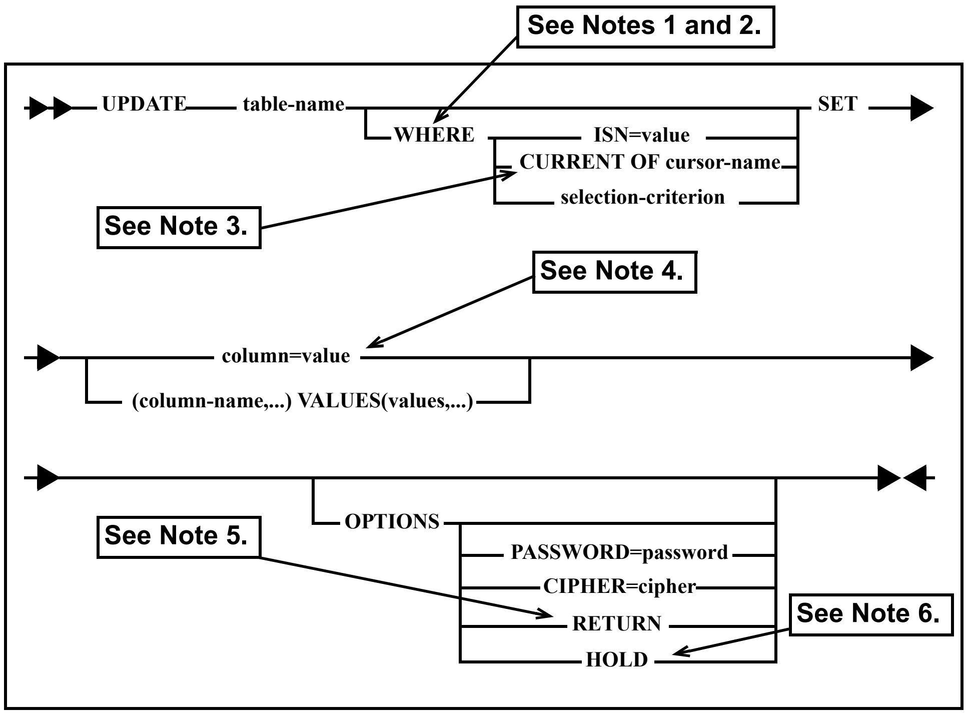
1. All information is accumulated for all executed statements after the TRACE ON is executed.
2. The keywords must be in uppercase.
3. The trace result set is returned for the TRACE OFF statement.

Example:

To trace the ADABAS buffers created for a READ ISN statement:

```
TRACE ON
READ ISN SELECT * FROM EMPLOYEES WHERE ISN = 100
TRACE OFF
```

UPDATE



Notes:

1. Without the WHERE clause, every row in the requested table is updated.
2. The WHERE criteria requires special consideration for use with MU and PE fields. For more information, see “Special Considerations for Searching for Particular Occurrences MU and PE Fields” on page 8-26.
3. When using the WHERE CURRENT OF cursor-name clause, unless you had already specified OPTIONS HOLD when you declared the cursor or you have already issued the HOLD cursor-name statement, you have to use OPTIONS HOLD in your UPDATE statement.
4. When updating an occurrence of an MU field defined as NU (null suppressed), there are special considerations for use. For more information, see “Special Considerations for Updating or Inserting into MU Fields Defined as NU” on page 8-26.
5. If a record is being updated at the same time it is being used by another user, and the UPDATE is issued without the OPTIONS RETURN clause, the UPDATE will wait until the record is released. If the OPTIONS RETURN clause is used, ADABAS will issue a response code of -145. If you partially update a record or a series of records, you should issue a ROLLBACK.
6. If you want to lock the record during the update after you issue a COMMIT or ROLLBACK, use the HOLD option located within the OPTIONS clause. The record(s) will be released after you issue a COMMIT or ROLLBACK.

Programming Tip:

The number of rows affected by the UPDATE statement can be returned by the `stmt.executeUpdate()` method; however, the maximum number that can be returned is 65,535.

Example:

For redefined fields in UPDATE statements, use the concatenation notation:

```
SET AA = “x|y”
```

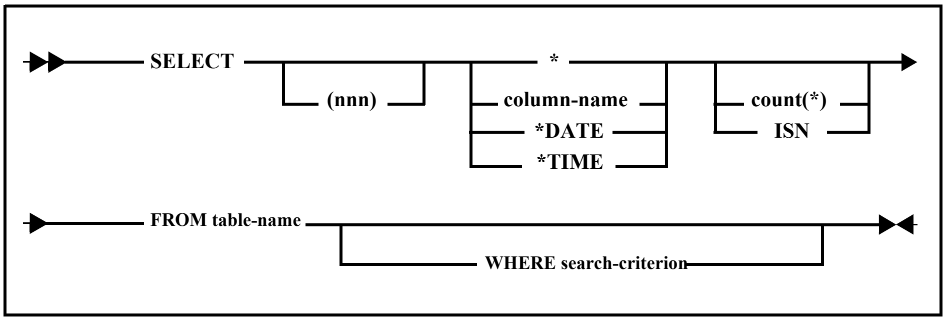
Where x and y represent each of the redefined fields for AA.

Statement Criterion

Within the SQL statements, the following criterion can be used:

- Selection criterion
- WHERE criterion
- Search criterion

Selection Criterion



Examples:

SELECT * FROM XYZ
Obtains all enabled columns from table XYZ.

SELECT COLUMN1 COLUMN2 FROM XYZ
Obtains only columns named COLUMN1 and COLUMN2 from table XYZ.

SELECT COUNT(*) FROM XYZ WHERE....
Returns the number of rows that meet the WHERE criteria. This statement is similar to the Natural statement **FIND NUMBER**. If column names are also specified, only columns from the first row that meet the criteria are returned.

SELECT ISN FROM XYZ WHERE...
Returns the ADABAS ISN number assigned to the row that meets the criteria.

SELECT(20) * FROM XYZ WHERE....
Limits the number of returned rows of data to 20. The number must be in parenthesis immediately following the SELECT keyword. Limits are not valid on statements using cursor processing.

Note:
The keywords COUNT(*) and ISN must be coded in the SELECT statement following any column names that are to be returned.

DATE and TIME

DATE and TIME allow you to obtain the current mainframe date and time and set the Natural date fields.

Obtaining Mainframe Current Date and/or Time

The current mainframe date and time can be obtained by using the *DATE and/or *TIME keywords in a SELECT statement. Because the current date and time is returned for each and every row returned from the query, you may want to limit the query as shown in the following example.

Example:

```
SELECT (1) *DATE *TIME FROM table_name
```

Setting Natural Date Fields

A Natural date field may be indicated to Shadow Mainframe Adapter Client for ADABAS by using the “D” date_format in the Data Mapping Facility input definitions for the SDADDM utility. If this format is used, the Mainframe Adapter Client can view or update the specified date column using the ODBC date format. Conversion from ODBC date to a natural date format is done automatically. The length of a “D” format type defined in the Data Mapping Facility must be 4.

Example:

The following statement would result in the given date being converted to a Natural date format and stored in the requested ADABAS file:

```
UPDATE table_name SET column_name = “1999/01/01”
```

The following statement sets the specified column name to the current date:

```
UPDATE table_name SET column_name = *DATE
```

MU and PE Criterion

Shadow Mainframe Adapter Client for ADABAS supports the following syntax options for PE and MU fields:

```
SELECT AI(*) FROM TABLE
SELECT AI(x:y) FROM TABLE
SELECT AI(x) FROM  TABLE
```

Shadow Mainframe Adapter Client for ADABAS also supports MU within PE with the syntax above. For example:

```
SELECT AI(*) FROM TABLE
SELECT AI(x:y, n:m) FROM TABLE
SELECT AI(x,*) FROM table
SELECT AI(x:y, *)
SELECT AI(*, x)
SELECT AI(*, x:y)
```

In addition to the support in SELECT and FIND SELECT statements, this same syntax will be supported in JOIN, READ ISN, READ PHYSICAL, READ LOGICAL, and cursor support.

The syntax provided will be similar to what Natural provides. To make this clear, the following examples of syntax have been provided, as taken from the Software AG Natural Programming Guide:

Examples

The following examples use the multiple-value field LANGUAGES and the periodic group CARS. The various values of the multiple-value field LANGUAGES can be referenced as follows:

`LANGUAGES(*)`	All values - the number of values depend on how map is generated.
`LANGUAGES(1)`	References the first value LANGUAGES (X). The value of the variable X determines the value to be referenced.
`LANGUAGES(1:3)`	References the first three values.
`LANGUAGES(6:10)`	References the sixth to tenth values.
`LANGUAGES(X:Y)`	The values of the variables X and Y determine the values to be referenced.

The various occurrences of the periodic group CARS can be referenced in the same manner:

`CAR(*)`	All values.
`CARS(1)`	References the first occurrence.
`CARS(X)`	The value of the variable X determines the occurrence to be referenced.
`CARS(1:2)`	References the first two occurrences.
`CARS(4:7)`	References the fourth to seventh occurrence.
`CARS(X:Y)`	The values of the variables X and Y determine the occurrences to be referenced.

The following examples use the multiple-value field SERVICING within the periodic group CARS. The various values of the multiple-value field can be referenced as follows:

`SERVICING(*)`	All values of SERVICING in all occurrences of CARS -- the number of values depend on how map is generated.
`SERVICING(1,1)`	References the first value of SERVICING in the first occurrence of CARS.
`SERVICING(1:5,1)`	References the first value of SERVICING in the first five occurrences of CARS.
`SERVICING(1:5,1:10)`	References the first ten values of SERVICING in the first five occurrences of CARS.

In order to use this support, you must regenerate the map using the PE_MU_COUNT option when running SDADEX. This is because this option will now generate a count field for each PE, MU, and MU within PE, and Shadow Mainframe Adapter Client for ADABAS will use the MU, PE, and MU within PE count field for this support.

Shadow Mainframe Adapter Client for ADABAS will base its assumption on the number of occurrences requested from the number of occurrences defined in the field definition (OPDE) by default when * is used. However, a user's requirement for applying the "filter" to return only the columns corresponding with the filled occurrences will be accepted. The following is a description of what has been implemented to support this requirement:

1. A new Shadow Mainframe Adapter Server parameter, ADABASPRUNEMUPE, has been introduced in the PRODADABAS group. The default for this parameter is NO, which specifies no filtering will occur. If this is set to NOTCOUNT, the filtering will be done to return only the columns corresponding with the filled occurrences. The _C (or C if you use the short name as column name in the map) count columns will also be returned as part of the resultset. If ADABASPRUNEMUPE is set to ALL, it functions the same way as ADABASPRUNEMUPE = NOTCOUNT ***except*** that the _C (or C if you use the short name as column name in the map) count columns will ***not*** be returned as part of the result set.

   If the resultset contains multiple rows, the number of columns returned will be the same as the maximum value in the count (i.e., the _C or C count column).

For example, assume the following result set:

AA	AIC	AI001	AI002	AI003
"key01"	1	"value11"	BLANK	BLANK
"key02"	2	"value21"	"value22"	BLANK
"key03"	0	BLANK	BLANK	
"key04"	1	"value41"	BLANK	BLANK

After the filtering with ADABASPRUNEMUPE=NOTCOUNT, the result set will be as follows:

AA	AIC	AI001	AI002
"key01"	1	"value11"	BLANK
"key02"	2	"value21"	"value22"
"key03"	0	BLANK	BLANK
"key04"	1	"value41"	BLANK

After the filtering with ADABASPRUNEMUPE=ALL, the resultset will be as follows:

AA	AI001	AI002
"key01"	"value11"	BLANK
"key02"	"value21"	"value22"
"key03"	BLANK	BLANK
"key04"	"value41"	BLANK

2. The filtering only takes effect for the case when * is used on an MU or PE field. For example, if AI is an MU, PE, or MU within PE, and you issue a SELECT AI(*), filtering will only be done for this case. If you select the columns individually (i.e., SELECT AI001, AI002...) or use the range (i.e., SELECT AI(1:3)), no filtering will be done.

3. Filtering will require the count field (i.e., the _C or C field). So, when you generate the map, you need to specify PE_MU_COUNT in SDADEX. Otherwise, Shadow Mainframe Adapter Client for ADABAS cannot and will not apply the filtering. If the map is generated without using

PE_MU_COUNT, if you use this syntax, Shadow Mainframe Adapter Client for ADABAS will reject the query.

4. If a range search is specified, the count column will not be returned because the user explicitly specifies a range. The following is an example of a range search:

   ```
   SELECT AA, AI(1:3) FROM TABLE WHERE PK >= 'key01'
   ```

   If the count field is needed, the user can include the count field in column selection criteria as follows:

   ```
   SELECT AA, AIC, AI(1:3) FROM TABLE WHERE PK >= 'key01'
   ```

5. A new trace message ("x COLUMNS PRUNED FROM RESULT SET") will be generated in the trace browse to indicate a filtering has taken place for the query issued by the user.

