



DataDirect[®]

SequeLink[®]

Developer's Reference

© 2000 MERANT. All rights reserved. Printed in the U.S.A.

DataDirect, INTERSOLV, MicroFocus, Middleware, Net Express, PVCS, SequeLink, and TechGnosis are registered trademarks, and Client/Server MiddleWare, DataDirect Connect ADO, DataDirect Connect Integrator, DataDirect Connect JDBC, DataDirect Connect ODBC, DataDirect Connect OLE DB, DataDirect Connect Premium, DataDirect Reflector, DataDirect SequeLink Integrator, MERANT, PVCS Dimensions, MERANT, PVCS Metrics, PVCS Replicator, PVCS TeamLink, PVCS Tracker, PVCS TrackerLink, PVCS Version Manager, PVCS VM Server, and WebDBLink are trademarks of MERANT. All other trademarks are the property of their respective owners.

ACKNOWLEDGEMENT. PVCS® Dimensions™ is implemented using the ORACLE® Relational database management system. ORACLE is a registered trademark of Oracle Corporation, Redwood City, California.

Portions created by Netscape are Copyright (C) 1998-1999 Netscape Communications Corporation. All Rights Reserved.

Portions created by Eric Young are Copyright (C) 1995-1998 Eric Young. All Rights Reserved.

No part of this publication, with the exception of the software product user documentation contained on a CD-ROM, may be copied, photocopied, reproduced, transmitted, transcribed, or reduced to any electronic medium or machine-readable form without prior written consent of MERANT.

Licensees may duplicate the software product user documentation contained on a CD-ROM, but only to the extent necessary to support the users authorized access to the software under the license agreement. Any reproduction of the documentation, regardless of whether the documentation is reproduced in whole or in part, must be accompanied by this copyright statement in its entirety, without modification.

U.S. GOVERNMENT RESTRICTED RIGHTS. It is acknowledged that the Software and the Documentation were developed at private expense, that no part is in the public domain, and that the Software and Documentation are Commercial Computer Software provided with RESTRICTED RIGHTS under Federal Acquisition Regulations and agency supplements to them. Use, duplication or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of The Rights in Technical Data and Computer Software clause at DFAR 252.227-7013 et. seq. or subparagraphs (c)(1) and (2) of the Commercial Computer Software Restricted Rights at FAR 52.227-19, as applicable. Contractor is MERANT, 9420 Key West Avenue, Rockville, Maryland 20850. Rights are reserved under copyright laws of the United States with respect to unpublished portions of the Software.

MERANT
9420 Key West Avenue
Rockville, Maryland 20850

Table of Contents

List of Tables	11
Preface	15
What Is DataDirect SequeLink?	15
Using This Book	15
Other SequeLink Documentation	17
Conventions Used in This Book	18
Typographical Conventions	18
Mouse Conventions	19
Keyboard Conventions	20
Environment-Specific Information	20
Ordering Printed Books	21
Contacting Technical Support	23
 Part 1: Developing ODBC Applications	
1 Using the SequeLink ODBC Client	27
About the SequeLink ODBC Client	27
Using the ODBC Administrator	28
Configuring ODBC Client Data Sources on Windows	28
Configuring ODBC User and System Client Data Sources	28
Configuring ODBC File Client Data Sources	32
ODBC Connection Dialogs	36
Testing ODBC Connections on Windows	42

Configuring ODBC Client Data Sources on UNIX	43
Configuring odbc.ini Files	43
Example: odbc.ini for Solaris	43
Setting Environment Variables	44
Using a Centralized odbc.ini File	45
Connecting Using a Connection String	45
ODBC Connection Attributes	46
2 Developing ODBC Applications	57
Required ODBC Libraries and Header Files	58
UNIX Compiler Requirements	59
ODBC API Functions	60
SQL Escape Sequences	63
Data Types and Isolation Levels	63
Threading	63
Threading Architecture	63
Cancelling Functions in Multithreaded Applications	64
Using Scrollable Cursors	66
Static and Keyset-Driven Cursors	66
Using Static Scrollable Cursors	67
Using Keyset-Driven Scrollable Cursors	67
Using Stored Procedures with Oracle	68
Specifying Application IDs	72
Specifying Application IDs Explicitly	72
Generating Application IDs Automatically	73
Error Handling	74
SequeLink ODBC Driver Errors	74
SequeLink Client Errors	75
SequeLink Server Errors	75
Database Errors	76

Developing Performance-Optimized ODBC Applications	77
Catalog Functions	77
Retrieving Data	83
ODBC Function Selection.	87
Designing ODBC Applications.	90
Updating Data	92

Part 2: Developing ADO Applications

3 Using the SequeLink ADO Client	97
About the SequeLink ADO Client.	97
Using the DataDirect Configuration Manager	98
Working with the DataDirect Configuration Manager. . .	100
Displaying Data Source Properties	101
Configuring ADO Client Data Sources	103
Creating an ADO Client Data Source	103
Modifying an ADO Client Data Source.	106
Renaming an ADO Client Data Source.	106
Deleting an ADO Client Data Source	107
Changing Data Source Directories	107
Copying an ADO Client Data Source	108
Testing ADO Connections	109
ADO Connection Dialogs.	110
Connecting with a Provider String	117
ADO Connection Attributes	118
4 Developing ADO Applications	123
OLE DB Objects and Interfaces	124
ErrorLookup	126
TCommand.	126
TDataSource.	127
TEnumerator	128
TMultipleResults	129
TRowset	129

TSession	130
TTransaction	132
TTransactionOptions.	132
Supported Schema Rowsets.	133
Supported OLE DB Property Groups	134
Data Source Property Group	135
Data Source Information Property Group.	135
Initialization Property Group.	148
Rowset Property Group	151
Session Property Group	162
OLE DB Interfaces Supported in ADO	162
Mapping ADO Methods and Properties	164
ADO Command Object.	164
Connection Object	172
Field Object.	184
Parameter Object	185
Property Object	186
Recordset Object	188
Data Shaping	197
Persisting Information	198
Using Rowsets.	198
Unicode Support.	199
Mapping Data Types	199
Specifying Application IDs	200
Specifying Application IDs Explicitly	200
Generating Application IDs Automatically.	200
Error Handling	201
SequeLink ADO Provider Errors.	201
SequeLink Client Errors	201
SequeLink Server Errors	202
Database Errors	202

Part 3: Developing JDBC Applications

5	Using the SequeLink Java Client	207
	About the SequeLink Java Client	207
	SequeLink JDBC Driver	208
	SequeLink Proxy Server	208
	JDBCSpy	209
	JDBCTest	210
	SequeLink Java Client Directory Structure	210
	Loading the SequeLink JDBC Driver	214
	Specifying SequeLink JDBC Driver Connection URLs	215
	Configuring JDBC Data Sources	217
	Creating and Managing JDBC Data Sources	218
	Using JNDI for Naming Databases	219
	Using Connection Pooling	219
	Using the Java Transaction API	221
	Specifying Connection Properties	223
	Using Connection URLs or the JDBC Driver Manager	223
	Using JDBC Data Sources	223
	JDBC Connection Properties	224
	Testing SequeLink JDBC Connections	229
	Using the SequeLink Java Client on a Java 2 Platform	229
6	Using JDBCTest	233
	About JDBCTest	233
	JDBCTest Tutorial	234
	Starting JDBCTest	234
	Configuring JDBCTest	236
	Connecting Using JDBCTest	236
	Executing a Simple Select Statement	239
	Executing a Prepared Statement	241

Retrieving Database Metadata	245
Scrolling Through a Result Set.	248
Batch Execution on a Prepared Statement.	251
Using Stored Procedures With Oracle.	254
7 Tracking JDBC Calls	261
About Spy	261
Loading the Spy JDBC Driver	262
Spy URL Syntax and Spy Attributes	263
Using Spy with JDBC Data Sources	264
Spy URL Examples.	265
Spy Log Example.	266
8 Developing JDBC Applications	269
JDBC 1.22 Support	270
JDBC 2.0 Support	271
JDBC 2.0 Optional Package Support	273
JDBC Compatibility.	274
SQL Escape Sequences	274
Data Types and Isolation Levels.	275
Threading	275
Threading Architecture	275
Cancelling Functions in Multithreaded Applications.	276
Using Scrollable Cursors.	277
Result Set Types	277
Concurrency Types	278
Using Scrollable Cursors.	279
Specifying Application IDs	280

Error Handling	281
SequeLink JDBC Driver Errors	281
SequeLink Server Errors	282
Database Errors	282
Fine-Tuning JDBC Application Performance	283
Reducing Download Time	283
Improving Character Set Conversion Performance	285
Fetching BigDecimal Objects	286

Part 4: Reference

A SQL Escape Sequences for ODBC and JDBC	289
Date, Time, and Timestamp Escape Sequences	290
Scalar Functions	290
String Functions	298
Numeric Functions	300
Date and Time Functions	302
System Functions	305
Like Predicate Escape Characters	305
Outer Join Escape Sequences	306
Procedure Call Escape Sequences	307
B Data Types and Isolation Levels	309
Supported Data Types	309
DB2 V4, V5	309
DB2 V6, V7	310
Informix 7	311
Informix 9	312
Microsoft SQL Server 7	314
Microsoft SQL Server2000	315
Oracle7	317
Oracle8	317
Sybase	319

10 Table of Contents

Isolation Levels	320
Index	321

List of Tables

Table 1-1.	ODBC Attributes	47
Table 2-1.	Sources for Required ODBC Development Tools.	58
Table 2-2.	Compiler Requirements for UNIX	59
Table 2-3.	Conformance of Supported Functions for 2.x ODBC Applications . .	60
Table 2-4.	Function Conformance for 3.x ODBC Applications.	62
Table 2-5.	Multithreading Functionality of the SequeLink ODBC Driver	64
Table 2-6.	Using SQLCancel in Multithreaded Applications	65
Table 2-7.	Support for Scrollable Cursors (ODBC)	66
Table 3-1.	DataDirect Configuration Manager: Parts and Functions for SequeLink ADO Data Sources.	100
Table 3-2.	ADO Connection Attributes	118
Table 4-1.	Objects and Interfaces Supported by the SequeLink ADO Provider	124
Table 4-2.	OLE DB Schema Rowsets Supported by the SequeLink ADO Provider	133
Table 4-3.	OLE DB Property Groups Supported by the SequeLink ADO Provider	134
Table 4-4.	OLE DB Data Source Property Supported by the SequeLink ADO Provider	135
Table 4-5.	OLE DB Data Source Information Properties Supported by the SequeLink ADO Provider	136
Table 4-6.	Initialization Properties Supported by the SequeLink ADO Provider	148
Table 4-7.	Rowset Properties Supported by the SequeLink ADO Provider.	152
Table 4-8.	Session Properties Supported by the SequeLink ADO Provider.	162

Table 4-9.	Supported OLE DB Interfaces Used by ADO	162
Table 4-10.	Mapping Built-in Methods of the ADO Command Object.	164
Table 4-11.	Dynamic Properties Used for the ADO Command Object	165
Table 4-12.	Mapping Collections for the ADO Command Object.	171
Table 4-13.	Mapping Methods Supported by the ADO Connection Object.	172
Table 4-14.	Mapping Built-in Properties Used by the ADO Connection Object	172
Table 4-15.	Dynamic Properties Supported for the ADO Connection Object.	173
Table 4-16.	Mapping Collections for the ADO Connection Object.	183
Table 4-17.	Mapping Built-in Properties Supported by the ADO Field Object	184
Table 4-18.	Mapping Built-in Properties Supported by the ADO Parameter Object	185
Table 4-19.	Mapping ADO Built-in Properties Supported for the Property Object	187
Table 4-20.	Mapping Methods Supported by the Recordset Object	188
Table 4-21.	Mapping Built-in Properties for the ADO Recordset Object	189
Table 4-22.	Dynamic Properties Used for the Recordset Object	191
Table 4-23.	Mapping Collections for the ADO Recordset Object	197
Table 5-1.	SequeLink Java Client Directory and Files.	210
Table 5-2.	Support for the Java Transaction API (JTA) by the SequeLink Java Client	221
Table 5-3.	JDBC Properties.	224
Table 6-1.	Starting JDBCTest with Java Virtual Machines Other Than the JDK.	235
Table 8-1.	JDBC 1.22 Functionality Supported by the SequeLink JDBC Driver.	270
Table 8-2.	JDBC 2.0 Functionality Supported by the SequeLink JDBC Driver	271
Table 8-3.	JDBC 2.0 Optional Package Functionality Supported by the SequeLink JDBC Driver	273

Table 8-4.	JDBC Compatibility	274
Table 8-5.	Using Cancel in Multithreaded Applications.	276
Table 8-6.	Support for Scrollable Cursors (JDBC)	278
Table A-1.	Scalar Functions Supported by the SequeLink ODBC Driver and SequeLink JDBC Driver	291
Table A-2.	Scalar String Functions	298
Table A-3.	Scalar Numeric Functions	301
Table A-4.	Scalar Time and Date Functions	303
Table A-5.	Scalar System Functions	305
Table A-6.	Outer Join Escape Sequences Supported by the SequeLink ODBC Driver and SequeLink JDBC Driver	306
Table B-1.	Data Types (DB2 V4, V5)	309
Table B-2.	Data Types (DB2 V6, V7)	310
Table B-3.	Data Types (Informix 7)	311
Table B-4.	Data Types (Informix 9)	312
Table B-5.	Data Types (Microsoft SQL Server 7)	314
Table B-6.	Data Types (Microsoft SQL Server2000).	315
Table B-7.	Data Types (Oracle7).	317
Table B-8.	Data Types (Oracle8).	317
Table B-9.	Data Types (Sybase 11 and 12).	319
Table B-10.	Isolation Levels	320

Preface

This book is your guide to developing client applications for your MERANT™ DataDirect® SequeLink® 5.1 environment. Read on to find out more about developing client applications to run in a SequeLink environment and how to use this book.

What Is DataDirect SequeLink?

DataDirect SequeLink is a middleware product that provides point-to-point connections from client to server for the latest data access standards, including Open Database Connectivity (ODBC), Java Database Connectivity (JDBC) applications, and ActiveX Data Objects (ADO) applications.

Using This Book

This book assumes that you are familiar with your operating system and its commands; the definition of directories; and accessing a database through an end-user application.

This book contains the following information:

Part 1: Developing ODBC Applications

- [Chapter 1 “Using the SequeLink ODBC Client” on page 27](#) provides information about using ODBC applications with the SequeLink ODBC Client.

- [Chapter 2 “Developing ODBC Applications” on page 57](#) provides information about developing ODBC applications for SequeLink environments.

Part 2: Developing ADO Applications

- [Chapter 3 “Using the SequeLink ADO Client” on page 97](#) provides information about using ADO applications with the SequeLink ADO Client.
- [Chapter 4 “Developing ADO Applications” on page 123](#) provides information about developing ADO applications for SequeLink environments.

Part 3: Developing JDBC Applications

- [Chapter 5 “Using the SequeLink Java Client” on page 207](#) provides information about using JDBC applications with the SequeLink Java Client.
- [Chapter 6 “Using JDBCtest” on page 233](#) introduces JDBCtest, a tool that allows you to test and learn the JDBC API, and contains a tutorial that takes you through a working example of its use.
- [Chapter 7 “Tracking JDBC Calls” on page 261](#) introduces Spy, a tool that allows you to track JDBC calls, and describes how to use it.
- [Chapter 8 “Developing JDBC Applications” on page 269](#) provides information about developing JDBC applications for SequeLink environments.

Part 4: Appendixes

- [Appendix A “SQL Escape Sequences for ODBC and JDBC” on page 289](#) describes the scalar functions supported for SequeLink. Your data store may not support all these functions.

- [Appendix B “Data Types and Isolation Levels” on page 309](#) lists the data types and isolation levels supported for each data store supported by SequeLink.

NOTE: This book refers the reader to Web URLs for more information about specific topics, including Web URLs not maintained by MERANT. Because it is the nature of Web content to change frequently, MERANT can guarantee only that the URLs referenced in this book were correct at the time of publishing.

Other SequeLink Documentation

The following documentation is provided on your SequeLink CD in PDF format. You can view the online documentation on the CD using the Acrobat Reader.

The following table provides a guide for finding information in your SequeLink documentation.

For information about...	Go to...
SequeLink concepts and planning your SequeLink environment	<i>Getting Started with SequeLink</i>
Installing the SequeLink middleware components	<i>SequeLink Installation Guide</i>
Administering your SequeLink environment	<i>SequeLink Administrator's Guide</i>
Developing ODBC, ADO, and JDBC applications for the SequeLink environment	<i>SequeLink Developer's Reference</i>
Troubleshooting and referencing error messages	<i>SequeLink Troubleshooting Guide and Reference</i>

DataDirect product documentation is also available in PDF and HTML formats on the MERANT DataDirect Web site:

<http://www.merant.com/products/datadirect/download/docs/dochome.asp>

Conventions Used in This Book

This section describes the typography, terminology, and other conventions used in this book.

Typographical Conventions

This book uses the following typographical conventions:

Convention	Explanation
<i>italics</i>	Introduces new terms that you may not be familiar with, and is used occasionally for emphasis.
bold	Emphasizes important information. Also indicates button, menu, and icon names on which you can act. For example, click Next .
UPPERCASE	Indicates the name of a file. For operating environments that use case-sensitive filenames, the correct capitalization is used in information specific to those environments. Also indicates keys or key combinations that you can use. For example, press the ENTER key.
monospace	Indicates syntax examples, values that you specify, or results that you receive.
<i>monospaced italics</i>	Indicates names that are placeholders for values you specify; for example, <i>filename</i> .

Convention	Explanation
forward slash /	Separates menus and their associated commands. For example, Select File / Copy means to select Copy from the File menu.
vertical rule	Indicates an OR separator to delineate items.
brackets []	Indicates optional items. For example, in the following statement: SELECT [DISTINCT], DISTINCT is an optional keyword.
braces { }	Indicates that you must select one item. For example, {yes no} means you must specify either yes or no.
ellipsis . . .	Indicates that the immediately preceding item can be repeated any number of times in succession. An ellipsis following a closing bracket indicates that all information in that unit can be repeated.

Mouse Conventions





This action...	Means to...
Click	Point to an object with the mouse pointer and press the left mouse button.
Double-click	Click the left mouse button twice.
Right-click	Click the right mouse button.
Drag	Press and hold the left mouse button while dragging item(s) to another part of the screen.
SHIFT+Click	Press and hold the SHIFT key; then click a selection. This lets you select a series of objects.
CTRL+Click	Press and hold the CTRL key; then click a selection. This lets you select objects randomly.

Keyboard Conventions

Select menu items by using the mouse or pressing ALT+ the key letter of the menu name or item.

Environment-Specific Information

This book supports users of various operating environments. Where it provides information that does not apply to all supported environments, the following symbols are used to identify that information.

Symbol	Environment
	<i>Windows</i> . Information specific to the Microsoft Windows 95, Windows 98, Windows NT, and Windows 2000 environments is identified by the Windows symbol.
	<i>Windows NT</i> . Information specific to the Microsoft Windows NT environment is identified by the Windows symbol and the letters “NT.”
	Windows 2000. Information specific to the Microsoft Windows 2000 environment is identified by the Windows symbol and the number “2000”.
	<i>UNIX</i> . Information specific to UNIX environments is identified by this symbol, which applies to all supported UNIX environments. UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Ltd.
OS/390	<i>OS/390</i> . Information specific to OS/390 environments is identified by the letters OS/390.

Ordering Printed Books

As part of your SequeLink license agreement, you may print and distribute as many copies of the SequeLink books as needed.

If you do not want to print each of these online books, you can order hard-copy versions from MERANT. To order, please complete the following order form and fax your request to MERANT at (919) 461-4526.

Order Form

Fax your request to MERANT at (919) 461-4526. The cost of shipping will be added to your order.

SequeLink Key: _____

Ordered By: _____ Phone: (____) ____ - _____

Company Name: _____

Mailing Address: _____

City: _____ State: _____ Zip: _____

Credit Card: Master Card[®] VISA[®] Discover[®] American Express[®]

Credit Card Number:

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

Expiration:

--	--

--	--

 Signature: _____
Mo Yr

SequeLink Book Title	Price	Quantity	Total
Documentation Set	\$100.00		
<i>Getting Started with SequeLink</i>	\$35.00		
<i>SequeLink Installation Guide</i>	\$35.00		
<i>SequeLink Administrator's Guide</i>	\$35.00		
<i>SequeLink Developer's Reference</i>	\$35.00		
<i>SequeLink Troubleshooting Guide and Reference</i>	\$35.00		

Shipping: Orders are shipped via Standard Airborne delivery from our Rockville Distribution Center. Items should arrive within 5 business days of receipt of order.

Contacting Technical Support

MERANT provides technical support for all registered users of SequeLink, including limited installation support, for the first 30 days. For support after that time, contact us using one of the following methods or purchase further support by enrolling in the SupportNet program. For more information about SupportNet, contact your sales representative.

The MERANT Web site provides the latest support information through SupportNet Online, our global service network that provides access to valuable tools and information. Our SupportNet users access information using the Web, automatic email notification, newsgroups, and regional user groups. SupportNet Online includes a knowledge base that allows you to search on keywords for technical bulletins and other information. You also can download product fixes for your DataDirect products.

World Wide Web

<http://support.merant.com>

E-Mail

USA, Canada, and Mexico

datadirect.answerline@merant.com

Australia and New Zealand

australia.answerline@merant.com

Japan

jpn.answerline@merant.co.jp

All other countries

int.datadirect.answerline@merant.com

Local Telephone Support

Local phone numbers can be found at:

<http://support.merant.com/websupport/contact/supportnetanswerline.asp>

Live Answerline telephone support is available 24 hours a day, seven days a week.

Fax and Mail Information

Fax US	1 919 461 4527
Fax International	+32 (0) 15 32 09 19
Mail	1500 Perimeter Park Drive, Suite 100, Morrisville, NC 27560 USA

When you contact us, please provide the following information:

- The **product serial number** located on the Product Registration Information card or on a product serial number card in your package. The number will be checked to verify your support eligibility. If you do not have a SupportNet contract, we will ask you to speak with a sales representative.
- Your **name and organization**. For a first-time call, you may be asked for full customer information, including location and contact details.
- The **version number** of your DataDirect product.
- The type and version of your **operating system**.
- Any **third-party software or other environment information** required to understand the problem.
- A **brief description of the problem**, including any error messages that you have received, **and the steps preceding the occurrence of the problem**. Depending on the complexity of the problem, you may be asked to submit an example so that we can recreate the problem.
- An assessment of the **severity level** of the problem.

Part 1: Developing ODBC Applications

This part contains the following chapters:

- [Chapter 1 “Using the SequeLink ODBC Client” on page 27](#) provides information about using ODBC applications with the SequeLink ODBC Client.
- [Chapter 2 “Developing ODBC Applications” on page 57](#) provides information about developing ODBC applications for SequeLink environments.

1 Using the SequeLink ODBC Client

This chapter provides information about using ODBC applications with the SequeLink ODBC Client.

About the SequeLink ODBC Client

The SequeLink ODBC Client supports ODBC applications through a component called the *SequeLink ODBC Driver*. On Windows and UNIX platforms, the SequeLink ODBC Driver is compliant with the Microsoft Open Database Connectivity (ODBC) 3.5 specification.

ODBC is an Application Program Interface (API) specification that allows applications to access multiple database systems using Structured Query Language (SQL). ODBC provides maximum interoperability—a single application can access many different database systems. This allows an ODBC developer to develop, compile, and ship an application, without targeting a specific type of data source. Users can then add the database drivers, which link the application to the database systems of their choice. The SequeLink ODBC Driver can connect all commercial ODBC-compliant applications with server databases.

For instructions on installing the SequeLink ODBC Client, refer to the *SequeLink Installation Guide*.

Using the ODBC Administrator



The first step in setting up an ODBC connection is creating an ODBC data source. The ODBC Administrator is installed automatically when you install the SequeLink ODBC Client on Windows. You use the ODBC Administrator to create and manage ODBC data sources.

To start the ODBC Administrator, click **Start**, then **Programs**. From the Programs menu, select **SequeLink ODBC Client 5.1**, and then select the **ODBC Administrator** application. The ODBC Data Source Administrator window appears listing resident data sources.

NOTE: An ODBC Administrator does not exist for UNIX; you must edit the `odbc.ini` file using a text editor. For instructions on creating ODBC client data sources for UNIX, see [“Configuring ODBC Client Data Sources on UNIX” on page 43](#).

Configuring ODBC Client Data Sources on Windows

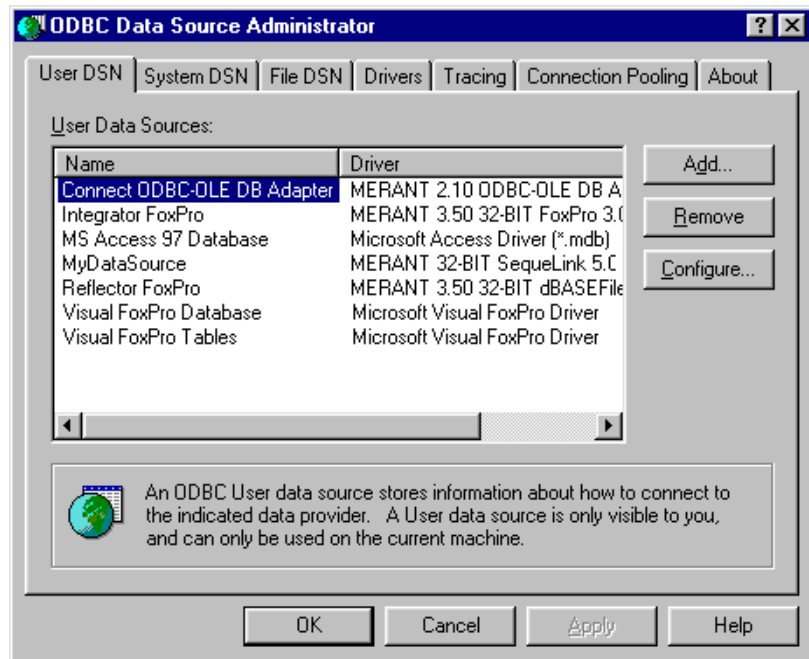


To configure client data sources for the SequeLink ODBC Client on Windows platforms, you use the ODBC Administrator.

Configuring ODBC User and System Client Data Sources

- 1 Start the ODBC Administrator. To start the ODBC Administrator, select **Start / Programs**. From the Programs menu, select **SequeLink ODBC Client 5.1**, and then select the **ODBC Administrator** application.

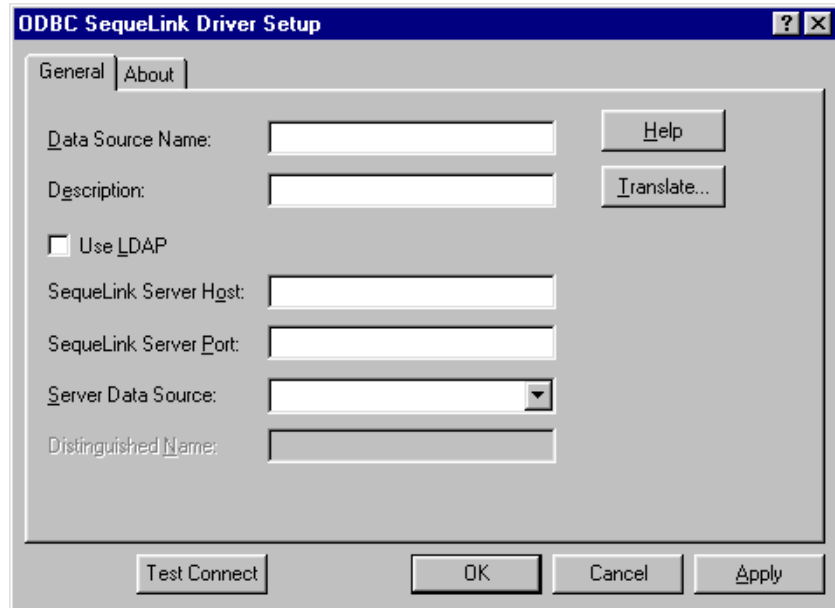
Click the **User DSN** tab or the **System DSN** tab to list user or system data sources, respectively.



- 2 To configure a new data source, click the **Add** button. A list of installed drivers appears. Select **MERANT 32-BIT SequeLink 5.1**; then, click **Finish**.

NOTE: To change an existing data source, select the data source you want to configure and click the **Configure** button.

The ODBC SequeLink Driver Setup window appears.



- 3 Provide the following information; then, click **OK**.

Data Source Name: Type a unique name that identifies this ODBC data source configuration. Examples include “Accounting” or “SequeLink to Oracle Data.”

Description: Optionally, type a description of the data source. For example, “My Accounting Database” or “Accounting Data in Oracle.”

SequeLink Server Host: Type the TCP/IP host name of the SequeLink service to which you want the SequeLink ODBC Client to connect.

SequeLink Server Port: Type the TCP/IP port the SequeLink service is listening on for incoming connection requests. The port you specify must be the same as the one that was specified for the SequeLink service when the SequeLink Server was installed; the default is 19996.

Server Data Source: Type the name of a server data source configured for the SequeLink service to use for the connection or select one from the drop-down list. This field is optional. If a server data source is not specified, the default server data source for that SequeLink service will be used for the connection.

Translate: Click **Translate** to select a translator. The Select Translator dialog box appears, listing the translators specified in the ODBC Translators section of the system information. SequeLink provides a translator named "OEM TO ANSI" that translates your data from the IBM PC character set to the ANSI character set. Select a translator; then, click **OK** to close this dialog box and perform the translation.

NOTE FOR LDAP USERS: To configure the SequeLink ODBC Client to retrieve connection information from a LDAP directory, select the **Use LDAP** check box. The fields change on the lower half of the screen to accommodate the information that is required to query a LDAP server for connection information. Provide the following information:

LDAP Server Host: Type the TCP/IP host name of the LDAP server.

LDAP Server Port: Type the TCP/IP port the LDAP server is listening on for incoming connection requests. If unspecified, the SequeLink ODBC Client will use the default LDAP port 389.

Distinguished Name (DN): Type an identifier that uniquely identifies the LDAP entry where connection information is stored.

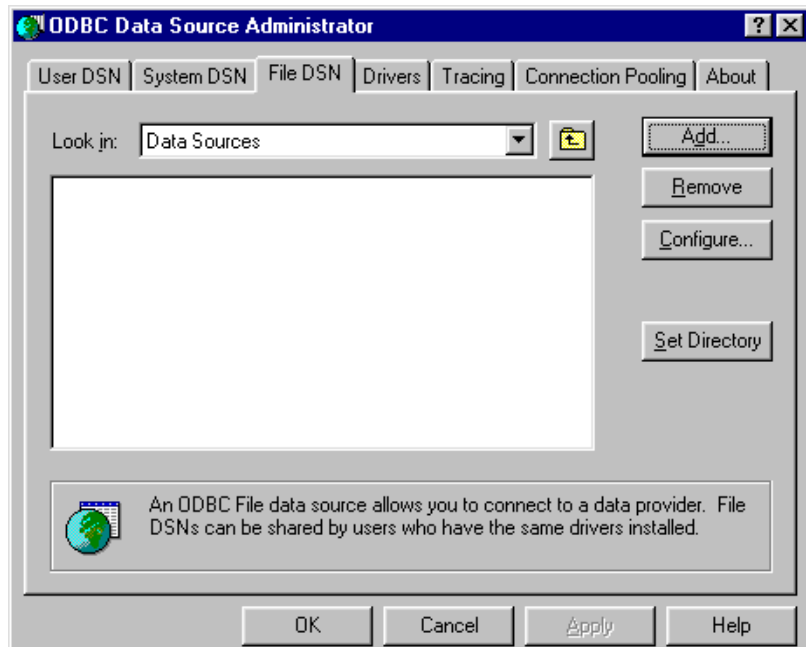
For more information about retrieving connection information from LDAP directories, refer to the *SequeLink Administrator's Guide*.

Configuring ODBC File Client Data Sources

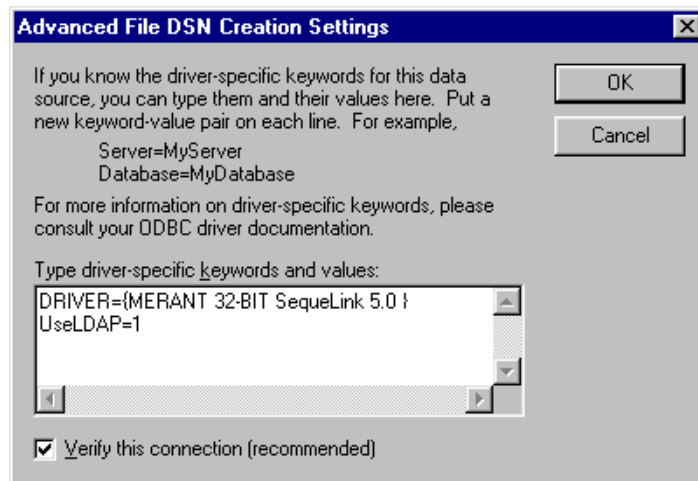
File data sources are data source files that can be stored on a file server, making the files available to any user who can access them.

To configure ODBC file client data sources:

- 1 Start the ODBC Administrator by clicking **Start**, then **Programs**. From the Programs menu, select **Sequelink ODBC Client 5.1**, and then select the **ODBC Administrator** application.
- 2 Click the **File DSN** tab. The File DSN tab lists any file data sources in the specified directory.

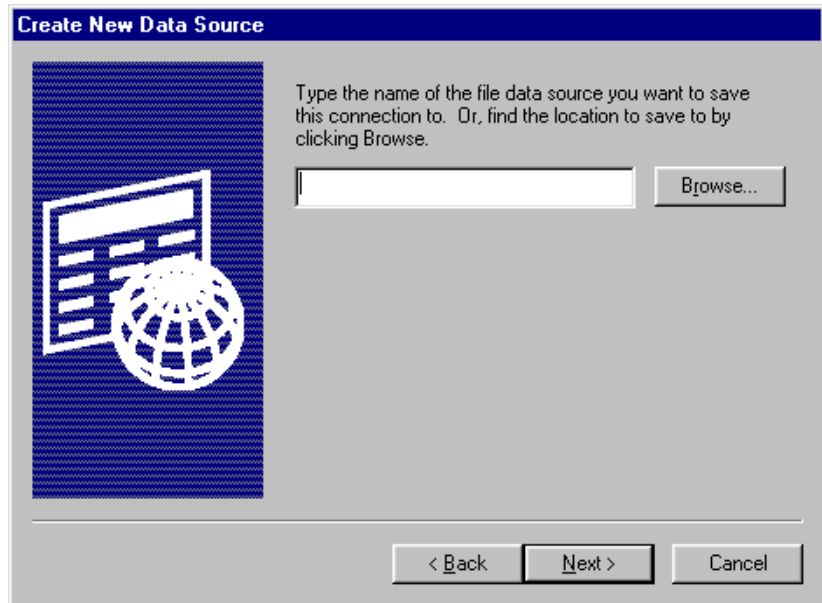


- 3 To configure a new data source, click the **Add** button. A list of installed drivers appears. Select **MERANT 32-BIT SequeLink 5.1**; then, perform one of the following actions:
 - To configure the file data source to connect directly to a SequeLink Server without retrieving connection information from an LDAP directory, click **OK**. Then, skip to [Step 5](#).
 - To configure the file data source to retrieve connection information from an LDAP directory, continue with the next step.
- 4 Click **Advanced**. The Advanced File DSN Creation Settings window appears.



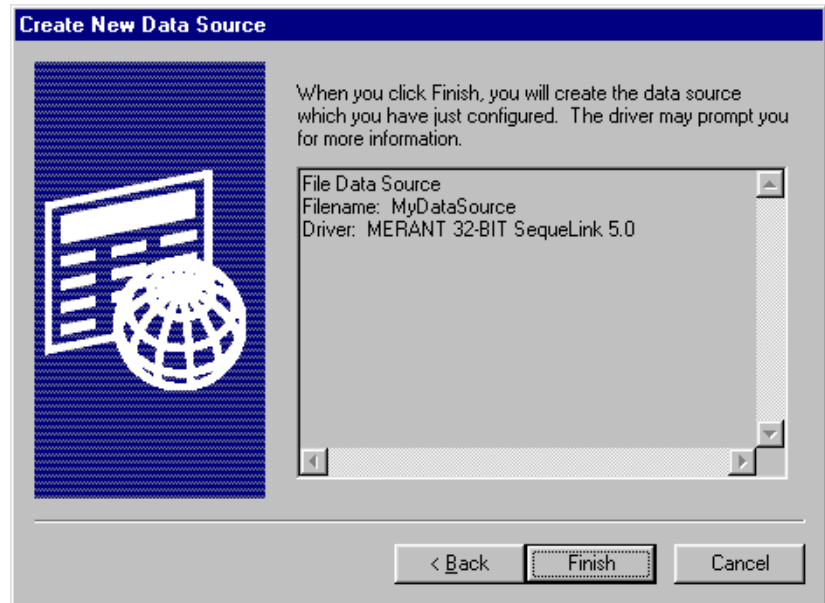
Type `UserLDAP=1` in the Type driver-specific keywords and values scrollable box; then, click **OK**. You are returned to the list of drivers. Click **Next** and continue with [Step 5](#).

- 5 The Create New Data Source window appears.



Type the name of the file data source you want to create or click **Browse** to select an existing file data source; then, click **Next**.

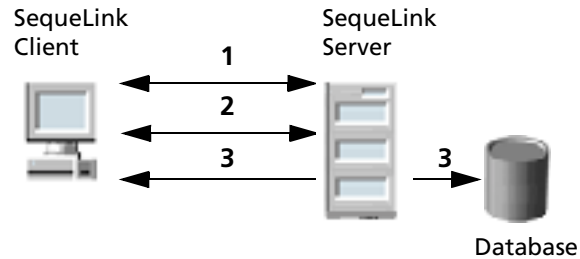
- 6 The Create New Data Source displays the settings you've configured for this data source.



- 7 Click **Finish** to create the file data source. A series of connection dialogs appear as described in [“ODBC Connection Dialogs” on page 36](#). The file data source will be saved after you enter the correct information in the connection dialogs.

ODBC Connection Dialogs

A SequeLink data access connection involves the following stages:



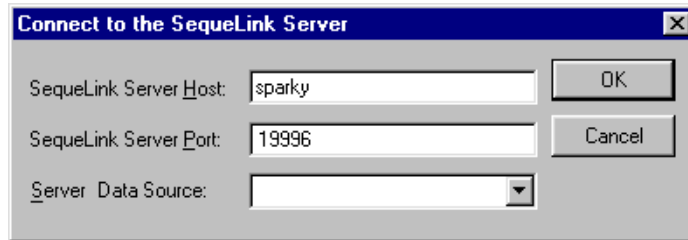
- 1 A network connection is established.
- 2 An authentication mechanism is used to establish the identity of the SequeLink Client to the SequeLink Server.
- 3 Based on information provided by the SequeLink Client application (for example, a database user name and password), a database connection is established.

Stage 1: Establishing a Network Connection

The first stage of the connection process involves establishing a network connection. The dialog that appears depends on whether the connection has been configured to connect directly to a SequeLink service or to retrieve connection information for the SequeLink service from a centralized LDAP directory.

Connecting Directly to a SequeLink Service

If the connection has been configured to connect directly to a SequeLink service, the Connect to the SequeLink Server dialog appears:



The screenshot shows a dialog box titled "Connect to the SequeLink Server". It has a standard Windows-style title bar with a close button (X) in the top right corner. The dialog contains three input fields on the left and two buttons on the right. The first input field is labeled "SequeLink Server Host:" and contains the text "sparky". The second input field is labeled "SequeLink Server Port:" and contains the text "19996". The third input field is labeled "Server Data Source:" and is a drop-down menu with a small downward-pointing arrow on its right side. To the right of the first two input fields is an "OK" button, and to the right of the second and third input fields is a "Cancel" button.

Provide the following information; then, click **OK**:

SequeLink Server Host: Type the TCP/IP host name of the SequeLink service.

SequeLink Server Port: Type the TCP/IP port on which the SequeLink service is listening. A default installation of SequeLink Server uses the port 19996.

Server Data Source: Type the name of a server data source to use for the connection or select one from the drop-down list. This step is optional. If a server data source is not specified, the default server data source for that service will be used for the connection.

Retrieving Connection Information from an LDAP Directory

If the connection has been configured to connect to an LDAP server to retrieve connection information from an LDAP directory, the Connect to the SequeLink Server dialog appears:



The screenshot shows a dialog box titled "Connect to the SequeLink Server". It has a standard Windows-style title bar with a close button (X) in the top right corner. The dialog contains three text input fields stacked vertically. The first field is labeled "LDAP Server Host" and contains the text "sparky". The second field is labeled "LDAP Server Port" and is empty. The third field is labeled "Distinguished Name" and is empty. To the right of the first field is an "OK" button, and to the right of the second field is a "Cancel" button.

Provide the following information; then, click **OK**:

LDAP Server Host: Type the TCP/IP host name of the LDAP server.

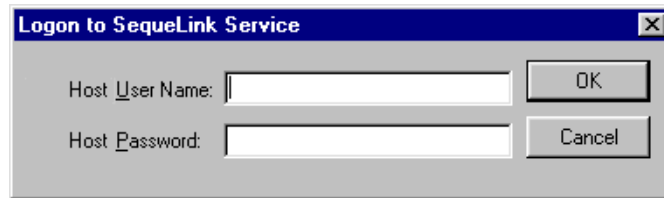
LDAP Server Port: Type the TCP/IP port on which the LDAP server is listening.

Distinguished Name: Type the Distinguished Name (DN) of the LDAP entry.

Stage 2: SequeLink Server Authentication

The second stage of the connection process involves authentication of the SequeLink Client to the SequeLink Server. The dialogs that appear depend on how authentication is configured for the SequeLink service.

- When ServiceAuthMethods=anonymous or ServiceAuthMethods=integrated_nt, no dialogs appear.
- When ServiceAuthMethods=OSLogon(HUID,HPWD) or ServiceAuthMethods=OSLogon(UID,PWD), the Logon to SequeLink Service dialog appears:



Provide the following information; then, click **OK**.

Host User Name: Type the host user name.

NOTE: When connecting to a Windows NT server, you must prefix the host user name with a server name, if authenticating to a local server, or a domain name (for example, SALES\DJONES). If the server name or domain name is omitted, the SequeLink Server will attempt to authenticate the user ID and password with the database account defined for the machine on which the SequeLink Server is running. If this validation fails, the SequeLink Server will attempt to authenticate the user ID and password with the database account defined for the domain of the machine on which the SequeLink Server is running.

Host Password: Type the host password.

- When ServiceAuthMethods=OSLogon(HUID,HPWD,NPWD) or ServiceAuthMethods=OSLogon(UID,PWD,NPWD) and the

password is expired, the Logon to SequeLink service dialog appears:

NOTE: If the password is not expired, the previously described dialog appears, prompting only for the host user name and host password.

Provide the following information; then, click **OK**.

Host User Name: Type the host user name.

NOTE: When connecting to a Windows NT server, you must prefix the host user name with a server name, if authenticating to a local server, or a domain name (for example, SALES\DJONES). If the server name or domain name is omitted, the SequeLink Server will attempt to authenticate the user ID and password with the database account defined for the machine on which the SequeLink Server is running. If this validation fails, the SequeLink Server will attempt to authenticate the user ID and password with the database account defined for the domain of the machine on which the SequeLink Server is running.

Host Password: Type the host password.

New Password: Type the new password to be used by the SequeLink password change mechanism.

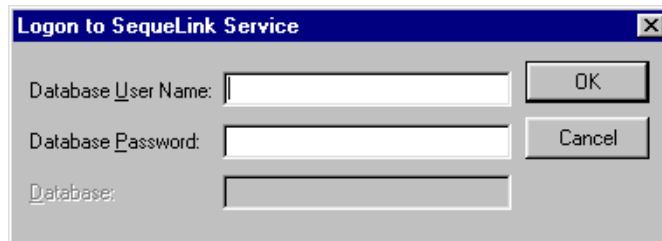
Confirm Password: Type again the new password to confirm it.

For more information about configuring authentication, refer to the *SequeLink Administrator's Guide*.

Stage 3: Data Store Logon

The last stage of the connection process involves logging on the data store. The dialogs that appear depend on the data store logon method configured for the SequeLink service:

- When `DataSourceLogonMethod=OSIntegrated`, no dialogs appear.
- When `DataSourceLogonMethod=DBMSLogon(UID,PWD)` or `DataSourceLogonMethod=DBMSLogon(DBUID,DBPWD)`, a data store-specific user name and password are required and the Logon to SequeLink Service dialog appears:



Provide the following information; then, click **OK**.

Database User Name: Type the database logon ID.

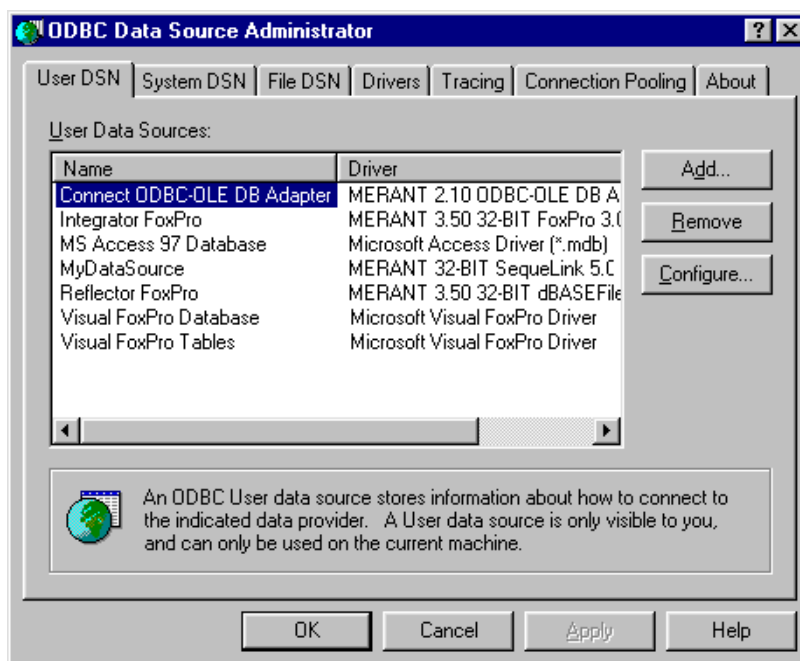
Database Password: Type the database password.

Database: Type the name of the database to which you want to connect. This field is disabled when the data store does not recognize the concept of databases.

For more information about configuring data store logon methods, refer to the *SequeLink Administrator's Guide*.

Testing ODBC Connections on Windows

- 1 On the SequeLink Client, start the ODBC Administrator. To start the ODBC Administrator, select **Start / Programs**. From the Programs menu, select **SequeLink ODBC Client 5.1**, and then select the **ODBC Administrator** application. The ODBC Data Source Administrator window appears listing resident data sources.



NOTE: An ODBC Administrator does not exist for UNIX. To create and manage ODBC data sources, you must edit the `odbc.ini` file using a text editor. For more information about the `odbc.ini`, see [“Configuring ODBC Client Data Sources on UNIX” on page 43](#).

- 2 Create an ODBC data source as described in [“Configuring ODBC Client Data Sources on Windows” on page 28](#), specifying the TCP/IP address and TCP/IP port of the SequeLink service.

- 3 Click the **Test Connect** button to test the connection. If successful, a dialog appears telling you the connection was successful. You are now ready to start using your ODBC applications with SequeLink.

Configuring ODBC Client Data Sources on UNIX



For UNIX, an ODBC Administrator does not exist. This section describes how to configure the `odbc.ini` file and how to set some required environment variables to use the SequeLink ODBC Client on UNIX.

Configuring `odbc.ini` Files

To configure an ODBC data source for UNIX, you must edit the `odbc.ini` file using the attributes in [Table 1-1, "ODBC Attributes," on page 47](#).

Example: `odbc.ini` for Solaris

The following code shows an example of an `odbc.ini` file for a SequeLink ODBC Client installed on a Solaris machine.

```
[ODBC Data Sources]
DataSourceName=MERANT 32-BIT SequeLink 5.1

[DataSourceName]
Driver=path_of_installdir/lib/ivslk14.so
Description=MERANT 32-BIT SequeLink 5.1
Host=
Port=
UseLDAP=0
```

```
DistinguishedName=
[ODBC]
Trace=0
TraceFile=odbctrace.out
TraceDll=path_of_installdir/lib/odbctrace.so
InstallDir=path_of_installdir
```

where *path_of_installdir* is the path to the SequeLink ODBC Client installation directory.

Setting Environment Variables

You must set several environment variables for the SequeLink ODBC Client on UNIX by executing a shell script located in the installation directory.

To execute the shell script:

- If you are using the Bourne or Korn shell, type:

```
. .sqlnk.sh
```

- If you are using the C shell, type:

```
source .sqlnk.csh
```

Executing this shell script sets the following environment variables:

ODBCINI	Specifies where the centralized <code>odbc.ini</code> file is located.
SQLNK_ODBC_HOME	Specifies the full path of the directory containing the SequeLink ODBC Client shared libraries.
LD_LIBRARY_PATH (Solaris)	Specifies the library search path so that the ODBC Driver Manager components and drivers can be located.
SHLIB_PATH (HP-UX)	
LIBPATH (AIX)	

Using a Centralized odbc.ini File

Because UNIX is a multi-user environment, you may want to use a single centralized odbc.ini file controlled by a system administrator. To do this, set the ODBCINI environment variable to point to the fully qualified pathname of the centralized file.

For example:

- In the Bourne or Korn shell:

```
ODBCINI=/opt/odbc/system_odbc.ini;export ODBCINI
```

- In the C shell:

```
setenv ODBCINI /opt/odbc/system_odbc.ini
```

The odbc.ini also requires a [ODBC] section that includes the InstallDir keyword. The value of the InstallDir keyword must be the path to the directory that contains the /lib and /messages directories. For example, if you choose the default installation directory, the following line must be in the [ODBC] section of the odbc.ini file:

```
InstallDir=/usr/slodbc51
```

Connecting Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which data source to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

You can specify long or short names in the connection string, which has the format:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

For example, a connection string for SequeLink may look like this:

```
DSN=Accounting;DB=EMP;UID=JOHN;PWD=XYZZY
```

or

```
DSN=Accounting;DB="X:IV;EMP";UID=JOHN;PWD=XYZZY
```

NOTE: If the database name (DB) contains a semicolon (;), you must place the name in quotes, as shown in the example above.

For a list of ODBC connection attributes and their valid values, see [“ODBC Connection Attributes” on page 46](#).

ODBC Connection Attributes

[Table 1-1, “ODBC Attributes,” on page 47](#) lists ODBC connection attributes in alphabetical order. The list includes long and short names and provides a description of each attribute. The defaults listed in [Table 1-1](#) are initial defaults that apply when no value is specified in the connection string or in the ODBC data source definition. If you specified a value for the attribute when

configuring the ODBC data source, that value is your default. In [Table 1-1](#), short names are shown enclosed within parentheses ().

Table 1-1. ODBC Attributes

Attribute	Description
ApplicationID (APPID)	<p>Specifies the application ID that identifies the client application to the SequeLink service. This attribute is only required when the SequeLink service you are connecting to has been configured to limit access to specific applications.</p> <p>For more information about using application IDs to limit access to SequeLink services, see “Specifying Application IDs” on page 72.</p>
AutomaticApplicationID (AUTOAPPID)	<p>Specifies an application ID that is automatically generated by the SequeLink ODBC Client to identify the client application to the SequeLink service. This attribute is only required when the SequeLink service you are connecting to has been configured to limit access to specific applications.</p> <p>For more information about using application IDs to limit access to SequeLink services, see “Specifying Application IDs” on page 72.</p>
BlockFetchForUpdate (BFFU)	<p>BlockFetchForUpdate={0 1}. Specifies a workaround connection attribute. When the isolation level is Read committed and a SELECT FOR UPDATE statement is issued against some data stores, the SequeLink ODBC Client does not lock the expected row.</p> <p>When set to 0, the appropriate row is locked.</p> <p>When set to 1 (the initial default), the appropriate row is not locked.</p> <p>NOTE: Specifying 0 will degrade the performance for SELECT FOR UPDATE statements because rows will be fetched one at a time.</p>
Database (DB)	<p>Specifies the name of the database to which you want to connect.</p>

Table 1-1. ODBC Attributes (cont.)

Attribute	Description
DSN (DSN)	Specifies a string that identifies an ODBC data source configuration. Examples include "Accounting" or "SequeLink to Oracle Data."
DBLogonID (DBUID)	Specifies the data store user name, which may be required depending on the server configuration.
DBPassword (DBPWD)	Specifies the data store password, which may be required depending on the server configuration.
DistinguishedName (DN)	Specifies the distinguished name identifying the LDAP entry from which connection information is retrieved. This attribute is required when UseLDAP=1.
EnableDescribeParam (EDP)	EnableDescribeParam={0 1}. Specifies a workaround connection attribute for connections to Oracle data stores only. When set to 0 (the initial default), support is turned off for SQLDescribeParam. When set to 1, support is turned on for SQLDescribeParam and will describe all parameters as SQL_CHAR with a precision of 999.

Table 1-1. ODBC Attributes (cont.)

Attribute	Description
FetchNextOnly (FNO)	<p>FetchNextOnly={TRUE FALSE}.</p> <p>Turns on a workaround for Visual Basic/Remote Data Objects (RDO) that circumvents a problem with FORWARD_ONLY cursors when the driver reports other values than FETCH_NEXT for SQLGetInfo(SQL_FETCH_DIRECTION).</p> <p>For example, if the driver only reports FETCH_NEXT, RDO performs SQLExecDirect, SQLBindCol, and SQLExtendedFetch(NEXT). If the driver supports more than FETCH_NEXT, RDO performs SQLExecDirect, SQLExtendedFetch(NEXT), and SQLGetData. This is only valid when the rowsize is 1, but RDO uses a larger rowsize in this situation.</p> <p>When set to TRUE, the driver will incorrectly report that only SQL_FETCH_NEXT is supported, which satisfies RDO.</p> <p>When set to FALSE (the initial default), the driver will correctly report other values than SQL_FETCH_NEXT.</p>
FixCharTrim (FCT)	<p>FixCharTrim={0 1}. Turns on a workaround for applications that have a problem using SQL_CHAR data padded with spaces. The SequeLink ODBC Driver returns SQL_CHAR data padded with spaces as mandated by the ODBC specification.</p> <p>When set to 0 (the initial default), the workaround is turned off.</p> <p>When set to 1, SQL_CHAR data that is not padded with spaces is returned.</p>

Table 1-1. ODBC Attributes (cont.)

Attribute	Description
GetOutputParams (GOP)	<p>Turns on a workaround that allows you to control when output parameters of stored procedures are returned to calling applications. This attribute uses a bitmask with the following options:</p> <p>When set to 1, output parameters are returned after an execute.</p> <p>When set to 2, output parameters are returned after a fetch is complete.</p> <p>When set to 4, output parameters are returned after moreresults no more rows.</p> <p>When set to 7 (the initial default), output parameters are returned after all of the above.</p> <p>The value for this connection attribute should be set to the cumulative value of all chosen options added together.</p> <p>NOTE: Set GetOutputParams=3 when executing stored procedures with output parameters in RDO (Visual Basic 5 and 6)</p>
HLogonID (HUID)	Specifies the host user name, which may be required depending on the server configuration.
HPassword (HPWD)	Specifies the host password, which may be required depending on the server configuration.
Host (HST)	<p>Specifies the TCP/IP address of the SequeLink Server, specified in dotted format or as a host name.</p> <p>LDAP: If LDAP is enabled, this attribute identifies the TCP/IP address of the LDAP server. This attribute can also be a list of LDAP servers separated by a blank space (for example, "ld1.foo.com ld2.foo.com ld3.foo.com"). If the first LDAP server in the list does not respond, the SequeLink ODBC Client will try to connect to the next LDAP server in the list.</p>

Table 1-1. ODBC Attributes (cont.)

Attribute	Description
LogonID (UID)	Specifies the host or data store user name, which may be required depending on the server configuration.
NewPassword (NPWD)	<p>Specifies the new host password to be used. If specified and applicable to the connection, the SequeLink password change mechanism is invoked. When the password has been changed successfully, the following warning is generated:</p> <pre data-bbox="696 597 1300 689">[MERANT] [ODBC SequeLink driver] [SequeLink Server] The user password was changed successfully</pre> <p>If unspecified and the SequeLink Server detects that the host password has expired, you will be prompted for a new host password.</p> <p>For more information about the SequeLink password change mechanism, refer to the <i>SequeLink Administrator's Guide</i>.</p>
Password (PWD)	Specifies the host or data store password, which may be required depending on the server configuration.
Port (PRT)	<p>Specifies the TCP/IP port on which the SequeLink Server is listening.</p> <p>LDAP: If LDAP is enabled, this attribute identifies the TCP/IP port on which the LDAP server is listening. If you do not specify a port, the default port for LDAP (389) will be used.</p>
ServerDataSource (SDSN)	Optionally, specifies a string that identifies the server data source to be used for the connection. If not specified, the configuration of the default server data source will be used for the connection.
SLKStaticCursorLongColBuffLen (SSCLCBL)	<p>Turns on a workaround that allows you to specify the amount of data (in KB) that is buffered for SQL_LONGVARCHAR and SQL_LONGVARBINARY columns with a static cursor.</p> <p>The default is 4.</p>

Table 1-1. ODBC Attributes (cont.)

Attribute	Description
UseLDAP (LDAP)	<p data-bbox="658 314 861 341">UseLDAP={0 1}.</p> <p data-bbox="658 359 1282 447">Determines whether the parameters to establish a connection to the SequeLink Server should be retrieved from LDAP.</p> <p data-bbox="658 465 1282 553">When set to 0 (the initial default), the SequeLink Client will connect directly to the specified SequeLink Server.</p> <p data-bbox="658 571 1282 822">When set to 1, the SequeLink Client will retrieve the TCP/IP host, TCP/IP port, and SequeLink server data source (optional) from an LDAP entry identified by a Distinguished Name (DN). Once the connection information is retrieved, the SequeLink Client will connect directly to the specified SequeLink Server. The DistinguishedName (DN) attribute is required.</p>
WorkArounds (WA)	<p data-bbox="658 840 1282 963">Turns on workarounds that allow you to take full advantage of the SequeLink ODBC Driver with ODBC applications requiring nonstandard or extended behavior.</p> <p data-bbox="658 980 1282 1102">IMPORTANT: Each of these options has potential side effects related to its use. An option should only be used to address the specific problem for which it was designed.</p>

Table 1-1. ODBC Attributes (cont.)

Attribute	Description
WorkArounds (cont.)	<p>When set to 1, the ODBC driver returns 1, allowing Microsoft Access to open tables as read-write. If an ODBC driver reports to Microsoft Access 2.0 that its SQL_CURSOR_COMMIT_BEHAVIOR or SQL_CURSOR_ROLLBACK_BEHAVIOR is 0, Microsoft Access opens tables as read-only.</p> <p>When set to 2, the driver reports that qualifiers are not supported. This option is provided because some applications cannot handle database qualifiers.</p> <p>When set to 4, the driver detects when Visual Basic requires multiple connections to a DBMS and has the multiple ODBC connections share a single physical connection to the DBMS. For DBMSs that support only a single connection, the second attempt fails.</p> <p>When set to 8, the driver returns 1. However, if an ODBC driver cannot detect the number of rows that are affected by an Insert, Update, or Delete statement, it may return -1 in SQLRowCount. Some products cannot handle this.</p> <p>When set to 16, the driver returns no INDEX_QUALIFIER, allowing Microsoft Access to open the table. If an ODBC driver in SQLStatistics reports to Microsoft Access 1.1 that an INDEX_QUALIFIER contains a period, Microsoft Access returns a tablename is not a valid name error.</p> <p>When set to 32, users of flat-file drivers are allowed to abort a long-running query by pressing the ESC key.</p>

Table 1-1. ODBC Attributes (cont.)

Attribute	Description
WorkArounds (cont.)	<p>When set to 64, the result is a column name of <i>Cposition</i> where <i>position</i> is the ordinal position in the result set. For example:</p> <pre data-bbox="658 427 1115 447">SELECT col1, col2+col3 FROM table1</pre> <p>produces the column names col1 and C2.</p> <p>SQLColAttributes/ SQL_COLUMN_NAME returns an empty string for result columns that are expressions. Use this option for applications that cannot handle empty strings in column names.</p> <p>When set to 256, SQLGetInfo/ SQL_ACTIVE_CONNECTIONS is forced to return as 1.</p> <p>When set to 512, the SQLSpecialColumns function returns a unique index as returned from SQLStatistics to prevent ROWID results.</p> <p>When set to 2048, SQLDriverConnect returns "Database=" instead of "DB=" in the returned connection string.</p> <p>When set to 65536, trailing zeros are stripped from decimal results, which prevents Microsoft Access from generating an error when decimal columns containing trailing zeros are included in the unique index.</p> <p>When set to 131072, all occurrences of the double quote character (") are turned into the accent grave character ('). Some applications always quote identifiers with double quotes. Double quoting causes problems for data sources that do not return SQLGetInfo/ SQL_IDENTIFIER_QUOTE_CHAR = ".</p>

Table 1-1. ODBC Attributes (cont.)

Attribute	Description
WorkArounds (cont.)	<p>When set to 524288, the precision and scale settings for SQL_DECIMAL parameters are overridden to precision 40 and scale 20.</p> <p>When set to 8388608, SQLGetInfo/SQL_DATABASE_NAME is returned as an empty string when SQLGetInfo/SQL_MAX_QUALIFIER_NAME_LEN is 0. This option should be used with Inprise/Borland tools, such as Delphi.</p> <p>When set to 536870912, SQLBindParameter is allowed to be called after SQLExecute to change the precision of previously bound parameters.</p> <p>When set to 1073741824, Microsoft Access assumes that ORDER BY columns do not have to be in the SELECT list. This option provides a workaround for data stores that always use ORDER BY columns.</p>

Table 1-1. ODBC Attributes (cont.)

Attribute	Description
WorkArounds2 (WA2)	<p data-bbox="658 314 1268 435">Turns on workarounds that allow you to take full advantage of the SequeLink ODBC Driver with ODBC applications requiring nonstandard or extended behavior.</p> <p data-bbox="658 453 1268 574">IMPORTANT: Each of these options has potential side effects related to its use. An option should only be used to address the specific problem for which it was designed.</p> <p data-bbox="658 591 1268 782">When set to 2, the driver ignores the ColumnSize/DecimalDigits specified by the application and uses the database defaults instead. Some applications incorrectly specify ColumnSize/DecimalDigits when binding timestamp parameters.</p> <p data-bbox="658 800 1268 1043">When set to 4, Microsoft Access uses the most recent native type mapping, as returned by SQLGetTypeInfo, for a specific SQL type. This option reverses the order in which types are returned, so that Microsoft Access will use the most appropriate native type. This option is recommended if you are using Microsoft Access against an Oracle8 data store.</p> <p data-bbox="658 1060 1268 1187">When set to 32, Microsoft Access requires that the characters "DSN=" are returned by SQLDriverConnect in the connection string output parameter.</p>

2 Developing ODBC Applications



This chapter provides information about developing ODBC applications for SequeLink environments, including:

- [“Required ODBC Libraries and Header Files” on page 58](#)
- [“UNIX Compiler Requirements” on page 59](#)
- [“ODBC API Functions” on page 60](#)
- [“SQL Escape Sequences” on page 63](#)
- [“Data Types and Isolation Levels” on page 63](#)
- [“Threading” on page 63](#)
- [“Using Scrollable Cursors” on page 66](#)
- [“Using Stored Procedures with Oracle” on page 68](#)
- [“Specifying Application IDs” on page 72](#)
- [“Error Handling” on page 74](#)
- [“Developing Performance-Optimized ODBC Applications” on page 77](#)

Required ODBC Libraries and Header Files

To develop ODBC applications, you must install the appropriate ODBC libraries and header files for your target platform, as shown in [Table 2-1](#).

Table 2-1. Sources for Required ODBC Development Tools

Platform	Required Headers and Libraries
 Windows 9x Windows NT Windows 2000	Microsoft Open Database Connectivity Software Development Kit (SDK).
 UNIX	The required header files and libraries are shipped with the SequeLink ODBC Client.

NOTE: We recommend that you obtain the Microsoft ODBC 3.x documentation.

UNIX Compiler Requirements



The SequeLink ODBC Client has specific compiler requirements on UNIX. Applications must be compiled using the guidelines shown in [Table 2-2](#).

Table 2-2. Compiler Requirements for UNIX

UNIX Platform	Compiler Requirements
AIX reentrant	C Set++ for AIX 3.1.4.0 (using the reentrant version, xLC_r)
AIX non-reentrant	C Set++ for AIX 3.1.4.0 (using the non-reentrant version, xLC)
Solaris	SUNWspro (SPARCompiler) 4.2 and 5.0
HP-UX 10 aCC	HP aC++ Compiler version A.01.12 or higher
HP-UX 11 aCC	HP aC++ Compiler version A.03.05 or higher

ODBC API Functions

The SequeLink ODBC Driver is ODBC Level 1–compliant, supporting all ODBC Core and Level 1 functions. Most Level 2 functions are also supported. [Table 2-3](#) and [Table 2-4](#) list supported ODBC 2.x and ODBC 3.x functions, respectively.

Table 2-3. Conformance of Supported Functions for 2.x ODBC Applications

Core Functions

SQLAllocConnect	SQLExecute
SQLAllocEnv	SQLFetch
SQLAllocStmt	SQLFreeConnect
SQLBindCol	SQLFreeEnv
SQLBindParameter	SQLFreeStmt
SQLCancel	SQLGetCursorName
SQLColAttributes	SQLNumResultCols
SQLConnect	SQLPrepare
SQLDescribeCol	SQLRowCount
SQLDisconnect	SQLSetCursorName
SQLError	SQLTransact

Level 1 Functions

SQLColumns	SQLParamData
SQLDriverConnect	SQLPutData
SQLGetConnectOption	SQLSetConnectOption
SQLGetData	SQLSetStmtOption
SQLGetFunctions	SQLSpecialColumns
SQLGetInfo	SQLStatistics
SQLGetStmtOption	SQLTables
SQLGetTypeInfo	

Table 2-3. Conformance of Supported Functions for 2.x ODBC Applications (cont.)

Level 2 Functions

SQLColumnPrivileges	SQLParamOptions
SQLDataSources	SQLPrimaryKeys
SQLDrivers	SQLProcedureColumns
SQLExtendedFetch	SQLProcedures
SQLForeignKeys	SQLSetPos
SQLMoreResults	SQLSetScrollOptions
SQLNativeSql	SQLTablePrivileges
SQLNumParams	

Table 2-4. Function Conformance for 3.x ODBC Applications

SQLAllocHandle	SQLFetchScroll	SQLParamOptions
SQLBindCol	SQLForeignKeys	SQLPrepare
SQLBindParameter	SQLFreeHandle	SQLPrimaryKeys
SQLBulkOperations	SQLFreeStmt	SQLProcedureColumns
SQLCancel	SQLGetConnectAttr	SQLProcedures
SQLCloseCursor	SQLGetCursorName	SQLPutData
SQLColAttribute	SQLGetData	SQLRowCount
SQLColumnPrivileges	SQLGetDescField	SQLSetConnectAttr
SQLColumns	SQLGetDescRec	SQLSetCursorName
SQLConnect	SQLGetDiagField	SQLSetDescField
SQLDataSources	SQLGetDiagRec	SQLSetDescRec
SQLDescribeCol	SQLGetEnvAttr	SQLSetEnvAttr
SQLDescribeParam	SQLGetFunctions	SQLSetPos
SQLDisconnect	SQLGetInfo	SQLSetScrollOptions
SQLDriverConnect	SQLGetStmtAttr	SQLSetStmtAttr
SQLDrivers	SQLGetTypeInfo	SQLSpecialColumns
SQLEndTran	SQLMoreResults	SQLStatistics
SQLExecDirect	SQLNativeSql	SQLTablePrivileges
SQLExecute	SQLNumParams	SQLTables
SQLExtendedFetch	SQLNumResultCols	
SQLFetch	SQLParamData	

SQL Escape Sequences

See [Appendix A “SQL Escape Sequences for ODBC and JDBC”](#) on [page 289](#) for information about the SQL escape sequences supported by the SequeLink ODBC Driver.

Data Types and Isolation Levels

The data types and isolation levels supported by the SequeLink ODBC Driver depend on the data store to which you are connecting. See [Appendix B “Data Types and Isolation Levels”](#) on [page 309](#) for database-specific information about data types and isolation levels.

Threading

The ODBC specification requires that all ODBC drivers must be thread-safe; that is, they must not fail when database requests are made on separate threads.

Threading Architecture

An ODBC driver can be based on one of the following architectures:

- *Not thread-safe.* The ODBC driver should not be used in a multi-threaded environment.
- *Thread-impaired.* The ODBC driver serializes all ODBC calls. All requests are handled one by one, without concurrent processing.

- *Thread per connection.* The ODBC driver processes requests concurrently with statement handles that do not share the same connection handle; however requests on the same connection are serialized.
- *Fully threaded.* All requests fully use the threaded model. The ODBC driver processes all requests on multiple statements concurrently.

The multithreading ability of the SequeLink ODBC Driver is platform dependent as shown in [Table 2-5](#).

Table 2-5. Multithreading Functionality of the SequeLink ODBC Driver

Platform	Capability
Windows 9x, Windows NT, and Windows 2000	Thread for each connection
Solaris	Thread for each connection
AIX reentrant	Thread for each connection
AIX non-reentrant	Not thread-safe
HP-UX 10 aCC	Thread for each connection using the DCE threading model
HP-UX 11 aCC	Thread for each connection using the HP-UX native threading model (p-threads)

Cancelling Functions in Multithreaded Applications

In a multithreaded application, the application can cancel a function that is running synchronously on a statement. To cancel the function, the application calls `SQLCancel` with the same statement handle as that used by the target function, but on a

different thread. Whether SQLCancel actually cancels the running function depends on the data store being accessed as shown in [Table 2-6](#). In [Table 2-6](#):

- *OK* means that SQLCancel can interrupt the running function.
- *Ignored* means that SQLCancel will have no effect on the running function.

In both cases, SQLCancel will return `SQL_SUCCESS`. If SQLCancel has been called from a different thread while there is a pending request, the original statement will return `SQL_ERROR` with the error message `Operation cancelled`.

Table 2-6. Using SQLCancel in Multithreaded Applications

Data Store	SQLCancel
DB2 V4, V5, V6 on OS/390	Ignored
DB2 V6 and V7 on Windows NT and UNIX	Ignored
Informix 7, 9	Ignored
Microsoft SQL Server 6.5, 7.0, 2000	OK
Oracle7 on Windows NT and Windows 2000	Ignored
Oracle7 on UNIX	OK
Oracle8 on Windows NT and Windows 2000	Ignored
Oracle8 on UNIX	OK
Sybase 11, 12	Ignored

Using Scrollable Cursors

Scrollable cursors can move backward and forward in a result set, allowing the application user to scroll back and forth through requested data. SequeLink supports two types of scrollable cursors—static and keyset-driven.

Static and Keyset-Driven Cursors

A *static cursor* is one that does not detect any changes made to the record after the cursor is opened. For example, if a static cursor fetches a row and another application then updates that row, the values would be unchanged when that row is fetched again. A *keyset-driven cursor* detects value changes to the record using keys that are saved when the cursor is opened to retrieve the current data values for each row.

The type of scrollable cursors that can be used depend on the data store to which you are connecting. [Table 2-7](#) shows the type of scrollable cursors supported for each database.

Table 2-7. Support for Scrollable Cursors (ODBC)

Database	Static	Keyset-Driven
DB2 on OS/390	✓	
DB2 on Windows NT and Windows 2000	✓	
Informix	✓	✓
Microsoft SQL Server (see note)	✓	✓
Oracle	✓	✓
Sybase (see note)	✓	✓

NOTE: To use keyset-driven cursors with Microsoft SQL Server and Sybase, the table must contain an identity column.

Using Static Scrollable Cursors

- The SequeLink ODBC Driver supports static cursors for all types of result set generating statements, including result sets generated by stored procedures.
- The SequeLink ODBC Driver supports LOB data for static cursors; however, by default, only the first 4096 bytes of the LOB column is buffered. For more information about specifying the amount of data that is buffered, see the `SLKStaticCursorLongColBuffLen` connection attribute in [Table 1-1, "ODBC Attributes," on page 47](#).

Using Keyset-Driven Scrollable Cursors

- The SequeLink ODBC Driver does not support using keyset-driven cursors on stored procedures or explicit batches.
- The SequeLink ODBC Driver cannot use keyset-driven cursors, when the Select statement contains any of the following SQL language constructions:
 - JOIN
 - Aggregate functions
 - GROUP BY

Using Stored Procedures with Oracle

SequeLink supports stored procedures against Oracle, including stored procedures in packages.

NOTE: Stored procedures in packages must be qualified with the package name, for example, EmployeePackage.EmployeeProc).

Also, SQLProcedures and SQLProcedureColumns can return information on procedures within PL/SQL packages, allowing ODBC applications to execute these procedures. This section contains an example that shows you how to fetch rows using Oracle PL/SQL procedures.

Example - Part 1

```
Create or replace package EmployeeInfo as
  Type EmployeeRec is record
  (
    Employee_Id      integer,
    Employee_Name    varchar2(25),
    Employee_Job     varchar2(25),
    Department_Name  varchar2(30),
    Employee_Salary  integer
  );
  Type EmployeeCursor is ref cursor return
  EmployeeRec;
End EmployeeInfo;

Create or replace procedure EmployeeInfoProc
(empname IN varchar2, empcursor IN OUT
EmployeeInfo.EmployeeCursor)
As
Begin
  Open empcursor For
  select empno, ename, job, dname, sal
  from emp, dept
```

```

where emp.deptno=dept.deptno and
ename like empname;
End;

```

NOTE: In this Oracle PL/SQL package, a record type and a cursor (result set) type is defined. The procedure contains an input parameter that can have a value, such as `Smi%`, to request information about employees whose last name starts with the letters 'Smi' (for example, Smith or Smithwick). The procedure also has one input/output parameter of the cursor type defined in the package.

Example - Part 2

This example shows an ODBC function call sequence executing the stored procedure.

```

SQLPrepare(..., '{call EmployeeInfoProc(?)}', ...)
    <- ODBC SQL syntax to executed stored procedures
SQLBindParameter(..., 'Smi%', ...)
    <- define the input variable for the input marker ?
    in the SQL stmt and assign the value 'Smi%' to it
SQLExecute()
    <- execute the stored procedure
SQLBindCol()
    <- Assign storage for result column 1 in the
    result set (Employee_Id)
SQLBindCol()
    <- Assign storage for result column 2 in the
    result set (Employee_Name)
SQLBindCol()
    <- Assign storage for result column 3 in the
    result set (Employee_Job)
SQLBindCol()
    <- Assign storage for result column 4 in the
    result set (Department_Name)
SQLBindCol()
    <- Assign storage for result column 5 in the
    result set (Employee_Salary)
SQLFetch()
    <- Fetch the first record from the result set
    generated by the stored procedure.

```

IMPORTANT: From the following procedure definition, you might think that, by having two parameters, the procedure must call `SQLBindParameter` twice:

```
Create or replace procedure EmployeeInfoProc  
(empname IN varchar2, empcursor IN OUT  
EmployeeInfo.EmployeeCursor)
```

Actually, it does not. The only way to create a result set from an Oracle stored procedure is to declare this result set, `empcursor`, as an input/output parameter. This can be seen in the result of `SQLProcedureColumns(..., 'EmployeeInfoProc', ...)` which an application can use to query the server about a stored procedure.

Here's an excerpt of a session using the tool ODBCTest:

SQLAllocStmt:

In: hdbc=0x004609F0, phstmt=VALID
Return: SQL_SUCCESS=0

SQLPrepare:

In: hstmt=#3 0x00305850, szSqlStr={call EmployeeInfoProc(?)}, cbSqlStr=26
Return: SQL_SUCCESS=0

SQLBindParameter:

In: hstmt=#3 0x00305850, ipar=1, fParamType=SQL_PARAM_INPUT=1,
fCType=SQL_C_CHAR=1,
fSqlType=SQL_CHAR=1, cbColDef=10, ibScale=0, rgbValue=VALID,
cbValueMax=300, pcbValue=VALID, SQL_LEN_DATA_AT_EXEC=FALSE
Return: SQL_SUCCESS=0

SQLExecute:

In: hstmt=#3 0x00305850
Return: SQL_SUCCESS=0

Get Data All:

"EMPNO", "ENAME", "JOB", "DNAME", "SAL"
7934, "MILLER", "CLERK", "ACCOUNTING", 1300.00
7654, "MARTIN", "SALESMAN", "SALES", 1250.00
2 rows fetched from 5 columns.

SQLProcedureColumns:

In: hstmt=#4 0x00305BD8, ...Qualifier=NULL, ...Qualifier=0,
Owner=SCOTT, ...Owner=5, ...Name=EMPLOYEEINFOPROC,
...Name=16, ...Name=NULL, ...Name=0
Return: SQL_SUCCESS=0

Get Data All:

"PROCEDURE_CAT", "PROCEDURE_SCHEM", "PROCEDURE_NAME", "COLUMN_NAME",
"COLUMN_TYPE", "DATA_TYPE", ..."TYPE_NAME", "COLUMN_SIZE",
"BUFFER_LENGTH", "DECIMAL_DIGITS", "NUM_PREC_RADIX", "NULLABLE",
"REMARKS", "COLUMN_DEF", "SQL_DATA_TYPE", "SQL_DATETIME_SUB",
"CHAR_OCTET_LENGTH", "ORDINAL_POSITION", "IS_NULLABLE"
"", "SCOTT", "EMPLOYEEINFOPROC", "EMPNAME", 1, 12, "VARCHAR2", 2000,
2000, <Null>, <Null>, 1, <Null>, <Null>, ...12, <Null>, 2000, 1, "YES"

Specifying Application IDs

Application IDs are alphanumeric strings passed by a SequeLink Client that identify the client application to a SequeLink service that has been configured to accept connections only from specific application IDs.

For more information about configuring SequeLink services to accept connections only from specific application IDs, refer to the *SequeLink Administrator's Guide*.

Specifying Application IDs Explicitly

ODBC client applications can identify themselves explicitly to the SequeLink service in any of the following ways:

- **Specifying the application ID in the ODBC connection string that is passed to `SQLDriverConnect`.** For example:

```
....;APPID=MyAppID;
```

or

```
....;ApplicationID=MyAppID;
```

where *MyAppID* is the application ID.

- **Specifying the application ID using `SQLSetConnectOption`.** Immediately after each call to `SQLConnect` or `SQLDriverConnect` connecting to the SequeLink ODBC Client, call `SQLSetConnectOption` as shown:

```
short res_code;  
res_code=SQLSetConnectOption(hdbc, 1053, "myAppId")
```

where *myAppId* is the application ID.

The `SQLSetConnectOption` is defined in `sql.h`. If an incorrect application ID is specified, the `SQLSetConnectOption` fails, and all subsequent SQL statements will fail.

- **Specifying the application ID using SQLSetConnectAttr.** Immediately after each call to `SQLConnect` or `SQLDriverConnect` connecting to the SequeLink ODBC Client, call `SQLSetConnectAttr` as shown:

```
SQLSetConnectAttr(hdbc, 1053, "myAppId", SQL_NTS)
```

where *myAppId* is the application ID.

The `SQLSetConnectAttr` is defined in `sql.h`. If an incorrect application ID is specified, the `SQLSetConnectAttr` fails and all subsequent SQL statements fail.

Generating Application IDs Automatically

ODBC client applications can turn on automatic application ID generation in any of the following ways:

- **Specifying the automatic application ID method in the ODBC connection string that is passed to `SQLDriverConnect`,** for example:

```
...;AutomaticApplicationID=x;
```

where *x* is either 1, 2, or 3.

- **Specifying `SQLSetConnectOption`.** Immediately after each call to `SQLConnect` or `SQLDriverConnect` connecting to the SequeLink ODBC Client, call `SQLSetConnectOption` as shown:

```
short res_code;
res_code=SQLSetConnectOption(hdbc, 1054, x)
```

where *x* is either 1, 2, or 3.

- **Specifying SQLSetConnectAttr.** Immediately after each call to `SQLConnect` or `SQLDriverConnect` connecting to the SequeLink ODBC Client, call `SQLSetConnectAttr` as shown:

```
SQLSetConnectAttr(hdbc, 1054, x, SQL_IS_INTEGER)
```

where *x* is either 1, 2, or 3.

Error Handling

The following types of errors can occur when you are using the SequeLink ODBC Client:

- SequeLink ODBC Driver errors
- SequeLink Client errors
- SequeLink Server errors
- Database errors

SequeLink ODBC Driver Errors

An error generated by the SequeLink ODBC Driver has the following format:

```
[MERANT] [ODBC SequeLink driver] message
```

For example:

```
[MERANT] [ODBC SequeLink driver] Invalid precision
specified.
```

The native error code is always zero (0).

If you receive this type of error, check the last ODBC call your application made. Contact your ODBC application vendor, or refer to the ODBC documentation available from Microsoft. The *ODBC 2.0 Programmer's Reference and Data Access SDK* and the *ODBC 3.0 Software Development Kit and Programmer's*

Reference are both available from Microsoft Press. For information on later versions of ODBC, refer to the documentation included in the ODBC SDK.