WHERE Criterion

WHERE Criterion for ADABAS Descriptor Searches

If you want WHERE criteria to allow only ADABAS descriptor searches, use DE_SEARCH_ONLY during data map creation.

WHERE Criterion for MU and PE Fields

When using the WHERE criterion with MU and PE fields, special considerations exist regarding the following:

- Searching for particular occurrences MU and PE fields
- Updating or inserting into MU fields defined as NU

Special Considerations for Searching for Particular Occurrences MU and PE Fields

Due to the way ADABAS allows searches in MU, when you specify a value for an MU occurrence in the WHERE clause, Shadow Mainframe Adapter Client for ADABAS will target all rows where any occurrence of that MU field match the value specified.

This works the same way for PE fields unless you are running at Shadow Connect v4.5 SVFX3268 or v4.8 SVFX0178 and above. At those levels of code, you can use the new parameter SEARCH_BY_PE_INDEX to allow the Mainframe Adapter Client to target rows that match a particular occurrence of the PE field.

Special Considerations for Updating or Inserting into MU Fields Defined as NU

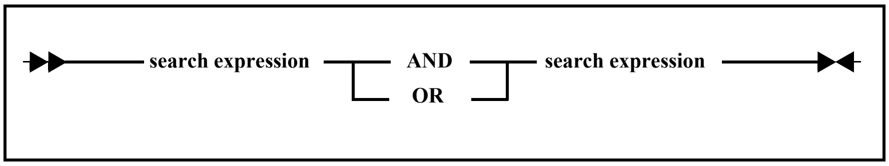
When updating or inserting into a particular occurrence of an MU field defined as NU (null suppressed), it is possible that another occurrence of the MU field may receive the update or insert. This is working as designed. In ADABAS, NU means that it is null suppressed. Therefore, for MU fields defined as NU (null suppressed), ADABAS suppresses 0 (zero) for numerics and blank for alpha. For example, if you update the second occurrence of an MU field defined as NU and the first occurrence does not exist, ADABAS suppresses the first occurrence and shifts the second occurrence downward to the first.

If you want positionality within an MU field and/or 0 (zero) or blank are valid values, then the MU must field must be defined as FI (fixed).

To check whether the MU field is defined as NU or FI, you can do the following:

- Issue a “SHOW FILE=nnnnn DBID=nnnn” (see “SHOW” on page 8-18).
- Review the ADAREP report.

Search Criterion

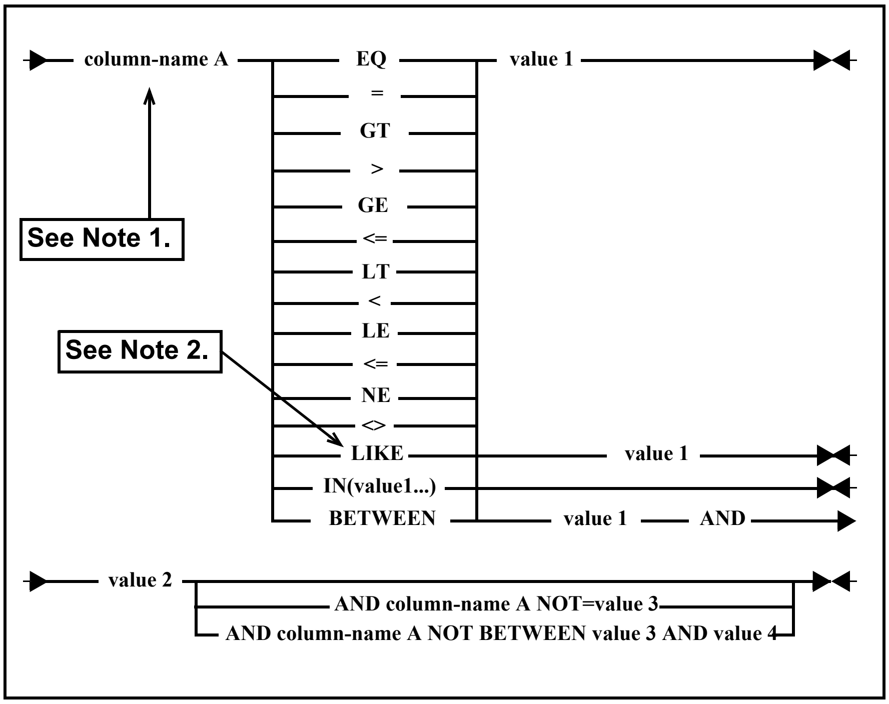


Note:

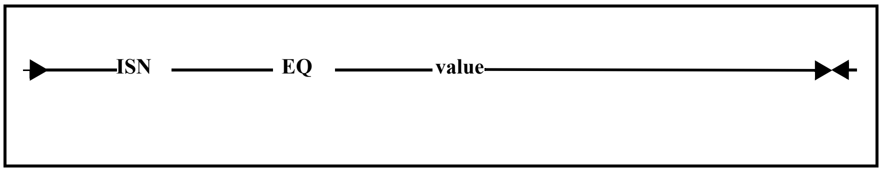
Parenthetical groupings are not supported; if used, they will be ignored.

Search Expression

The search expression is formatted as follows:



OR



Notes:

1. In most cases, column names in search criterion need to be ADABAS descriptor fields. The same column name must be specified in all locations designated by column-name A. It can also be ISN=value.

2. The LIKE operator will function like the EQ operator and ***does not*** allow wildcards.

3. If using table join support, only the simple operators (=, >, >=, <, <=, <>) are allowed.

Cursor Processing

If the DECLARE cursor-name CURSOR [FOR] clause is included in the SQL statement, the Mainframe Adapter Client application will be able to control the positioning of the cursor. Shadow Mainframe Adapter Client for ADABAS supports cursor processing as follows:

- The Mainframe Adapter Client application can open as many cursors as required at any given time.
- If the Mainframe Adapter Client application tries to declare a cursor name that is already in use, the new DECLARE CURSOR operation using the same cursor name will result in the cursor being returned to its original state and this same cursor name will take affect for the new DECLARE CURSOR operation.
- Cursors names must be 4 characters in length.
- For cursor names, it is recommended that alphanumeric characters be used and special character usage be avoided.
- The cursor name "NEON" is used internally by Shadow products and should not be used in Mainframe Adapter Client applications.
- Shadow Mainframe Adapter Client for ADABAS allows Mainframe Adapter Clients to specify the cursor processing in the following ways:
  - Using the FOR keyword (DECLARE cursor-name CURSOR FOR)
  - Omitting the FOR keyword (DECLARE cursor-name CURSOR)

Using the FOR Keyword

If the DECLARE cursor-name CURSOR FOR clause is included in the SQL statement, the Mainframe Adapter Client application will be able to control the positioning of the cursor by using OPEN cursor-name, FETCH cursor-name, and CLOSE cursor-name cursor processing.

Note:
Shadow Mainframe Adapter Client for ADABAS maintains the required context between Mainframe Adapter Client requests to allow single row return on each FETCH request.

Example:

```
DECLARE C001 CURSOR FOR SELECT * FROM EMPLOYEES
...
OPEN C001
...
FETCH_LOOP:
FETCH C001
CHECK RETURN > 0 EXIT LOOP
END_LOOP;
...
CLOSE C001
```

Omitting the FOR Keyword

Shadow Mainframe Adapter Client for ADABAS offers a quick way to keep the cursor position on the last row of the result set returned without going through OPEN cursor-name, FETCH cursor-name, and CLOSE cursor-name cursor processing. If the DECLARE cursor-name CURSOR clause (without the FOR keyword) is included in the SQL statement, the cursor position will be maintained on the last row returned.

This is designed for situations that return a single row where the cursor position is required for subsequent operations but the Mainframe Adapter Client does not want to use OPEN cursor-name and FETCH cursor-name logic. This saves a couple of network round-trips.

Example:

The following examples issue a SELECT from MYTAB where AA = “ABC123”, assuming one row will match the criteria and be returned. The Mainframe Adapter Client application will have the cursor position on that row for the subsequent UPDATE operation.

```
FIND DECLARE C001 CURSOR SELECT AA FROM MYTAB WHERE AA = "ABC123"
OPTIONS HOLD
UPDATE WHERE CURRENT OF C001 SET AB = "SOME VALUE"
CLOSE C001
```

Or

```
FIND DECLARE C001 CURSOR SELECT AA FROM MYTAB WHERE AA = "ABC123"
HOLD C001
UPDATE WHERE CURRENT OF C001 SET AB = "SOME VALUE"
CLOSE C001
```

Notes:

- You either have to place the OPTIONS HOLD clause in the DECLARE CURSOR statement (as shown in the first example) or issue the HOLD statement (as shown in the second example) to hold the record prior to updating the row where the cursor pointing. Otherwise, ADABAS will issue a -144 response code.
- If the record is already held by someone else and you do not want to wait for the record to be released, please use the RETURN option in the OPTIONS clause.

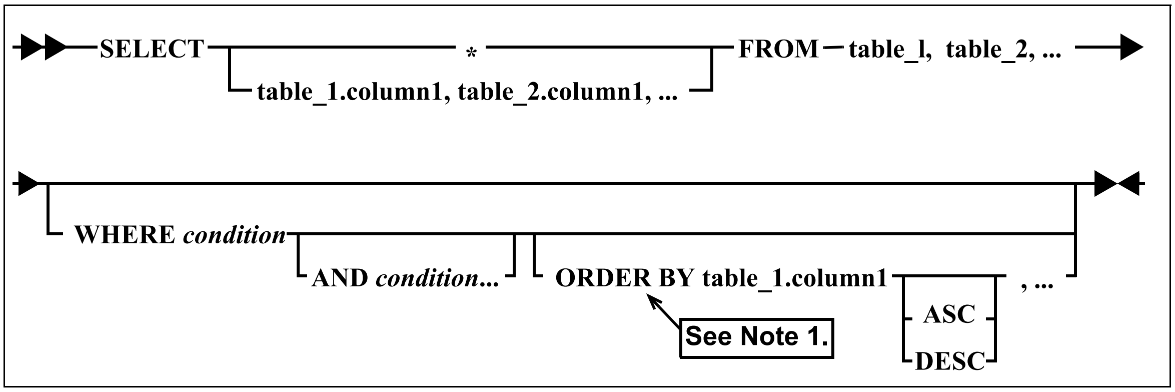
Please refer to the notes for “UPDATE” on page 8-19 for more information.

Using Table Joins to Obtain Data from Multiple ADABAS Files

Shadow Mainframe Adapter Client for ADABAS supports the selection of data from up to five ADABAS tables by allowing a logical joining of the tables through use of common data elements. Shadow Mainframe Adapter Client for ADABAS transforms the query into multiple nested SELECTs.

Statement Syntax

The SELECT statement syntax is as follows:



Where:

condition can be specified in either of the following formats:

table_1.column2 *operator* value
table_l.column2 *operator* table_2.column2 } ← See Note 2.
table.ISN=value ← See Note 3.

Notes:

1. Up to ten columns can be specified in the ORDER BY clause. When the ORDER BY clause is specified when doing a join, the Shadow Mainframe Adapter Server will internally sort the result set (based on the ORDER BY clause) before sending the result set back to the Mainframe Adapter Client; thus, to avoid overhead and maximize performance, it is strongly recommended that the ORDER BY be specified only when necessary and the size of the result set to be sorted should be minimized.

> ***Note:***
> The name of the column in the ORDER BY clause should be either a column name in the result set or a column position. This means that if the user uses a correlation ID for the column, the correlation ID or column position must also be used in the ORDER BY clause.

If the ORDER BY clause directional option is not explicitly issued to specify an ascending or descending order, the default value will be an ascending order; however a directional option can be explicitly specified for each column. The valid syntax for issuing the ORDER BY clause directional option is as follows:

- **For ascending order:** Either ASC or ASCENDING.
- **For descending order:** Either DESC or DESCENDING.

> ***Note:***
> ORDER BY support with table joins is only available with Shadow Mainframe Adapter Server v4.8 SVFX2554 and above.

2. Only the simple operators (=, <, >, <=, >=, <>) are allowed for table join support.

3. When using a table join, ISN can be the ***only*** criteria in the WHERE clause. The following examples show a valid use of ISN in a join statement and an invalid use of ISN in a join statement (a statement with more than one criteria in the WHERE clause).