SequeLink Client Errors

An error generated by the SequeLink ODBC Client has the following format:

```
[MERANT] [ODBC SequeLink driver] [SequeLink Client] message
```

For example:

```
[MERANT] [ODBC SequeLink driver] [SequeLink Client] The specified transliteration module is not found.
```

Use the native error code to look up details about the possible cause of the error. For a list of all error codes and messages, refer to the *SequeLink Troubleshooting Guide and Reference*.

SequeLink Server Errors

An error generated by SequeLink Server has the following format:

```
[MERANT] [ODBC SequeLink driver] [SequeLink Server] message
```

For example:

```
[MERANT] [ODBC SequeLink driver] [SequeLink Server] Only Select statements are allowed in this read-only connection.
```

Use the native error code to look up details about the possible cause of the error. For a list of all error codes and messages, refer to the *SequeLink Troubleshooting Guide and Reference*.

Database Errors

An error generated by the database has the following format:

```
[MERANT] [ODBC SequeLink driver] [...] message
```

For example:

```
[MERANT] [ODBC SequeLink driver] [Oracle] ORA-00942:table  
or view does not exist.
```

Use the native error code to look up details about the possible cause of the error. For these details, refer to your database documentation.

Developing Performance-Optimized ODBC Applications

This section provides general guidelines for optimizing system performance that have been compiled by examining how numerous shipping ODBC applications have been implemented. These guidelines are divided into the following categories:

- [“Catalog Functions” on page 77](#)
- [“Retrieving Data” on page 83](#)
- [“ODBC Function Selection” on page 87](#)
- [“Designing ODBC Applications” on page 90](#)
- [“Updating Data” on page 92](#)

Catalog Functions

The following ODBC functions are catalog functions:

- | | |
|-----------------------|-----------------------|
| ■ SQLColumns | ■ SQLProcedureColumns |
| ■ SQLColumnPrivileges | ■ SQLSpecialColumns |
| ■ SQLForeignKeys | ■ SQLStatistics |
| ■ SQLGetTypeInfo | ■ SQLTables |
| ■ SQLProcedures | ■ SQLTablePrivileges |

SQLGetTypeInfo is listed as a potentially performance-expensive ODBC function, because many drivers must query the server to obtain accurate information about which data types are supported (for example, to find dynamic data types such as user-defined types).

Catalog Functions Are Slow

Because catalog functions are slow compared to other ODBC functions, their frequent use can impair system performance. Although it is almost impossible to write an ODBC application without using catalog functions, you can improve performance by minimizing their use.

To return all result column information mandated by the ODBC specification, an ODBC driver may have to perform multiple queries, joins, subqueries, and unions to return the required result set for a single call to a catalog function. These particular elements of the SQL language are performance "hogs."

Applications should cache information from catalog functions so that multiple executions are unnecessary. For example, call `SQLGetTypeInfo` once in the application and cache the elements of the result set on which your application depends. It is unlikely that an application will use every element of the result set generated by a catalog function, so the cached information should not be difficult to maintain.

Passing Null Arguments

Passing null arguments to catalog functions generates time-consuming queries. In addition, network traffic may increase because of unnecessary result set information. Always supply as many non-null arguments to catalog functions as possible.

Because catalog functions are slow, applications should invoke them efficiently. Many applications pass the minimum number of non-null arguments required for the function to return a successful result set.

For example, consider a call to `SQLTables` where the application requests information about the table "Customers." Often, this call is coded as shown:

```
rc = SQLTables (NULL, NULL, NULL, NULL, "Customers", SQL_NTS, NULL);
```

A driver may process this `SQLTables` call into SQL as shown:

```
SELECT ... FROM SysTables WHERE TableName = 'Customers' UNION ALL
SELECT ... FROM SysViews WHERE ViewName = 'Customers' UNION ALL
SELECT ... FROM SysSynonyms WHERE SynName = 'Customers'
ORDER BY ...
```

Sometimes, not much information is known about the object for which you are requesting information. Any information that the application can send the driver when calling catalog functions can result in improved performance and reliability.

Using the previous example, suppose three "Customers" tables were returned in the result set:

- The first table was owned by the user.
- The second table was owned by the sales department.
- The third table was a view created by management.

It may not be obvious to the user which table to choose. If the application had specified the `OwnerName` argument for the `SQLTables` call, only one table would be returned and performance would improve, because less network traffic was required to return only one result row and unnecessary rows were filtered by the database. In addition, if the `TableType` argument can be supplied, the SQL sent to the server can be changed from a three-query union to a single `Select` statement as shown:

```
SELECT ... FROM SysTables
WHERE TableName = 'Customers' and Owner = 'Beth'
```

SQLColumns

Avoid using `SQLColumns` to determine table characteristics. Instead, use a dummy query with `SQLDescribeCol`.

Consider an application that allows the user to choose columns. Should the application use `SQLColumns` to return information about the columns to the user or prepare a dummy query and call `SQLDescribeCol`?

Case 1: SQLColumns Method

```
rc = SQLColumns (... "UnknownTable" ...);
// This call to SQLColumns will generate a query to
// the system catalogs... possibly a join which must be
// prepared, executed, and produce a result set
rc = SQLBindCol (...);
rc = SQLExtendedFetch (...);
// user must retrieve N rows from the server
// N = # result columns of UnknownTable
// result column information has now been obtained
```

Case 2: SQLDescribeCol Method

```
// prepare dummy query
rc = SQLPrepare (... "SELECT * from UnknownTable
WHERE 1 = 0" ...);
// query is never executed on the server - only prepared
rc = SQLNumResultCols (...);
for (irow = 1; irow <= NumColumns; irow++) {
    rc = SQLDescribeCol (...);
    // + optional calls to SQLColAttributes
}
// result column information has now been obtained
// Note we also know the column ordering within the table!
// This information cannot be
// assumed from the SQLColumns example.
```

In both cases, a query is sent to the server. In Case 1, the query must be evaluated and return a result set to the client. Case 2 is the better performing model.

To complicate this discussion, consider a database server that does not support natively preparing a SQL statement. The performance of Case 1 would not change, but the performance of Case 2 would improve slightly, because the dummy query is evaluated before being prepared. Because the Where clause of the query always evaluates to FALSE, the query generates no result rows and is processed without accessing table data. Again, Case 2 performs better than Case 1.

Managing the Retrieval of Database Meta-Information

Meta-information is information that describes the data stored in the database and can include information about the tables in the database, the columns in those tables, and the indexes that are defined for those tables. This data also is referred to as the database's *data dictionary* or *system catalog*.

Typically, ODBC, OLE DB, and JDBC applications extract and use information from the database's data dictionary using specific calls, such as the ODBC calls `SQLTables`, `SQLColumns`, and `SQLPrimaryKeys`. In large databases, the amount of meta-information that is retrieved can be considerable. Because some client applications cannot manage large amounts of information efficiently, system performance can be adversely affected.

Some ODBC, OLE DB, and JDBC calls have parameters that accept search patterns. You can use these parameters to limit the amount of meta-information that is retrieved; however, not every client application supports these parameters.

SequeLink allows you to use database data dictionary filters and database data dictionary views to limit the amount of meta-information that is retrieved.

Using Database Data Dictionary Filters

Database Data Dictionary filters limit the amount of meta-information that can be retrieved from the database's native data dictionary. Specifically, they limit the number of result rows that can be returned for SQLTables. The data dictionary filters override any call parameters that are passed by the application when it accesses the database's native data dictionary.

SequeLink provides the following types of database data dictionary filters, which must be defined on the server:

- Filter by catalog list
- Filter by schema list
- Filter by table type
- Filter by database (DB2 for OS/390 only)

For more information about setting the database data dictionary filters for a SequeLink service, refer to the *SequeLink Administrator's Guide*.

Using Database Data Dictionary Views (DB2 for OS/390 only)

The DataSourceDB2CatalogOwner service attribute on the server allows you to limit the meta-information that is returned by using views on the database data dictionary. Database Data Dictionary views are supported for DB2 for OS/390 only. For more information about setting the DataSourceDB2CatalogOwner service attribute, refer to the *SequeLink Administrator's Guide*.

Retrieving Data

This section provides general guidelines for retrieving data with ODBC applications.

Retrieving Long Data

Unless it is necessary, applications should not request long data (SQL_LONGVARCHAR and SQL_LONGVARBINARY data), because retrieving long data across a network is slow and resource-intensive.

Most users do not want to see long data. If the user does need to see these result items, the application can query the database again, specifying only the long columns in the select list. This method allows the average user to retrieve result sets without having to pay a high performance penalty for network traffic.

Although the best method is to exclude long data from the select list, some applications do not formulate the select list before sending the query to the ODBC driver (for example, some applications `select * from table_name ...`). If the select list contains long data, the driver must retrieve that data at fetch time, even if the application does not bind the long data in the result set. When possible, the application developer should use a method that does not retrieve all columns of the table.

Reducing the Size of Retrieved Data

To reduce network traffic and improve performance, you can reduce the size of data being retrieved to a manageable limit by calling `SQLSetStmtOption` with the `SQL_MAX_LENGTH` option. Although eliminating `SQL_LONGVARCHAR` and `SQL_LONGVARBINARY` data from the result set is ideal for performance optimization, sometimes, retrieving long data is necessary. When it is necessary, remember that most users do not want to see 100 KB, or more, of text on the screen. What

techniques, if any, are available to limit the amount of data retrieved?

Many application developers mistakenly assume that if they call `SQLGetData` with a container of size x , the ODBC driver only retrieves x bytes of information from the server. Because `SQLGetData` can be called multiple times for any one column, the driver optimizes its network use by retrieving long data in large chunks and returning it to the user when requested.

For example:

```
char CaseContainer[1000];
...
rc = SQLExecDirect (hstmt, "SELECT CaseHistory FROM Cases
    WHERE CaseNo = 71164", SQL_NTS);
...
rc = SQLFetch (hstmt);
rc = SQLGetData (hstmt, 1, CaseContainer, (SQLWORD)
    sizeof(CaseContainer), ...);
```

At this point, it is more probable that an ODBC driver will retrieve 64 KB of information from the server, rather than 1000 bytes. One 64-KB retrieval requires less network traffic than 64 1000-byte retrievals. Unfortunately, the application may not call `SQLGetData` again; thus, the first and only retrieval of `CaseHistory` would be slowed by the fact that 64 KB of data had to be sent across the network.

Many ODBC drivers allow you to limit the amount of data retrieved across the network by using the statement attribute `SQL_MAX_LENGTH`. This attribute allows the driver to communicate to the database server that only Z bytes of data are relevant to the client. The server responds by sending only the first Z bytes of data for all result columns. This optimization substantially reduces network traffic and improves performance. Our example returned only one row, but, consider the case where 100 rows are returned in the result set—the performance improvement would be substantial.

Using Bound Columns

Retrieving data using bound columns (SQLBindCol), instead of SQLGetData, reduces the ODBC call load and improves performance.

Consider the following example:

```
rc = SQLExecDirect (hstmt, "SELECT <20 columns>
    FROM Employees WHERE HireDate >= ?", SQL_NTS);
do {
rc = SQLFetch (hstmt);
// call SQLGetData 20 times
} while ((rc == SQL_SUCCESS) || (rc==
SQL_SUCCESS_WITH_INFO));
```

Suppose the query returns 90 result rows. More than 1890 ODBC calls are made (20 calls to SQLGetData x 90 result rows + 91 calls to SQLFetch).

Consider the same scenario that uses SQLBindCol, instead of SQLGetData:

```
rc = SQLExecDirect (hstmt, "SELECT <20 columns>
    FROM Employees WHERE HireDate >= ?", SQL_NTS);
// call SQLBindCol 20 times
do {
rc = SQLFetch (hstmt);
} while ((rc == SQL_SUCCESS) || (rc==
SQL_SUCCESS_WITH_INFO));
```

The number of ODBC calls made is reduced from more than 1890 to about 110 (20 calls to SQLBindCol + 91 calls to SQLFetch). In addition to reducing the call load, many ODBC drivers optimize how SQLBindCol is used by binding result information directly from the database server to the user's buffer. That is, instead of the ODBC driver retrieving information into a container and copying that information to the user's buffer, the ODBC driver simply requests the information from the server be placed directly into the user's buffer.

Using SQLExtendedFetch Instead of SQLFetch

By using `SQLExtendedFetch` to retrieve data, instead of `SQLFetch`, the ODBC call load decreases, resulting in better performance, and the code becomes less complex, resulting in more easily maintainable code.

Most ODBC drivers support `SQLExtendedFetch` for forward only cursors; yet, most ODBC applications use `SQLFetch` to retrieve data. Again, consider the previous example using `SQLExtendedFetch`, instead of `SQLFetch`:

```
rc = SQLSetStmtOption (hstmt, SQL_ROWSET_SIZE, 100);
// use arrays of 100 elements
rc = SQLExecDirect (hstmt, "SELECT <20 columns>
    FROM Employees WHERE HireDate >= ?", SQL_NTS);
// call SQLBindCol 1 time specifying row-wise binding
do {
rc = SQLExtendedFetch (hstmt, SQL_FETCH_NEXT, 0,
    &RowsFetched, RowStatus);
} while ((rc == SQL_SUCCESS) || (rc ==
SQL_SUCCESS_WITH_INFO));
```

In this second example, the number of ODBC calls made by the application is reduced from 110 to 4 (1 `SQLSetStmtOption` + 1 `SQLExecDirect` + 1 `SQLBindCol` + 1 `SQLExtendedFetch`)—a total savings from an initial call load of more than 1890 ODBC calls to 4 ODBC calls. In addition to reducing the call load, many ODBC drivers retrieve data from the server in arrays, further improving performance by reducing network traffic.

For ODBC drivers that do not support `SQLExtendedFetch`, the application can enable forward-only cursors using the ODBC cursor library (call `SQLSetConnectOption` using `SQL_ODBC_CURSORS/SQL_CUR_USE_IF_NEEDED`).

Although using the cursor library does not improve performance, it should not be detrimental to your application's response time when using forward-only cursors (no logging is required).

Furthermore, using the cursor library when `SQLExtendedFetch` is

not supported natively by the ODBC driver simplifies the code, because the application can depend on `SQLExtendedFetch` being available. The application does not require two algorithms (one using `SQLExtendedFetch` and another using `SQLFetch`).

ODBC Function Selection

This section provides general guidelines for selecting which ODBC functions will provide the best performance.

Using SQLPrepare/SQLExecute and SQLExecDirect

Using `SQLPrepare/SQLExecute` is not always as efficient as using `SQLExecDirect`. Use `SQLExecDirect` for queries that will be executed once and `SQLPrepare/SQLExecute` for queries that will be executed multiple times.

ODBC drivers are optimized based on the perceived use of the functions that are being executed. `SQLPrepare/SQLExecute` is optimized for multiple executions of a statement that uses parameter markers. `SQLExecDirect` is optimized for a single execution of a SQL statement. Unfortunately, more than 75 percent of all ODBC applications use `SQLPrepare/SQLExecute` exclusively.

Consider an ODBC driver that implements `SQLPrepare` by creating a stored procedure on the server which contains the prepared statement. Creating stored procedures has substantial overhead, but the statement will be executed multiple times. Although creating stored procedures is performance-expensive, processing is minimal because the query is parsed and optimization paths are stored when the procedure is created.

Using `SQLPrepare/SQLExecute` for a statement that will be executed only once results in unnecessary overhead.

Furthermore, applications that use SQLPrepare/SQLExecute for large, single-execution query batches will probably exhibit poor performance. Similarly, applications that use SQLExecDirect exclusively do not perform as well as those that use a logical combination of SQLPrepare/SQLExecute and SQLExecDirect sequences.

Using SQLPrepare and Multiple SQLExecute Calls

Applications that use SQLPrepare and multiple SQLExecute calls should use SQLParamOptions. Passing arrays of parameter values reduces the ODBC call load and network traffic.

Consider the following example that inserts data:

```
rc = SQLPrepare (hstmt, "INSERT INTO DailyLedger (...)
    VALUES (?, ?, ...)", SQL_NTS);
// bind parameters
...
do {
// read ledger values into bound parameter buffers
...
rc = SQLExecute (hstmt);      // insert row
} while ! (eof);
```

If there are 100 rows to insert, SQLExecute is called 100 times, resulting in 100 network requests to the server.

Alternatively, consider an algorithm that uses parameter arrays by calling `SQLParamOptions`:

```
rc = SQLPrepare (hstmt, "INSERT INTO DailyLedger (...)
    VALUES (?, ?, ...)", SQL_NTS);
rc = SQLParamOptions (hstmt, (UDWORD) 50, &CurrentRow);
// pass 50 parameters per execute
// bind parameters
...
do {
// read up to 50 ledger values into bound parameter buffers
...
rc = SQLExecute (hstmt);      // insert row
```

The call load is reduced from 100 to just 2 `SQLExecute` calls. Furthermore, network traffic is reduced considerably. Some ODBC drivers do not support `SQLParamOptions`. To achieve the best performance, applications should contain algorithms for using `SQLParamOptions`. `SQLParamOptions` is ideal for copying data into new tables or bulk loading tables.

Using the Cursor Library

If scrollable cursors are provided by the driver, do not use the cursor library automatically. The cursor library creates local temporary log files, which are performance-expensive to generate and provide worse performance than native scrollable cursors.

The cursor library provides support for static cursors, which simplifies the coding of applications that use scrollable cursors; however, the cursor library creates temporary log files on the user's local disk drive as it performs the task. Typically, disk input/output is one of the slowest operations on PCs. Although the cursor library is beneficial, applications should not choose automatically to use the cursor library when an ODBC driver supports scrollable cursors natively.

Typically, ODBC drivers that support scrollable cursors achieve better performance by requesting that the database server produce a scrollable result set, instead of emulating this ability by creating log files.

Many applications use:

```
rc = SQLSetConnectOption (hdbc, SQL_ODBC_CURSORS,
    SQL_CUR_USE_ODBC);
```

but should use:

```
rc = SQLSetConnectOption (hdbc, SQL_ODBC_CURSORS,
    SQL_CUR_USE_IF_NEEDED);
```

Designing ODBC Applications

This section provides general guidelines for designing ODBC applications that will help optimize system performance.

Managing Connections

Connection management affects application performance. Developers should optimize their applications by connecting once and using multiple statement handles, instead of performing multiple connections. Many ODBC applications contain poorly designed elements for connection management. Avoid connecting to a data source after establishing an initial connection.

Some ODBC applications are designed to call informational gathering routines that have no record of attached connection handles. For example, some applications establish a connection and call a routine in a separate DLL or shared library that reattaches and gathers information about the driver.

Although gathering ODBC driver information at connection is a good practice, it often is more efficient to gather it in one-step

rather than two steps. One popular commercial ODBC application connects a second time to gather driver information, and *never* disconnects the second connection. Applications that are designed as separate entities should pass the connected HDBC pointer to the data collection routine, instead of establishing a second connection.

Another bad practice is to connect and disconnect several times throughout your application to perform SQL statements. Connection handles can have multiple statement handles associated with them. Statement handles can provide memory storage for information about SQL statements; therefore applications do not need to allocate new connection handles to perform SQL statements. Applications should use *statement handles* to manage multiple SQL statements.

Connection and statement handling should be addressed before implementation. Spending time and thoughtfully handling connection management improves application performance and maintainability.

Committing Data

Committing data is extremely disk input/output intensive and slow. If the ODBC driver can support transactions, turn Autocommit off.

What does a commit involve? The database server must flush back to disk every data page containing updated or new data. This is not a sequential write, but a searched write to replace existing data in the table. By default, Autocommit is on when connecting to a data source, and Autocommit mode usually impairs performance because of the significant amount of disk input/output required to commit every operation.

Furthermore, some database servers do not provide an Autocommit mode. For this server type, the ODBC driver must explicitly issue a COMMIT statement and a BEGIN TRANSACTION

for every operation sent to the server. In addition to the large amount of disk input/output required to support Autocommit mode, a performance penalty is paid for up to three network requests for every statement issued by an application.

Updating Data

This section provides general guidelines for updating data in databases that will help optimize system performance.

Using Positional Updates and Deletes

Although positional updates do not apply to all types of applications, developers should attempt to use positional updates and deletes when it makes sense. Positional updates (using “update where current of cursor” or using `SQLSetPos`) allow the developer to signal the ODBC driver to “change the data here” by positioning the database cursor to the appropriate row to be changed. The developer is not forced to build a complex SQL statement and simply supplies the data that will be changed.

In addition to making the application more easily maintainable, positional updates usually result in improved performance. Because the database server is already positioned on the row for the Select statement in process, performance-expensive operations to locate the row to be changed are not needed. If the row must be located, the server usually has an internal pointer to the row available (for example, ROWID).

Using SQLSpecialColumns

Use `SQLSpecialColumns` to determine the optimal set of columns to use in the Where clause for updating data. Often,

pseudo-columns provide the fastest access to the data, and these columns can only be determined by using `SQLSpecialColumns`.

Some applications cannot be designed to take advantage of positional updates and deletes. These applications usually update data by using a `Where` clause consisting of some subset of the column values returned in the result set. Some applications may formulate the `Where` clause by using all searchable result columns or by calling `SQLStatistics` to find columns that may be part of a unique index. These methods usually work, but may result in fairly complex queries.

Consider the following:

```
rc = SQLExecDirect (hstmt, "SELECT first_name, last_name,
    ssn, address, city, state, zip FROM emp", SQL_NTS);
// fetchdata
...
rc = SQLExecDirect (hstmt, "UPDATE EMP SET ADDRESS = ?
    WHERE first_name = ? and last_name = ? and ssn = ?
    and address = ? and city = ? and state = ? and zip = ?",
    SQL_NTS);
// fairly complex query
```

Applications should call `SQLSpecialColumns/SQL_BEST_ROWID` to retrieve the optimal set of columns (possibly a pseudo-column) that identifies a specific record. Many databases support special columns that are not explicitly defined by the user in the table definition but are "hidden" columns of every table (for example, `ROWID` and `TID`). These pseudo-columns usually provide the fastest access to the data, because they usually point to the exact location of the record. Because pseudo-columns are not part of the explicit table definition, they are not returned from `SQLColumns`. To determine if pseudo-columns exist, call `SQLSpecialColumns`.

Consider the previous example again:

```

...
rc = SQLSpecialColumns (hstmt, ..... 'emp', ...);
...
rc = SQLExecDirect (hstmt, "SELECT first_name, last_name,
    ssn, address, city, state, zip, ROWID FROM emp",
    SQL_NTS);
// fetch data and probably "hide" ROWID from the user
...
rc = SQLExecDirect (hstmt, "UPDATE emp SET address = ?
    WHERE ROWID = ?", SQL_NTS);
// fastest access to the data!

```

If your data source does not contain special pseudo-columns, the result set of `SQLSpecialColumns` consists of the columns of the optimal unique index on the specified table (if a unique index exists); therefore, your application does not need to call `SQLStatistics` to find the smallest unique index.

Part 2: Developing ADO Applications

This part contains the following chapters:

- [Chapter 3 “Using the SequeLink ADO Client” on page 97](#) provides information about using ADO applications with the SequeLink ADO Client.
- [Chapter 4 “Developing ADO Applications” on page 123](#) provides information about developing ADO applications for SequeLink environments.

3 Using the SequeLink ADO Client

This chapter provides information about using ADO/OLE DB applications with the SequeLink ADO Client.

About the SequeLink ADO Client

The SequeLink ADO Client supports ADO/OLE DB applications through a component called the *SequeLink ADO Provider*. The SequeLink ADO Client is an ADO middleware data access component. It uses ADO technology to connect business applications to relational data stores (like Oracle). In most cases, minimal user interaction with the SequeLink ADO Client middleware is needed because it works in the background, providing connectivity transparently.

Some businesses write and support their own applications for data access. For example, application developers may need to write ADO or OLE DB-based data consumers that use the SequeLink ADO Provider. This book provides programming information for developers who want to control data source connections from within their applications.

After you install the SequeLink ADO Client, your OLE DB- or ADO-based business applications (data consumers) can automatically detect it on your system. Depending on the data consumer's design, the data consumer can connect directly to a data source that uses the SequeLink ADO Provider, or it can provide a way to select the data provider to make a connection.

You use the DataDirect Configuration Manager to define ADO data sources for the SequeLink ADO Provider. After you have defined data sources that use the provider, you can select the

provider from your data consumer to make a connection. See [“Configuring ADO Client Data Sources” on page 103](#) for instructions on creating and configuring data sources.

You use the MERANT DataDirect Configuration Manager to define ADO data sources for the SequeLink ADO Provider. The Configuration Manager contains a Setup Assistant for the SequeLink ADO Provider that helps you create an ADO data source.

After you have defined SequeLink ADO data sources, you can access them from your data consumer and connect to them. For more information, see [“Testing ADO Connections” on page 109](#).

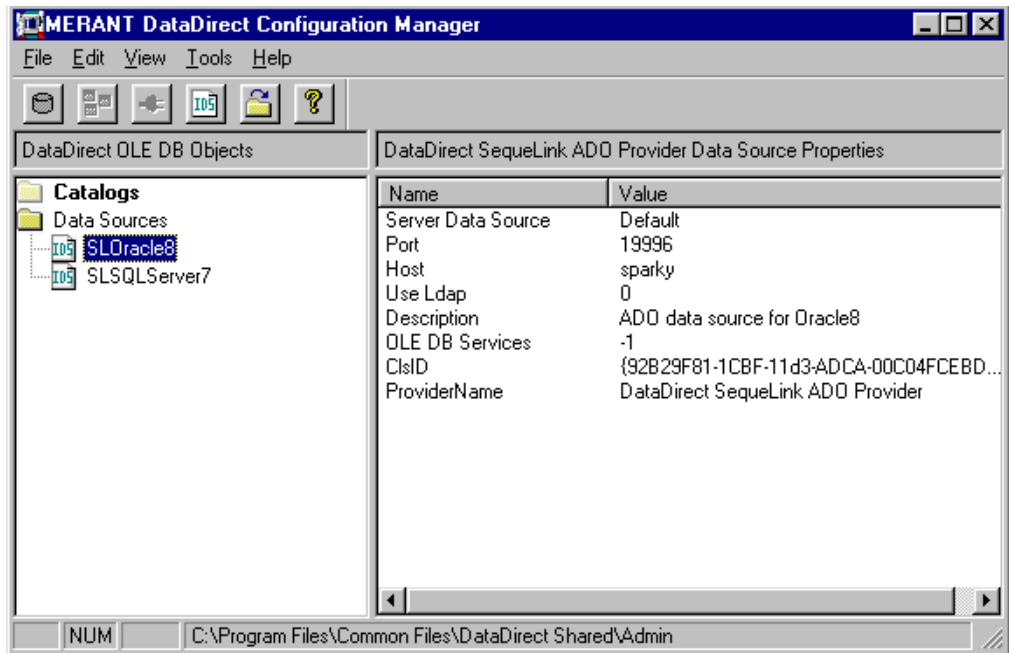
Using the DataDirect Configuration Manager

To create and configure data sources for the SequeLink ADO Client, you use the MERANT DataDirect Configuration Manager.

To start the Configuration Manager, select **Start / Programs**, and select **SequeLink ADO Client 5.1**. Then, select the **MERANT DataDirect Configuration Manager** application.

The MERANT DataDirect Configuration Manager window is divided into two panes. As [Figure 3-1 on page 99](#) shows, the left pane displays a folder containing defined ADO data sources. When you select a data source, the right pane displays the properties for the selected data source.

Figure 3-1. MERANT DataDirect Configuration Manager






Double-click the **Data Sources** folder to display any existing ADO data sources. The Configuration Manager displays the SequeLink ADO data sources contained in the current directory, which is shown in the status bar at the bottom of the Configuration Manager. The first time you start the Configuration Manager, the current directory defaults to the \Program Files\Common Files\DataDirect Shared\Admin directory.

Working with the DataDirect Configuration Manager

[Table 3-1](#) summarizes the parts and functions of the Configuration Manager that you use with SequeLink ADO data sources.

NOTE: Options that are not supported by the SequeLink ADO Provider are disabled in the toolbar and are omitted from this description.

Table 3-1. DataDirect Configuration Manager: Parts and Functions for SequeLink ADO Data Sources

Use this element...		To do this...
Toolbar		Create new data sources.
		Change the current directory.
		View online help.
Menu Bar	File	<ul style="list-style-type: none"> ■ Create a new data source. ■ Exit from the DataDirect Configuration Manager.
	Edit	<ul style="list-style-type: none"> ■ Delete a data source. ■ Rename a data source. ■ Modify a data source.

Shortcut Tip: Right-clicking an item in the left pane displays a pop-up menu that allows you to perform the same actions that are available from the toolbar and menu bar.

Table 3-1. DataDirect Configuration Manager: Parts and Functions for SequeLink ADO Data Sources (cont.)

Use this element...	To do this...
View	<ul style="list-style-type: none"> ■ View or hide the toolbar and status bar. ■ Refresh the MERANT Configuration Manager.
Tools	Options (change current directory)
Help	View online help.
Vertical splitter bar	Click on the bar and drag it to the right or left to change the size of the left and right panes.
Status bar	<ul style="list-style-type: none"> ■ Show the current keyboard state, including when Num Lock, Scroll Lock, and Caps Lock are turned on. ■ Show the current directory.

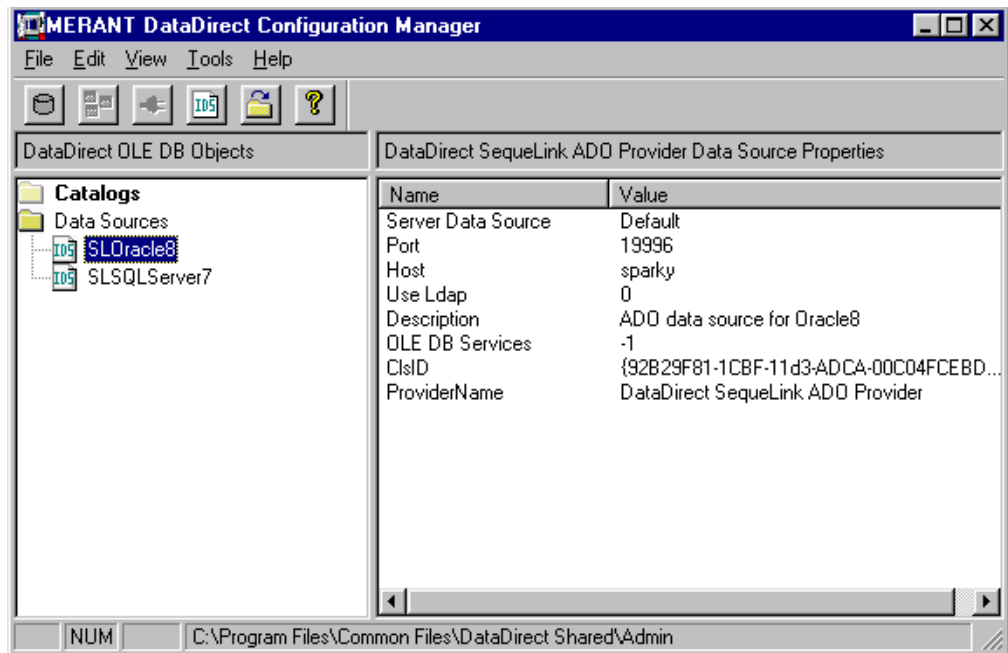
Shortcut Tip: Right-clicking an item in the left pane displays a pop-up menu that allows you to perform the same actions that are available from the toolbar and menu bar.

Displaying Data Source Properties

- 1 Start the Configuration Manager. To start the Configuration Manager, select **Start / Programs**, and select **SequeLink ADO Client 5.1**. Then, select the **MERANT DataDirect Configuration Manager** application.
- 2 Double-click the **Data Sources** folder to display any existing ADO data sources.

- 3 Highlight a data source in the list. The properties of the data source display in the right pane. For example, [Figure 3-1](#) shows the properties of an ADO data source named SLOracle8 displayed in the right pane.

Figure 3-2. MERANT DataDirect Configuration Manager: Displaying Data Source Properties



You can right-click a data source in the left pane to display a pop-up menu. The pop-up menu offers the same actions for the item that are available from the Edit menu.

To display a setup window for an existing data source, double-click an ADO data source in the Data Sources folder.

To create a new data source, highlight the **Data Sources** folder; then, select **File / New / Data Source** from the menu bar.

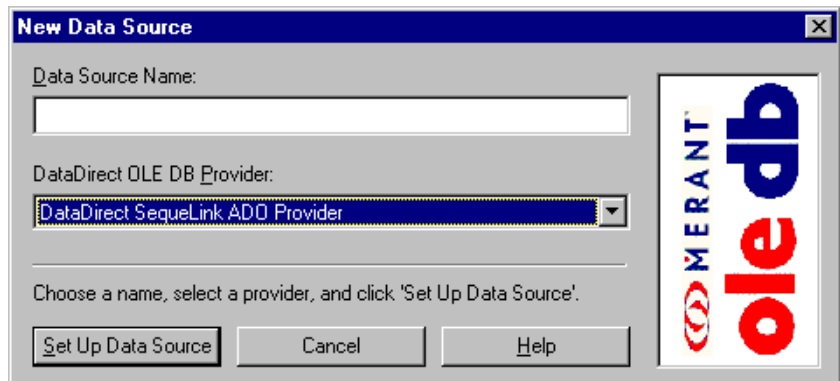
Configuring ADO Client Data Sources

The following sections provide instructions for configuring ADO client data sources:

- [“Creating an ADO Client Data Source” on page 103](#)
- [“Modifying an ADO Client Data Source” on page 106](#)
- [“Renaming an ADO Client Data Source” on page 106](#)
- [“Deleting an ADO Client Data Source” on page 107](#)
- [“Changing Data Source Directories” on page 107](#)
- [“Copying an ADO Client Data Source” on page 108](#)

Creating an ADO Client Data Source

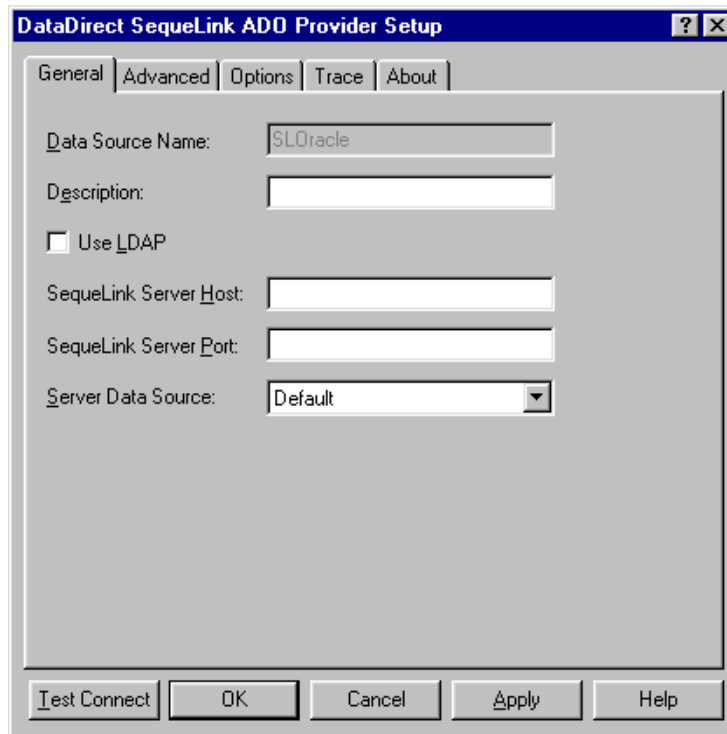
- 1 Start the DataDirect Configuration Manager. To start the Configuration Manager, select **Start / Programs**, and select **SequeLink ADO Client 5.1**. Then, select the **MERANT DataDirect Configuration Manager** application.
- 2 Select **File / New / Data Source** from the menu bar. The New Data Source window appears.



- 3 Type a name for the data source. All data sources located in the same directory must have unique names. If the name has

already been used for another data source, you will be prompted to enter a different name.

- 4 In the DataDirect OLE DB Providers drop-down list, select **DataDirect SequeLink ADO Provider**.
- 5 Click the **Set Up Data Source** button. The DataDirect SequeLink ADO Provider Setup window appears.



NOTE: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

- 6 Provide the following information.

Data Source Name: This is a read-only field that uniquely identifies this ADO data source configuration. Examples include "Accounting" or "SequeLink to Oracle Data."

Description: Optionally, type a description of the data source. For example, "My Accounting Database" or "Accounting Data in Oracle."

SequeLink Server Host: Type the TCP/IP host name of the SequeLink service to which you want the SequeLink ADO Client to connect. This field is available only if the Use LDAP check box is **not** selected.

SequeLink Server Port: Type the TCP/IP port the SequeLink service is listening on for incoming connection requests. The port you specify must be the same as the one that was specified for the SequeLink service when the SequeLink Server was installed; the default is 19996. This field is available only if the Use LDAP check box is **not** selected.

Server Data Source: Type the name of a server data source configured for the SequeLink service to use for the connection or select one from the drop-down list. This field is optional. If a server data source is not specified, the default server data source for that SequeLink service will be used for the connection. This field is available only if the Use LDAP check box is **not** selected.

NOTE FOR LDAP USERS: To configure the SequeLink ADO Client to retrieve connection information from an LDAP directory, select the **Use LDAP** check box. The fields change on the lower half of the screen to accommodate the information that is required to query an LDAP server for connection information. Provide the following information:

LDAP Server Host: Type the TCP/IP host name of the LDAP server.

LDAP Server Port: Type the TCP/IP port on which the LDAP server is listening for incoming connection requests. If unspecified, the SequeLink ADO Client will use the default LDAP port 389.

Distinguished Name (DN): Type an identifier that uniquely identifies the LDAP entry where connection information is stored.

For more information about retrieving connection information from LDAP directories, refer to the *SequeLink Administrator's Guide*.

NOTE: All data sources are saved to the current directory displayed in the Configuration Manager. For instructions on changing the current directory, see ["Changing Data Source Directories" on page 107](#).

Modifying an ADO Client Data Source

To modify the properties of a data source, double-click the data source in the Data Sources folder of the Configuration Manager to display the SequeLink ADO Provider Setup window. See ["Creating an ADO Client Data Source" on page 103](#) for a description of the fields you can change.

Renaming an ADO Client Data Source

You can rename data sources. You cannot rename or delete the Data Sources folder.

To rename a SequeLink ADO Provider data source:

- 1 Start the Configuration Manager. To start the Configuration Manager, select **Start / Programs**, and select **SequeLink ADO Client 5.1**. Then, select the **MERANT DataDirect Configuration Manager** application.
- 2 Select the data source you want to rename.

- 3 Select **Edit / Rename**. The data source name becomes an editable field.
- 4 Type the new name of the data source and press ENTER.

Deleting an ADO Client Data Source

- 1 Start the Configuration Manager. To start the Configuration Manager, select **Start / Programs**, and select **SequeLink ADO Client 5.1**. Then, select the **MERANT DataDirect Configuration Manager** application.
- 2 Select the data source you want to delete.
- 3 Select **Edit / Delete**.
- 4 A window appears prompting you to confirm the deletion. Click **Yes** to delete the selected data source.

Changing Data Source Directories

The Configuration Manager displays the SequeLink ADO data sources contained in the current directory, which is displayed in the status bar at the bottom of the Configuration Manager. The first time you start the Configuration Manager, the current directory defaults to the SequeLink ADO Client installation directory.

To change the current directory:

- 1 Click the **Change Current Directory** button on the tool bar.
- 2 Type the name of the new directory in the text field, or, click the **Browse** button to select a different directory.
- 3 Click **OK**.

After you change the current directory, the left pane of the Configuration Manager is automatically refreshed to display the data sources in the new directory.

The current directory remains active until you change it again. Any data sources you create are saved to the current directory.

Copying an ADO Client Data Source

Copying a data source can make it easier for you to configure new data sources that use the same properties as existing data sources. When you copy a data source, the copied data source retains all the properties of the original data source. After copying, you can modify the properties of the data source as needed.

To copy a data source:

- 1 In Windows Explorer, navigate to the directory that contains the data source you want to copy. All SequeLink ADO Provider data sources use .IDS as their file extension. For example, if the data source name appears as TEST in the Configuration Manager, the name of the data source file is TEST.IDS.

NOTE: The directory location of a data source displayed in the Configuration Manager appears in the status bar at the bottom of the Configuration Manager.

- 2 Copy the data source to the Windows Explorer clipboard; then, perform one of the following actions:
 - To copy to a different directory, navigate to the directory you want to copy to and paste the data source in that new directory. You can use the same data source name.
 - To copy to the same directory, paste the data source; then, rename the data source to a *unique* name. The copied data source will not be recognized by the Configuration

Manager unless you rename it with a name that is not used by any other the data sources in the directory.

- 3 To display the new data source in the Configuration Manager, perform one of the following actions:
 - If you copied the data source to a different directory, make that directory the current directory in the Configuration Manager by selecting **Tools / Options / Change current directory**. The new data source appears in the Data Sources folder.
 - If you copied the data source to the same directory and renamed the data source, select **View / Refresh** in the Configuration Manager. The new data source appears in the Data Sources folder.

Testing ADO Connections

You can connect to a data source using a Connection window, or using a provider string. For information about connecting using an ADO provider string, see [“Connecting with a Provider String” on page 117](#).

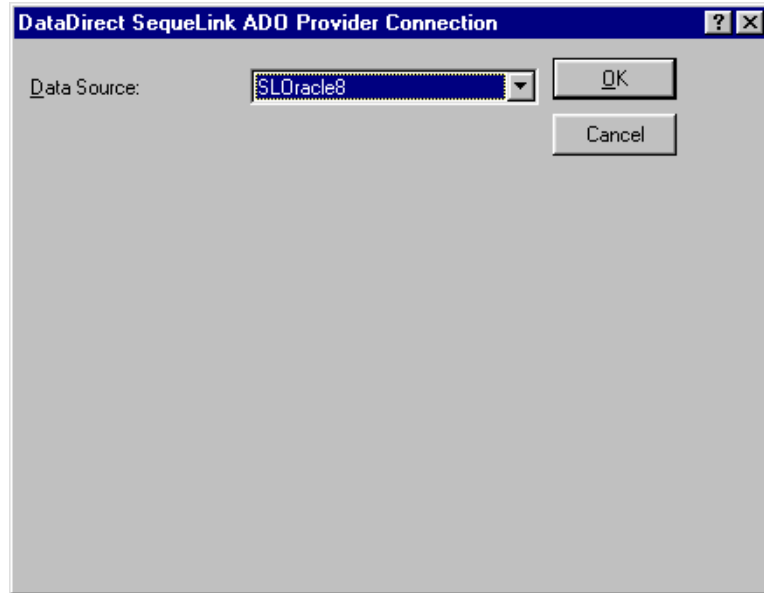
The SequeLink ADO Provider opens a Connection window when you perform either of the following actions:

- You request a connection to a SequeLink ADO Provider from within your data consumer, and your data consumer requests the SequeLink ADO Provider to prompt for missing connection parameters.
- You click **Test Connect** in a SequeLink ADO Provider setup window to test the connection to a data source you have set up.

For more information about ADO connection dialogs that may appear, see [“ADO Connection Dialogs” on page 110](#).

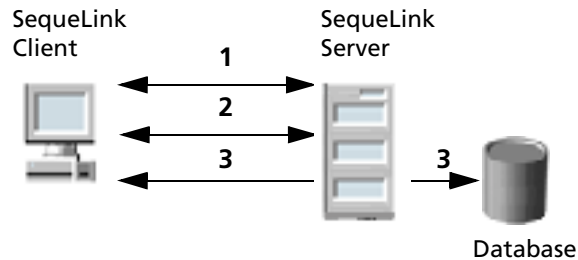
ADO Connection Dialogs

When your data consumer requests the SequeLink ADO Provider to prompt for missing connection parameters and an ADO data source has not been specified, the DataDirect SequeLink ADO Provider Connection window appears.



Select the data source that you want to use from the drop-down list. If you do not want to specify a data source name, select **None** from the drop-down list. In some cases, the data source name may be supplied automatically. Then, click **OK**.

The other connection dialogs that may appear involve prompting for information required to make a SequeLink data access connection. A SequeLink data access connection involves the following stages:



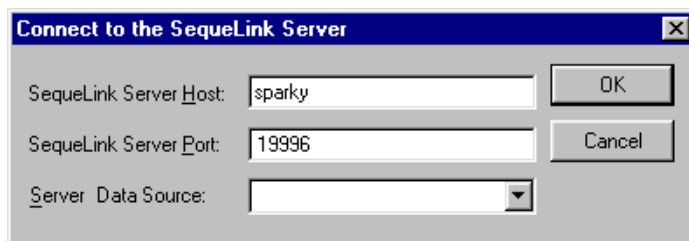
- 1 A network connection is established.
- 2 An authentication mechanism is used to establish the identity of the SequeLink Client to the SequeLink Server.
- 3 Based on information provided by the SequeLink Client application (for example, a database user name and password), a database connection is established.

Stage 1: Establishing a Network Connection

The first stage of the connection process involves establishing a network connection. The dialog that appears depends on whether the connection has been configured to connect directly to a SequeLink service or to retrieve connection information for the SequeLink service from a centralized LDAP directory.

Connecting Directly to a SequeLink Service

If the connection has been configured to connect directly to a SequeLink service, the following window appears:



The screenshot shows a dialog box titled "Connect to the SequeLink Server". It has a standard Windows-style title bar with a close button (X) in the top right corner. The dialog contains three input fields. The first is labeled "SequeLink Server Host" and contains the text "sparky". The second is labeled "SequeLink Server Port" and contains the text "19996". The third is labeled "Server Data Source" and is a drop-down menu. To the right of the "SequeLink Server Host" field is an "OK" button. To the right of the "SequeLink Server Port" field is a "Cancel" button.

Provide the following information; then, click **OK**.

SequeLink Server Host: Type the TCP/IP host name of the SequeLink service.

SequeLink Server Port: Type the TCP/IP port on which the SequeLink service is listening. A default installation of SequeLink Server uses the port 19996.

Server Data Source: Type the name of a server data source to use for the connection or select one from the drop-down list. This step is optional. If a server data source is not specified, the default server data source for that service will be used for the connection.

Retrieving Connection Information from an LDAP Directory

If the connection has been configured to connect to an LDAP server to retrieve connection information from an LDAP directory, the following window appears:



The screenshot shows a dialog box titled "Connect to the SequeLink Server". It has a standard Windows-style title bar with a close button (X) in the top right corner. The dialog contains three text input fields stacked vertically. The first field is labeled "LDAP Server Host:" and contains the text "sparky". The second field is labeled "LDAP Server Port:" and is empty. The third field is labeled "Distinguished Name:" and is empty. To the right of the first field is an "OK" button, and to the right of the second field is a "Cancel" button.

Provide the following information; then, click **OK**.

LDAP Server Host: Type the TCP/IP host name of the LDAP server.

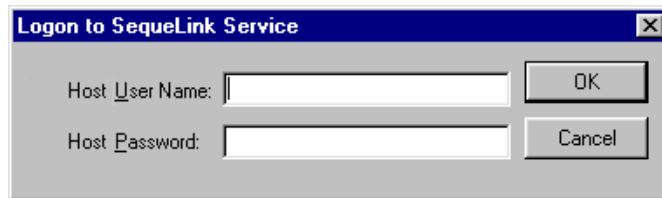
LDAP Server Port: Type the TCP/IP port on which the LDAP server is listening.

Distinguished Name: Type the Distinguished Name (DN) of the LDAP entry.

Stage 2: SequeLink Server Authentication

The second stage of the connection process involves authentication of the SequeLink Client to the SequeLink Server. The dialogs that appear depend on how authentication is configured for the SequeLink service.

- When ServiceAuthMethods=anonymous or ServiceAuthMethods=integrated_nt, no dialogs appear.
- When ServiceAuthMethods=OSLogon(HUID,HPWD) or ServiceAuthMethods=OSLogon(UID,PWD), the following dialog appears:



Provide the following information; then, click **OK**.

Host User Name: Type the host user name.

NOTE: When connecting to a Windows NT server, you must prefix the host user name with a server name, if authenticating to a local server, or a domain name (for example, SALES\DJONES). If the server name or domain name is omitted, the SequeLink Server will attempt to authenticate the user ID and password with the database account defined for the machine on which the SequeLink Server is running. If this validation fails, the SequeLink Server will attempt to authenticate the user ID and password with the database account defined for the domain of the machine on which the SequeLink Server is running.

Host Password: Type the host password.

- When ServiceAuthMethods=OSLogon(HUID,HPWD,NPWD) or ServiceAuthMethods=OSLogon(UID,PWD,NPWD) and the password is expired, the following dialog appears:

NOTE: If the password is not expired, the previous dialog appears. You are only prompted for the Host User Name and Host Password.

Provide the following information; then, click **OK**.

Host User Name: Type the host user name.

NOTE: When connecting to a Windows NT server, you must prefix the host user name with a server name, if authenticating to a local server, or a domain name (for example, SALES\DJONES). If the server name or domain name is omitted, the SequeLink Server will attempt to authenticate the user ID and password with the database account defined for the machine on which the SequeLink Server is running. If this validation fails, the SequeLink Server will attempt to authenticate the user ID and password with the database account defined for the domain of the machine on which the SequeLink Server is running

Host Password: Type the host password.

New Password: Type the new password to be used by the SequeLink password change mechanism.

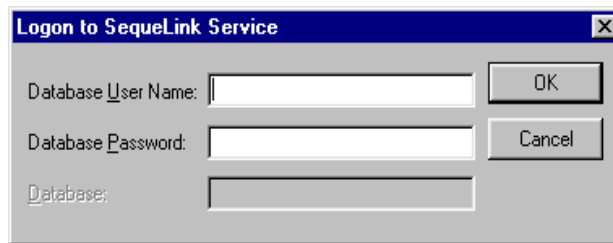
Confirm Password: Type again the new password to confirm it.

For more information about configuring authentication, refer to the *SequeLink Administrator's Guide*.

Stage 3: Data Store Logon

The last stage of the connection process involves logging on the data store. The dialogs that appear depend on the data store logon method configured for the SequeLink service:

- When `DataSourceLogonMethod=OSIntegrated`, no dialogs appear.
- When `DataSourceLogonMethod=DBMSLogon(UID,PWD)` or `DataSourceLogonMethod=DBMSLogon(DBUID,DBPWD)`, a data store-specific user name and password are required and the following dialog appears:



Provide the following information; then, click **OK**.

Database User Name: Type the database logon ID.

Database Password: Type the database password.

Database: Type the name of the database to which you want to connect. This field is disabled when the data store does not recognize the concept of databases.

For more information about configuring data store logon methods, refer to the *SequeLink Administrator's Guide*.

Connecting with a Provider String

Once a data source is defined through the DataDirect Configuration Manager and the SequeLink ADO Provider Setup Assistant, your application can connect directly to that data source. You can override the current settings for the data source when you connect using a *provider string*.

A provider string contains *attribute=value* pairs that control various aspects of the data provider's connection and interaction with the database. When an application names a specific data source to connect to, the application can also pass the data provider a provider string of *attribute=value* pairs. The data provider uses the values in the provider string instead of the default values defined for the data source in the system information.

Using provider strings allows application developers to configure connections for users programmatically and ensures that users have the optimum settings for working with the provider and database. Any values a user has set for a data source through the DataDirect Configuration Manager are overridden by corresponding values in the provider string for the current session only.

The provider string sets the DBPROP_INIT_PROVIDERSTRING initialization property and has the form:

```
"attribute=value;attribute=value;"
```

For a list of ADO connection attributes, see ["ADO Connection Attributes" on page 118](#).

ADO Connection Attributes

[Table 3-2](#) provides a list of ADO connection attributes supported by the SequeLink ADO Provider. It lists a description for each attribute. The defaults listed in [Table 3-2](#) are initial defaults that apply when no value is specified in the provider string or in the data source definition in the system information. If you specified a value for the attribute when configuring the data source in the Setup window, that value is your default.

Table 3-2. ADO Connection Attributes

Attribute	Description
Application ID	<p>Specifies the application ID that identifies the client application to the SequeLink service. This attribute is only required when the SequeLink service you are connecting to has been configured to limit access to specific applications.</p> <p>For more information about using application IDs to limit access to SequeLink services, see “Specifying Application IDs” on page 200.</p>
Automatic Application ID	<p>Specifies an application ID that is automatically generated by the SequeLink ADO Client to identify the client application to the SequeLink service. This attribute is only required when the SequeLink service you are connecting to has been configured to limit access to specific applications.</p> <p>For more information about using application IDs to limit access to SequeLink services, see “Specifying Application IDs” on page 200.</p>
Database	<p>Specifies the name of the database to which you want to connect.</p>
Database User Name	<p>Specifies the data store user name, which may be required depending on the server configuration.</p>
Database Password	<p>Specifies the data store password, which may be required depending on the server configuration.</p>

Table 3-2. ADO Connection Attributes (cont.)

Attribute	Description
Data Source	Specifies a string that identifies an ADO/OLE DB data source configuration. Examples include "Accounting" or "SequeLink to Oracle Data."
Default Length for Long Data	Turns on a workaround that allows you to specify the amount of data (in KB) that is buffered for SQL_LONGVARCHAR and SQL_LONGVARBINARY columns with a static cursor. The default is 4.
Distinguished Name	Specifies the distinguished name identifying the LDAP entry from which connection information is retrieved. This attribute is required when UseLDAP=1.
Host	Specifies the TCP/IP address of the SequeLink Server, specified in dotted format or as a host name. LDAP: If LDAP is enabled, this identifies the TCP/IP address of the LDAP server. This can also be a list of LDAP servers separated by a blank space (for example, "ld1.foo.com ld2.foo.com ld3.foo.com"). If the first LDAP server in the list does not respond, the SequeLink ADO Client will try to connect to the next LDAP server in the list.
Host Password	Specifies the host password, which may be required depending on the server configuration.
Host User Name	Specifies the host user name, which may be required depending on the server configuration.

Table 3-2. ADO Connection Attributes (cont.)

Attribute	Description
New Password	<p>Specifies the new host password to be used. If specified and applicable to the connection, the SequeLink password change mechanism is invoked. When the password has been changed successfully, the following warning is generated:</p> <pre data-bbox="572 487 1232 578">[MERANT] [SequeLink ADO Provider] [SequeLink Server] The user password was changed successfully</pre> <p>If unspecified and the SequeLink Server detects that the host password has expired, you will be prompted for a new host password.</p> <p>For more information about the SequeLink password change mechanism, refer to the <i>SequeLink Administrator's Guide</i>.</p>
Password	<p>Specifies the host or data store password, which may be required depending on the server configuration.</p>
Port	<p>Specifies the TCP/IP port on which the SequeLink Server is listening.</p> <p>LDAP: If LDAP is enabled, this identifies the TCP/IP port on which the LDAP server is listening. If you do not specify a port, the default port for LDAP (389) will be used.</p>
Server Data Source	<p>Optionally, identifies the server data source to be used for the connection. If not specified, the configuration of the default server data source will be used for the connection.</p>

Table 3-2. ADO Connection Attributes (cont.)

Attribute	Description
Use LDAP	<p data-bbox="605 314 808 343">UseLDAP={0 1}.</p> <p data-bbox="605 357 1225 453">Determines whether the parameters to establish a connection to the SequeLink Server should be retrieved from LDAP.</p> <p data-bbox="605 466 1279 527">When set to 0 (the initial default), the SequeLink Client will connect directly to the specified SequeLink Server.</p> <p data-bbox="605 541 1279 760">When set to 1, the SequeLink Client will retrieve the TCP/IP host, TCP/IP port, and SequeLink data source (optional) from an LDAP entry identified by a Distinguished Name (DN). Once the connection information is retrieved, the SequeLink Client will connect directly to the specified SequeLink Server. The DistinguishedName (DN) attribute is required.</p>
User ID	<p data-bbox="605 774 1279 838">Specifies the host or data store user name, which may be required depending on the server configuration.</p>

4 Developing ADO Applications

This chapter provides information about developing ADO applications for SequeLink environments including:

- [“OLE DB Objects and Interfaces” on page 124](#)
- [“Supported Schema Rowsets” on page 133](#)
- [“Supported OLE DB Property Groups” on page 134](#)
- [“OLE DB Interfaces Supported in ADO” on page 162](#)
- [“Mapping ADO Methods and Properties” on page 164](#)
- [“Data Shaping” on page 197](#)
- [“Persisting Information” on page 198](#)
- [“Using Rowsets” on page 198](#)
- [“Unicode Support” on page 199](#)
- [“Mapping Data Types” on page 199](#)
- [“Specifying Application IDs” on page 200](#)
- [“Error Handling” on page 201](#)

OLE DB Objects and Interfaces

The SequeLink ADO Provider supports Insert, Update, and Delete operations through the OLE DB Rowset interfaces and through the command interfaces (using SQL DML).

[Table 4-1](#) lists the OLE DB objects that the SequeLink ADO Provider supports, and the interfaces implemented for each object.

Table 4-1. Objects and Interfaces Supported by the SequeLink ADO Provider

OLE DB Object	Interface	
ErrorLookup (see also “ErrorLookup” on page 126.)	IErrorLookup	
TCommand (see also “TCommand” on page 126.)	IAccessor IColumnsInfo ICommand ICommandProperties ICommandText	IConvertType IColumnsRowset ICommandPrepare ICommandWithParameters ISupportErrorInfo
TDataSource (see also “TDataSource” on page 127.)	IDBCreateSession IDBInfo IDBInitialize IDBProperties	IPersist IPersistFile ISupportErrorInfo
TEumerator (see also “TEumerator” on page 128.)	IDBInitialize IDBProperties IParseDisplayName ISourcesRowset ISupportErrorInfo	

Table 4-1. Objects and Interfaces Supported by the SequeLink ADO Provider (cont.)

OLE DB Object	Interface	
TMultipleResults (see also "TMultipleResults" on page 129.)	IMultipleResults	ISupportErrorInfo
TRowset (see also "TRowset" on page 129.)	IAccessor	IRowsetIdentity
	IColumnsInfo	IRowsetInfo
	IConvertType	IRowsetLocate
	IColumnsRowset	IRowsetScroll
	IRowset	IRowsetUpdate
	IRowsetChange	ISupportErrorInfo
TSession (see also "TSession" on page 130.)	IDBCreateCommand	ISupportErrorInfo
	IDBSchemaRowset	ITransaction
	IGetDataSource	ITransactionJoin
	IOpenRowset	ITransactionLocal
	ISessionProperties	
TTransaction (see also "TTransaction" on page 132.)	ITransaction	
	ISupportErrorInfo	
TTransactionOptions (see also "TTransactionOptions" on page 132.)	ITransactionOptions	
	ISupportErrorInfo	

ErrorLookup

The ErrorLookup object contains detailed information about an OLE DB error. Unlike the other OLE DB objects that the SequeLink ADO Provider supports, ErrorLookup is not considered to be an OLE DB CoType.

ErrorLookup is used by OLE DB error objects to determine the values of the error message, source, help file path, and context ID based on the return code and a provider-specific error number.

TCommand

A command is an OLE DB object that is used to process a provider-specific text command, such as an SQL statement. Commands are generally used for data definition and data manipulation.

The SequeLink ADO Provider supports Insert, Update, and Delete operations by using SQL DML statements through the OLE DB command interfaces.

The SequeLink ADO Provider supports the following interfaces for TCommand:

- IAccessor. Provides methods for accessor management. An accessor is a collection of information that describes how data is stored in the consumer's buffer.
- IColumnsInfo. The simpler of two interfaces which can be used to expose information about columns of a rowset or prepared command. It provides a limited set of information in an array.
- IColumnsRowset. Supplies complete information about columns in a rowset.
- ICommand. Contains methods to execute commands.

- **ICommandPrepare**. Encapsulates command optimization, a separation of compile-time and run-time, as found in traditional relational database systems. The result of this optimization is a command execution plan.
- **ICommandProperties**. Specifies the properties from the Rowset property group that must be supported by the rowsets returned by `ICommand::Execute`.
- **ICommandText**. Gets and sets the command text.
- **ICommandWithParameters**. Encapsulates parameters. Parameter values are set when the command is processed.
- **IConvertType**. Contains a single method that gives information on the availability of type conversions on a command or on a rowset.
- **ISupportErrorInfo**. Indicates whether a specific OLE DB interface can return OLE Automation error objects.

TDataSource

A data source object is the initial object that a data provider instantiates for a consumer. A data source object can support multiple sessions and multiple transactions.

Generally, the consumer binds to the file moniker of a data source object or to a moniker returned by the enumerator object. This has the effect of calling `CoCreateInstance` on the class ID for the data provider.

The `SequeLink ADO Provider` supports the following interfaces for `TDataSource`:

- **IDBCreateSession**. Creates a new session from the data source object and returns the requested interface on the newly created session.
- **IDBInitialize**. Initializes the data source object.

- **IDBInfo**. Returns information about the keywords and literals that the SequeLink ADO Provider supports.
- **IDBProperties**. Sets and gets the values of properties on the data source object, and get information about all properties for the data source object.
- **IPersist**. Persists a data source object.
- **IPersistFile**. Persists a data source object to a file.
- **ISupportErrorInfo**. Indicates whether a specific OLE DB interface can return OLE Automation error objects.

TEnumerator

An enumerator is an object that searches for data sources and other enumerators. The OLE DB SDK provides an enumerator that lists all configured OLE DB data providers. A data provider's enumerator lists all of the data sources configured for the data provider.

The SequeLink ADO Provider supports the following interfaces for TEnumerator:

- **IParseDisplayName**. To instantiate a data source listed in the data source rowset, the consumer first calls `IParseDisplayName::ParseDisplayName` for the returned display name. This method returns a moniker, which the consumer can then bind to instantiate the data source object.
- **ISourcesRowset**. Returns a rowset of data sources visible from the current enumerator. This will be a rowset of data sources configured for the SequeLink ADO Provider.
- **ISupportErrorInfo**. Indicates whether a specific OLE DB interface can return OLE Automation error objects.

TMultipleResults

Multiple results can be returned:

- If the command text comprises multiple separate text commands, such as a batch of SQL statements
- If more than one set of parameters is passed to a command

A multiple results object retrieves multiple results. It is created by an execute operation.

The SequeLink ADO Provider supports the following interfaces for TMultipleResults:

- IMultipleResults. Retrieves multiple results created by a command. The SequeLink ADO Provider by default returns only a single result set for any command execution.
- ISupportErrorInfo. Indicates whether a specific OLE DB interface can return OLE Automation error objects.

TRowset

The SequeLink ADO Provider supports the following interfaces for TRowset:

- IAccessor. Provides methods for accessor management. An accessor is a collection of information that describes how data is stored in the consumer's buffer.
- IColumnsInfo. The simpler of two interfaces that can be used to expose information about columns of a rowset or prepared command. It provides a limited set of information in an array.
- IColumnsRowset. Supplies complete information about columns in a rowset.

- **IConvertType**. Contains a single method that gives information on the availability of type conversions on a command or on a rowset.
- **IRowset**. The base rowset interface. It provides methods for fetching rows sequentially, getting the data from those rows, and managing rows.
- **IRowsetChange**. Updates the values of columns in existing rows, deletes existing rows, and inserts new rows.
- **IRowsetIdentity**. Indicates that row instance identity is implemented on the rowset and enables testing for row identity. If a rowset supports this interface, any two row handles representing the same underlying row will always reflect the same data and state.
- **IRowsetInfo**. Provides a method to get properties from the Rowset property group.
- **IRowsetLocate**. Fetches arbitrary rows of a rowset.
- **IRowsetScroll**. Enables consumers to fetch rows at approximate positions in the rowset, get the approximate position of a row corresponding to a specified bookmark, and fetch rows starting from a fractional position in the rowset.
- **IRowsetUpdate**. Enables consumers to delay the transmission of changes to the data source. It also enables consumers to undo changes before transmission, get list of rows with pending changes, and so on.
- **ISupportErrorInfo**. Indicates whether a specific OLE DB interface can return OLE Automation error objects.

TSession

The primary function of a session is to define a transaction. From a session, a consumer can create a command or a rowset.

A session can be inside or outside of a transaction at any given time. When a session is created, it is outside of a transaction. Any work done in the session is automatically committed.

If a session supports `ITransactionLocal`, the consumer can call `ITransactionLocal::StartTransaction` to start an explicit transaction. The session is then in manual commit mode; any work in the session must be explicitly committed or ended abnormally.

The `SequeLink ADO Provider` supports the following interfaces for `TSession`:

- `IDBCreateCommand`. Consumers call this on a session to obtain a new command.
- `IDBSchemaRowset`. Provides advanced schema information about the data source.
- `IGetDataSource`. Obtains an interface pointer to the data source object.
- `IOpenRowset`. Enables consumers to open and work directly with tables in a data source by using `IOpenRowset::OpenRowset`, which generates a rowset of all rows in the table.
- `ISessionProperties`. Returns information about the properties a session supports and the current settings of those properties.
- `ISupportErrorInfo`. Indicates whether a specific OLE DB interface can return OLE Automation error objects.
- `ITransaction`. Used to start, commit, and abort transactions.
- `ITransactionJoin`. Used to control coordinated transactions. The consumer calls this interface to determine whether the provider supports coordinated transactions, and to enlist the session in a coordinated transaction. Do not call `ITransaction::JoinTransaction` if the session is already participating in either a local or coordinated transaction.

- **ITransactionLocal**. Used to commit and obtain status information about transactions at the current level of a nested transaction. This interface inherits from **ITransaction**.

TTransaction

A transaction is used to define persistent units of work within an application.

The SequeLink ADO Provider supports the following interfaces for **TTransaction**:

- **ISupportErrorInfo**. Indicates whether a specific OLE DB interface can return OLE Automation error objects.
- **ITransaction**. Used to start, commit, and abort transactions.

TTransactionOptions

The SequeLink ADO Provider supports extended transaction functionality. Therefore, it must support **TTransactionOptions**. The consumer can call **ITransactionLocal::GetOptionsObject** to obtain the transaction options object that can be used to specify configuration options for a subsequent call to **ITransactionLocal::StartTransaction**.

The SequeLink ADO Provider supports the following interfaces for **TTransactionOptions**:

- **ISupportErrorInfo**. Indicates whether a specific OLE DB interface can return OLE Automation error objects.
- **ITransactionOptions**. Gets and sets a suite of options associated with a transaction.

Supported Schema Rowsets

[Table 4-2](#) lists the OLE DB schema rowsets supported by the SequeLink ADO Provider.

Table 4-2. OLE DB Schema Rowsets Supported by the SequeLink ADO Provider

CATALOGS ^{(1), (2)}	PROCEDURE_PARAMETERS
COLUMN_PRIVILEGES ^{(1), (2)}	PROCEDURES
COLUMNS	PROVIDER_TYPES
FOREIGN_KEYS	SCHEMATA ^{(1), (2)}
INDEXES	STATISTICS
PRIMARY_KEYS	TABLE_PRIVILEGES ^{(1), (2)}
PROCEDURE_COLUMNS ^{(1), (2)}	TABLES

Notes:

¹ Not supported for DB2

² Not supported for Oracle

Supported OLE DB Property Groups

The data provider defines the properties that apply to data sources, and properties that provide a read-only set of information about the data provider and the data source.

To obtain a data provider's property values, a data consumer calls one of the methods listed in [Table 4-3](#).

Table 4-3. OLE DB Property Groups Supported by the SequeLink ADO Provider

Property Group	Method Used to Obtain Values
Data Source	IDBProperties::GetProperties
Data Source Information	IDBProperties::GetProperties
Initialization	IDBProperties::GetProperties
Rowset	ICommandProperties::GetProperties IRowsetInfo::GetProperties
Session	ISessionProperties::GetProperties

Data Source Property Group

The SequeLink ADO Provider supports the following property in the DBPROP_DATASOURCE property set. For more information, refer to your Microsoft OLE DB programming documentation.

Table 4-4. OLE DB Data Source Property Supported by the SequeLink ADO Provider

DBPROP_CURRENTCATALOG	The name of the current catalog. The data consumer can use the CATALOGS rowset to enumerate catalogs. If unspecified, the data provider uses the default catalog.
-----------------------	---

Data Source Information Property Group

[Table 4-5](#) lists the properties in the DBPROPSET_DATASOURCEINFO property set supported by the SequeLink ADO Provider. These properties are in the Data Source Information property group, are read-only properties, and constitute a set of static information about the data provider and data source. For more information about these properties, refer to your Microsoft OLE DB programming documentation.

NOTE: Some values are database-specific and depend on the SequeLink service you are using. These database-specific values are not listed in the table.

Table 4-5. OLE DB Data Source Information Properties Supported by the SequeLink ADO Provider

Property ID	Description and Default
DBPROP_ACTIVESESSIONS	Specifies the maximum number of sessions that can exist at one time. VALUE=0. There is no limit.
DBPROP_ASYNCCTXNABORT	Specifies whether the provider can abort transactions asynchronously. VALUE=VARIANT_FALSE
DBPROP_ASYNCCTXNCOMMIT	Specifies whether the provider can commit transactions asynchronously. VALUE=VARIANT_FALSE
DBPROP_BYREFACCESSORS	Specifies whether the SequeLink ADO Provider supports the DBACCESSOR_PASSBYREF flag in IAccessor::CreateAccessor. This applies to both row and parameter accessors. VALUE=VARIANT_FALSE.
DBPROP_CATALOGLOCATION	Specifies the position of the catalog name in a qualified table name in a text command. The value depends on the SequeLink service you are using. Possible values are: VALUE=DBPROPVAL_CL_START. The catalog name is at the start of the fully qualified name. VALUE=DBPROPVAL_CL_END. The catalog name is at the end of the fully qualified name.
DBPROP_CATALOGTERM	Specifies the name the data source uses for a catalog. VALUE="Database"

Table 4-5. OLE DB Data Source Information Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_CATALOGUSAGE	<p>Specifies how catalog names can be used in text commands. A combination of zero or one or more of the following:</p> <p>VALUE=DBPROPVAL_CU_DML_STATEMENTS. Catalog names are supported in all Data Manipulation Language (DML) statements.</p> <p>VALUE=DBPROPVAL_CU_TABLE_DEFINITION. Catalog names are supported in all table definition statements.</p> <p>VALUE=DBPROPVAL_CU_INDEX_DEFINITION. Catalog names are supported in all index definition statements.</p> <p>VALUE=DBPROPVAL_CU_PRIVILEGE_DEFINITION. Catalog names are supported in all privilege definition statements.</p>
DBPROP_COLUMNDEFINITION	<p>Specifies the valid clauses for defining a column.</p> <p>VALUE=DBPROPVAL_CD_NOTNULL. Columns can be created non-nullable.</p>
DBPROP_CONCATNULLBEHAVIOR	<p>Specifies how the data source handles the concatenation of null-valued character data type columns with non-NULL-valued character data type columns. The value depends on the SequeLink service you are using. Possible values are:</p> <p>VALUE=DBPROPVAL_CB_NULL. The result is null valued.</p> <p>VALUE=DBPROPVAL_CB_NON_NULL. The result is the concatenation of the non-NULL-valued column or columns.</p>

Table 4-5. OLE DB Data Source Information Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_CONNECTIONSTATUS	Specifies the status of the current connection. VALUE=DBPROPVAL_CS_INITIALIZED. The data source is in an initialized state and able to communicate with the data store.
DBPROP_DATASOURCENAME	Specifies the name of the data source used during connection. The data source is defined using the MERANT DataDirect Configuration Manager. For more information about configuring ADO data sources, see Chapter 3 "Using the SequeLink ADO Client" on page 97 .
DBPROP_DATASOURCEREADONLY	Specifies whether the data source is read-only. VALUE=VARIANT_FALSE. The data source can be updated.
DBPROP_DBMSNAME	Specifies the name of the data store accessed by the SequeLink ADO Provider. The value depends on the SequeLink service you are using.
DBPROP_DBMSVER	Specifies the version of the DBMS that the provider is currently accessing. This value depends on the SequeLink service you are using.
DBPROP_DSOTHRADMODEL	Specifies the threading model of the data source object. VALUE=DBPROP_RT_FREETHREAD. The SequeLink ADO Provider supports the free-threading model.

Table 4-5. OLE DB Data Source Information Properties Supported by the Sequelink ADO Provider (cont.)

Property ID	Description and Default
DPROP_GROUPBY	<p>Specifies the relationship between the columns in a GROUP BY clause and the nonaggregated columns in the select list. This value depends on the Sequelink service you are using. Possible values are:</p> <p>VALUE=DBPROPVAL_GB_EQUALS_SELECT. The GROUP BY clause must contain all nonaggregated columns in the select list. It cannot contain any other columns. For example, SELECT DEPT, MAX(SALARY) FROM EMPLOYEE GROUP BY DEPT</p> <p>VALUE=DBPROPVAL_GB_COLLATE. A COLLATE clause can be specified at the end of each grouping column.</p> <p>VALUE=DBPROPVAL_GB_CONTAINS_SELECT. The GROUP BY clause must contain all nonaggregated columns in the select list. It can contain columns that are not in the select list. For example, SELECT DEPT, MAX(SALARY) FROM EMPLOYEE GROUP BY DEPT, AGE.</p> <p>VALUE=DBPROPVAL_GB_NO_RELATION. The columns in the GROUP BY clause and the select list are not related. The meaning of nongrouped, nonaggregated columns in the select list is data source-dependent. For example, SELECT DEPT, SALARY FROM EMPLOYEE GROUP BY DEPT, AGE</p>
DBPROP_HETEROGENEOUSTABLES	<p>Specifies whether the provider can join tables from different catalogs or providers.</p> <p>VALUE=0. Heterogeneous joins are not supported.</p>

Table 4-5. OLE DB Data Source Information Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_IDENTIFIERCASE	<p>Specifies how identifiers handle case. Possible values are:</p> <p>VALUE=DBPROPVAL_IC_SENSITIVE. Case identifiers in SQL are case-sensitive and are stored in mixed case in the system catalog.</p> <p>VALUE=DBPROPVAL_IC_UPPER. Case identifiers in SQL are case-sensitive and are stored in upper case in the system catalog.</p> <p>VALUE=DBPROPVAL_IC_LOWER. Case identifiers in SQL are case-sensitive and are stored in lower case in the system catalog.</p> <p>VALUE=DBPROPVAL_IS_MIXED. Case identifiers in SQL are case-insensitive and are stored in mixed case in the system catalog.</p> <p>This value depends on the SequeLink service you are using.</p>
DBPROP_MAXINDEXSIZE	<p>Specifies the maximum number of bytes allowed in the combined columns of an index.</p> <p>VALUE=0. There is no limit.</p> <p>This value depends on the SequeLink service you are using.</p>
DBPROP_MAXROWSIZE	<p>Specifies the maximum length of a single row in a table.</p> <p>VALUE=0. There is no limit.</p>
DBPROP_MAXROWSIZEINCLUDESBLOB	<p>Specifies the maximum row size returned for the property includes the length of all BLOB data.</p> <p>VALUE=VARIANT_FALSE. The maximum row size does not include the length of all BLOB data.</p>

Table 4-5. OLE DB Data Source Information Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_MAXTABLEINSELECT	<p>Specifies the maximum number of tables allowed in the From clause of a Select statement.</p> <p>VALUE=0 is the default. A value of 0 means no limit.</p>
DBPROP_MULTIPLEPARAMSETS	<p>Specifies whether the provider supports multiple parameter sets.</p>
DBPROP_MULTIPLERESULTS	<p>Specifies whether the provider supports multiple results objects and the restrictions it places on these objects.</p> <p>VALUE=DBPROPVAL_MR_SUPPORTED. The provider supports multiple results objects.</p>
DBPROP_MULTIPLESTORAGEOBJECTS	<p>Specifies whether the provider supports multiple, open storage objects at the same time.</p>
DBPROP_MULTITABLEUPDATE	<p>Specifies whether the provider can update rowsets derived from multiple tables.</p> <p>VALUE=VARIANT_FALSE. The provider cannot update rowsets derived from multiple tables.</p>
DBPROP_NULLCOLLATION	<p>Specifies where NULLS are sorted in a list. The value depends on the SequeLink service you are using. Possible values are:</p> <p>VALUE=DBPROP_NC_END. NULLS are sorted at the end of the list, regardless of the sort order.</p> <p>VALUE=DBPROP_NC_HIGH. NULLS are sorted at the high end of the list.</p> <p>VALUE=DBPROP_NC_LOW. NULLS are sorted at the low end of the list.</p> <p>VALUE=DBPROP_NC_START. NULLS are sorted at the start of the list, regardless of the sort order.</p>

Table 4-5. OLE DB Data Source Information Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_OLEOBJECTS	<p>Specifies the way that the provider supports access to BLOBs and OLE objects stored in columns.</p> <p>VALUE=DBPROPVAL_OO_BLOB. The SequeLink ADO Provider supports access to BLOBs as structured storage objects. A data consumer determines which interfaces are supported through DBPROP_STRUCTUREDSTORAGE.</p>
DBPROP_OPENROWSET SUPPORT	<p>Describes support for opening objects through IOpenRowset.</p> <p>VALUE=DBPROPVAL_OR_S_TABLE. The SequeLink ADO Provider supports opening tables through IOpenRowset.</p>
DBPROP_ORDERBYCOLUMNSINSELECT	<p>Specifies the way that the provider orders columns in an Order By clause.</p> <p>VALUE=VARIANT_FALSE. Columns in an Order By clause do not have to be in the Select list.</p> <p>VALUE=VARIANT_TRUE. Columns in an Order By clause must be in the Select list.</p> <p>This value depends on the SequeLink service you are using.</p>
DBPROP_OUTPUTPARAMETERAVAILABILITY	<p>Specifies the time at which output parameter values become available.</p> <p>VALUE=DBPROPVAL_OA_ATROWRELEASE. If a command returns a single result that is a rowset, output parameter data is available at the time the rowset is completely released. If a command returns multiple results, output parameter data is available when IMultipleResults::GetResult returns DB_S_NORESULT or the multiple results object is completely released, depending on which occurs first. Before the output parameter data is available, the consumer's bound memory is in an indeterminate state.</p>

Table 4-5. OLE DB Data Source Information Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_PERSISTENTIDTYPE	<p>Specifies the type of DBID that the provider uses when storing (persisting) DBIDs for tables and columns.</p> <p>VALUE=DBPROPVAL_PT_NAME</p>
DBPROP_PREPAREABORTBEHAVIOR	<p>Specifies how aborting a transaction affects prepared commands.</p> <p>VALUE=DBPROPAL_CB_PRESERVE. Aborting a transaction preserves prepared commands. The application can run commands again without preparing them again.</p> <p>VALUE=DBPROPAL_CB_DELETE. Aborting a transaction deletes prepared commands. The application must prepare commands again before it can run them again.</p>
DBPROP_PREPARECOMMITBEHAVIOR	<p>Specifies how committing a transaction affects prepared commands.</p> <p>VALUE=DBPROPAL_CB_PRESERVE. Committing a transaction preserves prepared commands. The application can run commands again without preparing them again.</p> <p>VALUE=DBPROPAL_CB_DELETE. Committing a transaction deletes prepared commands. The application must prepare commands again before it can run them again.</p>
DBPROP_PROCEDURETERM	<p>Specifies a character string that contains the data source vendor's name for a procedure.</p> <p>This value depends on the SequeLink service you are using.</p>
DBPROP_PROVIDERFRIENDLYNAME	<p>Specifies the name of the provider, for example, MERANT SequeLink ADO Provider.</p>
DBPROP_PROVIDERNAME	<p>Specifies the filename of the SequeLink ADO Provider, where <i>nn</i> is the release level of the provider.</p> <p>VALUE=s1s1knn.DLL is the default.</p>

Table 4-5. OLE DB Data Source Information Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_PROVIDEROLEDBVER	<p>Specifies the version of the OLE DB specification supported by the SequeLink ADO Provider.</p> <p>VALUE=02.50</p>
DBPROP_PROVIDERVER	<p>Identifies the version of the SequeLink ADO Provider.</p> <p>VALUE=nnnn</p>
DBPROP_QUOTEDIDENTIFIERCASE	<p>Specifies how quoted identifiers manage case.</p> <p>VALUE=DBPROPVAL_IC_SENSITIVE. Quoted identifiers are case-sensitive and are stored in mixed case in the system catalog.</p> <p>VALUE=DBPROPVAL_IC_LOWER. Quoted identifiers are case-insensitive and are stored in lower case in the system catalog.</p> <p>VALUE=DBPROPVAL_IC_UPPER. Quoted identifiers in SQL are case-insensitive and are stored in upper case in the system catalog.</p> <p>VALUE=DBPROPVAL_IC_MIXED. Quoted identifiers in SQL are case-insensitive and are stored in mixed case in the system catalog.</p> <p>This value depends on the SequeLink service you are using.</p>
DBPROP_ROWSETCONVERSIONSONCOMMAND	<p>Specifies how callers to IConvertType::CanConvert can inquire on a command about conversions.</p> <p>VALUE=VARIANT_TRUE. Callers can inquire on a command about conversions supported on rowsets generated by the command.</p>
DBPROP_SCHEMATERM	<p>Specifies the name the data source uses for a schema.</p> <p>This value depends on the SequeLink service you are using.</p>

Table 4-5. OLE DB Data Source Information Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_SCHEMAUSAGE	<p>Specifies how schema names can be used in text commands. A combination of zero or one or more of the following:</p> <p>VALUE=DBPROPVAL_SU_DML_STATEMENTS. Schema names are supported in all Data Manipulation Language statements.</p> <p>VALUE=DBPROPVAL_SU_TABLE_DEFINITION. Schema names are supported in all table definition statements.</p> <p>VALUE=DBPROPVAL_SU_INDEX_DEFINITION. Schema names are supported in all index definition statements.</p> <p>VALUE= DBPROPVAL_SU_PRIVILEGE_DEFINITION. Schema names are supported in all privilege definition statements.</p>
DBPROP_SERVERNAME	<p>The value depends on the SequeLink service you are using.</p> <p>Specifies the name of the server. This can be the same as the DBPROP_INIT_DATASOURCE property if the server name is used to define the data source that the user specifies when connecting. Alternatively, if the provider connects through "friendly" data source names, this can be the actual name of the server.</p>

Table 4-5. OLE DB Data Source Information Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_SQLSUPPORT	<p>Specifies the level of SQL grammar the SequeLink ADO Provider supports. The effects are cumulative; that is, if the provider supports one level, it also sets the values for all lower levels. Possible values are:</p> <p>VALUE=DBPROPVAL_SQL_ODBC_CORE</p> <p>VALUE=DBPROPVAL_SQL_ODBC_MINIMUM</p> <p>VALUE=DBPROPVAL_SQL_ESCAPECLAUSES</p> <p>VALUE=DBPROPVAL_SQL_ANSI89_IEF</p> <p>VALUE=DBPROPVAL_SQL_ANSI92_ENTRY</p> <p>This value depends on the SequeLink service you are using.</p>
DBPROP_STRUCTUREDSTORAGE	<p>Specifies the interfaces the rowset supports on storage objects.</p> <p>VALUE=DBPROPVAL_SS_ISEQUENTIALSTREAM is the default.</p>
DBPROP_SUBQUERIES	<p>Specifies the predicates in text commands that support subqueries. A combination of zero or one or more of the following:</p> <p>VALUE=DBPROPVAL_SQ_COMPARISON</p> <p>VALUE=</p> <p>DBPROPVAL_SQ_CORRELATEDSUBQUERIES. This indicates that all predicates that support subqueries support correlated subqueries.</p> <p>VALUE=DBPROPVAL_SQ_EXISTS</p> <p>VALUE=DBPROPVAL_SQ_IN</p> <p>VALUE=DBPROPVAL_SQ_QUANTIFIED</p>

Table 4-5. OLE DB Data Source Information Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_SUPPORTEDTXNDDL	<p>Specifies whether Data Definition Language (DDL) statements are supported in transactions. Possible values are:</p> <p>VALUE=DBPROPVAL_TC_ALL. Transactions can contain DDL and DML statements in any order.</p> <p>VALUE=DBPROPVAL_TC_DDL_COMMIT. Transactions can only contain DML statements. DDL statements within a transaction cause the transaction to be committed.</p> <p>VALUE=DBPROPVAL_TC_DML. Transactions can only contain Data Manipulation Language (DML) statements. DDL statements within a transaction cause an error.</p> <p>This value depends on the SequeLink service you are using.</p>
DBPROP_SUPPORTEDTXNISOLEVELS	<p>Specifies the supported transaction isolation levels. One or a combination of zero or one or more of the following:</p> <p>VALUE=DBPROPVAL_TI_BROWSE</p> <p>VALUE=DBPROPVAL_TI_CHAOS</p> <p>VALUE=DBPROPVAL_TI_CURSORSTABILITY</p> <p>VALUE=DBPROPVAL_TI_ISOLATED</p> <p>VALUE=DBPROPVAL_TI_READCOMMITTED</p> <p>VALUE=DBPROPVAL_TI_READUNCOMMITTED</p> <p>VALUE=DBPROPVAL_TI_REPEATABLEREAD</p> <p>VALUE=DBPROPVAL_TI_SERIALIZABLE</p> <p>This value depends on the SequeLink service you are using.</p>
DBPROP_SUPPORTEDTXNISORETAIN	<p>Specifies the supported transaction isolation retention levels.</p> <p>VALUE=0</p>

Table 4-5. OLE DB Data Source Information Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_TABLETERM	Specifies the name the data source uses for a table. VALUE="Table"
DBPROP_USERNAME	Specifies a character string with the name used in a particular database. This can be different from the login name.