   Valid Join Statement

   ```
   SELECT EMPLOYEES.PERSONNEL_ID, VEHICLES.PERSONNEL_ID,
   VEHICLES.COLOR FROM VEHICLES, EMPLOYEES WHERE
   EMPLOYEES.PERSONNEL_ID = VEHICLES.PERSONNEL_ID AND
   EMPLOYEES.ISN = 177
   ```

   Invalid Join Statement

   ```
   SELECT EMPLOYEES.PERSONNEL_ID, VEHICLES.PERSONNEL_ID,
   VEHICLES.COLOR FROM VEHICLES, EMPLOYEES WHERE
   EMPLOYEES.PERSONNEL_ID = VEHICLES.PERSONNEL_ID AND
   EMPLOYEES.ISN = 177 AND EMPLOYEES.PERSONNEL_ID = "50005800"
   ```

Note:
For best performance, it is recommended that ISN be used on the "common" table. For example:

```
SELECT EMPLOYEES.PERSONNEL_ID,
VEHICLES.PERSONNEL_ID, VEHICLES.COLOR FROM
VEHICLES, EMPLOYEES WHERE EMPLOYEES.PERSONNEL_ID =
VEHICLES.PERSONNEL_ID AND EMPLOYEES.ISN = 177
```

Example:

The following SELECT statement:

```
SELECT * FROM EMPLOYEES, VEHICLES WHERE VEHICLE.PERSONNEL_ID =
EMPLOYEE.PERSONEL_ID AND EMPLOYEE.LAST_NAME = "JONES"
```

Results in the following statement execution:

```
   DECLARE C001 CURSOR FOR
       SELECT * FROM EMPLOYEES WHERE LAST_NAME = "JONES"
   OPEN C001
┌─ FETCH C001
│      READ ISN SELECT PERSONNEL_ID FROM EMPLOYEES WHERE CURRENT
│         OF C001
│      DECLARE C002 CURSOR FOR
│             SELECT * FROM VEHICLES WHERE VEHICLE_ID = "???????????"
│      OPEN C002
│      FETCH C002
│      LOOP
└─ LOOP
```

For each fetched row from cursor C001, the data returned for the column EMPLOYEES.PERSONNEL_ID is substituted in the C002 select statement. The fetch for C002 is done until all records are exhausted (ADABAS RSP 3) and the outer fetch C001 is then executed to retrieve the next row pointed to by the C001 cursor. The process continues until cursor C001 is exhausted (ADABAS RSP 3).

This logic is similar to the manner in which Natural would handle nested FIND statements.

Note:
Each row returned to the Mainframe Adapter Client would contain all columns from the EMPLOYEES table as well as all columns from the VEHICLES file.

Considerations for Using Table Joins

The following guidelines apply to using table joins for obtaining data from multiple ADABAS files:

- Only the SELECT and FIND SELECT verbs support multiple table joins.
- No updating of tables is allowed.
- The SELECT statement to obtain data from multiple ADABAS files requires that each column used in the SELECT be qualified by the ADABAS table name.
- The order for processing the individual SELECTs is based on table interdependencies within the original query.
- Conditions that specify values always take precedence.

Concatenation Notation

The following examples of concatenation notation are shown:

- Concatenating multiple fields with a superdescriptor
- Concatenating values in a redefined field

Concatenating Multiple Fields with a Superdescriptor

When using a superdescriptor that is comprised of multiple fields with dissimilar formats, use concatenation notation to create the superdescriptor value.

Note:
There must be the same number of defined type 6 format definitions in the mapping facility as there are field elements that comprise the superdescriptor.

Example:

Superdescriptor S1 is composed of the parent fields AA, AB and AC, as follows:

- AA is defined as A3.
- AB is defined as P5.
- AC is defined as A2.

To create the superdescriptor in the proper format to send to ADABAS, enter your SQL search criteria as follows:

```
WHERE S1 = “ABC|12345|DE”
```

Concatenating Values in a Redefined Field

When using redefined columns, the following considerations apply:

- For SELECT statements, the redefined column can be used as a column selection.
- For INSERT and UPDATE statements and within the WHERE clause of SELECT statements, the following applies:
  - If the redefined concatenation bar is not used, then the data value presented is allowed "as is."
  - If the redefined concatenation bar is used, then the data values are concatenated.
- If the redefinition is comprised of multiple parent fields with dissimilar formats, use concatenation notation to create the redefined value.

Example:

Field AA is composed of two redefined fields, AA_PART_1 and AA_PART_2, as follows:

```
REDEFINE_BEGIN
   REDEFINE_FILE = 173 REDEFINE_FIELD = AA
       (REDEFINE_COLUMN = AA_PART_1       REDEFINE_FORMAT = A
        REDEFINE_LENGTH = 5               REDEFINE_OFFSET = 0)
       (REDEFINE_COLUMN = AA_PART_2       REDEFINE_FORMAT = A
        REDEFINE_LENGTH = 5               REDEFINE_OFFSET = 5)
REDEFINE_END
```

For concatenation purposes, enter your SQL search criteria as follows:

```
SELECT AA_PART_1, AA_PART_2 FROM EMPLOYEE
SELECT * FROM EMPLOYEE WHERE AA = "ABCDE|12345"
```

Where `ABCDE` and `12345` each represent a value of the redefined field.

Note:
If there had been three redefined fields, you would have used three values (for example, `"ABCDE|12345|FGHIJ"`).

HEX Notation

Anywhere there is a value required and the value is a non-displaying item, use the following:

```
X'xx'
```

Where `xx` is a hexadecimal that must be given in pairs representing the high and low order nibbles of the byte(s).

Examples:

`X'C1'` is the character “A”.
`X'C1F0'` are the characters “A0”.

Part IV

Appendices

APPENDIX A: *Shadow Mainframe Adapter Client Keywords*

Shadow Mainframe Adapter Client, part of the Mainframe Adapter Client component of the Shadow Connect product, is controlled using certain keywords. These keywords are automatically given default values which can be changed depending on the Shadow function you are using.

This chapter covers the following topics:

- Setting a Keyword
- Keyword Descriptions
- Setting Keywords to Benefit Performance

Setting a Keyword

For details on how to set Shadow Mainframe Adapter Client keywords, see “Step 1: Configure the Data Source” on page 4-1 within Chapter 4, “Shadow Mainframe Adapter Client: Configuration,” of this guide.

Note:
In addition, certain application Mainframe Adapter Servers allow Shadow Mainframe Adapter Client keywords to be set via the application Mainframe Adapter Server graphical user interface.

Keyword Descriptions

Table A–1 on page A-2 lists the Shadow Mainframe Adapter Client keywords that can be set, along with a brief explanation of each and default settings.

Table A–1. Shadow Mainframe Adapter Client Keywords

Keyword	Description	Explanation	Default

ABCN	ADABAS Correlation Name Support	This keyword specifies whether the Shadow Mainframe Adapter Client for ADABAS will support column name correlation IDs for this particular connection.
		Possible values are as follows:
		• YES: Column name correlation IDs will be supported.
		• NO: Column name correlation IDs will not be supported.
		Support for column name correlation IDS can also be enabled for all connections to a particular Shadow Mainframe Adapter Server by setting the Shadow Mainframe Adapter Server parameter, ADABASCORRELATIONIDS. The Shadow Mainframe Adapter Client keyword adds flexibility, allowing users to enable this support on an application by application basis.
		Note: Setting this keyword to YES may cause a conflict with earlier versions of the Shadow Mainframe Adapter Client for ADABAS, which accepted SQL syntax statements of the following form (note that there are no commas to separate the operands):
		`SELECT AA AB FROM EMPQA1`
		With this keyword set to NO (or on earlier versions of the Shadow Mainframe Adapter Client for ADABAS), this will select two columns, AA and AB. With this keyword set to YES, AB will be considered a correlation name for AA.
		To select two columns when this keyword is set to YES, commas must be used to separate the two column names as follows:
		`SELECT AA, AB FROM EMPQA1`
			NO

ACMT	Number of Active Statements	This keyword is only applicable to the SQLGetInfo function called with an InfoType of SQL_ACTIVE_STATEMENTS, in which case the value will be as follows if this keyword is not explicitly set: • 0: If either the AF (MS Access Compatibility) keyword (see “AF” on page A-2) is set or the ZEAS (Zero Active Statements) keyword (see “ZEAS” on page A-32) is set. A value of 0 (zero) indicates an unknown or unlimited value. • 50: If the normal cursor pool is being used. • 400: If the extended cursor pool is being used. See “EXCU” on page A-12. However, this keyword can be set to a specific value.	0
AF	MS Access Compatibility	This keyword offers a solution to Microsoft Access compatibility issues. If this keyword is set to YES, all host decimal data values are returned to the Mainframe Adapter Client application without any trailing zeroes to the right of the decimal point. This keyword must be set to YES in order for Microsoft Access to work with DB2 decimal data. This keyword may be needed with any other product that uses the Microsoft Jet database engine, such as Visual Basic.	NO
ALCD	Always Convert Dynamic SQL	This keyword should ***not*** be used!	YES

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
ALPO	Always Get Password from ODBC.INI/ SHADOW.INI	This keyword controls whether the password will always be obtained from the `SHADOW.INI` file. If set to YES, the password will always be obtained from the `SHADOW.INI` file, and the password passed in the connection string will then be ignored. ***Note:*** See also "NOPW" on page A-19.	NO
ALUO	Always Get Userid from ODBC.INI/ SHADOW.INI	This keyword controls whether the userid will always be obtained from the `SHADOW.INI` file. If set to YES, the userid will always be obtained from the `SHADOW.INI` file, and the userid passed in the connection string will then be ignored. ***Note:*** See also "NOUS" on page A-19.	NO
ALUS	Alternate (Secondary) Userid	This keyword is used to set the host secondary userid for Mainframe Adapter Client applications. This keyword must be eight or fewer characters long. If this keyword is set to a non-blank, non-null value, a SET CURRENT SQLID statement is issued after DSNALI OPEN processing is completed.	
APNA	Application Name	This keyword is the application name. The application name is sent to the host as part of the logon information. The application name is normally used to group SQL statements within a plan. If the application name is not set, all of the SQL associated with a plan will be considered to be part of one large group. If the SQL used with one plan must be divided into subgroups (for conversion to static SQL), the application name must be set.	
APPL	Application	This keyword must be set to ODBC (which is the default).	ODBC
APRO	Application Read-Only	**(Read-only)** This keyword is an internal API parameter set via the following: • Below ODBC 3.0: SQLSetConnectOption function with an Attribute value of SQL_ACCESS_MODE. • ODBC 3.0: SQLSetConnectionAttr function with an Attribute value of SQL_ATTR_ACCESS_MODE.	
ASTM	Async Execution Timeout Value	This keyword controls the default wait time (in seconds) for an ODBC function that is executing asynchroneously. This keyword must be an integer.	2

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
AT	Accessible Tables	This keyword controls whether to restrict the list of tables for which the SQLTables function returns information. Many applications require information about the tables in the connected data source; however, in the case of DB2, this list of tables is unmanageably large unless you provide restrictions on the size. This keyword can be set as follows: • YES: When the SQLTables function is called, only the following tables will be returned: – Tables matching the table filter (see "DP" on page A-11). – Tables for which the current authorization ID has SELECT authority. This includes those in PUBLIC and PUBLIC AT ALL LOCATIONS but not those where SELECT authority is granted by virtue of a secondary authorization ID. • NO: All tables matching the table filter (see "DP" on page A-11) will be returned.	NO
AUST	Change Dynamic SQL to Auto-Static	This keyword is used to control whether or not Shadow Connect should try to convert dynamic SQL to auto-static SQL. Auto-static SQL converts literals in SQL statements to parameter markers for better query matching in Shadow Connect. In addition, auto-static SQL offers dynamic SQL caching, which means that for SELECT queries, the cursor is kept open across commits to keep the prepared SELECT statement in order to run it statically. For INSERT, UPDATE, and DELETE queries, the statements are lost at commit; thus, in order to take advantage of auto-static SQL for these types of queries, auto-commit must be disabled within the Mainframe Adapter Client application code and commits must be less frequent to exploit the use of the static SQL statements. The following behavior will result from setting this keyword: • YES: All dynamic SQL statements will be looked up in a lookaside buffer and converted to static SQL, if possible. • NO: Normal dynamic SQL processing will be performed. ***Note:*** Dynamic SQL can only be converted to auto-static SQL if this keyword is set to YES and if the Shadow Mainframe Adapter Server supports auto-static SQL, as determined by the Shadow Mainframe Adapter Server parameter AUTOSTATICSQL.	YES
AUSZ	Lookaside Buffer Size	**(Read-only)** This keyword displays the value of the Shadow Mainframe Adapter Server LOOKASIDESIZE parameter on the host.	
BIPA	Binary Passthrough	This keyword controls whether or not host binary data should be returned to Mainframe Adapter Client applications without being converted to hexadecimal. This keyword can be set as follows: • YES: Host binary data is passed through to Mainframe Adapter Client applications unchanged. • NO: Host binary data is converted to hexadecimal.	NO
BYDB	Bypass Double Quote	This keyword should be set to YES if double quotes should be bypassed. This keyword is provided to fix certain application bugs.	NO

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
CALK	CALL Lock Value	This keyword controls the type of lock associated with CALL statements on the host, as follows: • EXCLUSIVE • SHARE • UPDATE • NONE: The host code will assume that CALL statements do not obtain any host database locks.	EXCLUSIVE
CD	Convert Dynamic SQL	This keyword should ***not*** be used!	NO
CGFX	Cognos Impromptu Compatibility	This keyword can be set to YES to resolve certain problems with Cognos Impromptu. This keyword should not be set for any other reason.	YES
CLCT	Mainframe Adapter Client Certificate File	This keyword specifies the SSL Mainframe Adapter Client certificate file name.	
CLOR	Column Order Option	This keyword specifies how column names should be ordered when they are returned by catalog functions. Some functions explicitly specify an order; for functions that do not specify an order, this keyword is used by the catalog functions (SQLColumns). Possible values are as follows: • NAME • NUMBER	NAME
CMNP	Confirm New Password	If this keyword is set to YES, the user will be asked to confirm the new password string if the new password is set in the PWD keyword.	NO
CNDG	Connection Dialog Type	This keyword specifies what type of connection dialog should be displayed, as follows: • DYNAMIC: Either the SIMPLE or the DETAIL dialog is displayed, depending on how much information is needed. • SIMPLE • DETAIL • NEVER This keyword can be set using the data source settings or using a connection string.	DYNAMIC