Initialization Property Group

[Table 4-6](#) provides the supported initialization properties for the SequeLink ADO Provider. The properties are read/write. For more information, refer to the Microsoft OLE DB programming documentation.

Table 4-6. Initialization Properties Supported by the SequeLink ADO Provider

Property ID	Description and Default
DBPROP_AUTH_PASSWORD	Specifies the password to be used for connecting to the data source or enumerator. This corresponds to the Password connection attribute. For more information about connection attributes, see "ADO Connection Attributes" on page 118 .

Table 4-6. Initialization Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_AUTH_PERSIST_SENSITIVE_AUTHINFO	Specifies whether the SequeLink ADO Provider can store (persist) the password and other sensitive authentication information. VALUE=VARIANT_FALSE. The data source object cannot store the password or other sensitive authentication information.
DBPROP_AUTH_USERID	Specifies the User ID to be used for connecting to the data source. This corresponds to the User ID connection attribute. For more information about connection attributes, see “ADO Connection Attributes” on page 118 .
DBPROP_INIT_CATALOG	Specifies the name of the initial or default catalog to use when connecting to the data source.
DBPROP_INIT_DATASOURCE	Specifies the name of the data source or enumerator to which to connect. This corresponds to the Data Source Name field on the Setup window.
DBPROP_INIT_HWND	Specifies the window handle to use if the data source object or enumerator needs to prompt for additional information.
DBPROP_INIT_LCID	Specifies the preferred locale ID for the consumer.
DBPROP_INIT_MODE	A bitmask that specifies access permissions. VALUE=DB_MODE_READ. The default is readonly.

Table 4-6. Initialization Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_INIT_OLEDBSERVICES	<p>Specifies which OLE DB services to enable. A combination of zero or one or more of the following:</p> <p>VALUE=DBPROPVAL_OS_RESOURCEPOOLING Resources should be pooled.</p> <p>VALUE=DBPROPVAL_OS_TXENLISTMENT Sessions in an MTS environment should automatically be enlisted in a global transaction where required (implies DBPROPVAL_OS_RESOURCEPOOLING).</p> <p>VALUE=DBPROPVAL_OS_ENABLEALL is the default. All services should be invoked. By default, all services are enabled and invoked as requested. Individual services can be deselected by specifying the bitwise-OR of DBPROPVAL_OS_ENABLEALL along with the bitwise complement of any services to be deselected. For example, DBPROPVAL_OS_ENABLEALL ~DBPROPVAL_OS_TXENLISTMENT enables all services except automatic transaction enlistment in an MTS environment.</p>
DBPROP_INIT_PROMPT	<p>Specifies whether to prompt the user for missing or additional connection information. For more information about the ADO connection dialogs, see “ADO Connection Dialogs” on page 110.</p> <p>VALUE=DBPROMPT_COMPLETE. Display a connection dialog only if missing information is needed.</p>

Table 4-6. Initialization Properties Supported by the SequeLink ADO Provider (cont.)

Property ID	Description and Default
DBPROP_INIT_PROVIDERSTRING	<p data-bbox="758 314 1279 473">Specifies a provider-specific string that contains extra initialization information. For information about using the provider string, see "Connecting with a Provider String" on page 117.</p> <p data-bbox="758 487 1279 612">Consumers should use this property only for provider-specific connection information as described in Table 3-2 on page 118.</p>

Rowset Property Group

OLE DB data consumers can request certain properties to be satisfied by the rowsets that result from an OpenRowset or ICommand::Execute call. Common properties include the set of interfaces to be supported by the resulting rowset.

The SequeLink ADO Provider supports the IRowsetIdentity interface. This allows the data consumer to determine when two row handles represent the same underlying data.

[Table 4-7 "Rowset Properties Supported by the SequeLink ADO Provider"](#) on page 152 provides the properties that are included in the SequeLink ADO Provider properties group. The table shows the initial default values. Some of these properties can be changed at the Command level (through ICommandProperties->SetProperties) or Session level (through IOpenRowset). The resulting rowset will contain different values for these properties.

Table 4-7. Rowset Properties Supported by the SequeLink ADO Provider

Property IDs	Description and Default
DBPROP_IAccessor	This property is Read-only. VALUE=VARIANT_TRUE. The rowset supports IAccessor.
DBPROP_IColumnsInfo	This property is Read-only. VALUE=VARIANT_TRUE. The rowset supports IColumnsInfo.
DBPROP_IColumnsRowset	This property is Read/Write. VALUE=VARIANT_TRUE. The rowset supports IColumnsRowset.
DBPROP_IConvertType	This property is Read-only. VALUE=VARIANT_TRUE. The rowset supports IConvertType.
DBPROP_IMultipleResults	This property is Read/Write. VALUE=VARIANT_FALSE. The rowset does not support IMultipleResults.
DBPROP_IRowset	This property is Read-only. VALUE=VARIANT_TRUE. The rowset supports IRowset.
DBPROP_IRowsetChange	This property is Read-only. VALUE=VARIANT_TRUE. The rowset supports IRowsetChange.
DBPROP_IRowsetIdentity	This property is Read/Write. VALUE=VARIANT_FALSE. The rowset does not support IRowsetIdentity.
DBPROP_IRowsetInfo	This property is Read-only. VALUE=VARIANT_TRUE. The rowset supports IRowsetInfo.
DBPROP_IRowsetLocate	This property is Read/Write. VALUE=VARIANT_TRUE. The rowset supports IRowsetLocate.

Table 4-7. Rowset Properties Supported by the SequeLink ADO Provider (cont.)

Property IDs	Description and Default
DBPROP_IRowsetRefresh	This property is Read/Write. VALUE=VARIANT_FALSE. The rowset does not support IRowsetRefresh.
DBPROP_IRowsetScroll	This property is Read/Write. VALUE=VARIANT_FALSE. The rowset does not support IRowsetScroll.
DBPROP_IRowsetUpdate	This property is Read/Write. VALUE=VARIANT_FALSE. The rowset does not support IRowsetUpdate.
DBPROP_ISequentialStream	This property is Read/Write. VALUE=VARIANT_FALSE. The rowset does not support ISequentialStream.
DBPROP_ISupportErrorInfo	Used by the data consumer to determine whether an object can return OLE DB error objects. This property is Read/Write. VALUE=VARIANT_TRUE
DBPROP_ABORTPRESERVE	Specifies whether a rowset remains active after aborting a transaction. VALUE=VARIANT_TRUE. After aborting a transaction, the rowset remains active. That is, it is possible to fetch new rows, update, delete, and insert rows. VALUE=VARIANT_FALSE. After aborting a transaction, the only operations allowed on a rowset are to release row and accessor handles and to release the rowset. The value depends on the SequeLink service you are using.

Table 4-7. Rowset Properties Supported by the SequeLink ADO Provider (cont.)

Property IDs	Description and Default
DBPROP_ACCESSORDER	<p>Specifies the order in which a static or keyset cursor must access columns on the rowset.</p> <p>VALUE= DBPROPVAL_AO_SEQUENTIALSTORAGEOBJECTS</p> <p>Columns bound as storage objects can only be accessed in sequential order as determined by the column ordinal. Further, storage objects from one row must be retrieved before calling GetData on any columns in any subsequent row. Calling GetData on a column bound as a storage object returns DBSTATUS_E_UNAVAILABLE for any columns bound as storage objects if any of the following are true:</p>
DBPROP_BLOCKINGSTORAGEOBJECTS	<ul style="list-style-type: none"> ■ Columns beyond the column bound as a storage object are specified in the accessor. ■ Columns beyond the column bound as a storage object have been accessed in a previous call to GetData for that row. ■ GetData has been called for any columns on a row returned after the specified row. <p>Providers that never impose restrictions on column access ordering return DBPROPSTATUS_S_OK when this value is set. However, they upgrade the property to DBPROPVAL_AO_RANDOM such that calling GetProperty continues to return DBPROPVAL_AO_RANDOM for this property.</p> <p>Specifies whether storage objects can prevent use of other methods on the rowset.</p> <p>VALUE=VARIANT_FALSE. Instantiated storage objects do not prevent the use of other methods.</p>

Table 4-7. Rowset Properties Supported by the SequeLink ADO Provider (cont.)

Property IDs	Description and Default
DBPROP_BOOKMARKINFO	<p>Specifies additional information about bookmarks over the rowset.</p> <p>VALUE=DBPROPVAL_BI_CROSSROWSET (if bookmarks are supported)</p> <p>VALUE=0 (if bookmarks are not supported)</p> <p>If set, bookmark values returned by this rowset are valid across rowsets with the same metadata. If not set, bookmark values are specific to this rowset and are not guaranteed to return the same values in other rowsets, even those resulting from the same specification.</p>
DBPROP_BOOKMARKS	<p>Specifies whether the rowset supports bookmarks. This property is Read/Write.</p> <p>VALUE=FALSE. The rowset does not support bookmarks.</p>
DBPROP_BOOKMARKSKIPPED	<p>Specifies whether the rowset allows IRowsetLocate::GetRowsAt, IRowsetScroll::GetApproximatePosition, or IRowsetFind::FindNextRow to continue if a bookmark row was deleted, is a row to which the consumer does not have access rights, or is no longer a member of the rowset.</p> <p>VALUE=VARIANT_FALSE. GetRowsAt, GetApproximatePosition, or FindNextRow returns DB_E_BADBOOKMARK.</p>
DBPROP_BOOKMARKTYPE	<p>Specifies the type of bookmark supported by the rowset.</p> <p>VALUE=DBPROPVAL_BMK_NUMERIC. The bookmark type is numeric. Numeric bookmarks are based upon a row property that is not dependent on the values of the row's columns. For instance, they can be based on the absolute position of the row within a rowset, or on a row ID that the storage engine assigned to a tuple at its creation. The validity of numeric bookmarks is not changed by modifying the row's columns.</p>

Table 4-7. Rowset Properties Supported by the SequeLink ADO Provider (cont.)

Property IDs	Description and Default
DBPROP_CANFETCHBACKWARDS	<p>Specifies whether the rowset can fetch backwards. When the value is negative, these methods fetch rows backward from the specified row. This property is Read/Write.</p> <p>VALUE=VARIANT_FALSE. cRows must be non-negative.</p>
DBPROP_CANHOLDROWS	<p>Specifies whether the consumer can retrieve more rows while holding previously fetched rows with pending changes. This property is Read/Write.</p> <p>VALUE=VARIANT_FALSE. The rowset might require pending changes to be transmitted to the data store and all rows to be released before fetching additional rows, inserting new rows, or changing the next fetch position.</p>
DBPROP_CANSCROLLBACKWARDS	<p>Specifies whether the rowset can scroll backwards. This property is Read/Write.</p> <p>VALUE=VARIANT_FALSE. IRowsOffset must be non-negative.</p>
DBPROP_CHANGEINSERTEDROWS	<p>Specifies whether a newly inserted row is defined to be a row for which the insertion has been transmitted to the data source.</p> <p>VALUE=VARIANT_FALSE</p>
DBPROP_COLUMNRESTRICT	<p>Specifies whether access rights are restricted on a column-by-column basis.</p> <p>VALUE=VARIANT_TRUE. Access rights are restricted on a column-by-column basis.</p> <p>VALUE=VARIANT_FALSE</p>

Table 4-7. Rowset Properties Supported by the SequeLink ADO Provider (cont.)

Property IDs	Description and Default
DBPROP_COMMITPRESERVE	<p data-bbox="668 314 1305 378">Specifies what operations are allowed on a rowset following a transaction.</p> <p data-bbox="668 392 1305 517">VALUE=VARIANT_TRUE. After committing a transaction, the rowset remains active. That is, it is possible to fetch new rows, update, delete, and insert rows, and so on.</p> <p data-bbox="668 531 1305 656">VALUE=VARIANT_FALSE. After committing a transaction, the only operations allowed on a rowset are to release row and accessor handles and to release the rowset.</p> <p data-bbox="668 670 1305 725">The value is specific to the SequeLink Server you are using.</p>
DBPROP_DELAYSTORAGEOBJECTS	<p data-bbox="668 739 1305 803">Specifies the effect of delayed update mode on storage objects.</p> <p data-bbox="668 817 1305 881">VALUE=VARIANT_FALSE. Storage objects are used in immediate update mode. In particular:</p> <ul data-bbox="668 895 1305 1315" style="list-style-type: none"> <li data-bbox="668 895 1305 960">■ Changes to the object are immediately transmitted to the data source. <li data-bbox="668 973 1305 1020">■ Update has no effect on the object. <li data-bbox="668 1034 1305 1098">■ Undo does not undo changes made to the object since the row was last fetched or updated. <li data-bbox="668 1112 1305 1229">■ GetOriginalData retrieves the current value of the object, including changes made since the row was last fetched or updated. <li data-bbox="668 1242 1305 1315">■ In immediate update mode, this property has no effect on storage objects.
DBPROP_IMMOBILEROWS	<p data-bbox="668 1329 1305 1394">Specifies whether the rowset reorders inserted or updated rows.</p> <p data-bbox="668 1407 1305 1531">VALUE=VARIANT_TRUE. The rowset will not reorder inserted or updated rows. For IRowsetChange::InsertRow, rows will appear at the end of the rowset.</p>

Table 4-7. Rowset Properties Supported by the SequeLink ADO Provider (cont.)

Property IDs	Description and Default
DBPROP_LITERALBOOKMARKS	Specifies how bookmarks can be compared. VALUE=VARIANT_FALSE. Bookmarks can only be compared with IRowsetLocate::Compare.
DBPROP_LITERALIDENTITY	Specifies the method the data consumer uses to determine if two row handles point to the same row. VALUE=VARIANT_FALSE. The consumer must call IRowsetIdentity::IsSameRow to determine whether two row handles point to the same row.
DBPROP_LOCKMODE	Specifies the level of locking performed by the rowset. VALUE=DBPROPVAL_LM_NONE. The provider is not required to lock rows to ensure successful updates.
DBPROP_MAXOPENROWS	Specifies the maximum number of rows that can be active at the same time. VALUE=4096
DBPROP_MAXPENDINGROWS	Specifies the maximum number of rows that can have pending changes at the same time. This property is Read-only. VALUE=0 (because there is no limit)
DBPROP_MAXROWS	Specifies the maximum number of rows that can be returned in a rowset. This property is Read-write. VALUE=0 (because there is no limit)
DBPROP_MEMORYUSAGE	Estimates the amount of memory that the rowset can use. VALUE=0 (there is no limit)
DBPROP_OTHERINSERT	Specifies whether the rows inserted by others are visible to the rowset. VALUE=VARIANT_FALSE. The rowset cannot see updates and deletes made by others.

Table 4-7. Rowset Properties Supported by the SequeLink ADO Provider (cont.)

Property IDs	Description and Default
DBPROP_OTHERUPDATEDELETE	<p>Specifies whether the rowset can see updates and deletes made by others.</p> <p>VALUE=VARIANT_FALSE. The rowset cannot see updates and deletes made by others.</p>
DBPROP_OWNINSERT	<p>VALUE=VARIANT_FALSE. The rowset cannot see rows inserted by consumers of the rowset unless the command is executed again.</p>
DBPROP_OWNUPTATEDDELETE	<p>VALUE=VARIANT_FALSE. The rowset cannot see updates and deletes made by consumers of the rowset unless the command is executed again.</p>
DBPROP_REENTRANTEVENTS	<p>VALUE=VARIANT_TRUE. The provider supports reentrancy during callbacks to the IRowsetNotify interface. The provider might not support reentrancy on all rowset methods. These methods return DB_E_NOTREENTRANT.</p>
DBPROP_REMOVEDELETED	<p>Specifies whether static cursors remove deleted rows.</p> <p>VALUE=VARIANT_FALSE. Static cursors do not remove deleted rows.</p>
DBPROP_REPORTMULTIPLECHANGES	<p>Specifies whether an update or delete can affect multiple rows.</p> <p>VALUE=VARIANT_FALSE. An update or delete always affects a single row or the provider cannot detect whether it affects multiple rows.</p>
DBPROP_RETURNPENDINGINSERTS	<p>Specifies whether the methods that fetch rows can return pending insert rows.</p> <p>VALUE=VARIANT_FALSE. The methods that fetch rows cannot return pending insert rows.</p>
DBPROP_ROWRESTRICT	<p>VALUE=VARIANT_FALSE. Access rights are not restricted on a row-by-row basis. If the rowset supports IRowsetChange, SetData can be called for any row.</p>
DBPROP_ROWTHREADMODEL	<p>Specifies the threading model of the rowsets generated by the command.</p> <p>VALUE=DBPROPVAL_RT_FREETHREAD</p>

Table 4-7. Rowset Properties Supported by the SequeLink ADO Provider (cont.)

Property IDs	Description and Default
DBPROP_SERVERCURSOR	VALUE=VARIANT_FALSE. The provider determines where to locate the cursor. The consumer can determine the location of the cursor by checking the value of this property on the rowset.
DBPROP_STRONGIDENTITY	VALUE=VARIANT_FALSE. There is no guarantee that the handles of newly inserted rows can be compared successfully. In this case, IRowsetIdentity::IsSameRow might return DB_E_NEWLYINSERTED.
DBPROP_TRANSACTEDOBJECT	VALUE=VARIANT_FALSE. Any object created on the specified column is not transacted. That is, all changes to the object are permanent once they are made visible to the data source. If this property is set on a column that does not contain an object, it is ignored.
DBPROP_UNIQUEROWS	Specifies whether rows in a rowset can be uniquely identified by their column values. VALUE=VARIANT_FALSE. Rows in the rowset may or may not be uniquely identified by their column values.

Table 4-7. Rowset Properties Supported by the SequeLink ADO Provider (cont.)

Property IDs	Description and Default
DBPROP_UPDATABILITY	<p data-bbox="668 314 1326 538">Specifies the supported methods on IRowsetChange. This property should be used in conjunction with DBPROP_IRowsetChange. If DBPROP_IRowsetChange is VARIANT_TRUE and DBPROP_UPDATABILITY is not set, then it is provider-specific what methods are supported on IRowsetChange.</p> <p data-bbox="668 552 1326 642">If DBPROP_UPDATABILITY is specified, then the provider must not support any methods whose bits are not set.</p> <p data-bbox="668 656 1326 720">A combination of zero or one or more of the following:</p> <p data-bbox="668 734 1326 763">Value=DBPROPVAL_UP_CHANGE. SetData is supported.</p> <p data-bbox="668 777 1326 841">Value=DBPROPVAL_UP_DELETE. DeleteRows is supported.</p> <p data-bbox="668 855 1326 911">Value=DBPROPVAL_UP_INSERT. InsertRow is supported.</p>

Session Property Group

[Table 4-8](#) lists the properties the SequeLink ADO Provider supports in the DBPROPSET_SESSION property set. For more information, refer to your Microsoft OLE DB programming information.

Table 4-8. Session Properties Supported by the SequeLink ADO Provider

Property IDs	Description and Default
DBPROP_SESS_ AUTOCOMMITISOLEVELS	Specifies the transaction isolation level while in auto-commit mode. VALUE=NULL

OLE DB Interfaces Supported in ADO

[Table 4-9](#) lists the OLE DB interfaces that ADO supports and describes whether the interface is required by ADO. The SequeLink ADO Provider supports additional OLE DB interfaces that are not used by ADO.

For a description of the OLE DB objects and interfaces supported by the SequeLink ADO Provider, see [“OLE DB Objects and Interfaces” on page 124](#).

Table 4-9. Supported OLE DB Interfaces Used by ADO

OLE DB Interface	Use by ADO
IAccessor	Required
IColumnsInfo	Required
IColumnsRowset	If available

Table 4-9. Supported OLE DB Interfaces Used by ADO (cont.)

OLE DB Interface	Use by ADO
ICommand	If available
ICommandPrepare	If available
ICommandProperties	If available
ICommandText	If available
ICommandWithParameters	If available
IConvertType	Required
IDBCreateCommand	If available
IDBCreateSession	Required
IDBInitialize	Required
IDBProperties	Required
IOpenRowset	Required
IRowset	Required
IRowsetChange	If available
IRowsetIndex	If available
IRowsetInfo	Required
IRowsetLocate	If available
IRowsetScroll	If available
IRowsetUpdate	If available
ISourcesRowset	If available
ITransaction	If available
ITransactionLocal	If available
ITransactionOptions	If available

Mapping ADO Methods and Properties

This section maps the methods, properties, and collections of ADO objects to the OLE DB methods supported by the SequeLink ADO Provider.

Some ADO methods do not have a comparable OLE DB method. When more than one OLE DB method can be used, ADO uses the method that requires the least amount of system resources. For example, if an ADO method can use either `ICommand` or `IOpenRowset`, it uses the less performance-intensive `IOpenRowset`.

ADO Command Object

The `Command` object can be used to specify a database query in the language native to the database server. For a relational data provider, this is usually a SQL statement.

The `Execute` method for the ADO `Command` object maps to the OLE DB method, `ICommand::Execute`.

[Table 4-10](#) lists the supported ADO methods for the `Connection` object and maps them to the corresponding OLE DB methods.

Table 4-10. Mapping Built-in Methods of the ADO Command Object

ADO Method	OLE DB Method
<code>CommandText</code>	<code>ICommandText::SetCommandText</code>
<code>CommandTimeout</code>	<code>ICommandProperties::SetCommandProperties</code>
<code>Prepared</code>	<code>ICommandPrepare::Prepare</code> <code>ICommandPrepare::Unprepare</code>

[Table 4-11](#) lists the dynamic properties that are supported by the SequeLink ADO Provider for the Command object.

Table 4-11. Dynamic Properties Used for the ADO Command Object

ADO Property	Description and Default
Access Order	<p>Specifies the order in which a static or keyset cursor must access columns on the rowset.</p> <p>VALUE=2. Columns can be accessed in any order.</p>
Blocking Storage Objects	<p>Specifies whether instantiated storage objects can prevent the use of other methods on the rowset.</p> <p>VALUE=False</p>
Change Inserted Rows	<p>Specifies whether a newly inserted row is defined to be a row for which the insertion has been transmitted to the data source, as opposed to a pending insert row.</p> <p>VALUE=False. The value can only be set to True if the rowset is using a keyset-driven cursor.</p>
Column Privileges	<p>Specifies whether access rights are restricted on a column-by-column basis.</p> <p>VALUE=False. Access rights are not restricted on a column-by-column basis.</p> <p>VALUE=True. Access rights are restricted on a column-by-column basis.</p>
Command Time Out	<p>Specifies the number of seconds before a command times out.</p> <p>VALUE=0</p> <p>NOTE: This property is supported for Microsoft SQL Server only.</p>

Table 4-11. Dynamic Properties Used for the ADO Command Object (cont.)

ADO Property	Description and Default
Fetch Backward	Specifies whether the rowset can fetch backwards. When the value is negative, these methods fetch rows backward from the specified row. This property is Read/Write. VALUE=False
Hold Rows	Specifies whether the consumer can retrieve more rows while holding previously fetched rows with pending changes. This property is Read/Write. VALUE=True. Access rights are restricted on a column-by-column basis.
IAccessor	Specifies whether the rowset supports IAccessor. This property is always set to True. VALUE=True
IColumnsInfo	Specifies whether the rowset supports IColumnsInfo. This property is Read-only. VALUE=True
IColumnsRowset	Specifies whether the rowset supports IColumnsRowset. This property is always set to True. VALUE=True
IConvertType	Specifies whether the rowset supports IConvertType. This property is always set to True. VALUE=True
IRowset	Specifies whether the rowset supports IRowset. This property is always set to True. VALUE=True
IRowsetChange	Specifies whether the rowset supports IRowsetChange. VALUE=False

Table 4-11. Dynamic Properties Used for the ADO Command Object (cont.)

ADO Property	Description and Default
IRowsetInfo	Specifies whether the rowset supports IRowsetInfo. VALUE=True
Literal Row Identity	Specifies the method the data consumer uses to determine if two row handles point to the same row. VALUE=False. The consumer must call IRowsetIdentity::IsSameRow to determine whether two row handles point to the same row.
Lock Mode	Specifies the level of locking the rowset performs. VALUE=1. The provider is not required to lock rows at any time to ensure successful updates.
Maximum Open Rows	Specifies the maximum number of rows that can be active at the same time. VALUE=4096
Maximum Pending Rows	Specifies the maximum number of rows that can have pending changes at the same time. VALUE=0. There is no limit.
Maximum Rows	Specifies the maximum number of rows that can be returned in a rowset. VALUE=0. There is no limit.
Memory Usage	Estimates the amount of memory that can be used by the rowset. VALUE=0. There is no limit.
Objects Transacted	Specifies whether an object created on a specified column is transacted. If set on a column that does not contain an object, this property is ignored. VALUE=True

Table 4-11. Dynamic Properties Used for the ADO Command Object (cont.)

ADO Property	Description and Default
Others' Changes Visible	<p>Specifies whether the rowset can see updates and deletes made by others.</p> <p>VALUE=False. The rowset cannot see updates and deletes made by others.</p> <p>VALUE=True. The rowset can see updates and deletes made by others.</p>
Others' Inserts Visible	<p>Specifies whether the rowset can see inserts made by others.</p> <p>VALUE=False. The rowset cannot see inserts made by others.</p>
Own Changes Visible	<p>Specifies whether a rowset can see its own updates and deletes.</p> <p>VALUE=False. The rowset cannot see updates and deletes made by consumers of the rowset unless the command is executed again.</p>
Own Inserts Visible	<p>Specifies whether a rowset can see its own inserts.</p> <p>VALUE=False. The rowset can see the rows inserted by consumers only after the command is run again.</p>
Preserve on Abort	<p>Specifies whether a rowset remains active after aborting a transaction.</p> <p>VALUE=True. After aborting a transaction, the rowset remains active. That is, it is possible to fetch new rows, update, delete, and insert rows, and so on.</p> <p>VALUE=False. After aborting a transaction, the only operations allowed on a rowset are to release row and accessor handles and to release the rowset.</p>

Table 4-11. Dynamic Properties Used for the ADO Command Object (cont.)

ADO Property	Description and Default
Preserve on Commit	<p>Specifies what operations are allowed on a rowset following a transaction.</p> <p>VALUE=True. After committing a transaction, the rowset can fetch new rows, update, delete, and insert rows, and so on.</p> <p>VALUE=False. After committing a transaction, the only operations allowed on a rowset are to release row and accessor handles and to release the rowset.</p>
Quick Restart	<p>VALUE=True. IRowset::RestartPosition is relatively quick to execute. It does not again execute the command that created the rowset.</p>
Remove Deleted Rows	<p>Specifies whether static cursors remove deleted rows.</p> <p>VALUE=False</p>
Report Multiple Changes	<p>Specifies whether an update or delete can affect multiple rows.</p> <p>VALUE=False. An update or delete always affects a single row or the provider cannot detect whether it affects multiple rows.</p>
Return Pending Inserts	<p>Specifies whether the methods that fetch rows can return pending insert rows.</p> <p>VALUE=False</p>
Row Privileges	<p>Specifies access restrictions for rows.</p> <p>VALUE=False. The data provider does not set access restrictions for rows.</p>
Row Threading Model	<p>Specifies the threading model of the rowsets generated by the command.</p> <p>VALUE=1. The SequeLink ADO Provider uses the free-threaded model.</p>

Table 4-11. Dynamic Properties Used for the ADO Command Object (cont.)

ADO Property	Description and Default
Scroll Backward	Specifies whether the rowset can scroll backwards. This property is Read/Write. VALUE=False. IRowOffset must be non-negative.
Server Cursor	Specifies whether to materialize the cursor on the server. VALUE=False. The provider determines where to locate the cursor.
Strong Row Identity	Specifies whether the handles of newly inserted rows can be compared successfully. VALUE=False. There is no guarantee that the handles of newly inserted rows can be compared successfully.
Unique Rows	Specifies whether rows in a rowset can be uniquely identified by their column values. VALUE=False. Rows in the rowset may or may not be uniquely identified by their column values.
Updatability	Specifies the supported methods on IRowsetChange. This property should be used in conjunction with IRowsetChange. VALUE=0
Use Bookmarks	Specifies whether the rowset supports bookmarks. VALUE=False

The SequeLink ADO Provider supports the following standard ADO Command collections as listed in [Table 4-12](#).

Table 4-12. Mapping Collections for the ADO Command Object

Collection	OLE DB Method
Parameters	Get: ICommandWithParameters::GetParameterInfo or DBSCHEMA_PROCEDURE_PARAMETERS schema rowset Set: ICommandWithParameters::SetParameterInfo
Properties	ICommandProperties::GetProperties <i>and</i> ICommandProperties::GetProperties IDBProperties::GetPropertyInfo

Connection Object

The ADO Connection object represents a single session with an OLE DB data source. It defines a physical connection to the data source for a data provider.

[Table 4-13](#) lists the supported ADO methods for the Connection object and maps them to the corresponding OLE DB methods.

Table 4-13. Mapping Methods Supported by the ADO Connection Object

ADO Method	OLE DB Method
BeginTrans	ITransactionLocal::StartTransaction
CommitTrans	ITransactionLocal::Commit
Execute	ICommand::Execute <i>or</i> IOpenRowset::OpenRowset
Open	IDBInitialize::Initialize IDBCreateSession::Create Session
RollBackTrans	ITransactionLocal::Abort
OpenSchema	IDBSchemaRowset::GetRowset

The SequeLink ADO Provider supports the built-in properties for the Command object listed in [Table 4-14](#), which also maps the ADO properties to the corresponding OLE DB methods.

Table 4-14. Mapping Built-in Properties Used by the ADO Connection Object

ADO Property	OLE DB Method
Attributes	ITransactionLocal::StartTransaction
CommandTimeout	ICommandProperties::SetCommandProperties (DBPROP_COMMANDTIMEOUT)

Table 4-14. Mapping Built-in Properties Used by the ADO Connection Object (cont.)

ADO Property	OLE DB Method
DefaultDatabase	IDBProperties::GetProperties (DBPROP_CURRENTCATALOG) <i>and</i> IDBProperties::SetProperties (DBPROP_CURRENTCATALOG)
IsolationLevel	ITransactionLocal::StartTransaction
Mode	IDBProperties::GetProperties (DBPROP_INIT_MODE) <i>and</i> IDBProperties::SetProperties (DBPROP_INIT_MODE)
Provider	ISourcesRowset::GetSourcesRowset

[Table 4-15](#) lists the dynamic properties supported for the ADO Connection object.

Table 4-15. Dynamic Properties Supported for the ADO Connection Object

ADO Property	Description and Default Value
Active Sessions	Specifies the maximum number of sessions that can exist at one time. VALUE=0. There is no limit.
Asynchable Abort	Specifies whether the provider can abort transactions asynchronously. VALUE=False
Asynchable Commit	Specifies whether the provider can commit transactions asynchronously. VALUE=False

Table 4-15. Dynamic Properties Supported for the ADO Connection Object (cont.)

ADO Property	Description and Default Value
Autocommit Isolation Levels	<p>Specifies the transaction isolation level while in auto-commit mode. A combination of zero or one or more of the following:</p> <p>VALUE=DBPROPVAL_TI_BROWSE</p> <p>VALUE=DBPROPVAL_TI_CURSORSTABILITY</p> <p>VALUE=DBPROPVAL_TI_ISOLATED</p> <p>VALUE=DBPROPVAL_TI_READCOMMITTED</p> <p>VALUE=DBPROPVAL_TI_READUNCOMMITTED</p> <p>VALUE=DBPROPVAL_TI_REPEATABLEREAD</p> <p>VALUE=DBPROPVAL_TI_SERIALIZABLE</p>
Catalog Location	<p>Specifies the position of the catalog name in a qualified table name in a text command.</p> <p>VALUE=1. The catalog name is at the start of the fully qualified name.</p> <p>VALUE=2. The catalog name is at the end of the fully qualified name.</p>
Catalog Term	<p>Specifies the name the data source uses for a catalog.</p> <p>VALUE=Database</p>
Catalog Usage	<p>A bitmask specifying how catalog names can be used in text commands. A combination of zero or one or more of the following:</p> <p>VALUE=1. Catalog names are supported in all Data Manipulation Language statements.</p> <p>VALUE=2. Catalog names are supported in all table definition statements.</p> <p>VALUE=4. Catalog names are supported in all index definition statements.</p> <p>VALUE=8. Catalog names are supported in all privilege definition statements.</p>
Column Definition	<p>Defines the valid clauses for defining a column.</p> <p>VALUE=1. Columns can be created non-nullable.</p>

Table 4-15. Dynamic Properties Supported for the ADO Connection Object (cont.)

ADO Property	Description and Default Value
COM Object Support	<p>Specifies the way that the data provider supports access to BLOBs and OLE objects stored in columns.</p> <p>VALUE=1. The data providers support access to BLOBs as structured storage objects. A data consumer determines which interfaces are supported through the Structured Storage property.</p>
Connection Status	<p>Specifies the status of the current connection.</p> <p>VALUE=1. The data source is in an initialized state and able to communicate with the data store.</p>
Current Catalog	<p>Specifies the name of the current catalog. The data consumer can use the CATALOGS rowset to enumerate catalogs. If not set, the data provider uses the default catalog.</p>
Data Source	<p>Specifies the name of the data source or enumerator to which to connect.</p> <p>This corresponds to the Data Source Name field on the Setup window.</p>
Data Source Name	<p>Specifies the name of the data source (server) used during the connection process.</p>
Data Source Object Threading Model	<p>Specifies the threading model of the data source object.</p> <p>VALUE=1. The data providers use the free-threading model.</p>
DBMS Name	<p>Specifies the name of the product accessed by the SequeLink ADO Provider.</p>
DBMS Version	<p>Specifies the version of the product that the provider is currently accessing, where xx is the specific version of the product.</p>
Extended Properties	<p>A provider-specific string that contains extra initialization information. Consumers should use this property only for provider-specific connection information.</p> <p>For information on using the provider string with the SequeLink ADO Provider, see "Connecting with a Provider String" on page 117.</p>

Table 4-15. Dynamic Properties Supported for the ADO Connection Object (cont.)

ADO Property	Description and Default Value
GROUP BY Support	Specifies the relationship between the columns in a Group By clause and the nonaggregated columns in the select list.
Heterogeneous Table Support	Specifies whether the provider can join tables from different catalogs or providers. VALUE=1. The provider can join tables from different catalogs. VALUE=0. The provider cannot join tables from different catalogs or providers.
Identifier Case Sensitivity	Specifies how identifiers treat case. VALUE=1. Identifiers in SQL are case-sensitive and are stored in upper case in the system catalog. VALUE=2. Identifiers in SQL are case-insensitive and are stored in lower case in the system catalog. VALUE=4. Identifiers in SQL are case-sensitive and are stored in mixed case in the system catalog. VALUE=8. Identifiers in SQL are case-insensitive and are stored in mixed case in the system catalog.
Initial Catalog	Specifies the name of the initial, or default, catalog to use when connecting to the data source.
Isolation Levels	Specifies the supported transaction isolation levels. A combination of zero or one or more of the following: VALUE=DBPROPVAL_TI_BROWSE VALUE=DBPROPVAL_TI_CURSORSTABILITY VALUE=DBPROPVAL_TI_ISOLATED VALUE=DBPROPVAL_TI_READCOMMITTED VALUE=DBPROPVAL_TI_READUNCOMMITTED VALUE=DBPROPVAL_TI_REPEATABLEREAD VALUE=DBPROPVAL_TI_SERIALIZABLE
Isolation Retention	Specifies the supported transaction isolation retention levels. VALUE=0

Table 4-15. Dynamic Properties Supported for the ADO Connection Object (cont.)

ADO Property	Description and Default Value
Locale Identifier	Specifies the preferred locale ID for the consumer.
Maximum Index Size	Specifies the maximum number of bytes allowed in the combined columns of an index. VALUE=0. There is no limit.
Maximum Open Chapters	Specifies the maximum number of chapters that can be open at any time. VALUE=0
Maximum Row Size	Specifies the maximum length of a single row in a table. VALUE=0. There is no limit.
Maximum Row Size Includes BLOB	Specifies whether the maximum row size returned for the property includes the length of all BLOB data. VALUE=False. The maximum row size does not include the length of all BLOB data.
Maximum Tables in SELECT	Specifies the maximum number of tables allowed in the FROM clause of a Select statement. A value of 0 means no limit or the limit is not known. VALUE=0. There is no limit.
Mode	Specifies a bitmask that specifies access permissions. VALUE=3. The default access is read-write.
Multi-Table Update	Specifies whether the provider can update rowsets derived from multiple tables. VALUE=False
Multiple Connections	Specifies whether the provider must spawn multiple connections to support concurrent command, session, and rowset objects. VALUE=True
Multiple Parameter Sets	Specifies whether the provider supports multiple parameter sets at the same time. VALUE=True

Table 4-15. Dynamic Properties Supported for the ADO Connection Object (cont.)

ADO Property	Description and Default Value
Multiple Results	<p>Specifies whether the provider supports multiple results objects and the restrictions it places on these objects.</p> <p>VALUE=1. The provider supports multiple results objects.</p>
Multiple Storage Objects	<p>Specifies whether the provider supports multiple, open storage objects at the same time.</p> <p>VALUE=True. The provider supports more than one open storage object at a time.</p>
NULL Collation Order	Specifies where nulls are sorted into a list.
NULL Concatenation Behavior	<p>Defines how the data source handles the concatenation of null-valued character data type columns with character data type columns that are not null-valued.</p>
OLE DB Services	<p>Specifies which OLE DB services to enable.</p> <p>Value=0. The provider does not enable the OLE DB services.</p>
OLE DB Version	<p>Specifies the version of OLE DB supported by the data provider.</p> <p>VALUE=02.50</p>
Open Rowset Support	<p>Describes support for opening objects through IOpenRowset.</p> <p>VALUE=0. All providers support opening tables through IOpenRowset.</p>
ORDER BY Columns In Select List	Specifies the way that the provider orders columns in an Order By clause.

Table 4-15. Dynamic Properties Supported for the ADO Connection Object (cont.)

ADO Property	Description and Default Value
Output Parameter Availability	<p>Specifies the time at which output parameter values become available.</p> <p>VALUE=1. Output parameters are not supported.</p> <p>VALUE=2. Output parameter data is available immediately after ICommand::Execute returns.</p> <p>VALUE=4. If a command returns a single result that is a rowset, output parameter data is available at the time the rowset is completely released.</p>
Pass By Ref Accessors	<p>Specifies whether the provider supports the DBACCESSOR_PASSBYREF flag in IAccessor::CreateAccessor.</p> <p>VALUE=False</p>
Password	<p>Specifies the password to be used for connecting to the data source or enumerator.</p>
Prepare Abort Behavior	<p>Specifies how aborting a transaction affects prepared commands.</p> <p>VALUE=1. Aborting a transaction deletes prepared commands. The application must prepare commands again before it can run them again.</p> <p>VALUE=2. Aborting a transaction preserves prepared commands. The application can run commands again without preparing them again.</p>
Prepare Commit Behavior	<p>Specifies how committing a transaction affects prepared commands.</p> <p>VALUE=1. Committing a transaction deletes prepared commands. The application must prepare commands again before it can run them again.</p> <p>VALUE=2. Committing a transaction preserves prepared commands. The application can run commands again without preparing them again.</p>
Procedure Term	<p>Specifies a character string that contains the data source vendor's name for a procedure.</p>
Prompt	<p>Specifies whether to prompt the user for additional information during initialization.</p>

Table 4-15. Dynamic Properties Supported for the ADO Connection Object (cont.)

ADO Property	Description and Default Value
Provider Friendly Name	Specifies the name of the provider, "DataDirect SequeLink ADO Provider".
Provider Name	Specifies the filename of the provider. VALUE= <code>slslnn.DLL</code> , where <i>nn</i> is the release level of the provider.
Provider Version	Specifies the version of the DataDirect data provider. VALUE= <code>05.10.0000</code>
Quoted Identifier Sensitivity	Specifies how quoted identifiers treat case. VALUE=1. Quoted identifiers in SQL are case-sensitive and are stored in upper case in the system catalog. VALUE=2. Quoted identifiers in SQL are case-insensitive and are stored in lower case in the system catalog. VALUE=4. Quoted identifiers in SQL are case-sensitive and are stored in mixed case in the system catalog. VALUE=8. Quoted identifiers in SQL are case-insensitive and are stored in mixed case.
Read Only Data Source	Specifies whether the data source is read-only. VALUE=0. The data source can be updated.
Rowset Conversions on Command	Specifies how callers to <code>IConvertType::CanConvert</code> can inquire on a command about conversions. VALUE= <code>True</code> . Callers can inquire on a command about conversions supported on rowsets generated by the command.
Schema Term	Specifies the name the data source uses for a schema.

Table 4-15. Dynamic Properties Supported for the ADO Connection Object (cont.)

ADO Property	Description and Default Value
Schema Usage	<p>Specifies how schema names can be used in text commands. A combination of the following:</p> <p>VALUE=DBPROPVAL_SU_DML_STATEMENTS. Schema names are supported in all Data Manipulation Language statements.</p> <p>VALUE=DBPROPVAL_SU_TABLE_DEFINITION. Schema names are supported in all table definition statements.</p> <p>VALUE=DBPROPVAL_SU_INDEX_DEFINITION. Schema names are supported in all index definition statements.</p> <p>VALUE=DBPROPVAL_SU_PRIVILEGE_DEFINITION. Schema names are supported in all privilege definition statements.</p>
Server Name	<p>Specifies the name of the server. This can be the same as the Data Source property if the server name is used to define the data source that the user specifies when connecting. Alternatively, if the provider connects through "friendly" data source names, this can be the actual name of the server.</p>

Table 4-15. Dynamic Properties Supported for the ADO Connection Object (cont.)

ADO Property	Description and Default Value
SQL Support	<p data-bbox="582 314 1239 470">Specifies the level of SQL grammar that the provider supports. The effects are cumulative, that is, if the provider supports one level, it also sets the values for all lower levels. A combination of zero or one or more of the following:</p> <p data-bbox="582 487 951 517">VALUE=DBPROPVAL_SQL_NONE</p> <p data-bbox="582 531 1022 560">VALUE=DBPROPVAL_SQL_ODBC_CORE</p> <p data-bbox="582 574 1068 604">VALUE=DBPROPVAL_SQL_ODBC_MINIMUM</p> <p data-bbox="582 618 1082 647">VALUE=DBPROPVAL_SQL_ODBC_EXTENDED</p> <p data-bbox="582 661 1082 690">VALUE=DBPROPVAL_SQL_ESCAPECLAUSES</p> <p data-bbox="582 704 1068 734">VALUE=DBPROPVAL_SQL_ANSI92_ENTRY</p> <p data-bbox="582 748 1068 777">VALUE=DBPROPVAL_SQL_ANSI92_ENTRY</p> <p data-bbox="582 791 1139 821">VALUE=DBPROPVAL_SQL_FIPS_TRANSITIONAL</p> <p data-bbox="582 835 1168 864">VALUE=DBPROPVAL_SQL_ANSI92_INTERMEDIATE</p> <p data-bbox="582 878 1053 907">VALUE=DBPROPVAL_SQL_ANSI92_FULLL</p> <p data-bbox="582 921 1039 951">VALUE=DBPROPVAL_SQL_ANSI89_IEF</p> <p data-bbox="582 965 1039 994">VALUE=DBPROPVAL_SQL_SUBMINIMUM</p>
Structured Storage	<p data-bbox="582 1003 1179 1060">Specifies what interfaces the rowset supports on storage objects.</p> <p data-bbox="582 1078 1065 1137">VALUE=1. The provider supports DBPROPVAL_SS_ISEQUENTIALSTREAM.</p>
Subquery Support	<p data-bbox="582 1154 1233 1246">Specifies the predicates in text commands that support subqueries. A combination of zero or one or more of the following:</p> <p data-bbox="582 1263 1168 1293">VALUE=DBPROPVAL_SQ_CORRELATEDSUBQUERIES</p> <p data-bbox="582 1307 1022 1336">VALUE=DBPROPVAL_SQ_COMPARISON</p> <p data-bbox="582 1350 965 1380">VALUE=DBPROPVAL_SQ_EXISTS</p> <p data-bbox="582 1394 908 1423">VALUE=DBPROPVAL_SQ_IN</p> <p data-bbox="582 1437 1022 1466">VALUE=DBPROPVAL_SQ_QUANTIFIED</p>
Table Term	Specifies the name the data source uses for a table.

Table 4-15. Dynamic Properties Supported for the ADO Connection Object (cont.)

ADO Property	Description and Default Value
Transaction DDL	Specifies whether SQL Data Definition Language (DDL) statements are supported in transactions.
User Name	Specifies a character string with the name used in a particular database. This can be different from the login name.
Window Handle	Specifies the window handle to use if the data source object or enumerator needs to prompt for additional information.

The SequeLink ADO Provider supports the following standard ADO Connection collections as listed in [Table 4-16](#).

Table 4-16. Mapping Collections for the ADO Connection Object

Collection	OLE DB Method
Properties	IDBProperties::GetPropertyInfo and IDBProperties::SetPropertyInfo

Field Object

The Fields object contains information about a single column of data in the Recordset object. You access a Fields collection and the Fields objects it contains through the Recordset object.

The SequeLink ADO Provider supports the built-in properties for the ADO Field object listed in [Table 4-17](#), which also maps the ADO properties to the corresponding OLE DB methods.

Table 4-17. Mapping Built-in Properties Supported by the ADO Field Object

ADO Property	OLE DB Method
ActualSize	IAccessor::CreateAccessor and IRowset::GetData
Attributes	IColumnsInfo::GetColumnInfo
DefinedSize	IColumnsInfo::GetColumnInfo
Name	IColumnsInfo::GetColumnInfo
NumericScale	IColumnsInfo::GetColumnInfo
OriginalValue	IRowsetUpdate::GetOriginalData
Precision	IColumnsInfo::GetColumnInfo
Value	IAccessor::CreateAccessor and IRowset::GetData and IRowset::SetData

Parameter Object

A Parameter object represents a parameter for a Command object based on a parameterized query or stored procedure. The functionality of the data provider determines which collections, methods and properties for the Parameter object are available.

The Parameter object supports the ADO AppendChunk method.

The SequeLink ADO Provider supports the built-in properties for the ADO Parameter object listed in [Table 4-18](#), which also maps the ADO properties to the corresponding OLE DB methods supported by the SequeLink ADO Provider.

Table 4-18. Mapping Built-in Properties Supported by the ADO Parameter Object

ADO Property	OLE DB Method
Attributes	Get: ICommandWithParameters::GetParameterInfo or DBSCHEMA_PROCEDURE_PARAMETERS schema rowset Set: ICommandWithParameters::SetParameterInfo
Direction	Get: ICommandWithParameters::GetParameterInfo or DBSCHEMA_PROCEDURE_PARAMETERS schema rowset Set: ICommandWithParameters::SetParameterInfo
Name	Get: ICommandWithParameters::GetParameterInfo or DBSCHEMA_PROCEDURE_PARAMETERS schema rowset
NumericScale	Get: ICommandWithParameters::GetParameterInfo or DBSCHEMA_PROCEDURE_PARAMETERS schema rowset Set: ICommandWithParameters:: SetParameterInfo
Precision	Get: ICommandWithParameters::GetParameterInfo or DBSCHEMA_PROCEDURE_PARAMETERS schema rowset Set: ICommandWithParameters::SetParameterInfo

Table 4-18. Mapping Built-in Properties Supported by the ADO Parameter Object (cont.)

ADO Property	OLE DB Method
Size	Get: ICommandWithParameters::GetParameterInfo or DBSCHEMA_PROCEDURE_PARAMETERS schema rowset Set: ICommandWithParameters::SetParameterInfo
Type	Get: ICommandWithParameters::GetParameterInfo or DBSCHEMA_PROCEDURE_PARAMETERS schema rowset Set: ICommandWithParameters::SetParameterInfo
Value	IAccessor::CreateAccessor <i>and</i> ICommand::Execute

Property Object

A Property object represents a dynamic characteristic of an ADO object defined by the provider. Properties cannot be deleted.

A property can be built-in or dynamic. Built-in properties are implemented in ADO and can be used by any new object. You can change the values for built-in properties, but you cannot change their characteristics.

Dynamic properties are defined by the underlying data provider, and appear in the Properties collection for the object. They can be referenced only through the collection.

Table 4-19 maps the built-in properties of the Property object to the OLE DB methods supported by the SequeLink ADO Provider.

Table 4-19. Mapping ADO Built-in Properties Supported for the Property Object

ADO Property	OLE DB Method
Attributes	Get: ICommandWithParameters::GetParameterInfo or DBSCHEMA_PROCEDURE_PARAMETERS schema rowset Set: ICommandWithParameters::SetParameterInfo
Name	Get: ICommandWithParameters::GetParameterInfo or DBSCHEMA_PROCEDURE_PARAMETERS schema rowset
Type	Get: ICommandWithParameters::GetParameterInfo or DBSCHEMA_PROCEDURE_PARAMETERS schema rowset Set: ICommandWithParameters::SetParameterInfo
Value	IAccessor::CreateAccessor and ICommand::Execute

Recordset Object

The Recordset object is the set of records resulting from a query against a database and a cursor, which is the interface to the records. If you create a Connection object before you open a Recordset object, multiple Recordset objects can be opened on the same connection.

[Table 4-20](#) maps the methods of the Recordset object to the OLE DB methods supported by the SequeLink ADO Provider.

Table 4-20. Mapping Methods Supported by the Recordset Object

ADO Method	OLE DB Method
AddNew	IRowsetChange::InsertRow
CancelBatch	IRowsetUpdate::Undo
Clone	IRowsetLocate
Close	IAccessor::ReleaseAccessor IRowset::ReleaseRows
Delete	IRowsetChange::DeleteRows
GetRows	IAccessor::CreateAccessor IRowsetLocate::GetRowsAt IRowset::GetNextRows IRowset::GetData
Move	IRowsetLocate::GetRowsAt IRowset::GetNextRows
MoveFirst	IRowsetLocate::GetRowsAt IRowset::RestartPosition
MoveLast	IRowsetLocate::GetRowsAt
MoveNext	IRowsetLocate::GetRowsAt IRowset::GetNextRows
MovePrevious	IRowsetLocate::GetRowsAt IRowset::GetNextRows

Table 4-20. Mapping Methods Supported by the Recordset Object (cont.)

ADO Method	OLE DB Method
NextRecordSet	IMultipleResults::GetResult
Open	IOpenRowset::OpenRowset ICommand::Execute
Requery	IOpenRowset::OpenRowset ICommand::Execute
Supports	IRowsetInfo::GetProperties
Update	IRowsetChange::SetData IRowsetUpdate::Update
UpdateBatch	IRowsetUpdate::Update

The SequeLink ADO Provider supports the built-in properties for the Recordset object listed in [Table 4-21](#), which also maps the ADO properties to the corresponding OLE DB methods.

Table 4-21. Mapping Built-in Properties for the ADO Recordset Object

ADO Property	OLE DB Method
AbsolutePage	Get: IRowsetScroll::GetApproximatePosition Set: IRowsetScroll::GetRowsAtRatio or IRowsetLocate::GetRowsAt
AbsolutePosition	Get: IRowsetScroll::GetApproximatePosition Set: IRowsetScroll::GetRowsAtRatio or IRowsetLocate::GetRowsAt
ActiveConnection	IDBInitialize::Initialize <i>and</i> IDBCreateSession::CreateSession
Bookmark	IAccessor::CreateAccessor <i>and</i> IRowsetLocate::GetRowsAt

Table 4-21. Mapping Built-in Properties for the ADO Recordset Object (cont.)

ADO Property	OLE DB Method
CacheSize	cRows argument to IRowsetLocate::GetRowsAt or IRowset::GetNextRows
CursorType	ICommandProperties::SetProperties
EditMode	IRowsetUpdate::GetRowStatus
Filter	IRowsetUpdate::GetPendingRows and IRowsetLocate::GetRowsByBookmark
LockType	ICommandProperties::SetProperties
MaxRecords	IOpenRowset::OpenRowset (DBPROP_MAXROWS) or ICommandProperties::SetCommandProperties (DBPROP_MAXROWS)
PageCount	IRowsetScroll::GetApproximatePosition
PageSize	Used for absolute positioning with PageCount and AbsolutePage properties
RecordCount	IRowsetScroll::GetApproximatePosition
Status	IRowsetUpdate::GetRowStatus

Table 4-22 lists the dynamic properties supported by the SequeLink ADO Provider for the Recordset object.

Table 4-22. Dynamic Properties Used for the Recordset Object

ADO Property	Description and Default
Access Order	Specifies the order in which a static or keyset cursor must access columns on the rowset. VALUE=2. Columns can be accessed in any order.
Blocking Storage Objects	Specifies whether instantiated storage objects can prevent use of other methods on the rowset. VALUE=False Instantiated storage objects do not prevent the use of other methods.
Bookmark Information	Specifies additional information about bookmarks over the rowset.
Bookmark Type	Specifies the type of bookmark supported by the rowset.
Bookmarks Ordered	Specifies whether bookmarks can be compared byte-by-byte or must be compared with IRowsetLocate::Compare.
Change Inserted Rows	Specifies whether the consumer can call IRowsetChange::DeleteRows or IRowsetChange::SetData for newly inserted rows. VALUE=False. DeleteRows returns a status of DBROWSTATUS_E_NEWLYINSERTED for the newly inserted row and SetData returns DB_E_NEWLYINSERTED.
Column Privileges	Specifies whether access rights are restricted on a column-by-column basis. VALUE=True. Access rights are restricted on a column-by-column basis. VALUE=False. Access rights are not restricted on a column-by-column basis. If the rowset exposes IRowsetChange, SetData can be called for any column in the rowset.

Table 4-22. Dynamic Properties Used for the Recordset Object (cont.)

ADO Property	Description and Default
Command Timeout	Specifies the number of seconds before a command times out. VALUE=0 (default is unlimited)
Defer Column	Specifies whether data in a column is not fetched until an accessor is used on the column.
Delay Storage Object Updates	Specifies the effect of delayed update mode on storage objects.
Fetch Backward	Specifies whether the rowset can fetch backwards. When the value is negative, these methods fetch rows backward from the specified row. This property is Read/Write. VALUE=False. cRows must be non-negative.
Hold Rows	Specifies whether the consumer can retrieve more rows while holding previously fetched rows with pending changes. This property is Read/Write. VALUE=True. The data consumer can retrieve more rows. VALUE=False. The data consumer requires pending changes to be transmitted to the data source object and all rows to be released before fetching additional rows, inserting new rows, or changing the next fetch position.
IAccessor	Specifies whether the rowset supports IAccessor. This property is always set to True. VALUE=True
IColumnsInfo	Specifies whether the rowset supports IColumnsInfo. This property is Read-only. VALUE=True
IColumnsRowset	Specifies whether the rowset supports IColumnsRowset. This property is always set to True. VALUE=True

Table 4-22. Dynamic Properties Used for the Recordset Object (cont.)

ADO Property	Description and Default
IConvertType	Specifies whether the rowset supports IConvertType. This property is always set to True. VALUE=True
IRowset	Specifies whether the rowset supports IRowset. This property is always set to True. VALUE=True
IRowsetChange	Specifies whether the rowset supports IRowsetChange. VALUE=False
IRowsetInfo	Supplies information about a rowset. VALUE=True
IRowsetLocate	Specifies whether bookmarks are enabled. VALUE=False
Immobile Rows	Specifies whether the rowset reorders inserted or updated rows. VALUE=True
Literal Bookmarks	Specifies how bookmarks can be compared. VALUE=False. Bookmarks can only be compared with IRowsetLocate::Compare.
Literal Row Identity	Specifies the method the data consumer uses to determine if two row handles point to the same row. VALUE=False. The consumer must call IRowsetIdentity::IsSameRow to determine whether two row handles point to the same row.
Lock Mode	Specifies the level of locking performed by the rowset. VALUE=1. The provider is not required to lock rows to ensure successful updates.
Maximum Open Rows	Specifies the maximum number of rows that can be active at the same time. VALUE=4096

Table 4-22. Dynamic Properties Used for the Recordset Object (cont.)

ADO Property	Description and Default
Maximum Pending Rows	Specifies the maximum number of rows that can have pending changes at the same time. VALUE=0. There is no limit.
Maximum Rows	Specifies the maximum number of rows that can be returned in a rowset. VALUE=0. There is no limit.
Memory Usage	Estimates the amount of memory that the rowset can use. VALUE=0. There is no limit.
Objects Transacted	Specifies whether an object created on a specified column is transacted. If this property is set on a column that does not contain an object, it is ignored.
Others' Changes Visible	Specifies whether the rowset can see updates and deletes made by others. VALUE=False
Others' Inserts Visible	Specifies whether a rowset can see inserts made by others. VALUE=False. The rowset cannot see the rows inserted by others.
Own Changes Visible	Specifies whether a rowset can see its own updates and deletes. VALUE=False. The rowset cannot see updates and deletes made by consumers of the rowset unless the command is executed again.
Own Inserts Visible	Specifies whether a rowset can see its own inserts. VALUE=False. The rowset can see the rows inserted by consumers only after the command is run again.

Table 4-22. Dynamic Properties Used for the Recordset Object (cont.)

ADO Property	Description and Default
Preserve on Abort	<p data-bbox="619 319 1205 381">Specifies whether a rowset remains active after aborting a transaction.</p> <p data-bbox="619 395 1282 487">VALUE=True. After aborting a transaction, the rowset remains active. That is, it is possible to fetch new rows, update, delete, and insert rows, and so on.</p> <p data-bbox="619 501 1275 591">VALUE=False. After aborting a transaction, the only operations allowed on a rowset are to release row and accessor handles and to release the rowset.</p>
Preserve on Commit	<p data-bbox="619 609 1246 671">Specifies what operations are allowed on a rowset following a transaction.</p> <p data-bbox="619 685 1282 777">VALUE=True. After committing a transaction, the rowset remains active. That is, it is possible to fetch new rows, update, delete, and insert rows, and so on.</p> <p data-bbox="619 791 1268 881">VALUE=False. After committing a transaction, the only operations allowed on a rowset are to release row and accessor handles and to release the rowset.</p>
Quick Restart	<p data-bbox="619 899 1275 991">IRowset::RestartPosition is not expensive to execute and does not execute the command that created the rowset again.</p>
Remove Deleted Rows	<p data-bbox="619 1008 1232 1071">Specifies whether static and keyset-driven cursors remove deleted rows.</p> <p data-bbox="619 1085 796 1107">VALUE=False</p>
Report Multiple Changes	<p data-bbox="619 1124 1225 1187">Specifies whether an update or delete can affect multiple rows.</p> <p data-bbox="619 1201 1260 1291">VALUE=False. An update or delete always affects a single row or the provider cannot detect whether it affects multiple rows.</p>
Return Pending Inserts	<p data-bbox="619 1308 1246 1371">Specifies whether the methods that fetch rows can return pending insert rows.</p> <p data-bbox="619 1385 796 1407">VALUE=False</p>
Row Privileges	<p data-bbox="619 1425 1260 1487">Specifies whether the provider restricts access rights on a row-to-row basis.</p> <p data-bbox="619 1501 796 1524">VALUE=False</p>

Table 4-22. Dynamic Properties Used for the Recordset Object (cont.)

ADO Property	Description and Default
Row Threading Model	<p>Specifies the threading model of the rowsets generated by the command.</p> <p>VALUE=1. The provider supports the free-threaded model.</p>
Scroll Backward	<p>Specifies whether the rowset can scroll backwards. This property is Read/Write.</p> <p>VALUE=False. IRowsOffset must be non-negative.</p>
Server Cursor	<p>Specifies whether to materialize the cursor on the server.</p> <p>VALUE=False. The provider determines where to locate the cursor.</p>
Skip Deleted Bookmarks	<p>Specifies whether the rowset allows a Get or Find process to continue if a bookmark row was deleted, is a row to which the data consumer does not have access rights, or is no longer a member of the rowset.</p>
Strong Row Identity	<p>Specifies whether the handles of newly inserted rows can be compared successfully.</p> <p>VALUE=False. There is no guarantee that the handles of newly inserted rows can be compared successfully.</p>
Unique Rows	<p>Specifies whether rows in a rowset can be uniquely identified by their column values.</p> <p>VALUE=False. Rows in the rowset may or may not be uniquely identified by their column values.</p>
Updatability	<p>Specifies the supported methods on IRowsetChange. This property should be used in conjunction with IRowsetChange. If IRowsetChange is TRUE and Updatability is not set, then it is provider-specific which methods are supported on IRowsetChange.</p> <p>VALUE=0</p>
Use Bookmarks	<p>Specifies whether the rowset supports bookmarks.</p> <p>VALUE=False. The rowset does not support bookmarks.</p>

The SequeLink ADO Provider supports the following standard ADO Recordset collections as listed in [Table 4-23](#).

Table 4-23. Mapping Collections for the ADO Recordset Object

Collection Object	OLE DB Method
Fields	IColumnsInfo
Properties	IRowset::GetPropertyInfo IRowset::GetProperties IRowset::SetProperties

Data Shaping

Data shaping allows you to create hierarchical recordsets with data exposed by an ADO/OLE DB data provider. This is done through the MSDataShape OLE DB provider, which is part of the MDAC. MSDataShape acts as a service component to the SequeLink ADO Provider to expose data shaping functionality.

To perform queries, MSDataShape uses a Shape language, which is functionally similar to SQL. For more information about the Shape language, refer to your MDAC documentation.

You specify the provider in the Connection object connect string as `Data Provider=DataDirect SequeLink ADO Provider`. The provider supplying data shaping support is specified in the Connection object `Provider` property as `MSDataShape`.

For example, the following code fragment can be used to create hierarchical recordsets with data exposed by the SequeLink ADO Provider using the SequeLink ADO data source named *HR*:

```
Dim cnn As New ADODB.Connection
cnn.Provider = "MSDataShape"
cnn.Open
```

```
"Shape Provider = DataDirect SequeLink ADO Provider;  
DataSourceName = HR;  
User ID = Mary Smith;  
Password = human"
```

Persisting Information

A data source object can be persisted (saved). The SequeLink ADO Provider uses the `IPersist` and the `IPersistFile` interfaces to persist the class ID and the values of data source properties set by the data consumer. With the `IPersistFile` interface, the data consumer saves the information to a file.

When the data consumer loads the persisted data source, the data provider retrieves the saved information. All of the initialization properties return to the state that was current when the data source was persisted. The stored values overwrite the values of any properties the data consumer might have set.

Using Rowsets

ADO/OLE DB data providers use rowsets to expose data in tabular form. The SequeLink ADO Provider supports the `IOpenRowset` interface, which retrieves all data from a table for a consumer. In addition, the provider supports the `ICommand` interface, which allows a consumer to get a rowset that meets a specific criteria.

For information on the schema rowsets supported, see the ["Supported Schema Rowsets" on page 133](#).

For more information about rowsets, refer to your Microsoft OLE DB programming documentation.

Unicode Support

The SequeLink ADO Provider supports Unicode.

When the string is data, the SequeLink ADO Provider returns the string to the data consumer as ANSI strings. If the data consumer requests a different data type through the bindings, the SequeLink ADO Provider performs the appropriate conversion.

When the string is not data, the SequeLink ADO Provider converts the string to Unicode format before returning it to the data consumer. This is required to conform to the OLE DB specification.

Mapping Data Types

Refer to [Appendix B “Data Types and Isolation Levels” on page 309](#) for information on the way the underlying data provider’s data types map to the standard OLE DB data types. `IColumns::GetColumnInfo` and `ICommandWithParameters::GetParameterInfo` are used to report OLE DB data types.

NOTE: Always use four-digit years for conversions from variant types to date/time types. Using two-digit years is not supported and will result in undefined behavior.

Specifying Application IDs

Application IDs are alphanumeric strings passed by a SequeLink Client that identify the client application to a SequeLink service that has been configured to accept connections only from specific application IDs.

For more information about configuring SequeLink services to accept connections only from specific application IDs, refer to the *SequeLink Administrator's Guide*.

Specifying Application IDs Explicitly

Using the SequeLink ADO Client, the client application specifies the following *key-value* pair in the DBPROP_INIT_PROVIDERSTRING property of the DBPROPSET_DBINITALL property set:

```
ApplicationID=MyAppID;
```

where *myAppID* is the application ID.

Generating Application IDs Automatically

Using the SequeLink ADO Client, the client application specifies the following *key-value* pairs in the DBPROP_INIT_PROVIDERSTRING property of the DBPROPSET_DBINITALL property set:

```
Automatic Application ID=x
```

where *x* is either 1, 2, or 3.

Error Handling

The following types of errors can occur when you are using the SequeLink ADO Client:

- SequeLink ADO Provider errors
- SequeLink Client errors
- SequeLink Server errors
- Database errors

SequeLink ADO Provider Errors

An error generated by the SequeLink ADO Provider has the following format:

```
[MERANT] [SequeLink ADO provider] message
```

For example:

```
[MERANT] [SequeLink ADO provider] Invalid precision specified.
```

The native error code is always zero (0).

If you receive this type of error, check the last ADO call your application made. Contact your ADO or OLE DB application vendor, or refer to the ADO and OLE DB documentation available from Microsoft.

SequeLink Client Errors

An error generated by the SequeLink ADO Client has the following format:

```
[MERANT] [SequeLink ADO provider] [SequeLink Client] message
```

For example:

```
[MERANT] [SequeLink ADO provider] [SequeLink Client] Memory allocation error occurred.
```

Use the native error code to look up details about the possible cause of the error. For a list of all error codes and messages, refer to the *SequeLink Troubleshooting Guide and Reference*.

SequeLink Server Errors

An error generated by SequeLink Server has the following format:

```
[MERANT] [SequeLink ADO provider] [SequeLink Server] message
```

For example:

```
[MERANT] [SequeLink ADO provider] [SequeLink Server] Only Select statements are allowed in this read-only connection.
```

Use the native error code to look up details about the possible cause of the error. For a list of all error codes and messages, refer to the *SequeLink Troubleshooting Guide and Reference*.

Database Errors

An error generated by the database has the following format:

```
[MERANT] [SequeLink ADO provider] [...] message
```

For example:

```
[MERANT] [SequeLink ADO provider] [Oracle] ORA-00942:table or view does not exist.
```

Use the native error code to look up details about the possible cause of the error. For these details, refer to your database documentation.

Part 3: Developing JDBC Applications

This part contains the following chapters:

- [Chapter 5 “Using the SequeLink Java Client” on page 207](#) provides information about using JDBC applications with the SequeLink Java Client.
- [Chapter 6 “Using JDBCTest” on page 233](#) introduces JDBCTest, a tool that allows you to test and learn the JDBC API, and contains a tutorial that takes you through a working example of its use.
- [Chapter 7 “Tracking JDBC Calls” on page 261](#) introduces Spy, a tool that allows you to track JDBC calls, and describes how to use it.
- [Chapter 8 “Developing JDBC Applications” on page 269](#) provides information about developing JDBC applications for SequeLink environments.

5 Using the SequeLink Java Client

This chapter provides information about using JDBC applications with the SequeLink Java Client.

About the SequeLink Java Client

The SequeLink Java Client provides JDBC access through any Java-enabled applet, application, or application server. It delivers high-performance point-to-point and n-tier access to industry-leading data stores across the Internet and intranets. The SequeLink Java Client is optimized for the Java environment, allowing you to incorporate Java technology and extend the functionality and performance of your existing system. The SequeLink Java Client includes:

- SequeLink JDBC Driver
- SequeLink Proxy Server
- JDBC Spy
- JDBC Test

SequeLink JDBC Driver

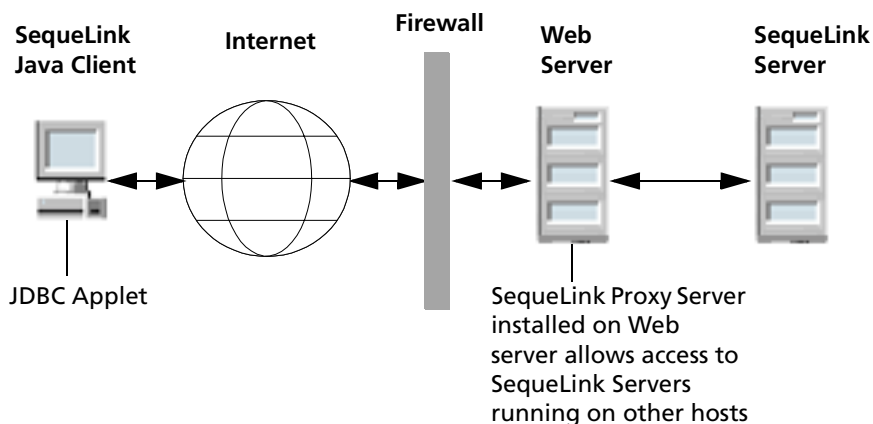
The SequeLink JDBC Driver is compliant with the Java Database Connectivity (JDBC) 2.0 specification. The SequeLink JDBC Driver also supports the JDBC 2.0 Optional Package, which provides the following functionality:

- Java Naming Directory Interface (JNDI) for naming databases
- Connection Pooling
- Distributed Transactions Management support

SequeLink Proxy Server

Installing the SequeLink Proxy Server on the Web server from which your JDBC applets are downloaded allows untrusted applets to connect to SequeLink Servers on hosts other than the Web server as shown in [Figure 5-1 “SequeLink Proxy Server Installed on a Web Server” on page 208](#).

Figure 5-1. SequeLink Proxy Server Installed on a Web Server



In addition, you can use Secure Socket Layer (SSL) encryption with the proxy server to encrypt data between the SequeLink Proxy Server and the SequeLink Java Client. You can also use SSL with a Java application running on your Intranet to secure data over your entire network by installing the SequeLink Proxy Server on the same machine as the SequeLink Server. For example, you may want to use SSL to encrypt the data sent between an application server and the data store serviced by a SequeLink Server on another machine. For more information about SSL, refer to the *SequeLink Administrator's Guide*.

JDBCSpy

Spy is a tracking tool for JDBC calls. Spy passes calls issued by an application to an underlying JDBC driver and logs detailed information about those calls. It provides the following advantages:

- Logging is JDBC 1.22- and JDBC 2.0-compliant, including support for the JDBC 2.0 Optional Package
- Logging is consistent, regardless of the JDBC driver used.
- All parameters and function results for JDBC calls can be logged.
- Even if your JDBC driver does not support logging, you can still log JDBC calls.
- Logging can be enabled, without changing the application, using the `SequeLinkDataSource` object.

JDBCTest

JDBCTest contains menu selections that correspond to specific JDBC functions—for example, connecting to a database or passing a SQL statement. It allows you to:

- Execute a single JDBC method or execute multiple JDBC methods simultaneously, so that you can easily perform some common tasks, such as returning result sets
- Display the results of all JDBC function calls in one window, while displaying fully commented, Java JDBC code in an alternate window

SequeLink Java Client Directory Structure

[Table 5-1](#) shows the SequeLink Java Client directory after installation and provides a description of the files.

Table 5-1. SequeLink Java Client Directory and Files

Directories and Files	Description
driver/examples/CheckAgainstCertificateFromFile.java driver/examples/CheckAgainstCertificateFromJar.java driver/examples/KeyStoreCertificateChecker.java	Contains Java source files that provide examples of certificate checkers.
driver/examples/JNDI_FILESYSTEM_Example.java driver/examples/JNDI_LDAP_Example.java	Contains Java source files that allows you to create JDBC data sources. These source files must be adapted for your environment, and subsequently compiled and run.

Table 5-1. Sequelink Java Client Directory and Files (cont.)

Directories and Files	Description
driver/lib/sljc.jar	JAR file containing all classes of the Sequelink JDBC Driver implementing the JDBC 2.0 Core API. To load the driver, add this path to your CLASSPATH variable.
driver/lib/sljcx.jar	JAR file containing all classes of the Sequelink JDBC Driver implementing the JDBC 2.0 Optional Package. To use the JDBC 2.0 Optional Package, add this path to your CLASSPATH variable.
driver/lib/slrsa_rc4.jar driver/lib/slrsa_rc4_11.jar	JAR files containing implementations of the RSA and RC4 algorithms. NOTE: slrsa_rc4_11.jar is the applet/JDK 1.1 version.
driver/lib/slssl.jar driver/lib/slssl_11.jar	JAR files containing all classes of the Sequelink JDBC Driver that implement Secure Socket Layer (SSL) encryption. NOTE: slssl_11.jar is the applet/JDK 1.1 version.
jdbctest/classes	Contains all the JDBCTest classes. To use JDBCTest, add this path to your CLASSPATH variable.
jdbctest/jdbctest.bat	Batch file that starts JDBCTest.
jdbctest/jdbctest.html	HTML file that starts JDBCTest as an applet.
jdbctest/jdbctest.sh	UNIX shell script that starts JDBCTest.
install.idb	Support file for the uninstaller.

Table 5-1. Sequelink Java Client Directory and Files (cont.)

Directories and Files	Description
proxy/cmdsrv.exe	Executable that registers the Sequelink Proxy Server as a Windows service.
proxy/decrypt.bat proxy/encrypt.bat	Batch files that decrypt and encrypt the private key of the Sequelink Proxy Server certificate.
proxy/decrypt.sh proxy/encrypt.sh	UNIX shell scripts that decrypt and encrypt the private key of the Sequelink Proxy Server certificate.
proxy/proxyserver.bat	Batch file that starts the Sequelink Proxy Server.
proxy/proxyserver.sh	UNIX shell script that starts the Sequelink Proxy Server.
proxy/cert/	Contains demo certificates.
proxy/demos/com/merant/sequelink/demo/ demo.properties proxy/demos/com/merant/sequelink/demo/ GenerateDemoCertificates\$DN.class proxy/demos/com/merant/sequelink/demo/ GenerateDemoCertificates.class	Contains Java files you can use to generate certificates.
proxy/demos/com/merant/sequelink/demo/ KeyTool.class	Contains a Java class file that extracts certificates from a Java2 KeyStore and converts certificates to different formats.
proxy/lib/slproxy.jar	Jar file containing all classes for the Sequelink Proxy Server.
proxy/lib/slrsa_rc4.jar proxy/lib/slrsa_rc4_11.jar	JAR files containing implementations of the RSA and RC4 algorithms. NOTE: slrsa_rc4_11.jar is the applet/JDK 1.1 version.

Table 5-1. SequeLink Java Client Directory and Files (cont.)

Directories and Files	Description
driver/lib/slssl.jar driver/lib/slssl_11.jar	JAR files containing all classes of the SequeLink Proxy Server that implement Secure Socket Layer (SSL) encryption. NOTE: slssl_11.jar is the applet/JDK 1.1 version.
proxy/log	The directory that contains all messages logged by the SequeLink Proxy Server.
spy/lib/spy.jar	JAR file containing all Spy classes. To use Spy, add this path to your CLASSPATH variable.
sun/lib/jdbc2_0-stdext.jar	JAR file containing redistributable Sun Microsystems components for the JDBC 2.0 Optional Package.
sun/lib/jndi.jar	JAR file containing redistributable Sun Microsystems components for JNDI 1.2.
sun/lib/jta-spec1_0_1.jar	JAR file containing redistributable Sun Microsystems components for JTA 1.0.1.
sun/lib/fs/fscontext.jar sun/lib/fs/providerutil.jar	JAR files containing redistributable Sun Microsystems components for the File System JNDI Provider.
sun/lib/ldap/jaas.jar sun/lib/ldap/ldap.jar sun/lib/ldap/ldapbp.jar sun/lib/ldap/providerutil.jar	JAR files containing redistributable Sun Microsystems components for the LDAP JNDI Provider.
uninstall.class	Executable that uninstalls SequeLink Java Client.

Loading the SequeLink JDBC Driver

To use the SequeLink JDBC Driver, you first must register it with the JDBC Driver Manager. You can register the SequeLink JDBC Driver in any of the following ways:

- **Method 1:** Set the Java property `sql.drivers` using the Java `-D` option. The `sql.drivers` property is defined as a colon-separated list of driver class names. For example:

```
com.merant.sequelink.jdbc.SequeLinkDriver:  
sun.jdbc.odbc.JdbcOdbcDriver
```

The `sql.drivers` property can be set like other Java properties, using the `-D` option. For example:

```
java -Dsql.drivers=  
com.merant.sequelink.jdbc.SequeLinkDriver
```

- **Method 2:** Set the Java property `sql.drivers` from within your Java application or applet. To do this, code the following lines in your JDBC application, and call `DriverManager.getConnection()`:

```
Properties p = System.getProperties();  
p.put ("sql.drivers",  
"com.merant.sequelink.jdbc.SequeLinkDriver");  
System.setProperties (p);
```

- **Method 3:** Explicitly load the driver class using the standard `Class.forName()` method. To do this, code the following lines and call `DriverManager.getConnection()`:

```
Class.forName ("com.merant.sequelink.jdbc.  
SequeLinkDriver");
```

Specifying SequeLink JDBC Driver Connection URLs

The connection URL format depends on whether you are using SSL encryption. For more information about SSL encryption, refer to the *SequeLink Administrator's Guide*.

If not using SSL encryption, the connection URL format is:

```
jdbc:sequelink://hostname:port[;key=value]...
```

If using SSL encryption, the connection URL format is:

```
jdbc:sequelink:ssl://hostname:port[;key=value]...
```

where:

<i>hostname</i>	is the TCP/IP address or TCP/IP host name of the server to which you are connecting. NOTE: Untrusted applets cannot open a socket to a machine other than the originating host. For more information about untrusted applets, refer to the <i>SequeLink Administrator's Guide</i> .
<i>port</i>	is the TCP/IP port on which the SequeLink server is listening. A default installation of SequeLink Server uses the port 19996.
<i>key=value</i>	specifies connection properties. For a list of connection properties and their valid values, see "JDBC Connection Properties" on page 224.