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
CNMD	Connection Mode	This keyword controls the connection mode used by applications. The connection mode determines how long each physical connection (session or conversation) lasts and if SQL operations are blocked together. The possible connection modes are as follows: • PERMANENT: Use a permanent connection and send each SQL operation standalone to the Mainframe Adapter Server. • BLOCK: The session is permanent; however, SQL operations are blocked and sent together. • TRANSACTION: Each SQL operation is sent standalone but the session is terminated at the end of each Logical Unit Of Work (LUOW). • TRANSBLOCK: The session is terminated at the end of each LUOW, and SQL operations are blocked and sent together. • MESSAGE: SQL operations are blocked and sent together using messages. No session is ever maintained.	PERMANENT
CNNA	Connection Name	This keyword is used to specify the connection name. The use of the connection name is application specific. The name can be up to eight (8) bytes long; however, the application may or may not use all of the bytes. The connection name is padded on the right with blanks.	
CNPL		This keyword should ***not*** be used!	OFF
CNTM	Connection Timeout Value	This keyword controls how many seconds the Mainframe Adapter Client will wait for a connection to the host Mainframe Adapter Server to complete. If this keyword is set to 0 (zero), then the system default value will be used. Otherwise, the specified value will be used. This value should never be negative but can be 0 (zero). This value has no effect in UDP messaging. ***Note:*** The use of this keyword may cause some unpredictable results in many environments. This option is only available in the TCP/IP and the MQ link types (see “LINK” on page A-16), but it is applicable to both the permanent and the messaging modes (see “CNMD” on page A-6).	0

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
COFX	COUNT() Fix Type	This keyword controls how COUNT(column name) SQL functions are modified. Microsoft Access uses the COUNT(column name) function two ways, both of which are wrong: 1. In some cases, COUNT(column name) really means COUNT (DISTINCT column name). 2. In other cases, COUNT(column name) means COUNT(*). This keyword can be set as follows: • DISTINCT: The DISTINCT keyword will be inserted (as in case 1, above). • ASTERISK: The column name will be replaced with a "*" (as in case 2, above). • NONE: No changes will be made. ***Note:*** COUNT(column name) functions are never changed unless the MS Access Compatibility keyword is set to YES. See "AF" on page A-2.	DISTINCT
COID	Correlation ID Support	This keyword controls whether or not SQLGetInfo should return information about correlation IDs. When this keyword is set, SQLGetInfo will behave as follows: • YES: SQLGetInfo will return information about correlation IDs. • NO: No information about correlation IDs will be returned. This keyword must be set to NO to enable some tools to work.	YES
CPFX	Catalog Prefix	This keyword specifies the PROCEDURE QUALIFIER (in ODBC 2.0 terminology) or the PROCEDURE CAT (in ODBC 3.0 terminology). Possible values are as follows: • SYSIBM: Accept the default value to use the standard system catalogs (SYSIBM). • SHADOW: Specify SHADOW only if you are using the Shadow-optimized catalog feature, which requires DB2. SHADOW should represent the table owner (in DB2 terminology, the authorization ID) for a set of DB2 tables that are optimized to support catalog queries. ***Note:*** Shadow-optimized catalogs are considered essential for smooth operation of pre-packaged applications with the Shadow product. • SDBMAP: Specify SDBMAP to return metadata related to data maps in the Shadow Data Mapping Facility (DMF) for CICS and IMS pseudo-stored procedures.	SYSIBM

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
CRBH	Cursor Commit/ Rollback Behavior	This keyword controls what value to return for the cursor commit and rollback behavior SQLGetInfo request, as follows: • DELETE: This will close cursors and delete prepared statements. To use the cursor again, the application must re-prepare and re-execute the statement. • CLOSE: This will only close cursors. For prepared statements, the application can call SQLExecute on the statement without calling SQLPrepare again. • PRESERVE	CLOSE
CRIN	Create Table Index Automatically	This keyword controls whether an index is automatically created when you create a table with a primary key or unique constraint. This index enforces the uniqueness. To disable this option, set the value to NO.	YES
CVNL	Convert Nulls to Blanks	This keyword controls whether or not nulls (zero bytes) in character string data should be returned from the Mainframe Adapter Server to the Mainframe Adapter Client, or should be converted to blanks. Both fixed length and variable length character data are affected by this keyword. This keyword can be set as follows: • YES: Nulls will be converted to blanks. • NO: Nulls will not be converted.	NO
CVTS	Convert Timestamps	This keyword controls whether or not timestamp values should be converted to other types, as follows: • YES: Timestamps that appear to be dates will be converted to dates and timestamps that appear to be times will be converted to times. These conversions are required to circumvent bugs in several products including MS Access and Crystal Reports. • NO: Timestamps will not be modified.	NO
D2VR	DB2 Version String	This keyword can overwrite the host returned DB2 version string data. This string should be in the following format: x.y.z Where x, y and z are all numeric digits, as follows: • x (required) is the DB2 version byte. • y (required) is the DB2 modification level. • z (optional) is the DB2 release level. If the release level is not set, then 0 is assumed. ***Examples:*** • 2.3 is for DB2 version 2, modification level 3. • 4.1.1 is for DB2 version 4, modification level 1, and release 1.	
DACN	Disable SQLCancel Tracing	This keyword controls whether to disable SQLCancel tracing. SQLCancel creates a separate connection to the host Mainframe Adapter Server, and when the SQLCancel operation is completed, this connection is freed. Setting this keyword to YES prevents all tracing of the operations used in creating and terminating the new connection. This is done to avoid cluttering the trace file.	YES

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
DAOP	SQLDescribeParam Support	This keyword controls if Shadow Mainframe Adapter Client should support SQLDescribeParam in a limited form. This keyword can be set to the following values: • Mainframe Adapter Server: The Shadow Mainframe Adapter Server code will decide whether this support is enabled. • YES: SQLDescribeParam is supported to a limited extent. • NO: SQLDescribeParam is not supported at all.	Mainframe Adapter Server
DBD	Default Database	This keyword specifies the database where the table will be created when issuing CREATE statements.	
DBMD	DBCS Mode	**(Only for DBCS or GRAPHIC data)** This keyword is used to specify how DBCS (double byte character set) data is handled: • DEFAULT: (Default) When processing DBCS or GRAPHIC data, leaving the default value of DEFAULT will cause an error. • BINARY: All DBCS data will be treated as binary data. • CHAR: All DBCS data will be treated as character data, even if it is stored in GRAPHIC columns. • GRAPHIC: It is assumed that the DBCS data will be treated as SQL_GRAPHIC, SQL_VARGRAPHIC, or SQL_LONGVARGRAPHIC SQL types. DBCS data will still be converted from host formats to Mainframe Adapter Client formats; however, in BINARY mode, data will be described as binary even though it is not stored in binary columns on the host. ***Note:*** This keyword only applies if you are using DBCS or GRAPHIC data, in which case you must choose how to handle the DBCS data, as attempting to use DEFAULT will result in an error. The three choices are to treat it as binary data (BINARY), treat it as character data (CHAR), or treat it as graphic data (GRAPHIC).	DEFAULT
DBQO	DBCS Remove Blanks	This keyword controls whether or not blanks are removed from within quoted strings for double byte languages (such as Chinese, Japanese, and Korean). Possible values are as follows: • YES: Blanks that are within quoted strings in SQL being sent to the host will be removed. • NO: Blanks that are inside quoted strings in SQL being sent to the host will not be changed.	YES
DBTY	DBMS Type	This keyword provides SQL access to databases by specifying the means by which users will be accessing data. Supported values are as follows: • DB2 • Oracle • ADABAS • VSAM • VSAMCICS • DATA	DB2

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
DCCM	Decimal Point is Comma	This keyword controls if the decimal point is a comma or not, as follows: • YES: A comma is considered to be the decimal point. This keyword should only be set to YES if the host uses a comma as the decimal point. This keyword will cause commas to be converted to periods before data is passed to the auto-static SQL conversion facility. • NO: A period is considered to be the decimal point.	NO
DENU	Decimal as Numeric	This keyword determines whether DECIMAL and NUMERIC columns should be reported as DECIMAL columns or NUMERIC columns (because both types of columns are treated as the same by the Shadow Mainframe Adapter Client). Possible values are as follows: • YES: All DECIMAL and NUMERIC columns will be reported as NUMERIC. • NO: All DECIMAL and NUMERIC columns will be reported as DECIMAL.	NO
Description	Description	This optional keyword can contain a string value describing this data source (user-defined).	
DFCL	Default Column Names	This keyword controls whether or not Shadow Mainframe Adapter Client will assign default column names (of the form COLnnn, where nnn is the column number) when the DBMS does not return names for the columns. If set to YES, default column names will be assigned.	NO
DFPR	Defer Prepare for CALLs	This keyword controls whether or not to defer prepare of CALL statements. Possible values are as follows: • YES: Prepare is always deferred for CALL statements. If this keyword is set to YES, prepare is deferred for CALLs that do not have parameter markers. • NO: Prepare is only deferred for CALLs that have one or more parameter markers. In other words, prepare is always deferred for CALLs that have parameter markers.	NO
DFSC	Default Schema	This keyword controls the default schema for stored procedures with an implicit schema name. Possible values are as follows: • 0: The procedures are executed as Shadow RPCs. • 1: Shadow Mainframe Adapter Server uses the RPCDEFAULTSCHEMA parameter value as the schema name. • Specific schema name: If the value is a specific schema name, it is used as the default schema via the SET CURRENT PATH setting each time a connection is established. ***Note:*** This keyword is only supported at a Shadow Mainframe Adapter Server functional level of 4 or higher (see “FULV” on page A-13).	0

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
DIPO	Disable Prepare/ Open Optimization	This keyword controls whether to disable the prepare/open optimization. Possible values are as follows: • YES: Disables the prepare/open optimization feature, consequently adding a network round-trip (for the SQLExecute). The RDO layer on top of VB issues two SQLPrepares and only one SQLExecute for each select statement. This causes two result sets to be returned for every query. For those Mainframe Adapter Client applications that expect the SQLPrepare to be executed as soon as it requested (for example, applications that require the metadata in the SQLCA), this keyword should be set to YES. • NO: Enables prepare/open optimization, offering support for deferred prepare, meaning that the SQLPrepare and SQLExecute can be combined into single call that is sent at SQLExecute time. Prepare/open optimization minimizes network flow and improves performance.	NO
DP	Table Owner Filter	This keyword specifies the table filter to use for the SQLTables function, which returns a list of available tables in the remote database. This field, along with the accessible tables feature (see "AT" on page A-4), is used to restrict the information returned by the SQLTables function. When the application requests a list of tables, Shadow Mainframe Adapter Client constructs this list based on the value of the table filter. Only tables that fall under the first-level name (userid) are returned. This filter can include the two SQL wildcard characters "%" and "_", as follows: • "%" substitutes for zero or more characters. • "_" substitutes for exactly one character. The table filter defaults to the userid value, which means Shadow Mainframe Adapter Client only returns tables that you own or created under your userid. ***Note:*** More than one table filter can be specified by comma delimiting the entries.	<USERID>
DPSO	Duplicate Socket Descriptor	**(UNIX only)** This keyword is a non-negative integer that is used to return a new and unused file descriptor for the connecting socket. The new socket number will be a descriptor number that is either equal to or greater than this number. The old descriptor will be closed after the replacement. This keyword is only enabled for the UNIX platforms and is designed to allow C run-time applications to get around the 255 maximum open file handle limitation in studio library. For example, a good value to set this keyword to is 256.	0
DRIVER	Driver	This keyword is used for DSN-less connections to specify the name of the Shadow Mainframe Adapter Client driver being used.	