JDBC Connection URL Examples:

The following examples show some typical SequeLink JDBC Driver connection URLs:

```
jdbc:sequelink://sequelinkhost:19996;
```

```
jdbc:sequelink://189.23.5.25:19996;user=john;  
password=whatever
```

```
jdbc:sequelink://189.23.5.132:19996;databaseName=stores7
```

```
jdbc:sequelink://189.23.5.68:19996;databaseName=pubs;  
HUser=john;HPassword=whatever
```

```
jdbc:sequelink://sequelinkhost:4006;  
databaseName=pubs;DBUser=john;DBPassword=whatever
```

```
jdbc:sequelink:ssl://mysecurehost:9500;  
cipherSuites=SSL_DH_anon_WITH_RC4_128_MD5
```

```
jdbc:sequelink:ssl://mysecurehost:9502;  
cipherSuites=SSL_DHE_RSA_WITH_DES_CBC_SHA;  
certificateChecker=CheckAgainstCertificateFromJar
```

The preceding examples do not show the user and password connection properties. Typically, these properties are specified in the connection properties stored in the `java.util.Properties` object, which is supplied as a parameter to the `getConnection` method.

Configuring JDBC Data Sources

Using JDBC data sources provides flexibility to make environment changes and reduces the time it takes to reconfigure your infrastructure when a change is made. For example, if a SequeLink service is reconfigured (for example, moved to another machine, port, and so on), the SequeLink administrator can change and run the configuration source file described in [“Creating and Managing JDBC Data Sources” on page 218](#), reassigning the logical name of the JDBC data source to the changed data source configuration. As a result, the client application code does not have to change, because it only refers to the logical name of the JDBC data source.

SequeLink supports the following JDBC data source implementations defined by the JDBC 2.0 Optional Package:

- JNDI for Naming Databases
- Connection pooling
- Distributed Transaction Management Support

NOTES:

- You must include the `javax.sql.*` and `javax.naming.*` classes to create and use JDBC data sources. The SequeLink Java Client provides all the necessary JAR files that contain the required classes and interfaces.
- In addition, you must include the `javax.transaction.xa.*` class to use and implement distributed transactions.

Creating and Managing JDBC Data Sources

JDBC data sources are implemented using a SequeLink class `com.merant.sequelink.jdbcx.datasource.SequeLinkDataSource`. This single data source implementation implements the following interfaces defined in the JDBC 2.0 Optional Package:

- `javax.sql.DataSource`
- `javax.sql.ConnectionPoolDataSource`
- `javax.sql.XADataSource`

The SequeLink Data Source implementation implements both the `java.io.Serializable` and `javax.naming.Referenceable` interfaces. The interface that is used depends on the service provider you are using and how the `SequeLinkDataSource` object is saved in your JNDI environment.

Your SequeLink Java Client installation contains the following examples that show how to create and use JDBC data sources:

- `JNDI_LDAP_Example.java`. Use this example to create a JDBC data source and save it in your LDAP directory, using the JNDI Provider for LDAP.
- `JNDI_FILESYSTEM_Example.java`. Use this example to create a JDBC data source and save it in your local file system, using the File System JNDI Provider.

Using JNDI for Naming Databases

Instead of using connection URLs, client applications can access a JNDI-named data source using a logical name to retrieve the `javax.sql.DataSource` object. This object loads the SequeLink JDBC Driver and establishes the connection to the SequeLink service.

Once a JDBC data source has been registered with JNDI, it can be used by your JDBC application as shown in the following example:

```
Context ctx = new InitialContext();
DataSource ds = (DataSource)ctx.lookup("jdbc/EmployeeDB");
Connection con = ds.getConnection("scott", "tiger");
```

In this example, the JNDI environment is first initialized. Next, the initial naming context is used to find the logical name of the JDBC data source. The `Context.lookup()` method returns a reference to a Java object, which is narrowed to a `javax.sql.DataSource` object. Finally, the `DataSource.getConnection()` method is called to establish a connection with the SequeLink service.

For instructions on creating JDBC data sources, see [“Creating and Managing JDBC Data Sources” on page 218](#).

Using Connection Pooling

Connection pooling allows you to reuse connections rather than create a new one every time the SequeLink Client needs to establish a data access connection. Connection pooling manages connection sharing across different user requests to maintain performance and reduce the number of new connections that must be created. For example, compare the transaction sequences shown in [“Example A: Without Connection Pooling” on page 220](#) and [“Example B: With Connection Pooling” on page 220](#).

Example A: Without Connection Pooling

- 1 The client application creates a connection.
- 2 The client application sends a data access query.
- 3 The client application obtains the result set of the query.
- 4 The client application displays the result set to the end user.
- 5 The client application ends the connection.

Example B: With Connection Pooling

- 1 The client checks the connection pool for an unused connection.
- 2 If an unused connection exists, it is returned by the pool implementation; otherwise, it creates a new connection.
- 3 The client application sends a data access query.
- 4 The client application obtains the result set of the query.
- 5 The client application displays the result set to the end user.
- 6 The client application returns the connection to the pool.

NOTE: The client application still calls "close()", but the connection remains open and the pool is notified of the close request.

The pool implementation creates "real" database connections using the `getPooledConnection()` method of `ConnectionPoolDataSource`. Then, the pool implementation registers itself as a listener to the `PooledConnection`. When a client application requests a connection, the pool implementation is notified by the `ConnectionEventListener` interface that the connection is free and available for reuse. The pool implementation is also notified by the `ConnectionEventListener` interface when the client somehow corrupts the database connection, so that the pool implementation can remove that connection from the pool.

Once a JDBC data source has been registered with JNDI, it can be used by your JDBC application as shown in the following example, typically through a third-party connection pool tool:

```
Context ctx = new InitialContext();
ConnectionPoolDataSource ds = (ConnectionPoolDataSource)
ctx.lookup("jdbc/EmployeeDB");
pooledConnection pcon = ds.getPooledConnection("scott",
"tiger");
```

In this example, the JNDI environment is first initialized. Next, the initial naming context is used to find the logical name of the JDBC data source. The `Context.lookup()` method returns a reference to a Java object, which is narrowed to a `javax.sql.ConnectionPoolDataSource` object. Finally, the `ConnectionPoolDataSource.getPooledConnection()` method is called to establish a connection with the SequeLink service.

For instructions on creating JDBC data sources, see [“Creating and Managing JDBC Data Sources” on page 218](#).

Using the Java Transaction API

[Table 5-2](#) lists which databases are supported for the Java Transaction API (JTA) by the SequeLink Java Client.

Table 5-2. Support for the Java Transaction API (JTA) by the SequeLink Java Client

Database	JTA Supported?
Oracle7	No
Oracle8	Yes
Informix 7	No
Informix 9	Yes
DB2 V4 on OS/390	No
DB2 V5, V6 on OS/390	Yes

Table 5-2. Support for the Java Transaction API (JTA) by the SequeLink Java Client (cont.)

Database	JTA Supported?
DB2 V5, V6.1, 7.1 on UNIX, Windows NT, Windows 2000	Yes
Sybase 11	No
Sybase 12	Yes
SQL Server 7	Yes
SQL Server 2000	Yes

Once a JDBC data source has been registered with JNDI, it can be used by your JDBC application as shown in the following example, typically through application server software:

```
Context ctx = new InitialContext();
XADataSource ds = (XADataSource)
ctx.lookup("jdbc/EmployeeDB");
XAConnection xacon = ds.getXAConnection("scott", "tiger");
```

In this example, the JNDI environment is first initialized. Next, the initial naming context is used to find the logical name of the JDBC data source. The `Context.lookup()` method returns a reference to a Java object, which is narrowed to a `javax.sql.XADataSource` object. Finally, the `XADataSource.getXAConnection()` method is called to establish a connection with the SequeLink service.

For instructions on creating JDBC data sources, see [“Creating and Managing JDBC Data Sources” on page 218](#).

Specifying Connection Properties

You can specify connection properties using a connection URL, the JDBC Driver Manager, or JDBC data sources. The properties you can specify depend on the connection method you choose. For a list of the connection properties, see [“JDBC Connection Properties” on page 224](#).

Using Connection URLs or the JDBC Driver Manager

In order of precedence, you can specify connection properties using:

- `getConnection(url, user, password)`, where *user* and *password* are specified using the `getConnection` method defined in `java.sql.DriverManager`
- `java.util.properties` object
- Connection URL specified using the URL parameter of the `getConnection` method defined in `java.sql.DriverManager`
- Server data sources specified using the SequeLink Manager

Using JDBC Data Sources

In order of precedence, you can specify connection properties using:

- `getConnection(user, password)`, where *user* and *password* are specified using the `getConnection` method defined in `javax.sql.DataSource`

- JDBC DataSource object
- Server data sources specified using the SequeLink Manager

JDBC Connection Properties

[Table 5-3](#) lists the JDBC connection properties supported by the SequeLink JDBC Driver, describes each property, and specifies the methods with which it can be specified.

Table 5-3. JDBC Properties

Property	Description
blockFetchForUpdate	<p>BlockFetchForUpdate={0 1}. Specifies a workaround connection attribute. When the isolation level is Read Committed and a SELECT FOR UPDATE statement is issued against some data stores, the SequeLink Java Client does not lock the expected row.</p> <p>When set to 0, the appropriate row is locked.</p> <p>When set to 1 (the initial default), the appropriate row is not locked.</p> <p>WARNING: Specifying 1 will degrade the performance for SELECT FOR UPDATE statements because rows will be fetched one at a time.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL ■ java.util.properties ■ server data source

Table 5-3. JDBC Properties (cont.)

Property	Description
certificateChecker	<p>The fully qualified class name of a user-defined server certificate checker class. When the SequeLink Client and SequeLink Server have agreed on a SSL cipher suite that requires a server certificate, this class is used to verify the server certificate on behalf of the client. The class must be an implementation of the <code>com.merant.sequelink.cert.CertificateCheckerInterface</code> interface.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL ■ <code>java.util.properties</code> <p>For more information about certificate checker classes, refer to the <i>SequeLink Administrator's Guide</i>.</p>
cipherSuites	<p>The Secure Socket Layer (SSL) cipher suites with which the SequeLink Java Client can use to connect. This property is required when <code>networkProtocol=ssl</code>.</p> <p>For a list of supported cipher suites, refer to the <i>SequeLink Administrator's Guide</i>.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL ■ <code>java.util.properties</code>
databaseName	<p>The name of the data store to which you want to connect.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL ■ <code>java.util.properties</code> ■ server data source

Table 5-3. JDBC Properties (cont.)

Property	Description
DBUser	<p>The data store user name, which may be required depending on the server configuration.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL ■ java.util.properties
DBPassword	<p>The data store password, which may be required depending on the server configuration.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL ■ java.util.properties
HUser	<p>The host user name, which may be required depending on the server configuration.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL ■ java.util.properties
HPassword	<p>The host password, which may be required depending on the server configuration.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL ■ java.util.properties
networkProtocol	<p>networkProtocol={socket ssl}. Specifies the protocol to be used.</p> <p>When set to socket (the initial default), SSL encryption is not used.</p> <p>When set to SSL, SSL encryption is used.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL ■ java.util.properties

Table 5-3. JDBC Properties (cont.)

Property	Description
newPassword	<p>The new host password to be used. If specified and applicable to the connection, the SequeLink password change mechanism is invoked. When the password has been changed successfully, the following warning is returned:</p> <pre data-bbox="576 487 1225 579">[MERANT] [SequeLink JDBC driver] [SequeLink Server] The user password was changed successfully</pre> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL ■ java.util.properties <p>For more information about the SequeLink password change mechanism, refer to the <i>SequeLink Administrator's Guide</i>.</p>
password	<p>The host or data store password, which may be required depending on the server configuration.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ getConnection ■ JDBC data source ■ URL ■ java.util.properties
portNumber	<p>The TCP/IP port on which the SequeLink service is listening.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL
serverName	<p>The TCP/IP address of the SequeLink server in dotted format or host name format.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL

Table 5-3. JDBC Properties (cont.)

Property	Description
SLKStaticCursorLongColBuffLen	<p>The amount of data (in KB) that is buffered for SQL_LONGVARCHAR and SQL_LONGVARBINARY columns with an insensitive result set.</p> <p>The default is 4.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL ■ java.util.properties ■ server data source
serverDataSource	<p>A property that specifies a string to identify the server data source to be used for the connection. If unspecified, the configuration of the default server data source will be used for the connection.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ JDBC data source ■ URL ■ java.util.properties
user	<p>The host or data store user name, which may be required depending on the server configuration.</p> <p>This property can be specified using:</p> <ul style="list-style-type: none"> ■ getConnection ■ JDBC data source ■ URL ■ java.util.properties

Testing SequeLink JDBC Connections

For instructions on connecting with the SequeLink Java Client using JDBCtest, see [Chapter 6 “Using JDBCtest” on page 233](#).

Using the SequeLink Java Client on a Java 2 Platform

When using the SequeLink JDBC Driver on a Java 2 Platform with the standard security manager enabled, you must give the driver some additional permissions. Refer to your Java 2 Platform documentation for more information about the Java 2 Platform security model and permissions.

You can run an application on a Java 2 Platform with the standard security manager using:

```
"java -Djava.security.manager application_class_name"
```

where *application_class_name* is the class name of the application.

Web browser applets running in the Java 2 plug-in are always running in a JVM with the standard security manager enabled. To enable the necessary permission, you must add them to the security policy file of the Java 2 Platform. This security policy file can be found in the `jre\lib\security` subdirectory of the Java 2 Platform installation directory.

To use JDBC data sources, all code bases must have the following permissions:

```
// permissions granted to all domains
grant {
// DataSource access
permission java.util.PropertyPermission "java.naming.*", "read,write";
// Adjust the server host specification for your environment
permission java.net.socketPermission "*.merant.be:0-65535", "connect";
};
```

To use insensitive scrollable cursors, all code bases must have access to temporary files:

```
// permissions granted to all domains
grant {
// Permission to create and delete temporary files.
// Adjust the temporary directory for your environment.
permission java.io.FilePermission "C:\\TEMP\\-", "read,write,delete";
};
```

To use SSL or other data privacy functionality, the following permissions are required for the SequeLink Java Client code base only:

```
// permissions granted to the SequeLink Java Client code base only
grant codeBase "file:/slje/lib/-" {
// Security providers
// Only needed when using SSL or other data privacy functionality
// (e.g. fixed key DES/3DES)
permission java.security.SecurityPermission "putProviderProperty.IAIK";
permission java.security.SecurityPermission "insertProvider.IAIK";
permission java.security.SecurityPermission "putProviderProperty.SLJCE";
permission java.security.SecurityPermission "insertProvider.SLJCE";
};
```

Applets that connect to another server other than the one they are downloaded from must have the following permission:

```
// permissions granted to the SequeLink Java Client code base only
grant codeBase "file:/slje/lib/-" {
// TCP/IP

// Adjust the server host specification for your environment
permission java.net.SocketPermission "*.merant.be:0-65535", "connect";
};
```

NOTES:

- Make sure that you adjust the code base of the SequeLink Java Client for your environment. For an applet, this will probably start with "http://" or "https://".
- Make sure you adjust the server host specification and location of temporary files for your environment.

6 Using JDBCTest

This chapter provides information about JDBCTest, a tool that allows you to test and learn the JDBC API, and contains a tutorial that takes you through a working example of its use.

About JDBCTest

JDBCTest contains menu selections that correspond to specific JDBC functions—for example, connecting to a database or passing a SQL statement. It allows you to:

- Execute a single JDBC method or execute multiple JDBC methods simultaneously, so that you can easily perform some common tasks, such as returning result sets
- Display the results of all JDBC function calls in one window, while displaying fully commented, Java JDBC code in an alternate window

NOTE: JDBCTest does not support the JDBC 2.0 Optional Package.

JDBCTest Tutorial

This JDBCTest tutorial explains how to use the most important features of JDBCTest (and the JDBC API) and assumes that you can connect to an Oracle database with the standard available demo table or fine-tune the sample SQL statements shown in this example as appropriate for your environment.

NOTE: The step-by-step examples used in this tutorial do not show typical clean-up routines (for example, closing result sets and connections). These steps have been omitted to simplify the examples. Do not forget to add these steps when you use equivalent code in your applications.

Starting JDBCTest

How you start JDBCTest depends on whether you want to start it as an application or applet, and your Java Virtual Machine:



- **As a Java application on Windows:** Run the `jdbctest.bat` file located in the `jdbctest` directory.



- **As a Java application on UNIX:** Run the `jdbctest.sh` shell script located in the `jdbctest` directory.
- **As an applet:** Start your applet viewer or Web browser and open the HTML file `jdbctest.html` located in the `jdbctest` directory.
- **Using a Java Virtual Machine Other Than the JDK:** JDBCTest is a Java application. For instructions on starting a Java application with your Java Virtual Machine, refer to your Java Virtual Machine documentation. Regardless of the Java

Virtual Machine, you must make sure that your CLASSPATH includes:

- The directory containing the JDBCTest classes:
install_dir/jdbctest/classes
- The JAR file containing the SequeLink JDBC Driver classes:
install_dir/driver/lib/sljc.jar

where *install_dir* is your SequeLink Java Client installation directory. The class file containing the JDBCTest application is JDBCTest.class.

JDBCTest accepts a parameter that identifies a configuration file containing default options. By default, this file is config.txt. For more information about this configuration file, see [“Configuring JDBCTest” on page 236](#).

[Table 6-1](#) shows some examples of starting JDBCTest with some popular Java Virtual Machines, where *install_dir* is your SequeLink Java Client installation directory:

Table 6-1. Starting JDBCTest with Java Virtual Machines Other Than the JDK

Java Virtual Machine	Start Syntax
Java Runtime Environment 1.1.x (JRE)	<code>jre -cp install_dir/jdbctest/classes;install_dir/driver/ lib/sljc.jar JDBCTest Config.txt</code>
Java Runtime Environment 1.2 (JRE)	<code>java -classpath install_dir/jdbctest/classes;install_dir/driver/ lib/sljc.jar JDBCTest Config.txt</code>
Java 2 Platform (formerly Java Development Kit 1.2)	<code>java -classpath install_dir/jdbctest/classes;install_dir/driver/ lib/sljc.jar JDBCTest Config.txt</code>
JView from Microsoft	<code>Jview /cp:p install_dir/jdbctest/classes;install_dir/driver/ lib/sljc.jar JDBCTest Config.txt</code>

Configuring JDBCTest

The default JDBCTest configuration file is *install_dir/jdbctest/classes/Config.txt* where *install_dir* is your SequeLink Java Client installation directory. This file can be edited as appropriate for your environment using any text editor. All parameters are configurable, but the most commonly configured parameters are:

Databases	A list of comma separated JDBC URLs. You can use one of these URLs as a template when you make a JDBC connection. The default Config.txt file contains example URLs for most databases.
DefaultDatabase	The default JDBC URL to be used when you make a connection.

Connecting Using JDBCTest

- 1 Start JDBCTest as a Java application or applet.



- *As a Java application on Windows:* Run the jdbctest.bat file located in the jdbctest directory.



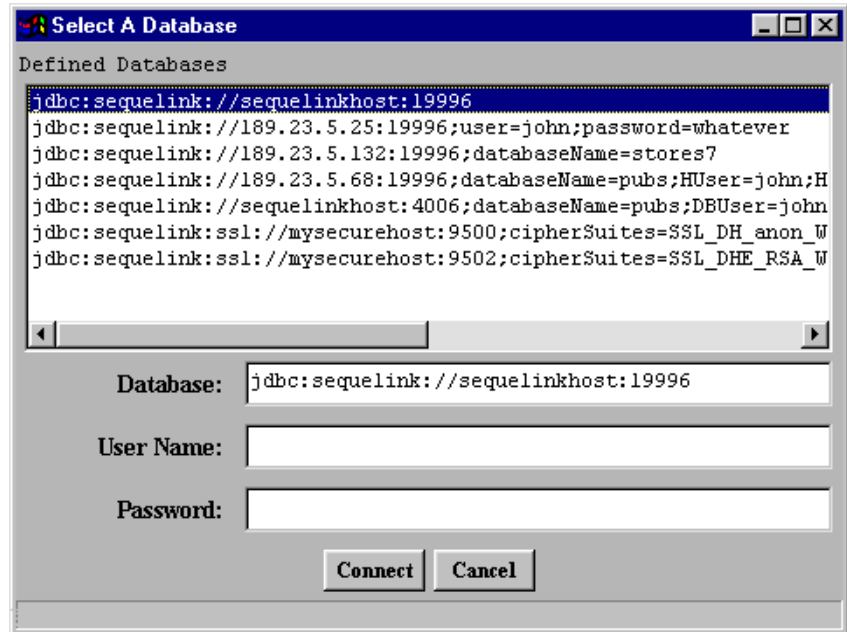
- *As a Java application on UNIX:* Run the jdbctest.sh shell script located in the jdbctest directory.
- *As an applet:* Start your applet viewer or Web browser and open the HTML file jdbctest.html located in the jdbctest directory.

- 2 From the JDBCTest Welcome window, click the **Press Here To Continue** button. The JDBCTest window appears.
- 3 Select **Driver / Register Driver**. JDBCTest prompts you for the JDBC driver you want to load.

- 4 In the Please Supply a Driver URL field, make sure that the following driver is specified; then, click **OK**.

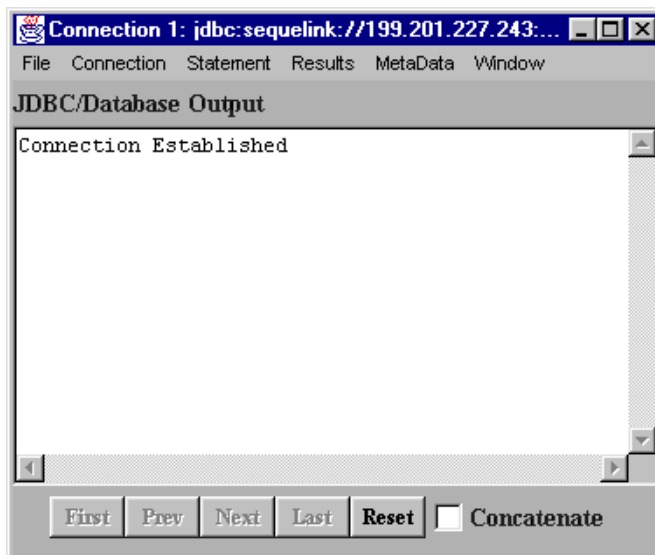
```
com.merant.sequelink.jdbc.SequeLinkDriver
```

- 5 Select **Connection / Connect To DB**. The Select A Database window appears with a list of default SequeLink JDBC Driver connection URLs.



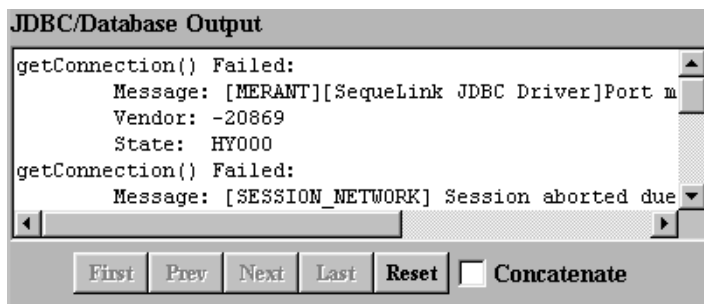
- 6 Select one of the default SequeLink JDBC Driver connection URLs. In the Database field, modify the default values of the connection URL appropriately for your environment.
- 7 In the User Name and Password fields, type the required user and password connection properties; then, click the **Connect** button. For information about JDBC connection properties, see ["JDBC Connection Properties"](#) on page 224.

- 8 If the connection was successful, the Connection window shows the Connection Established message in the JDBC/Database Output scroll box.



If the connection was successful, you can start using your JDBC applications with SequeLink.

If the connection was unsuccessful, you are returned to the JDBCTest window. The `getConnection()` Failed: message appears in the JDBC/Database Output scroll box. If your connection failed, refer to the *SequeLink Troubleshooting Guide and Reference*.



The connection window shows the following information:

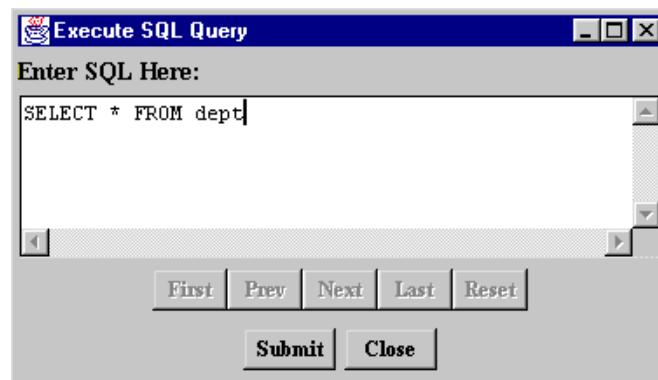
- In the JDBC/Database Output scroll box, a report indicating whether the last action succeeded or failed
- In the Java Code scroll box, the actual Java code used to implement the last action

TIP: Select the **Concatenate** check box to see the Java code of all previous actions; otherwise, the Java code of only the last action will be shown. The Concatenate check box is selected by default, which degrades performance, particularly when displaying large resultSets.

Executing a Simple Select Statement

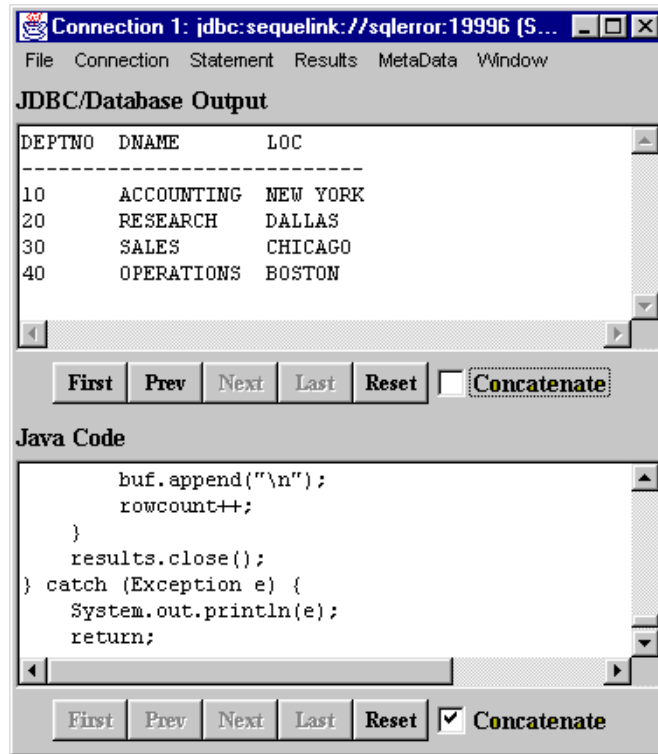
This example explains how to execute a simple Select statement and retrieve the results.

- 1 Select **Connection / Create Statement**. The connection window indicates that the creation of the statement was successful.
- 2 Select **Statement / Execute Stmt Query**. JDBCTest prompts for a SQL statement.
- 3 Specify the Select statement you want to execute.



Click **Submit**; then, click **Close**.

- 4 Select **Results / Show All Results**. The data from your result set is displayed.

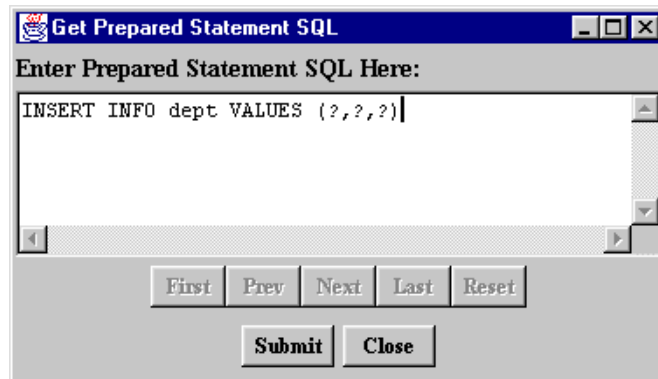


- 5 Scroll through the code in the Java Code scroll box to see which JDBC calls have been implemented by JDBCTest.

Executing a Prepared Statement

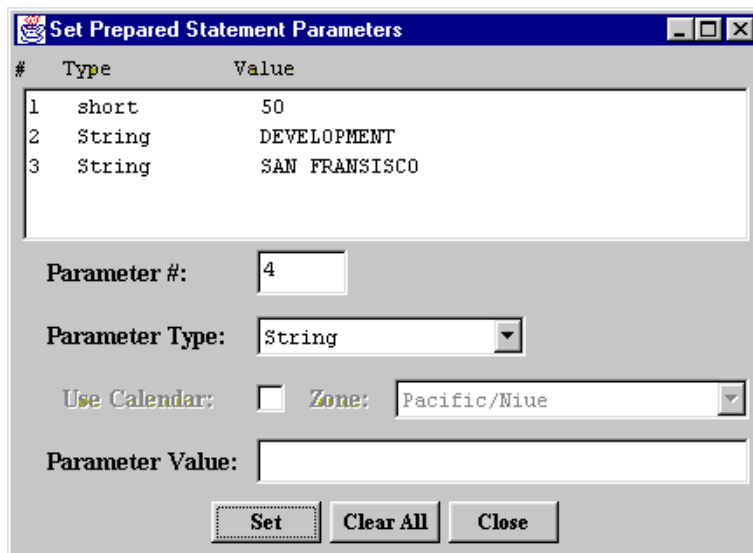
This example explains how to execute a parameterized statement multiple times.

- 1 Select **Connection / Create Prepared Statement**. JDBCTest prompts you for a SQL statement.
- 2 Specify the Insert statement you want to execute.



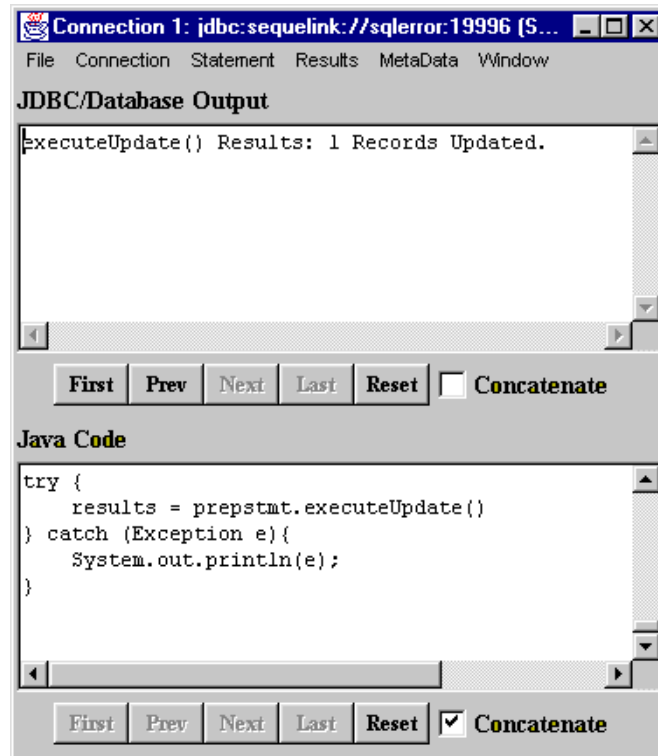
Click **Submit**; then, click **Close**.

- 3 Select **Statement / Set Prepared Parameters**. To set the values and type for each parameter:
 - a Type the parameter number.
 - b Select the parameter type.
 - c Type the parameter value.
 - d Click **Set** to pass this information to the JDBC driver.



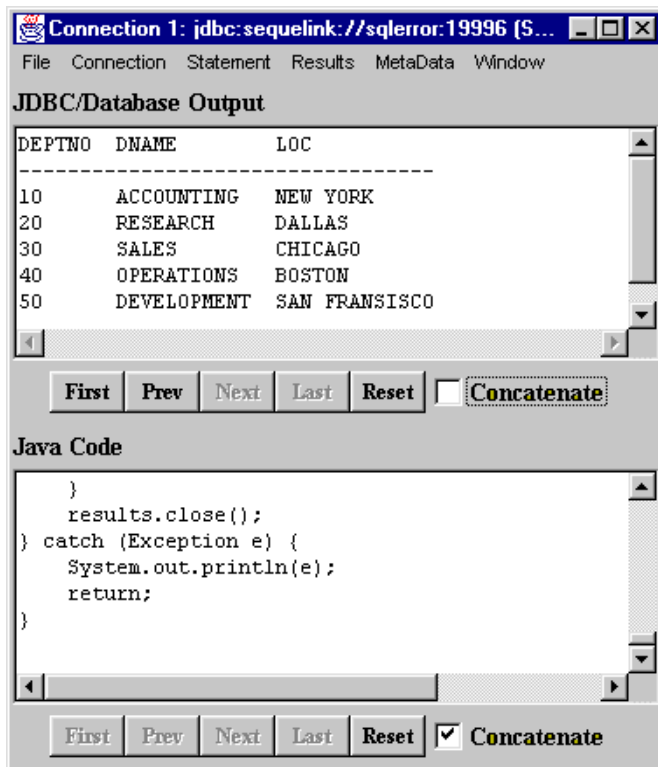
Click **Close**.

- 4 Select **Statement / Execute Stmt Update**. As expected, the JDBC/Database Output scroll box indicates that one row has been inserted.



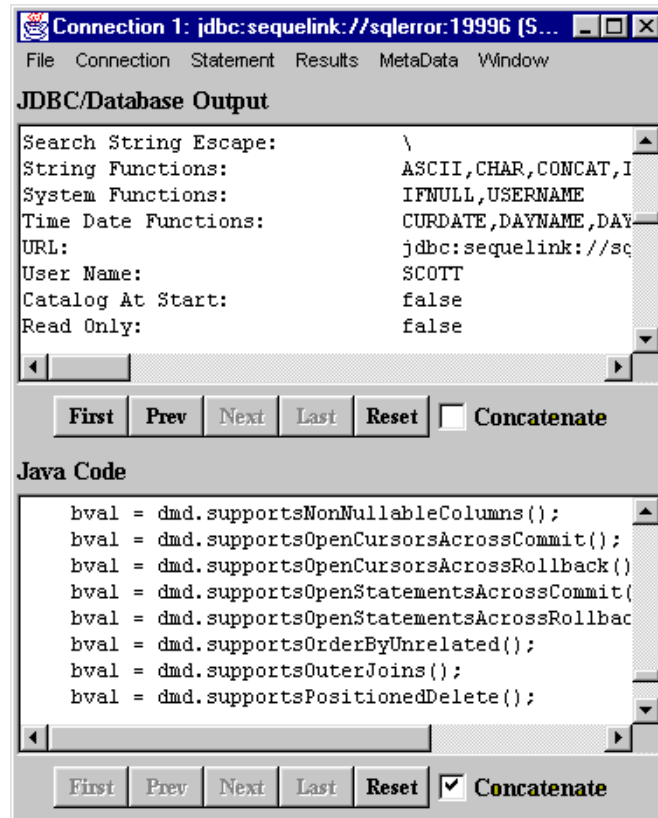
- 5 If you want to insert multiple records, repeat [Step 3](#) and [Step 4](#) for each record.

- 6 If you repeat the steps described in “Executing a Simple Select Statement” on page 239, you will see that the previously inserted records are also returned.



Retrieving Database Metadata

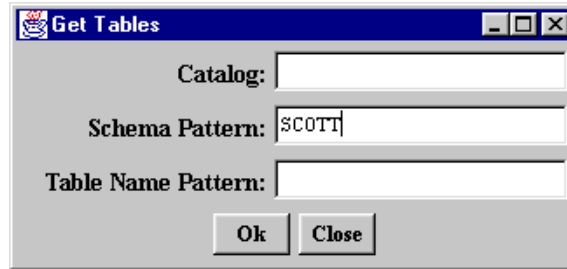
- 1 Select **Connection / Get DB Meta Data**.
- 2 Select **MetaData / Show Meta Data**. Information about the JDBC driver and the database you are connected to is returned.



- 3 Scroll through the Java code in the Java Code scroll box to find out which JDBC calls have been implemented by JDBCTest.

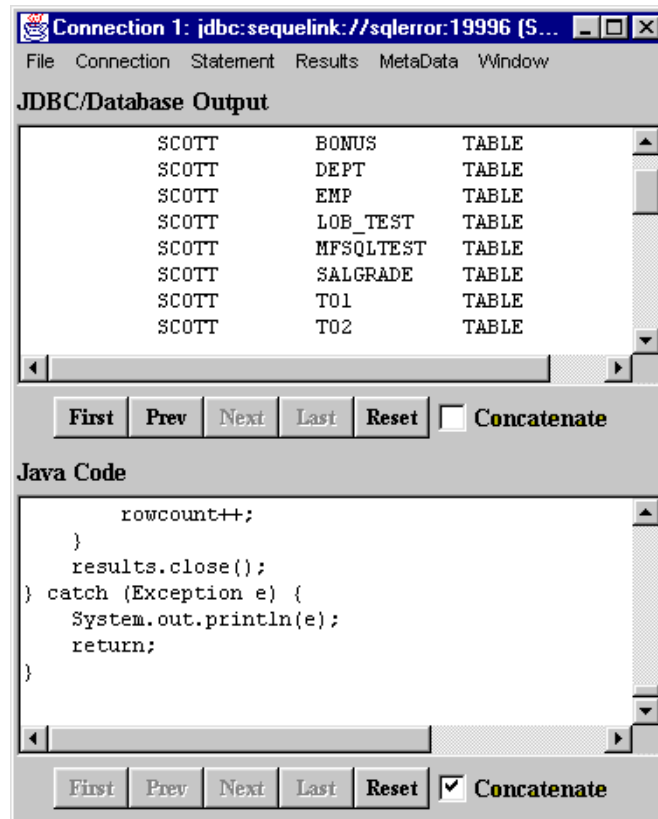
The metadata also allows you to query the database catalog (enumerate the tables in the database, for example). In this example, we will query all tables that are owned by the user *SCOTT*.

- 4 Select **MetaData / Tables**.
- 5 In the Schema Pattern field, type `SCOTT`.



Click **Ok**. The Connection window indicates that `getTables()` succeeded.

- 6 Select **Results / Show All Results**. All tables owned by *SCOTT* are returned.



Scrolling Through a Result Set

NOTE: Scrollable result sets are supported by JDBC 2.0 and require a Java 2 Platform (JDK 1.2)-compatible Java Virtual Machine.

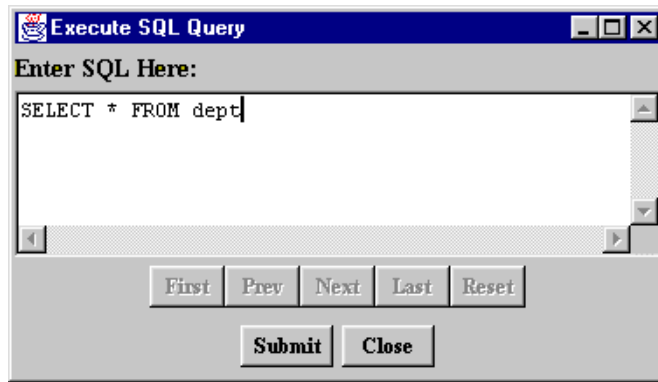
- 1 Select **Connection / Create JDBC 2.0 Statement**. JDBC Test prompts for a result set type and concurrency.
- 2 In the `resultSetType` field, select **TYPE_SCROLL_SENSITIVE**.
- 3 In the `resultSetConcurrency` field, select **CONCUR_READ_ONLY**.



Click **Submit**; then, click **Close**.

- 4 Select **Statement / Execute Stmt Query**.

- 5 Specify the Select statement you want to execute.

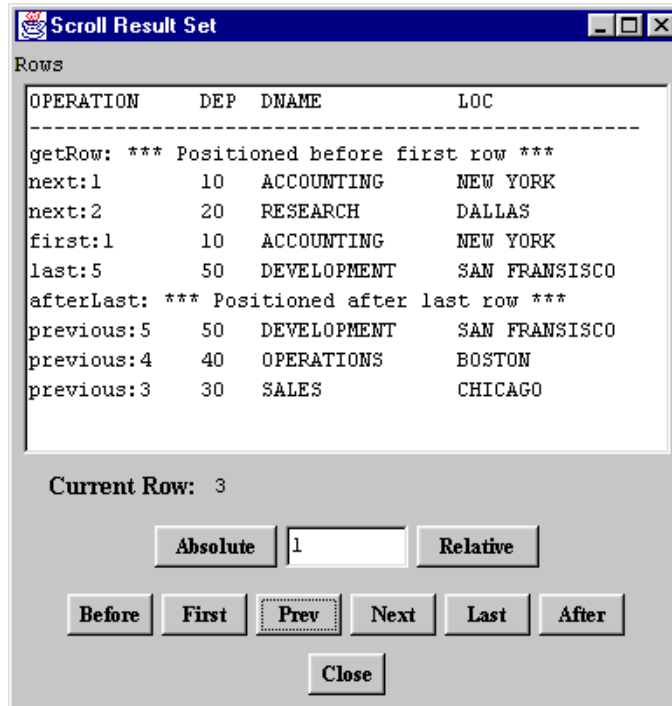


Click **Submit**; then, click **Close**.