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
DRPM		This keyword controls whether or not Shadow Mainframe Adapter Client should prompt for missing logon information. This keyword can have one of the following values: • NOPROMPT: (Default) SQL_DRIVER_NOPROMPT. • COMPLETE: SQL_DRIVER_COMPLETE. • PROMPT: SQL_DRIVER_PROMPT.	NOPROMPT
DSFL	DTS Plan File	This keyword should ***not*** be used!	
DSN	Data Source Name	This keyword specifies the name of the data source.	
DTCH	Convert DB2 Date to CHAR	When this keyword is set to YES, the DB2 date keyword will be converted to SQL_CHAR.	NO
DTFM	Date Format	This keyword is used to specify how ODBC dates are converted to character strings, as follows: • ODBC: The standard ODBC/ISO format yyyy-mm-dd will be used. • UK: The dd-mm-yyyy format will be used. • EUR: The dd.mm.yyyy format will be used. • USA: The mm/dd/yyyy format will be used. • USA2: The mm/dd/yy format will be used.	ODBC
EXCC	SQLExecute Connection Check	This keyword is used to force a connection check during SQLExecute processing. Possible values are as follows: • YES: SQLExecute will always return an error condition if the connection has been reset or shutdown by the Mainframe Adapter Server. • NO: (Default) If prepare/open optimization is in effect (see “DIPO” on page A-11), SQLExecute will bypass the connection check and return SQL_SUCCESS. This keyword can be used by Shadow Mainframe Adapter Client users when prepare/open optimization is in effect to detect a stale connection during `statement.executeQuery()` processing instead of result set processing.	NO
EXCU	Extended Cursor Pool	This keyword controls whether or not the extended cursor pool should be used. Possible values are as follows: • YES: The extended cursor pool will be used. The extended cursor pool is much larger than the standard cursor pool; however, the extended cursor pool cannot be used with auto-static SQL (see “AUST” on page A-4). • NO: The normal cursor pool will be used.	NO

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
FALG	Fast Logon	This keyword controls whether or not fast logon should be used. Possible values are as follows: • YES: Logon processing will be done with a minimum of network I/O. This is only possible if the host Mainframe Adapter Server supports fast logon. Failures will result if fast logon is requested and the host Mainframe Adapter Server does not support it. The ability of the Mainframe Adapter Server to handle fast logon must be verified before this option is used. • NO: Normal logon processing will be used	NO
FEDL	Unload Communication DLLs	This keyword is used to determine whether the communication DLL (i.e., `WINSOCK.DLL` or `CPIC.DLL`) should be unloaded after an application disconnects. This keyword can be specified as follows: • YES: Shadow Connect always unloads the communication DLL after the application disconnects. • NO: Unloading the communication DLL while the communication service is still active will cause errors. Setting this keyword to NO will prevent the DLL from being unloaded by Shadow Mainframe Adapter Client.	YES
FULV	Function Level Value	**(Read-only)** This keyword is set at connection time based on the maintenance level of the Shadow Sever code.	
FXFL	Fix Floating Point Rounding	This keyword is available to fix certain floating point rounding errors. These errors are typically seen when a floating point number is converted to a decimal or integer value. Possible values are as follows: • YES: Floating point values will be adjusted before they are converted. • NO: No adjustments will be made to floating point values.	NO
FXIS	Fix INSERT Statements	This keyword controls whether or not the VALUES clause of each INSERT statement should be scanned for dates, times, and timestamps, and whether or not quotes should be added. Possible values are as follows: • YES: Quotes will be added to dates, times, and timestamps as needed. • NO: INSERT statements will be not modified.	NO
FXOR	Fix Oracle 8.0.5 Bug	This field is used to fix the Oracle 8.0.5 bug.	NO
GAJI	GAIJI Extension Support	When set to YES, this keyword enables GAIJI extension support in a codepage.	YES
GATB	GAIJI Extension Table Name	This keyword sets the GAIJI extension table name. There is no default name. If this keyword is left blank, a Mainframe Adapter Server-side default table name will be used. This keyword has a maximum size of 4 characters.	

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
GHDS	GRAPHIC Data Processing	This keyword should be set to YES when inserting DBCS data into DB2 GRAPHIC columns. When set to YES, Shadow will determine whether a CHAR parameter is actually a GRAPHIC parameter so that Shadow can handle it correctly.	Conditional
GXSR	GRAPHIC Literal Processing	This keyword controls if some string literals should be converted to parameters. This conversion is needed for some applications operating in DBCS modes. If this keyword is set to YES, then all string literals not proceeded by “G” or “g” will be converted to parameter markers. ***Note:*** This keyword is ignored if Shadow Mainframe Adapter Client is not in a DBCS environment.	NO
HD	Retain Cursor	This keyword controls how DB2 COMMIT operations affect cursors in the data source. Possible values are as follows: • YES: Cursors will be preserved in the same position as before the COMMIT operation took place. This enables the application to execute or fetch without preparing the statement again. ***Note:*** Setting this keyword to YES also causes Shadow Mainframe Adapter Server to issue COMMITs after every SELECT statement as long as auto-commit is enabled by the application. • NO: The application will close and delete cursors after a COMMIT operation, meaning you must prepare and execute the next statement from scratch.	YES
HODB	Host Debugging Values	This keyword is used to control the debugging of host programs (generally stored procedures). The host program language must be specified correctly so that the program can be invoked with the correct debugging options. The supported values are as follows: • NONE • COBOL • PLI • C • C++	NONE
HOST	Host Name	This keyword specifies the host name or IP address for the connection. The value can contain a hostname string or a TCP/IP address in dot-notation format. Failover Shadow Mainframe Adapter Servers can be specified by listing host/port combinations as follows: HOST=Host1,Host2,Host3;PORT=1200,1210,1500 Based on the host/port combinations specified, Shadow Mainframe Adapter Client would try to connect to Host1:1200, Host2:1210, and Host3:1500, in that order. Specified host names can be the same host. Note that this syntax should ***not*** be used to specify Shadow Mainframe Adapter Server group directors.	

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
IDQO	Identifier Quote Option	This keyword is used to show what identifier quote character should be returned to the application using SQLGetInfo with SQL_IDENTIFER_QUOTE_CHAR. This keyword is needed to fix certain application bugs. Possible values are as follows: • BLANK • SINGLE • DOUBLE	BLANK
IGHI	Ignore High Bound Column Errors	This keyword controls whether or not SQLFetch should ignore high bound column errors. Possible values are as follows: • YES: SQLFetch will ignore column binding errors. This keyword must be set to YES to enable some tools (such as Microsoft Excel) to work. • NO: Column binding errors will be handled normally.	NO
IGUN	Ignore Underscore Characters	This keyword controls whether or not the underscore character ("_") should be considered as a wildcard or a regular character. This keyword is used to solve problems in which a table or procedure name actually has an underscore in the name and the underscore is misinterpreted as a wildcard. This keyword can be set as follows: • YES: Underscore will be handled as a normal byte. • NO: Underscore will be treated as a wildcard character that matches any other single byte.	NO
KEQU	Keep Literal Quotes	This keyword is used to keep quotes around and embedded in string literals, as follows: • YES: Quotes around and embedded in string literals will be retained. • NO: Quotes will be removed from string literals. ***Note:*** This only applies to string literals passed to stored procedures, including MDI remote stored procedures (RSPs).	NO
LAFX	Lotus Approach Compatibility	This keyword can be set to YES to resolve certain problems with Lotus Approach. This keyword should not be set for any other reason.	NO
LBCB	Convert LONG VAR Data to LOB	This keyword allows any SQL_LONGVARCHAR or SQL_LONGVARBINARY data to be treated as LOB data (either CLOB or BLOB).	YES
LGFX	Long Data Fix	This keyword controls whether or not a large number should be returned for the length and precision of LONG VARCHAR keywords, rather than the actual length and precision. This fix is needed to resolve certain problems in GetChunk processing. This keyword can be set as follows: • YES: A large number will be returned for the length and precision of LONG VARCHAR keywords. • NO: The actual values will be returned for the length and precision of LONG VARCHAR keywords.	NO

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
LGID	Language ID	This keyword is used to specify the language to be used. The possible values are as follows: • ARB: Arabic • CHS: Simplified Chinese • CHT: Traditional Chinese • DAN: Danish • DFT: Default • DEU: German • ENC: U.S. English Compat • ENG: U.K. English • ENU: U.S. English • ESN: Modern Spanish • ESP: Castilian Spanish • FIN: Finish • FRA: French • FRC: Canadian French • ISL: Icelandic • ITA: Italian • JPL: Japanese • JPX: Japanese Latin Ext • KOR: Korean • MDI: Micro Decisionware • NLD: Dutch • NOR: Norwegian • PPS: PeopleSoft English • PTG: Portuguese • SVE: Swedish • TUR: Turkish	ENU
LGMG	Return Logon Messages	This keyword controls whether or not non-error logon messages should be returned to Mainframe Adapter Client applications. Possible values are as follows: • YES: If a message is returned to the application, the return code will be set to SQL_SUCCESS_WITH_INFO and the error message text will be available using the Solderer function (or some higher level API). • NO: If a message is returned to the application, it will be displayed using a dialog box.	NO
LGPA	Convert Strings to Params	This keyword controls whether or not long strings should be converted to LONG VARCHAR parameters. This conversion is needed because some applications produce literals that are longer than can be handled by the host database. Possible values are as follows: • YES: All long strings will be converted to parameter markers. • NO: SQL strings will not be scanned for long strings.	NO
LINK	Link Type	This keyword specifies the connection information. Supported values are as follows: • TCPIP • LU62 • MQSeries	TCPIP
LKES	Add ESCAPE Option	This keyword causes an ESCAPE clause to be added after each LIKE clause if the following conditions are met: • This keyword is set to YES. • The application did not provide its own ESCAPE clause. • The LIKE clause contains one or more characters that need to be escaped.	YES
LKTH	Use Thread Locks	**(Read-only)** This keyword is set depending on whether thread-safe mode is being utilized.	
LNDN	LAN Domain Name	**(Read-only)** This is an internal, read-only keyword that may not be set by the user.	

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
LNID	LAN Userid	**(Read-only)** This is an internal, read-only keyword that may not be set by the user.	
LVCT	Catalog Level Value	**(Read-only)** This keyword value is negotiated internally between Shadow Mainframe Adapter Client and the Shadow Mainframe Adapter Server and represents the level of support between the two components.	
LZCM	LZ Compression	If this keyword is set to YES and the Shadow Mainframe Adapter Server supports this feature, a variant of Lempel-Ziv compression will be used to compress all data flow between Shadow Mainframe Adapter Client and the Shadow Mainframe Adapter Server.	YES
MDBO	MDI Text/ Keywords	This keyword controls if variable text and keywords can be used together with MDI RSPs, as follows: • YES: Variable text can be used with MDI keywords. • NO: Variable text can not be used with keywords.	NO
MDQO	MDI Keep Quotes	This keyword controls if quotes should be retained around MDI keyword values. Possible values are as follows: • YES: Quotes will be included in keyword values and keyword value lengths. • NO: Quotes will be stripped from MDI keyword values.	NO
MGDG	Message Digest	This keyword is not used.	
MGTY	Message Type	This keyword controls the type of messages used for message oriented connections. It is only used if the messaging connection mode is employed (see “CNMD” on page A-6). Supported values are as follows: • NETWORK: A TCP/IP or LU 6.2 session or conversation is created for each message. • UDP: When the link type is set to TCP/IP (see “LINK” on page A-16), users can also choose the UDP message type as the messaging protocol. Using UDP datagrams for messaging usually incurs lower network overhead; however, UDP communication is not reliable and is discouraged when connecting over a WAN. • HTTP: HTTP and HTTPS methods will tunnel the communication session inside the HTTP protocol stream to allow the session to be established over firewalls and HTTP proxies. • HTTPS: HTTPS is the Netscape secured HTTP protocol which implements SSL and can be used when a secured channel between the two endpoints is required.	NETWORK
MR	Maximum Rows	This keyword can be used to specify an integer value to which to limit the number of rows returned from a single query. If no value is entered, no restriction is placed on the number of rows returned.	0

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
MSMT	MTS Security SID Type	This keyword specifies the security SID option when using the MS Transaction Mainframe Adapter Server based network authentication/single signon model. Possible values are as follows: • Direct Caller SID • Direct Creator SID • Original Caller SID • Original Creator SID	Direct Caller SID
MT	Enable Multitasking	This keyword can be set to YES to enable the multitasking capabilities of Shadow Connect. This feature will keep users' screens from locking up while calls are pending. ***Note:*** This feature should be used with extreme caution! Many applications will not operate correctly with this feature activated.	NO
MUNT	MultiNet TCP/IP Compatibility	This keyword can be set to YES to support applications using the Multi-Net TCP/IP product for host access.	NO
MXBU	Maximum Buffer Size	This keyword sets the Shadow Mainframe Adapter Client maximum communication buffer size for all data exchanges. The default for this keyword is 0 which currently sets the maximum buffer size to 100,000 bytes (32K bytes in 16-bit Windows). ***Note:*** This value will be used to negotiate with the Shadow Mainframe Adapter Server side limit. The actual runtime buffer size could be smaller but not larger.	0
MXDG	Floating to Character Digits	This keyword controls the maximum number of digits that should be generated for a floating point to character conversion. This keyword should only be specified if the default value is not acceptable.	6
MXRT	UDP Maximum Retry Attempts	This keyword controls the maximum number of UDP retries for an unacknowledged request. This value must be a non-negative number. If this keyword is set to 0 (zero), the UDP request retry feature is disabled.	10
NEONTRACE	NEONTRACE	This keyword can be used to control the Shadow Mainframe Adapter Client Trace Facility tracing options. For more information, see Chapter 6, "Shadow Mainframe Adapter Client Trace Facility."	
NBIO	Non-Blocking Network I/O	This keyword controls whether or not TCP/IP network I/O operations should be blocking. Possible values are as follows: • YES: Non-blocking network I/O operations will be used, and the Windows message queue will be used to wait for I/O operation completion. • NO: Blocking network I/O operations will be used.	NO
NLGR	German NLS Support	This keyword can be set to YES to resolve certain problems handling SQL in the German language environment. This keyword should not be set for any other reason.	NO

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
NO3D	Disable CTL3D	This keyword is used to indicate whether or not CTL3D processing will occur. Setting this option to YES will prevent the use of the CTL3D DLL with the connection dialog. This option must be set to YES for some applications if an old copy of CTL3D is installed.	NO
NOAS	Disable Async Option	This keyword is used to indicate whether or not asynchronous execution will be allowed. Setting this option to YES will prevent all asynchronous processing.	YES
NODA	Return Code if No Rows Affected	This keyword controls the return code when an INSERT/UPDATE/DELETE does not affect any rows. The following values are supported: • INFO: The return code will be SQL_SUCESS_WITH_INFO. • NODATA: The return code will be SQL_NO_DATA_FOUND. • SUCCESS: The return code will be SQL_SUCCESS. • ERROR: The return code will be SQL_ERROR.	INFO
NONL	No Nulls	This keyword controls whether or not zero length strings should be returned. Possible values are as follows: • YES: Zero length strings will be replaced by one blank. • NO: Zero length strings will not be modified.	NO
NOPM	Disable All Prompts	This keyword controls whether to disable all interactive prompts or informational message boxes. By setting this keyword to YES, all interactive prompts informational message boxes will be disabled. This feature is required when Shadow Mainframe Adapter Client is being called from an NT service, a UNIX daemon process, or any Mainframe Adapter Server type application that cannot be interrupted.	NO
NOPW	Don’t Get Passwords from ODBC.INI/ SHADOW.INI	This keyword controls whether the password should not be obtained from the `SHADOW.INI` file. Possible values are as follows: • YES: Any password value in the `SHADOW.INI` will be ignored; the password must be provided by the connection string or the connection dialogs. • NO: The password will be obtained from the `SHADOW.INI` file in some cases. ***Note:*** See also “ALPO” on page A-3.	NO
NOUS	Don’t Get Userids from ODBC.INI/ SHADOW.INI	This keyword controls whether the userid should not be obtained from the `SHADOW.INI` file. Possible values are as follows: • YES: Any userid value in the `SHADOW.INI` will be ignored; the userid must be provided by the connection string or the connection dialogs. • NO: The userid will be obtained from the `SHADOW.INI` file in some cases. ***Note:*** See also “ALUO” on page A-3.	NO

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
O3MD	ODBC 3.0 Meta Data Support	This keyword controls whether Shadow Mainframe Adapter Client will request catalog data conforming to the ODBC 3.0 specification. Possible values are as follows: • YES: Shadow Mainframe Adapter Client requests catalog data as defined in the ODBC 3.0 specification. When using Shadow Mainframe Adapter Server optimized catalogs, the proper Shadow Mainframe Adapter Server catalog level support must be in place to complete this support (the Shadow Mainframe Adapter Server parameter ODBCCATALOGLEVEL must be set to 4). If the Mainframe Adapter Client requests it, but the host installation does not have the correct catalog level turned on, the connection will fail. • NO: Shadow Mainframe Adapter Client requests catalog data as defined in the ODBC 1.0 specification.	NO
ODBC.INI	ODBC.INI	**(Read-only)** This keyword is used to display the location and name of the `ODBC.INI` file found and used for the connection on UNIX platforms.	
ODPG	ODBC Mainframe Adapter Client Code Page	This keyword is used to specify the ODBC Mainframe Adapter Client code page. The following values are supported: • DEFAULT: The default is a set of UNIX code pages for UNIX and Windows Latin 1 (ANSI) for Windows 95/NT. • LATIN1: Forces the use of the Windows Latin 1 code page in any environment. • UNIX: Forces the use of the UNIX code pages in any environment. ***Note:*** Windows Latin 1 is also known as ISO 8859.	DEFAULT
OJFX	Fix DB2 Outer Joins	If this keyword is set to YES, parentheses around ON clauses of DB2 outer joins will be removed. While this syntax is valid ANSI SQL, DB2 V5 cannot handle it. This kind of SQL is generated by tools such as Brio.	NO
OLEDB	Caller is an OLE DB Provider	This keyword indicates whether Shadow Mainframe Adapter Client is operating as an ODE DB provider. ***Note:*** This keyword can only bet set through the connection string.	NO
ONWY	One-Way Messages	This keyword controls whether or not responses to messages should be sent to the Mainframe Adapter Client. Possible values are as follows: • YES: One-way message processing will be used. This means that no responses will be returned to Mainframe Adapter Client applications. • NO: Normal responses will be returned for messages. ***Note:*** This keyword can only be used with messages.	NO

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
OPRW	Optimal Row Count	This keyword limits the number of rows that will be returned from the host each time a request for rows is made. Since any number of requests for rows can be made for one query, this value will have no effect on the total number or rows returned by a query. This value can be used to control the size of the buffers used to return rows from the host.	0
OPTM	Operation Timeout Value	This keyword controls the timeout value (in seconds) for all Mainframe Adapter Client network operations (i.e. query preparation, execution, retrieving result set) ***after*** the connection has been established. This value should never be negative, but it can be zero. This value has no effect in UDP messaging. ***Note:*** The use of this keyword may cause some unpredictable results in many environments. This option is only available in the TCP/IP and the MQ link types (see “LINK” on page A-16), but it is applicable to both the permanent and the messaging modes (see “CNMD” on page A-6).	0
PAHN	Parameter Handling	This keyword is used to control how parameters are handled when they are passed by applications. In some cases, special processing is needed to fix values passed by some applications. Possible values are as follows: • NONE: Parameter values passed by applications are not changed. • INPUT: Required for Crystal Reports, because Crystal Reports passes parameters using obsolete values that must be modified before they can be used. • LENGTH: Fixes some application parameter length errors. • ADO: Fixes some ADO related errors.	NONE
PAOP	SQLParamOptions Support	This keyword controls if Shadow Mainframe Adapter Client should support SQLParamOptions in a limited form. This keyword can be set to the following values: • Mainframe Adapter Server: The Shadow Mainframe Adapter Server code will decide whether this support is enabled. • YES: SQLParamOptions is supported to a limited extent. • NO: SQLParamOptions is not supported at all.	Mainframe Adapter Server
PBFU	PowerBuilder Compatibility	This keyword can be set to YES to resolve certain problems with PowerBuilder. This keyword should not be set for any other reason.	NO

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
PLAN	Plan Name	This keyword specifies the plan name that Shadow Connect uses on the host to make a connection to DB2. The format of this keyword is as follows: • The first two letters of the name are always "SD." • The third letter is always the last letter of the Shadow Mainframe Adapter Server subsystem name, which is "B" by default. • The fourth letter designates the isolation level of the plan, as follows: C Cursor Stability SQL_TXN_READ_COMMITTED R Repeatable Read SQL_TXN_SERIALIZABLE S Read Stability SQL_TXN_REPEATABLE_READ U Uncommitted Read SQL_TXN_READ_UNCOMMITTED ***Note:*** If the fourth character is any character other than "R," "S," or "U," it is assumed the plan was bound with an isolation level of Cursor Stability. • The final four characters are always a version number (such as "1010" or "1040"). For example, the default plan name value is SDBC1010. This default is release-dependent. ***Note:*** If the Mainframe Adapter Client sets this keyword to either DFLT or DEFAULT, then the DEFAULTDB2PLANNAME parameter that is set on the Shadow Mainframe Adapter Server will be used for that connection.	SDBC1010
PORT	Port Number	This keyword specifies the Shadow Mainframe Adapter Server listener port. Failover Shadow Mainframe Adapter Servers can be specified by listing host/port combinations as follows: HOST=Host1,Host2,Host3;PORT=1200,1210,1500 Based on the host/port combinations specified, Shadow Mainframe Adapter Client would try to connect to Host1:1200, Host2:1210, and Host3:1500, in that order. Specified host names can be the same host. Note that this syntax should ***not*** be used to specify Shadow Mainframe Adapter Server group directors.	1200
PRAD	Precision Adjustment	This keyword controls the precision-adjustment Shadow Mainframe Adapter Client makes when passing a numeric value as a literal. When passing a literal numeric value, Shadow Mainframe Adapter Client will adjust the precision to be an odd number by adding one (1) to even precision values. For example, the value 12.34 will be sent with a precision of 5. This adjustment sometimes causes the rejection of acceptable values; thus, this precision-adjustment can be disabled by setting this keyword to NO.	YES

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
PRES	Process Escapes	This keyword is used to turn off ESCAPE clause processing. Possible values are as follows: • YES: Perform ESCAPE clause processing. • NO: Turn off ESCAPE clause processing. This may result in improved performance by reducing CPU overhead.	YES
PRNF	Procedure Name Filter	This keyword is used to filter the procedure names returned by the product. This keyword is used only if it is set to a non-blank value and if the host application does not provide a procedure name filter string. The procedure name filter string from either the host application or from this keyword restricts the information returned by procedure catalog inquiries (SQLProcedures and SQLProcedureColumns). Only rows that match the procedure name pattern provided will be returned. If this keyword is set to a single percent sign (“%”), then all procedures will be returned. There is no default value for this keyword.	
PROF	Procedure Owner Filter	This keyword is used to filter the procedure owners returned by the product. This keyword is used only if it is set to a non-blank value and if the host application does not provide a procedure owner filter string. The procedure owner filter string from either the host application or from this keyword restricts the information returned by procedure catalog inquiries (SQLProcedures and SQLProcedureColumns). Only rows with owner IDs that match the procedure owner pattern provided will be returned. If this keyword is set to a single percent sign (“%”), then all procedures will be returned. This keyword defaults to the current userid or alternate userid.	
PROW	Procedure Owner Handling	This keyword is used to control how procedure owner values are returned to applications, as follows: • NONE: The procedure owner value is not changed. • NULL: The actual owner value is used as a prefix to the procedure name, and the owner keyword is returned as a null value. • EMPTY: The actual owner value is used as a prefix to the procedure name, and the owner keyword is returned as an empty string.	NONE
PWD	Password	This keyword specifies the mainframe password with which to connect to the host.	
PXPW	Proxy Mainframe Adapter Server Password	When HTTP tunneling messaging mode is enabled and the HTTP proxy Mainframe Adapter Server requires user authentication, this keyword allows the user to specify the proxy user password string. At this time, Basic HTTP Access authentication is the only method supported.	

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
PXSR	Proxy Mainframe Adapter Server Information	When HTTP tunneling messaging mode is enabled and the HTTP stream must first connect to a proxy Mainframe Adapter Server, this keyword should be set to identify the proxy Mainframe Adapter Server's hostname (or IP address) and the port number. The accepted formats for this keyword are as follows: • MY_PROXY:80 • 10.22.33.44:1200 The first part of the value is the hostname or the IP address of the proxy Mainframe Adapter Server and the second part is the port number on which the proxy Mainframe Adapter Server is listening. ***Note:*** For some proxy Mainframe Adapter Server setups, user authentication might be required.	
PXUS	Proxy Mainframe Adapter Server Userid	When HTTP tunneling messaging mode is enabled and the HTTP proxy Mainframe Adapter Server requires user authentication, this keyword allows the user to specify the proxy userid (or name) string. At this time, Basic HTTP Access authentication is the only method supported	
QUNA	Qualifier Name Separator	This keyword should be set to YES if SQLGetInfo must return a period when obtaining SQL_QUALIFIER_NAME_SEPARATOR. This keyword is provided to fix certain application bugs.	NO
RDFX	RDO Compatibility	This keyword can be set to YES to resolve certain problems with Remote Data Objects. This keyword should not be set for any other reason.	NO
RDON	Read Only	This keyword is set to YES to make the data source read-only. Setting the data source to read-only will not actually prevent update operations; however, it will prevent any index information from being returned to the application. This will generally prevent any updates from being attempted. ***Note:*** Setting this keyword to YES will greatly improve the performance of some applications using the standard DB2 catalogs.	NO
RMEQ	Remove Equals	This keyword controls if the equals byte should be removed from MDI keyword names as part of MDI RSP (Remote Stored Procedure) invocation. Possible values are as follows: • YES: The equals byte will be removed from each keyword name. • NO: The equals will not be removed from the keyword name. ***Note:*** This keyword only applies to MDI RSPs invoked using TSQL 0/1/2 syntax. It does not apply to MDI RSPs invoked using the extended ODBC CALL syntax.	NO
RO	Optimized Fetch	This keyword controls whether or not to use the block fetch feature of Shadow Mainframe Adapter Client. By using the block fetch feature, the increase in performance is significant. ***Note:*** When the block fetch feature is enabled, cursor-based deletes (DELETE WHERE CURSOR OF) cannot be used.	YES

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
RPGT	Allow Repeating SQLGetData	This keyword controls whether to allow repeating SQLGetData calls for the same unbound column while the cursor is still on that result set row. For variable length columns, applications can retrieve the column data in pieces. After the data is completely retrieved, the next SQLGetData call will retrieve the data from the beginning again. ***Note:*** LOB type columns cannot be retreived repeatedly.	YES
RSBK	Result Set Cache Size	This keyword specifies the number of SQL statements contained by the result set cache (see “RSCH” on page A-25). ***Note:*** A higher value translates into more disk space.	256
RSCH	Result Set Caching	If this keyword is set to YES, Shadow Mainframe Adapter Client will cache result sets of SQL statements. When the same SQL statement is encountered, results are retrieved from a local cache if the underlying tables have not changed since the last cache update.	NO
RSIN	Result Set Caching Interval	This keyword specifies an interval (in seconds) within which stale results are tolerated. This keyword can be specified as follows: • Any positive number: The result set will be returned without host processing if the following conditions are met: – If result set caching is enabled (see “RSCH” on page A-25) and a previously cached SQL statement is encountered. – The number of seconds since the result set was stored is within this interval. • 0 (zero): All statements will be submitted to the host, where the underlying tables will be checked for changes; this will guarantee no stale results. • -1: All result sets will always retrieved from a local cache (if one is available). ***Note:*** Tolerances greater than zero save network round-trips, but a value of zero is still useful since the result set is still retrieved from a local cache if the underlying tables haven’t changed.	0
RTGL	Return Global Tables	This keyword controls whether or not global temporary tables (GTTs) will be returned as normal tables. Possible values are as follows: • YES: GTTs will be treated and described as normal tables. • NO: GTTs will be handled as GTTs. This keyword is provided to allow standard tools to be used with GTTs. Many of these tools bypass GTTs that are described as GTTs; however, the same tools will work correctly with GTTs that are described as normal tables.	YES

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
RTSY	Return System Tables	This keyword controls whether or not system tables (SYSIBM) will be returned as normal tables. Possible values are as follows: • YES: System tables will be treated and described as normal tables. • NO: System tables will be handled as system tables. This keyword is provided to allow standard tools to be used with system tables. Many of these tools bypass system tables that are described as system tables. However, the same tools will work correctly with system tables that are described as normal tables.	NO
RTTM	UDP Datagram Retry Timeout	This keyword controls the initial timeout period (in seconds) before a UDP request is retransmitted. This timeout value is doubled on successive retries; however, this value will not increase without bound—the maximum timeout allowable is 30 seconds. ***Note:*** If the UDP request retry feature is disabled (see “MXRT” on page A-18), this keyword will have no effect.	30
RTTP	Return TABLE_TYPE Column for Meta Data Calls	This keyword controls whether the TABLE_TYPE column is returned for the SQLPrimaryKeys and the SQLStatistics calls. If this keyword is set to YES, the TABLE_TYPE column will be returned as the last column of the result set.	YES
SBMD	SBCS Mode	**(Only for DBCS or GRAPHIC data)** This keyword is used to specify how SBCS (single byte character set) data is handled. SBCS strings are assumed to contain a mixture of single byte characters and double byte characters. Each set of double byte characters starts with an SO byte and ends with an SI byte. This keyword can be set as follows: • DEFAULT: (Default) When processing DBCS or GRAPHIC data, leaving the default value of DEFAULT will cause an error. • BLANK: All SO/SI bytes will be converted to blanks. • DELETE: All SO/SI bytes will be deleted from each string. ***Note:*** This keyword only applies if you are using DBCS or GRAPHIC data, in which case you must choose how to handle the SO/SI (shift out/shift in) bytes, as attempting to use DEFAULT will result in an error. The two choices are to convert them to blanks (BLANK) or remove them (DELETE).	DEFAULT
SECN	Application Mainframe Adapter Server	This keyword is used to select the initial application Mainframe Adapter Server value on the host. It must be sixteen or fewer characters long. If this keyword is set to a non-blank, non-null value, the current application will be connected to the specified application Mainframe Adapter Server after DSNALI OPEN processing is completed.	

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
SECU	Network Authentication	This keyword controls the network authentication model. Support of network authentication models allows users to connect without being prompted for their password, assuming they have already been authenticated via their NT domain and their userid is the same on the OS/390 or z/OS system. It is required that both the authenticated LANID and TSOID that are used during logon processing are the same and are defined in the OS/390 or z/OS security database for RACF, ACF2 or Top Secret. The following values are supported: • NONE • Domain-based: When the domain-based authentication is selected, Shadow Mainframe Adapter Client will verify if the userid associated with the current process has been authenticated by a domain-based system. For Windows platforms, this requires the user to first log on to a NT domain. For UNIX platforms, the local machine must be a member of a NIS domain, and the password database used to authenticate the user must be NIS-mapped. • MS Transaction Mainframe Adapter Server: The Microsoft Transaction Mainframe Adapter Server (MTS) based authentication model allows an MTS Mainframe Adapter Server COM component to pass on the account name of the Mainframe Adapter Client applications invoking the object. • MyNet: This setting is used with the CKS MyNet single signon product support. • SNA: This setting works with the MS SNA Mainframe Adapter Server single signon facility. The Mainframe Adapter Client must be a MS/SNA CPI-C Mainframe Adapter Client. ***(Continued on next page)***	NONE
SECU *(Continued)*	Network Authentication	***(Continued from previous page)*** ***Note:*** In order to support the functionality of this keyword, the Shadow Mainframe Adapter Server parameter Mainframe Adapter ClientLOGON must be set to YES. On the Shadow Mainframe Adapter Server, a RACINIT call is still performed to properly set up the ACEE information for the connection; however, the call will specify that the connection is trusted.	NONE
SEDG	Current Degree	This keyword is used to set the initial current degree value on the host. The only possible values for this keyword are as follows: • ANY • 1 No other values are supported at this time. If this keyword is set to a non-blank, non-null value, a SET CURRENT DEGREE statement is issued after DSNALI OPEN processing is completed.	

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
SEPK	Current Packageset	This keyword is used to set the initial current packageset value on the host. This value must be eighteen or fewer characters long. If this keyword is set to a non-blank, non-null value, a SET CURRENT PACKAGESET statement is issued after DSNALI OPEN processing is completed. The SET statement assigns the specified value to the CURRENT PACKAGESET special register.	
SERL	Current Rules	This keyword is used to set the initial current rules value on the host. The only possible values for this keyword are as follows: • DB2 • STD No other values are supported at this time. If this keyword is set to a non-blank, non-null value, a SET CURRENT RULES statement is issued after DSNALI OPEN processing is completed.	
Mainframe Adapter Server	Mainframe Adapter Server	This keyword can be set to bypass a FILEDSN problem with Microsoft Access. ***Note:*** This keyword can be set through the connection string, but it is not available via jConfig tool dialog boxes.	
SEVL	System Engineering Value	This keyword can be set to various values to obtain diagnostic and debugging data. This keyword should only be used at the specific request of the Customer Support staff.	0
SKEY	Customer Secret Key	This keyword is not for general use.	
SRMD	Instance Message Digest	This keyword is not used.	
SQVA	SQL Mainframe Adapter Server Support	This keyword is used to control whether or not SQLTypeInfo calls should show that variable data is 255 bytes long. DB2 variable data is actually only 254 bytes long; however, some applications do not support that length. Possible values for this keyword are as follows: • YES: SQLTypeInfo will return 255 as the length of variable data. If this keyword is set to YES, the behavior of SQLCancel is modified to circumvent vendor implementation errors. • NO: SQLTypeInfo will return 254 as the length of variable data.	NO
SSLX	Secured Channel Protocol	This keyword allows the user to select a secured channel protocol to encrypt and authenticate the Mainframe Adapter Client/Mainframe Adapter Server session, as follows: • Standard: The default is the standard Shadow Connect secured data stream, which provides optimal performance but does not support strong encryption and authentication. • SSL2: This is based on the SSL version 2 standard. • SSL3: This is based on the SSL version 3 standard; however, this value is not supported at this time.	Standard

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
STFX	Fix String Length	This keyword can be set to YES to fix a specific bug that can occur in some products, like Microsoft Access, where the string length is incorrectly set to zero when the SQL_CHAR keyword only contains blanks. ***Note:*** If the keyword AF (MS Access Compatibility) is set to YES (see "AF" on page A-2), this keyword does not need to be set. The keyword AF includes the functionality of keyword STFX.	NO
STZO	Suppress Decimal Trailing Zeros	This keyword controls whether or not trailing zeros should be removed from decimal keywords. If this keyword is set to YES, trailing zeros will be removed. ***Note:*** If the keyword AF (MS Access Compatibility) is set to YES (see "AF" on page A-2), this keyword does not need to be set. The keyword AF includes the functionality of keyword STZO.	NO
SUBSYS	Subsystem	This keyword specifies the DB2 subsystem. This keyword should only be set if using DB2; otherwise this keyword must be set to NONE.	
SYFU	Sybase Compatibility	If this keyword is set to YES, a set of Sybase fixes will be used. These fixes bypass bugs in the Sybase ODBC support.	NO
SYOP	Table Owner - Synonym Option	This keyword is specifies how table owners should be handled for tables that are actually synonyms. Some of these choices are required to make specific desktop productivity tools work with Shadow Mainframe Adapter Client. Possible values are as follows: • ZERO • NORMAL • NULL • FORCEZERO	ZERO
TBFL	Table Name Filter	This keyword is used to filter the table names returned by the product. This keyword is only used if it is set to a non-blank value and if the host application does not provide a table name filter string. The table name filter string from either the host application or from this keyword restricts the information returned by catalog inquiries (SQLTables, SQLColumns, SQLTablePrivileges). Only rows that match the table name pattern provided will be returned. If this keyword is set to a single percent sign ("%"), all tables will be returned. There is no default value for this keyword. ***Examples:*** • TBFL=STAFF will cause only information from the table labeled "STAFF" to be displayed. • TBFL=STA% will cause information from all tables that begin with "STA" to be displayed. • TBFL=STAFF BOOK% will cause information from the table labeled "STAFF" and all tables that begin with "BOOK" to be displayed.	

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
TCLM	Table & Column Name Modification	This keyword controls adjustment of table and column names for DB2, as follows: • NONE: The names are considered correct and are not processed. • BEA1: The BEA formula converts names to 18 characters by using the first character and last 17 characters for names longer than 18 characters.	NONE
TQOP	Table Qualifier Option	This keyword specifies how table qualifiers should be returned to application programs. Some databases use table qualifiers as part of the overall name of each table. In other words, two tables can have exactly the same name in all other respects, if their table qualifiers are different. There are three possible values: • NORMAL • NULL • ZERO	NORMAL
TRLT	Truncate Literal String	This keyword controls whether literal strings are truncated. Setting this keyword to NO will not truncate the literal string even if the string length is greater than 20.	YES
TRNA	Transaction Name	This keyword is used to specify the transaction name. The use of the transaction name is application specific. It can be up to eight (8) bytes long. The transaction name is padded on the right with blanks.	
TSCH	Convert DB2 Time to CHAR	When this keyword is set to YES, the DB2 timestamp and time keyword will be converted to SQL_CHAR. ***Note:*** If the keyword AF (MS Access Compatibility) is set to YES (see "AF" on page A-2), this keyword does not need to be set. The keyword AF includes the functionality of keyword TSCH.	NO
UPCH	Upcase All Character Data	This keyword controls whether or not all character data sent to the host should be converted to uppercase. Possible values are as follows: • YES: All character data will be converted to uppercase. • NO: Character data will not be modified.	NO
UPDB	Upcase Double Quote String	This keyword controls whether or not strings in double quotes should be converted to uppercase. Strings must be converted to uppercase in some cases because DB2 treats table names in double quotes as literals. Possible values are as follows: • YES: Strings in double quotes will be converted to uppercase. • NO: Strings in double quotes will not be modified.	NO
UPNL	Upcase All Non-Literals	This keyword controls whether all SQL keywords that are non-literals will be converted to uppercase. ***Note:*** Although the default for this keyword is NO in most cases, if the following conditions are met, this keyword will default to YES: • If the AUST keyword is set to NO (see "AUST" on page A-40). • If the LGID keyword is set to JPL or JPX (see "LGID" on page A-16).	NO

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
USERID	Userid	This keyword specifies the mainframe userid with which to connect to the host.	
USERPARM	Host User Parm	This keyword is sent to the host as part of the logon information. The host uses this keyword to complete logon to the host security system and/or host databases.	
VAFX	Visual Age Compatibility	This keyword can be set to YES to resolve certain problems with Visual Age. This keyword should not be set for any other reason.	NO
WACU	Display Wait Cursor	This keyword controls whether or not to update the cursor (with an hourglass). Users may observe a dramatic improvement in performance by setting this option to NO, which results in no updates to the cursor and no display of the hourglass.	NO
WGPH	Wollongong Pathworks 3.1 Support	This keyword must be set to YES when running in the Wollongong Pathworks 3.1 16-bit TCP/IP environment	NO
WIUR	Add WITH UR to Queries	This keyword controls whether or not a WITH UR clause should be added to queries. The WITH UR clause reduces the amount of CPU time needed for some queries. There is also the possibility it could change the results of these queries by allowing uncommitted data to be read. Possible values are as follows: • YES: A WITH UR clause will be added to queries. • NO: A WITH UR clause will not be added to queries.	NO
WRPR	Enable SQLPrepare for ADABAS/ VSAM	This keyword controls whether or not Shadow Mainframe Adapter Client should enable SQLPrepare support for ADABAS and VSAM statements that have parameter markers. Possible values are as follows: • YES: Enable SQLPrepare support for statements that have one or more parameter markers. • NO: Prepare will be deferred for statements that have parameter markers. ***Note:*** For the statements that do not have parameter markers, regardless of the setting of this keyword, SQLPrepare is always supported and the result-set is returned at SQLPrepare.	YES
XAEN	X/OPEN XA Support	This keyword should be set if the target Mainframe Adapter Server will participate in an X/OPEN XA transaction. Possible values are as follows: • None: This value implies that no part of the transaction will use the XA protocol. • TWO-PHASE: This value should be used in a situation where the transaction involves more than one resource manager and the target Mainframe Adapter Server supports the two-phase commit protocol. The value TWO-PHASE also allows for the one-phase commit scenario whenever the coordinator thinks it is appropriate.	None

Table A–1. Shadow Mainframe Adapter Client Keywords (Continued)

Keyword	Description	Explanation	Default
XAOP	X/OPEN XA Transaction Manager	This keyword is used to set the XA transaction manager type. This option must be set correctly when participating in a distributed transaction coordinated by a monitor. Supported values are as follows: • None: This value implies that this is not a distributed transaction. • TUXEDO-TMS: Used for BEA Tuxedo coordinated transactions to identify that the connection will be owned by the Tuxedo TMS Mainframe Adapter Server. • TUXEDO-SQL: Used for BEA Tuxedo coordinated transactions to identify that the connection will be owned by the Tuxedo SQL application Mainframe Adapter Server. • MSDTC: MSDTC is the Microsoft Distributed Transaction Manager used mostly by the MTS applications. • JTS: Used for any transaction manager compatible with the Java Transaction API (JTA).	None
ZEAS	Zero Active Statements	This keyword controls whether or not zero should be returned for the number of active statements. Possible values are as follows: • YES: Zero will be returned for the number of active statements supported by Shadow Mainframe Adapter Client. • NO: The actual number of supported active statements will be returned.	NO
ZECL	Zero Column Names	This keyword must be set to YES if column names should be set to binary zeros. This is a performance optimization for production applications.	NO

Setting Keywords to Benefit Performance

There are several keywords that can be set to improve performance and to benefit users of certain tools. This section will highlight these keywords and their benefits for various applications.

3-Tier Applications

- **NOPM (Disable All Prompts)**

  You must set this keyword to YES when Shadow Mainframe Adapter Client is being called from an NT service, a UNIX daemon process, or any other Mainframe Adapter Server-type application that cannot be interrupted. This will prevent interactive prompts or informational message boxes, such as those informing you about license or password expiration status, from interrupting an application.

General Use

- **LGPA (Convert Strings to Params)**

  Setting this keyword to YES results in the conversion of strings greater than 254 bits to parameter markers. This is necessary for those applications that produce literals that are too long to be handled by the host database.

  Without this setting, you will get the SQLCODE error -102, indicating that the host database could not handle the long string.

- **DIPO (Disable Prepare/Open Optimization)**

  Setting this keyword to NO enables the prepare/open optimization feature. The prepare/open optimization feature saves a network round-trip for applications that use the ODBC functions SQLPrepare and SQLExecute instead of SQLExecDirect. Using SQLPrepare to prepare a SQL statement and SQLExecute to execute that SQL statement is a two trip operation. With prepare/open optimization, Shadow Connect has combined these two functions into a single network operation. If the application issues a SQLPrepare, Shadow Mainframe Adapter Client, in conjunction with Shadow Mainframe Adapter Server, prepares *and executes* the SQL statement. As a result, when the Mainframe Adapter Client issues the SQLExecute, Shadow Mainframe Adapter Client immediately returns SQL_SUCCESS without performing the additional network round-trip.

  This works great for most applications; however, if the applications issue SQLPrepare calls “under-the-covers” to access meta data information, the SQL statements will never be executed. To disable the prepare/open optimization feature, set the DIPO keyword (Disable Prepare/Open Optimization) to YES.

- **FALG (Fast Logon)**

  When set to YES, this keyword allows for logon processing with a minimum of network I/O. However, since this feature can only be used by host Mainframe Adapter Servers that support fast logon, it will fail with older versions of the Shadow Mainframe Adapter Server.

- **USERPARM (Host User Parm)**

  The information stored by this keyword is sent to the host as part of the logon information and is also inserted into the SMF records. This keyword is used to complete logon to the host security system and/or host database.

- **OPTM (Operation Timeout Value)**

  This keyword controls the timeout value (in seconds) for all Mainframe Adapter Client network operations after the connection has been established. When set to the default value of 0, it will not cancel a long query; however, when set to a number greater than 0, it will cancel the connection after the operation has exceeded the timeout value.

- **OPRW (Optimal Row Count)**

  This keyword allows you to specify the number of rows returned per block fetch. This will limit the number of rows that will be returned from the host each time a request for rows is made. This value will have no effect on the total number of rows returned by a query, but it will control the number per block fetch.

- **RDON (Read Only)**

  Setting this keyword to YES will greatly improve the performance of some applications using the standard DB2 catalogs. It will make the data source read-only, preventing any index information from being returned to the application. This will generally prevent any update attempts, although it will not actually prevent update operations.

- **LGMG (Return Logon Messages)**

  When this keyword is set to YES, if a logon returns a message, a return code of SQL_SUCCESS_WITH_INFO will be set. This way, a dialog box is not displayed, and the message text will be available using the SQLError function.

- **TQOP (Table Qualifier Option)**

  This keyword is used to specify how table qualifiers should be returned to application tables. Typically, an application will issue SELECT statements that contain the database name, the qualifier, and the table name. Possible values for this keyword are NORMAL, NULL, and ZERO. When this keyword is set to ZERO, Shadow Mainframe Adapter Client will return strings of zero length for the database name when calling the SQL tables.

APPENDIX B:

National Language Support

This chapter describes the National Language Support (NLS) of Shadow Mainframe Adapter Client, part of the Mainframe Adapter Client component of Shadow Connect. Topics include the following:

- Languages Supported
- Setting the Language

Languages Supported

Table B–1 below lists the languages that Shadow Mainframe Adapter Client supports, along with the code pages used for each.

Table B–1. Shadow Mainframe Adapter Client NLS Support

Language	Windows Language Code	EBCDIC Code Page Number	Windows Code Page Number
Arabic	ARB	N/A	1256
Canadian French	FRC	037	1252
Danish	DAN	277	1252
Dutch	NLD	037	1252
English (UK)	ENG	285	1252
English (US)	ENU	037	1252
Finnish	FIN	278	1252
French	FRA	297	1252
German	DEU	273	1252
Icelandic	ISL	871	1252
Italian	ITA	280	1252
Japanese	JPL	290	932
Korean	KOR	037	1252
MDI	MDI	N/A	1252
Norwegian	NOR	277	1252
PeopleSoft	PPS	N/A	1252
Portuguese	PTG	037	1252
Portuguese (Brazilian)	PTB	037	1252

Table B–1. Shadow Mainframe Adapter Client NLS Support (Continued)

Language	Windows Language Code	EBCDIC Code Page Number	Windows Code Page Number
Spanish (Modern)	ESN	284	1252
Spanish (Castilian)	ESP	284	1252
Swedish	SWE	278	1252

Setting the Language

Shadow Mainframe Adapter Client accommodates your choice of language in a transparent manner. By default, Shadow Mainframe Adapter Client uses the language setting you have already specified for your machine; however, this language value can be overwritten using connection string or the data source settings.

Shadow Mainframe Adapter Client attempts to obtain the language information from three sources. The three sources and the priority order by which the language is set are as follows:

- Connection string.
- Data source settings.
- Locale settings.

Note:
If the language has not been set by any of the methods mentioned above, Shadow Mainframe Adapter Client will default to English (US).

1. **Connection String:** The connection string is passed by an application. To specify a language using a connection string, the LGID keyword must be coded. An example Mainframe Adapter Clientiion string follows:

   ```
   jdbc:neon:<data-source-name>;
   HOST=MKT.NEONSYS.COM;PORT=1200;UID=UID1;PWD=PWD1;
   SUBSYS=DSN1;PLAN=SDBC1010;CPFX=SYSIBM;LGID=DAN
   ```

2. **JDBC Data Source Settings:** The language information may also be specified in the data source settings, which are configured by using the jConfig tool to configure the data source settings by selecting the data source and navigating to **Configure →Parameters**. Then, from the **Parameter Settings** list, select **Language ID** (keyword LGID) and select an appropriate value from the **Value** list. See Figure B–1.

Figure B–1. Optional Parameters Information -- Setting the Language ID

Doc Reference:
For more information about configuring a JDBC data source, see “Configuring a JDBC Data Source” on page 4-4 within Chapter 4, “Shadow Mainframe Adapter Client: Configuration,” of this guide.

3. **Locale Setting:** The locale setting can also be used to set the language as follows:

   - **Windows:** To set the language for the Shadow Mainframe Adapter Client, go into the **Control Panel**, select the **Regional Options** icon, and select the language using the drop-down menu. This causes the new value to be stored in the `WIN.INI` file.

   - **UNIX:** You will need to adjust your machine-specific locale environment variable (see your system administrator for details).

APPENDIX C: *JDBC Compliance*

Shadow Mainframe Adapter Client, part of the Mainframe Adapter Client component of the Shadow Connect product, offers JDBC compliance as detailed in this appendix.

JDBC Compliance

CallableStatement - *Partially Supported*

Methods ***not*** supported include:

- `getArray (int)`
- `getBlob (int)`
- `getClob (int)`
- `getObject (int, java.util.Map)`
- `getRef (int)`
- `registerOutParameter (int, int, String)`

Connection - *Partially Supported*

Methods ***not*** supported include:

- `getTypeMap ()`
- `setTransactionIsolation (int)`

> ***Note:***
> The transaction isolation level is set by selecting the right plan at connection time. Once it’s selected, there is no way of changing it.

- `setTypeMap (java.util.Map)`

ConnectionPoolDataSource - *Fully Supported*

DatabaseMetaData - *Fully Supported*

Methods that return hard-coded values include:

- `deletesAreDetected (int)` - Always returns false
- `getExtraNameCharacters ()` - Always returns null
- `getUDTs (String, String, String, int[])` - Always returns null
- `insertsAreDetected (int)` - Always returns false
- `othersDeletesAreVisible (int)` - Always returns false
- `othersInsertsAreVisible (int)` - Always returns false
- `othersUpdatesAreVisible (int)` - Always returns false

- `ownDeletesAreVisible (int)` - Always returns false
- `ownInsertsAreVisible (int)` - Always returns false
- `ownUpdatesAreVisible (int)` - Always returns false
- `supportsANSI92EntryLevelSQL ()` - Always returns true
- `supportsANSI92FullSQL ()` - Always returns false
- `supportsANSI92IntermediateSQL ()` - Always returns false
- `updatesAreDetected (int)` - Always returns false

DataSource - *Fully Supported*

Driver - *Fully Supported*

PooledConnection - *Fully Supported*

PreparedStatement - *Partially Supported*

Methods ***not*** supported include:

- `setArray (int, Array)`
- `setBlob (int, Blob)`
- `setClob (int, Clob)`
- `setNull (int, int, String)`
- `setRef (int, Ref)`

ResultSet - *Partially Supported*

Methods ***not*** supported include:

- `absolute (int)`
- `afterLast ()`
- `beforeFirst ()`
- `cancelRowUpdates ()`
- `deleteRow ()`
- `first ()`
- `getArray (int)`
- `getArray (String)`
- `getBlob (int)`
- `getBlob (String)`
- `getClob (int)`
- `getClob (String)`
- `getFetchSize ()`
- `getObject (int, java.util.Map)`
- `getObject (String, java.util.Map)`
- `getRef (int)`
- `getRef (String)`
- `getRow ()`
- `insertRow ()`
- `isAfterLast ()`
- `isBeforeFirst ()`
- `isFirst ()`

- `isLast ()`
- `last ()`
- `moveToCurrentRow ()`
- `moveToInsertRow ()`
- `previous ()`
- `refreshRow ()`
- `relative (int)`
- `rowDeleted ()`
- `rowInserted ()`
- `rowUpdated ()`
- `setFetchSize (int)`
- `updateXXX` - Not all of the updates methods are supported

ResultSetMetaData - *Fully Supported*

Statement - *Partially Supported*

Methods ***not*** supported include:

- `cancel ()` - Asynchronous execution is not supported
- `getFetchSize ()`
- `getQueryTimeout ()`
- `setEscapeProcessing (boolean)`
- `setFetchSize (int)`

XAConnection - *Fully Supported*

XADataSource - *Fully Supported*

XAResource - *Fully Supported*

Xid - *Fully Supported*