- 6 Select **Results / Scroll Results**. The Scroll Result Set window indicates that the cursor is positioned before the first row.



- 7 Click the **Absolute**, **Relative**, **Before**, **First**, **Prev**, **Next**, **Last**, and **After** buttons as appropriate to navigate through the result set. After each action, the Scroll Result Set window displays the data at the current position of the cursor.



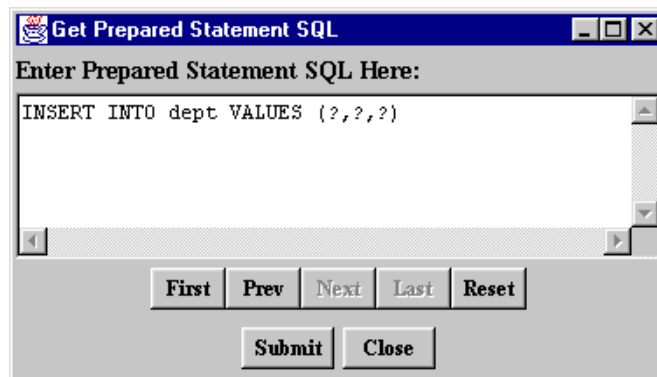
- 8 Click **Close**.

Batch Execution on a Prepared Statement

Batch execution on a prepared statement allows you to update or insert multiple records simultaneously. In some cases, this can significantly improve system performance, because fewer round-trips to the database are required.

NOTE: Batch execution on a prepared statement is supported by the JDBC 2.0 specification and requires a Java 2 Platform (JDK 1.2)-compatible Java Virtual Machine.

- 1 Select **Connection / Create Prepared Statement**.
- 2 Specify the Insert statement you want to execute.



Click **Submit**; then, click **Close**.

- 3 Select **Statement / Add Stmt Batch**.

- 4 For each parameter:
 - a Type the parameter number.
 - b Select the parameter type.
 - c Type the parameter value.
 - d Click **Set**.

#	Type	Value
1	short	60
2	String	MARKETING
3	String	NEW YORK

Parameter #:

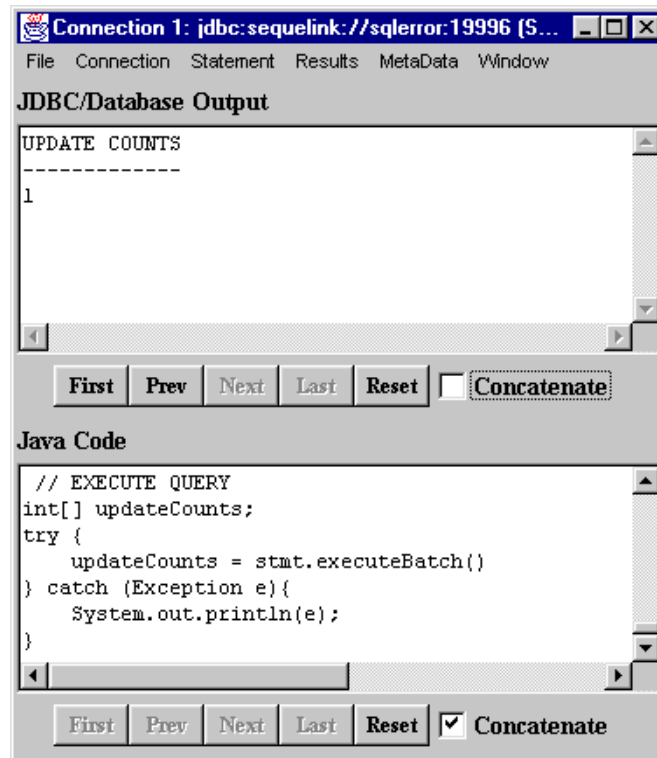
Parameter Type:

Use Calendar: Zone:

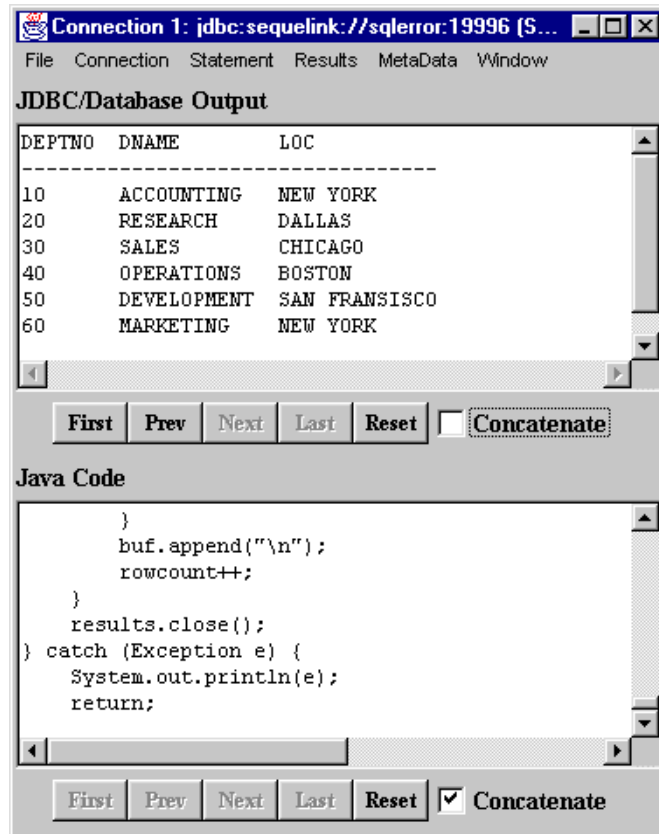
Parameter Value:

- 5 Click **Add** to add the specified set of parameters to the batch.
- 6 To add multiple parameter sets to the batch, repeat [Step 3](#) as many times as you need. When you are finished adding parameter sets to the batch, click **Close**.

- 7 Select **Statement / Execute Stmt Batch**. JDBCTest displays the rowcount for each of the elements in the batch.



- 8 If you re-execute the Select statement from “[Executing a Simple Select Statement](#)” on page 239, you will see that the previously inserted records are returned.



Using Stored Procedures With Oracle

SequeLink supports stored procedures with Oracle7 and Oracle8 databases. Also, with Oracle8, `getProcedures()` and `getProcedurecolumns()` can return information on procedures within PL/SQL packages, allowing JDBC applications to execute these procedures. This example shows you how to fetch rows using Oracle PL/SQL procedures.

Creating the Stored Procedure

In the following Oracle PL/SQL package, a record type and a cursor (result set) type are defined. The procedure contains an input parameter that can have a value, such as `Smi%`, to request information about employees whose last name starts with the letters 'Smi' (for example, Smith or Smithwick). The procedure also has one input/output parameter of the cursor type defined in the package.

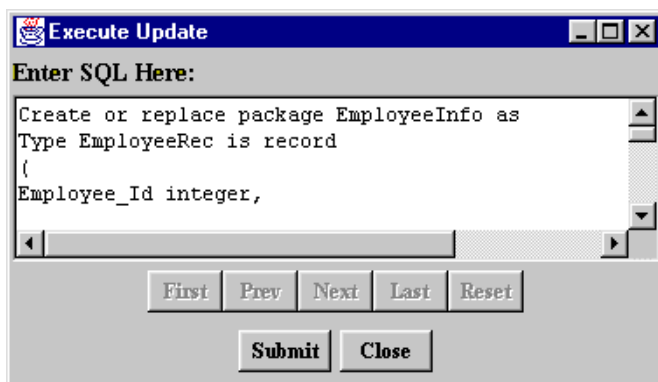
```

Create or replace package EmployeeInfo as
Type EmployeeRec is record
(
Employee_Id integer,
Employee_Name varchar2(25),
Employee_Job varchar2(25),
Department_Name varchar2(30),
Employee_Salary integer
);
Type EmployeeCursor is ref cursor return EmployeeRec;
End EmployeeInfo;
Create or replace procedure EmployeeInfoProc
(empname IN varchar2, empcursor IN OUT EmployeeInfo.EmployeeCursor)
As
Begin
Open empcursor For
select empno, ename, job, dname, sal from emp, dept
where emp.deptno=dept.deptno and ename like empname;
End;
```

To create the example stored procedure:

- 1 Select Connection / Create Statement.**
- 2 Select Statement / Execute Stmt Update.** JDBCTest prompts you for a SQL statement.

- 3 In the Enter SQL Here scroll box, type the name of the preceding PL/SQL package.



Click **Submit**; then, **Close**. The connection window indicates that the statement is executed successfully, and the PL/SQL package is created.

Executing the Stored Procedure

This example explains the JDBC function call sequence required to execute a stored procedure. We will request information for all employees starting with “M%”.

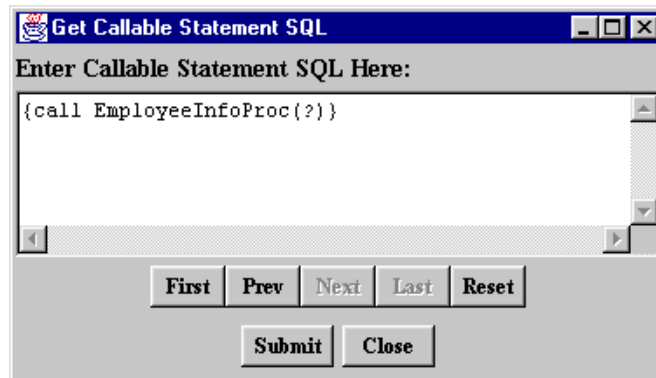
You might think that from the following procedure definition:

```
Create or replace procedure EmployeeInfoProc (empname IN varchar2, empcursor
IN OUT EmployeeInfo.EmployeeCursor)
```

having two parameters will require the application to call `setString()` twice. Actually, it does not. The only way to create a result set from an Oracle stored procedure is to declare this result set, *empcursor*, as an input/output parameter. So, the JDBC API treats this stored procedure as if it has only one input parameter, and returns a result set.

To execute the example stored procedure:

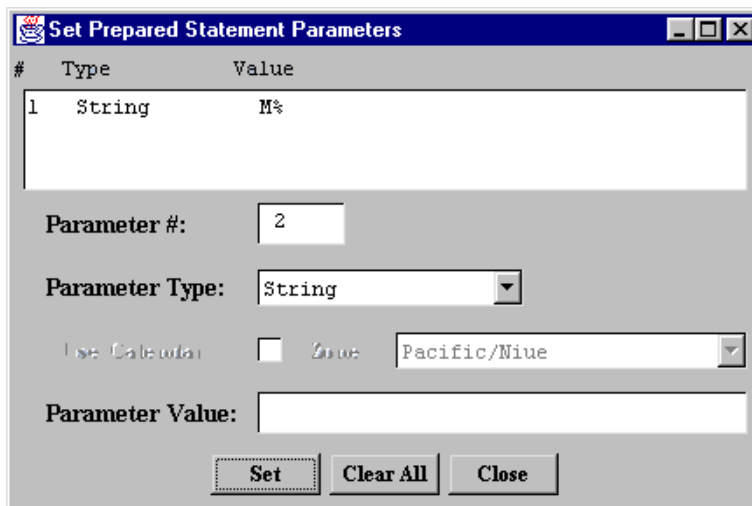
- 1 Select **Connection / Create Callable Statement**. JDBCTest prompts for the SQL statement to execute the stored procedure.
- 2 Using JDBC SQL syntax, specify the stored procedure you want to execute.



Click **Submit**; then, click **Close**.

- 3 Select **Statement / Set Prepared Parameters**.

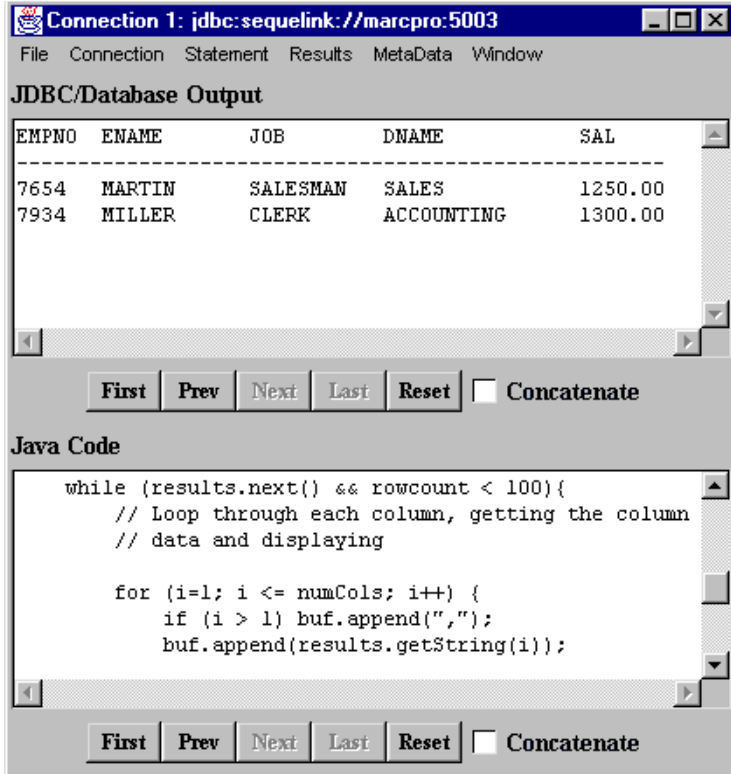
- 4 For each parameter:
 - a Type the parameter number.
 - b Select the parameter type.
 - c Type the parameter value.
 - d Click **Set**.



Click **Close**.

- 5 Select **Statement / Execute Stmt Query**. The connection window indicates that the stored procedure has been executed successfully.

6 Select Results / Show All Results.



Connection 1: jdbc:sequelink://marcpro:5003

File Connection Statement Results MetaData Window

JDBC/Database Output

EMPNO	ENAME	JOB	DNAME	SAL
7654	MARTIN	SALESMAN	SALES	1250.00
7934	MILLER	CLERK	ACCOUNTING	1300.00

First Prev Next Last Reset Concatenate

Java Code

```
while (results.next() && rowcount < 100){
    // Loop through each column, getting the column
    // data and displaying

    for (i=1; i <= numCols; i++) {
        if (i > 1) buf.append(",");
        buf.append(results.getString(i));
    }
}
```

First Prev Next Last Reset Concatenate

7 Tracking JDBC Calls

This chapter introduces Spy, a tool that allows you to track JDBC calls, and describes how to use it.

About Spy

Spy is a tracking tool for JDBC calls. Spy passes calls issued by an application to an underlying JDBC driver and logs detailed information about those calls. It provides the following advantages:

- Logging is JDBC 1.22- and JDBC 2.0-compliant, including support for the JDBC 2.0 Optional Package
- Logging is consistent, regardless of the JDBC driver used.
- All parameters and function results for JDBC calls can be logged.
- Even if your JDBC driver does not support logging, you can still log JDBC calls.
- Logging can be enabled, without changing the application, using the `SequeLinkDataSource` object.

Loading the Spy JDBC Driver

To use the Spy JDBC Driver, you first must register it with the JDBC Driver Manager. You can register the Spy JDBC Driver in any of the following ways:

- **Method 1:** Set the Java property `sql.drivers` using the Java `-D` option. The `sql.drivers` property is defined as a colon-separated list of driver class names. For example:

```
com.merant.jdbcspy.SpyDriver:  
sun.jdbc.odbc.JdbcOdbcDriver
```

The `sql.drivers` property can be set like other Java properties, using the `-D` option. For example:

```
java -Dsql.drivers=com.merant.jdbcspy.SpyDriver
```

- **Method 2:** Set the Java property `sql.drivers` from within your Java application or applet. To do this, code the following lines in your Java application or applet, and call `DriverManager.getConnection()`:

```
Properties p = System.getProperties();  
p.put ("sql.drivers", "com.merant.jdbcspy.SpyDriver");  
System.setProperties (p);
```

- **Method 3:** Explicitly load the driver class using the standard `Class.forName()` method. To do this, code the following line and call `DriverManager.getConnection()`:

```
Class.forName ("com.merant.jdbcspy.SpyDriver");
```

Spy URL Syntax and Spy Attributes

Spy uses the following format as a connection URL:

```
jdbc:spy:{original-url}; [key=value] . . .
```

where *original-url* is the connection URL of the underlying JDBC driver.

In addition, you can specify the following options:

<code>log=System.out</code>	Redirects logging to the Java output standard.
<code>log=(file)<i>filename</i></code>	Redirects logging to the file specified by <i>filename</i> . By default, Spy will use the stream specified in <code>DriverManager.setLogStream()</code> .
<code>load=<i>classname</i></code>	Loads the driver specified by <i>classname</i> . The default value is <code>com.merant.sequelink.jdbc.SequelinkDriver</code> .
<code>linelimit=<i>numberofchars</i></code>	The maximum number of characters, specified by <i>numberofchars</i> , that Spy will log on one line. The default is 0 (no maximum limit).
<code>logIS={yes no nosingleread}</code>	Specifies whether Spy logs activity on <code>InputStreams</code> . <code>nosingleread</code> =turns on logging for <code>InputStreams</code> (but not <code>InputStream.read()</code> messages), allowing logging on <code>InputStreams</code> without generating large log files full of single-byte read messages. The default is no.

logTName={yes | no}

Specifies whether Spy logs the name of the current thread. The default is no.

Using Spy with JDBC Data Sources

The SequeLink JDBC Driver implements the following features defined by the JDBC 2.0 Optional Package:

- JNDI for Naming Databases
- Connection Pooling
- Distributed Transaction Management (DTC)

You can use Spy to track JDBC calls with all of these features. The `com.merant.sequelink.jdbcx.datasource.SequeLinkDataSource` class supports the `SpyAttributes` connection attribute, which specifies a semi-colon-separated list of Spy attributes as described in [“Spy URL Syntax and Spy Attributes” on page 263](#). For more information about configuring JDBC data sources, see [“JDBC Connection URL Examples:” on page 216](#).

The following examples create a `SequeLinkDataSource` and specifies that all JDBC calls must be logged in the file `/tmp/spy.log`, including the name of the current thread.

```
...
SequeLinkDataSource sds=new SequeLinkDataSource();
sds.setServerName("MyServer");
sds.setPortNumber(1234);
sds.setSpyAttributes("log=(file)/tmp/spy.log;logTName=
yes");
Connection conn=sds.getConnection("scott","tiger");
...
```

Spy URL Examples

Example A:

```
jdbc:spy:{jdbc:odbc:Oracle7};load=sun.jdbc.odbc.JdbcOdbcDriver;  
log=(file)C:\temp\spy.log
```

Using this example, Spy would:

- 1 Load the JDBC-ODBC bridge.
- 2 Log all JDBC activity to the file c:\temp\spy.log.

Example B:

```
jdbc:spy:{jdbc:sequelink://...};log=System.out;linelimit=80
```

Using this example, Spy would:

- 1 Load the SequeLink JDBC Driver.
- 2 Log all JDBC activity to the standard output file.
- 3 Log a maximum of 80 characters for each line.

Spy Log Example

NOTE: Numbers in bold superscript are note indicators. See the notes following the example for the referenced text.

```
All rights reserved.1
registerDriver:driver[className=com.merant.jdbcspy.SpyDriver,
context=null,com.merant.jdbcspy.SpyDriver@1ec49f]2

*Driver.connect(jdbc:spy:{jdbc:sequelink://QANT:4003;databaseName=Oracle;})
    trying driver[className=com.merant.jdbcspy.SpyDriver,
context=null,com.merant.jdbcspy.SpyDriver@1ec49f]3

spy>> Driver.connect(String url, Properties info)
spy>> url = jdbc:spy:{jdbc:sequelink://QANT:4003;databaseName=Oracle;
OSUser=qauser;OSPassword=null12}
spy>> info = {password=tiger, user=scott}
spy>> OK (Connection[1])4

getConnection returning driver[className=com.merant.jdbcspy.SpyDriver,
context=null,com.merant.jdbcspy.SpyDriver@1ec49f]5

spy>> Connection[1].getWarnings()
spy>> OK6

spy>> Connection[1].createStatement
spy>> OK (Statement[1])7

spy>> Statement[1].executeQuery(String sql)
spy>> sql = select empno,ename,job from emp where empno=7369
spy>> OK (ResultSet[1])8

spy>> ResultSet[1].getMetaData()
spy>> OK (ResultSetMetaData[1])9

spy>> ResultSetMetaData[1].getColumnCount()
spy>> OK (3)10
```

```
spy>> ResultSetMetaData[1].getColumnLabel(int column)
spy>> column = 1
spy>> OK (EMPNO)11

spy>> ResultSetMetaData[1].getColumnLabel(int column)
spy>> column = 2
spy>> OK (ENAME)12

spy>> ResultSetMetaData[1].getColumnLabel(int column)
spy>> column = 3
spy>> OK (JOB)13

spy>> ResultSet[1].next()
spy>> OK (true)14

spy>> ResultSet[1].getString(int columnIndex)
spy>> columnIndex = 1
spy>> OK (7369)15

spy>> ResultSet[1].getString(int columnIndex)
spy>> columnIndex = 2
spy>> OK (SMITH)16

spy>> ResultSet[1].getString(int columnIndex)
spy>> columnIndex = 3
spy>> OK (CLERK)17

spy>> ResultSet[1].next()
spy>> OK (false)18

spy>> ResultSet[1].close()
spy>> OK19

spy>> Connection[1].close()
spy>> OK20
```

NOTES:

1: The Spy driver is registered. The `spy>>` prefix indicates that this line has been logged by Spy.

2: The JDBC Driver Manager logs a message each time a JDBC driver is registered.

3: This is the logging of the JDBC Driver Manager. It logs a message each time a JDBC application makes a connection.

4: The application connects with the specified URL. The User Name and Password are specified using properties.

5: This is the logging of the JDBC Driver Manager. It logs a message each time a successful connection is made.

6: The application checks to see if there are any warnings. In this example, no warnings are present.

7 and 8: The statement `"select empno,ename,job from emp where empno=7369"` is created.

9, 10, 11, 12, and 13: Some metadata is requested.

14, 15, 16, and 17: The first row is fetched.

18: The application attempts to fetch the second row, but the database returned only one row for this query.

19: After fetching all data, the result set is closed.

20: The application finishes and disconnects.

8 Developing JDBC Applications

This chapter provides information about developing JDBC applications for SequeLink environments including:

- ["JDBC 1.22 Support" on page 270](#)
- ["JDBC 2.0 Support" on page 271](#)
- ["JDBC 2.0 Optional Package Support" on page 273](#)
- ["JDBC Compatibility" on page 274](#)
- ["SQL Escape Sequences" on page 274](#)
- ["Data Types and Isolation Levels" on page 275](#)
- ["Threading" on page 275](#)
- ["Using Scrollable Cursors" on page 277](#)
- ["Specifying Application IDs" on page 280](#)
- ["Error Handling" on page 281](#)
- ["Fine-Tuning JDBC Application Performance" on page 283](#)

JDBC 1.22 Support

[Table 8-1](#) lists the JDBC 1.22 functionality supported by the SequeLink JDBC Driver.

Table 8-1. JDBC 1.22 Functionality Supported by the SequeLink JDBC Driver

JDBC Functionality	Description
java.sql.CallableStatement	Fully implemented
java.sql.Connection	Fully implemented
java.sql.DatabaseMetaData	Fully implemented
java.sql.Driver	Fully implemented by class com.merant.sequelink.jdbc.SequeLinkDriver
java.sql.PreparedStatement	Fully implemented
java.sql.ResultSet	Fully implemented
java.sql.ResultSetMetaData	Fully implemented
java.sql.Statement	Fully implemented

JDBC 2.0 Support

[Table 8-2](#) lists the JDBC 2.0 functionality supported by the SequeLink JDBC Driver. For information about the JDBC 2.0 Optional Package functionality supported by the SequeLink JDBC Driver, see [“JDBC 2.0 Optional Package Support”](#) on [page 273](#).

Table 8-2. JDBC 2.0 Functionality Supported by the SequeLink JDBC Driver

JDBC Functionality	Description
java.sql.CallableStatement	<ul style="list-style-type: none"> ■ getBigDecimal() returning a BigDecimal with full precision ■ Calendar extensions for getDate(), getTime(), and getTimeStamp
java.sql.Connection	<ul style="list-style-type: none"> ■ createStatement() with result set type and concurrency (see note) ■ prepareCall() with result set type and concurrency (see note) ■ prepareStatement() with result set type and concurrency (see note)
java.sql.DatabaseMetaData	Fully implemented
java.sql.Driver	Fully implemented
java.sql.PreparedStatement	<ul style="list-style-type: none"> ■ Batches ■ setCharacterStream() ■ Calendar extensions for setDate(), setTime(), and setTimeStamp()

NOTE: For more information about scrollable cursors, see [“Specifying Application IDs”](#) on [page 280](#).

Table 8-2. JDBC 2.0 Functionality Supported by the SequeLink JDBC Driver (cont.)

JDBC Functionality	Description
java.sql.ResultSet	<ul style="list-style-type: none">■ Scrolling through result sets (see note)■ <code>getBigDecimal()</code> returning a <code>BigDecimal</code> with full precision■ <code>getCharacterstream()</code>■ <code>getStatement()</code>■ Calendar extensions for <code>getDate()</code>, <code>getTime()</code>, and <code>getTimeStamp()</code>
java.sql.ResultSetMetaData	Fully implemented
java.sql.Statement	Fully implemented

NOTE: For more information about scrollable cursors, see [“Specifying Application IDs” on page 280](#).

JDBC 2.0 Optional Package Support

[Table 8-3](#) lists the JDBC 2.0 Optional Package functionality supported by the SequeLink JDBC Driver.

Table 8-3. JDBC 2.0 Optional Package Functionality Supported by the SequeLink JDBC Driver

JDBC Functionality	Support
javax.sql.ConnectionEventListener	Fully implemented
javax.sql.ConnectionPoolDataSource	Fully implemented
javax.sql.DataSource	Fully implemented
javax.sql.PooledConnection	Fully implemented
javax.sql.XAConnection	Fully implemented
javax.sql.XADataSource	Fully implemented
javax.sql.xa.XAResource	Fully implemented

JDBC Compatibility

[Table 8-4](#) shows compatibility between the JDBC application versions, Java Virtual Machines, and the SequeLink JDBC Driver.

Table 8-4. JDBC Compatibility

JDBC Version Used (see note)	Java Virtual Machine	Compatible?	Comments
1.22	1.0.2	No	The SequeLink JDBC Driver does not support JDK 1.0.2.
1.22	1.1.x	Yes	
1.22	1.2	Yes	
2.0	1.0.2	No	The SequeLink JDBC Driver does not support JDK 1.0.2.
2.0	1.1.x	No	A JDBC 2.0 application requires the JDBC 2.0 interface.
2.0	1.2	Yes	

NOTE: Is the application using JDBC 1.22 or JDBC 2.0 features?

SQL Escape Sequences

See [Appendix A “SQL Escape Sequences for ODBC and JDBC” on page 289](#) for information about the SQL escape sequences supported by the SequeLink JDBC Driver.

Data Types and Isolation Levels

The data types and isolation levels supported by the SequeLink JDBC Driver depend on the data store to which you are connecting. See [Appendix B “Data Types and Isolation Levels” on page 309](#) for database-specific information about data types and isolation levels.

Threading

The SequeLink JDBC Driver is completely thread safe; that is, it will not fail when database requests are made on separate threads.

Threading Architecture

A JDBC driver can be based on one of the following architectures:

- *Thread impaired.* The JDBC driver serializes all JDBC calls. All requests are handled one by one, without concurrent processing.
- *Thread per connection.* The JDBC driver processes requests concurrently with statements that do not share the same connection; however requests on the same connection are serialized. The SequeLink JDBC Driver uses this architecture.
- *Fully threaded.* All requests use the threaded model. The JDBC driver processes all requests on multiple statements concurrently.

Cancelling Functions in Multithreaded Applications

In a multithreaded application, a thread can use the `cancel` method to cancel a statement that is being executed by another thread. Whether the `cancel` method actually cancels the statement depends on the data store being accessed as shown in [Table 8-5](#). In [Table 8-5](#):

- *OK* means that `cancel` can interrupt the running statement.
- *Ignored* means that `cancel` will have no effect on the running statement.

In both cases, the `cancel` method will return `SQL_SUCCESS`. If the `cancel` method has been called from a different thread while there is a pending request, the original statement will return `SQL_ERROR` with the error message `Operation cancelled`.

Table 8-5. Using Cancel in Multithreaded Applications

Data Store	SQLCancel
DB2 V4, V5, V6R1 on OS/390	Ignored
DB2 V6R1, V7R1	Ignored
Informix 7, 9	Ignored
Microsoft SQL Server 6.5, 7.0, 2000	OK
Oracle7 on Windows NT and Windows 2000	Ignored
Oracle7 on UNIX	OK
Oracle8 on Windows NT and Windows 2000	Ignored
Oracle8 on UNIX	OK
Sybase 11, 12	Ignored

NOTE: Cancel functionality is not supported when the connection uses Secure Socket Layer (SSL) encryption.

Using Scrollable Cursors

Scrollable cursors can move backward and forward in a result set, allowing the application to scroll back and forth through retrieved data.

Result Set Types

JDBC 2.0 defines the following result set types:

- Forward-only
- Scroll-insensitive
- Scroll-sensitive

Forward-only result sets allow you to move forward, but not backward, through the data. The application only can move forward using the `next()` method.

Typically, a *scroll-insensitive result set* ignores changes that are made while it is open. It provides a static view of the underlying data it contains. The membership, order, and column values of rows are fixed when the result set is created.

In contrast, a *scroll-sensitive result set* provides a dynamic view of the underlying data, reflecting changes that are made while it is open. The membership and ordering of rows in the result set may be fixed, depending on how it is implemented.

The type of result sets that can be used depend on the data store to which you are connecting. [Table 8-6](#) shows the type of result sets supported for each database.

Table 8-6. Support for Scrollable Cursors (JDBC)

Database	Scroll-Insensitive	Scroll-Sensitive
DB2 on OS/390	✓	
Informix	✓	✓
Microsoft SQL Server (see note)	✓	✓
Oracle	✓	✓
Sybase (see note)	✓	✓

NOTE: To use scroll-sensitive cursors with Microsoft SQL Server and Sybase, the table must contain an identity column.

Concurrency Types

JDBC 2.0 defines the following concurrency types for a result set:

- Read-only
- Updatable

A *read-only* result set does not allow its contents to be updated. Read-only result sets can increase the overall level of concurrency between transactions, because multiple read-only locks can be held on a data item simultaneously.

An *updatable* result set allows its contents to be updated and may use database write locks to mediate access to the same data item by different transactions. Because only a single write lock may be held at one time on a data item, updatable result sets can reduce concurrency.

An optimistic concurrency control scheme may be appropriate if you can accurately predict that conflicting access to data will seldom occur. Typically, optimistic concurrency control implementations compare rows by a value or by a version number to determine if an update conflict has occurred.

Using Scrollable Cursors

- The SequeLink JDBC Driver supports forward-only and scroll-insensitive result sets against all data stores.
- Scroll-sensitive result sets on stored procedures or explicit batches are not supported.
- Scroll-sensitive result sets are not supported when the Select statement contains any of the following SQL language constructions:
 - JOIN
 - Aggregate functions
 - GROUP BY
- The SequeLink JDBC Driver does not support updatable result sets.

NOTE: When the SequeLink JDBC Driver cannot support the requested result set type or concurrency, it will automatically downgrade it and generate one or multiple SQLWarnings with detailed information.

Specifying Application IDs

Application IDs are alphanumeric strings passed by a SequeLink Client that identify the client application to a SequeLink service that has been configured to accept connections only from specific application IDs.

After establishing a connection with the SequeLink JDBC Driver, immediately invoke `setApplicationId`. The `setApplicationId` method is defined on the interface `com.merant.SlExtensionInterface`, and uses the following method prototype:

```
public void setApplicationId(String s) throws SQLException
```

You can set the application ID as shown in the following example:

```
import java.sql.*;
import com.merant.SlExtensionInterface;

...
Connection con = DriverManager.getConnection(...);

String appId = "myAppID";
if (con instanceof SlExtensionInterface)
{
    SlExtensionInterface slCon = (SlExtensionInterface)con;
    slCon.setApplicationId(myAppID);
}
```

where *myAppID* is the application ID.

For more information about configuring SequeLink services to accept connections only from specific application IDs, refer to the *SequeLink Administrator's Guide*.

Error Handling

The SequeLink JDBC Driver reports errors to the calling application by returning `SQLExceptions`. Errors can be generated by the following components:

- SequeLink JDBC Driver
- SequeLink Server
- Database

SequeLink JDBC Driver Errors

An error generated by the SequeLink JDBC Driver has the following format:

```
[MERANT] [SequeLink JDBC Driver] message
```

For example:

```
[MERANT] [SequeLink JDBC Driver] Timeout expired.
```

Use the native error code to look up details about the possible cause of the error. For a list of all error codes and messages, refer to the *SequeLink Troubleshooting Guide and Reference*. Sometimes, you may need to check the last JDBC call your application made and refer to the JDBC specification for recommended action.

SequeLink Server Errors

An error generated by SequeLink Server has the following format:

```
[MERANT] [SequeLink JDBC Driver] [SequeLink Server] message
```

For example:

```
[MERANT] [SequeLink JDBC Driver] [SequeLink Server] Only  
Select statements are allowed in this read-only connection.
```

Use the native error code to look up details about the possible cause of the error. For a list of all error codes and messages, refer to the *SequeLink Troubleshooting Guide and Reference*.

Database Errors

An error generated by the database has the following format:

```
[MERANT] [SequeLink JDBC Driver] [...] message
```

For example:

```
[MERANT] [SequeLink JDBC Driver] [Oracle] ORA-00942:table  
or view does not exist.
```

Use the native error code to look up details about the possible cause of the error. For these details, refer to your database documentation.

Fine-Tuning JDBC Application Performance

This section provides some tips for fine-tuning the performance of your JDBC applications.

Reducing Download Time

Generally, the time that it takes for applets to download is determined by the following factors:

- *Number of classes that are loaded.* Each class that is downloaded results in an HTTP request to your Web server. The more requests and transfers that are made, the slower the download.
- *Size of the byte code that is loaded.* The more bytes that are transferred, the slower the download.

JDK 1.1-compatible Java Virtual Machines support JAR files, which reduces the number of HTTP requests because all the class files are packaged together in the JAR file. The JAR format also allows you to compress the packaged files, which further optimizes the download.

To reduce download time by using JAR files:

- 1 Package all classes of your applet into a JAR file.
- 2 Copy the JAR file into the directory indicated by the codebase tag.
- 3 Specify the JAR file in the archive tag.

For example:

```
<html>
<applet
width=100 height=100
code=JDBCTestApplet
```

```

codebase=.
archive=myapplet.jar>
<param name=ConfigFile value=Config.txt>
</applet>

```

The SequeLink JDBC Driver is packaged into the following JAR files:

- `sljc.jar` contains all classes of the SequeLink JDBC Driver.
- `sljcx.jar` contains all classes of the SequeLink JDBC Driver implementing the JDBC 2.0 Optional Package.
- `slssl.jar` or `slssl_11.jar` contains all classes of the SequeLink JDBC Driver implementation of SSL. This file is only required if you will be using SSL encryption.
- `slrsa_rc4.jar` or `slrsa_rc4_11.jar` contains all classes of the SequeLink JDBC Driver implementing the RSA and RC4 algorithms.

NOTE: These files are not installed by the SequeLink Java installer. You must download them from the MERANT web site and copy them as described in the *SequeLink Administrator's Guide*.

To use the SequeLink JDBC Driver from within your applet, specify these JAR files in the archive tag as shown:

```

<html>
<applet
width=100 height=100
code=JDBCTestApplet
codebase=.
archive=myapplet.jar,sljc.jar,slssl_11.jar>
<param name=ConfigFile value=Config.txt>
</applet>

```

By default, the JAR files shipped with SequeLink Java Client are uncompressed. If download speed is crucial in your environment, you may want to repackage the SequeLink JDBC Driver classes into a compressed JAR file.

NOTE: You may experience problems using compressed JAR files with early versions of Microsoft Internet Explorer 4.x and Netscape Communicator 4.x.

Improving Character Set Conversion Performance

Java uses Unicode as its native character encoding method; however, most databases use a different character encoding method. Transliteration is defined by the SequeLink service using the `ServiceCodePage` attribute. The standard character set conversion implementation is 100% Pure Java; however, system performance may not be optimal.

If you are using transliteration and want to improve your driver performance, replace the `VMTransliterator.class` file with the `VMTransliterator.java` file. Both files are located in `install_dir/driver/src/com/merant/sequelink/ssp/translit/` where `install_dir` is the SequeLink Java Client installation directory.

NOTE: `Transliteration.class` uses classes from the `sun.io` package provided by Sun Microsystems. Because this transliteration class is not 100% Pure Java, it is not bundled with the default implementation of the SequeLink JDBC Driver. By using classes from the `sun.io` package, character set conversion performance can improve significantly; however, Sun Microsystems reserves the right to remove or change the `sun.io` classes in future releases of the Java Virtual Machine, which could result in compatibility problems with this character set conversion implementation.

Fetching BigDecimal Objects

JDBC 1.22 defines `getBigDecimal()` with a scale parameter. When the SequeLink JDBC Driver fetches a `BigDecimal` object from a database, it rescales it using the scale specified by the application. This additional processing can downgrade system performance, particularly when large numbers of `BigDecimal` objects are fetched by your application.

To eliminate this additional rescaling, JDBC 2.0 defines an overloaded version of `getBigDecimal`, without the scale parameter. This method allows the SequeLink JDBC Driver to return the `BigDecimal` object with the original precision.

Part 4: Reference

This part contains the following appendixes:

- [Appendix A “SQL Escape Sequences for ODBC and JDBC” on page 289](#) describes the scalar functions supported for SequeLink. Your data store may not support all these functions.
- [Appendix B “Data Types and Isolation Levels” on page 309](#) lists the data types and isolation levels supported for each data store supported by SequeLink.

A SQL Escape Sequences for ODBC and JDBC

A number of language features, such as outer joins and scalar function calls, are commonly implemented by DBMSs. The syntax for these features is often DBMS-specific, even when a standard syntax has been defined. ODBC and JDBC define escape sequences that contain standard syntaxes for the following language features:

- Date, time, timestamp, and datetime interval literals
- Scalar functions such as numeric, string, and data type conversion functions
- LIKE predicate escape characters
- Outer joins
- Procedure calls

The escape sequence used by ODBC and JDBC is:

```
{extension}
```

The escape sequence is recognized and parsed by the SequeLink ODBC Driver or SequeLink JDBC Driver, which replaces the escape sequences with data store-specific grammar.

Date, Time, and Timestamp Escape Sequences

The escape sequence for date, time, and timestamp literals is:

```
{literal-type 'value'}
```

where *literal-type* is one of the following:

literal-type	Description	Value Format
d	Date	yyy-mm-dd
t	Time	hh:mm:ss [1]
ts	Timestamp	yyyy-mm-dd hh:mm:ss [.f...]

Example:

```
UPDATE Orders SET OpenDate={d '1995-01-15'}
WHERE OrderID=1023
```

Scalar Functions

You can use scalar functions in SQL statements with the following syntax:

```
{fn scalar-function}
```

where *scalar-function* is a scalar function supported by the SequeLink ODBC Driver and the SequeLink JDBC Driver, as shown in [Table A-1 “Scalar Functions Supported by the SequeLink ODBC Driver and SequeLink JDBC Driver” on page 291](#).

Example:

```
SELECT {fn UCASE(NAME)} FROM EMP
```

Table A-1. Scalar Functions Supported by the SequeLink ODBC Driver and SequeLink JDBC Driver

Data Store	String Functions	Numeric Functions	Timedate Functions	System Functions
DB2 V4 on OS/390	CHAR_LENGTH CHARACTER_LENGTH CONCAT LEFT LENGTH RIGHT SUBSTRING	Not supported	CURDATE CURRENT_DATE CURTIME DAYOFMONTH HOUR MINUTE MONTH NOW QUARTER SECOND YEAR	IFNULL USERNAME
DB2 V5 on OS/390	BOTH LEADING SUBSTRING TRAILING	Not supported	CURDATE CURRENT_DATE CURTIME DAYOFMONTH HOUR MINUTE MONTH NOW QUARTER SECOND YEAR	IFNULL USERNAME
DB2 V4, V5 on Windows and UNIX	CONCAT LEFT LENGTH RIGHT SUBSTRING	Not supported	CURDATE CURRENT DATE CURTIME DAYOFMONTH HOUR MINUTE MONTH NOW QUARTER SECOND YEAR	IFNULL USERNAME

Table A-1. Scalar Functions Supported by the SequeLink ODBC Driver and SequeLink JDBC Driver (cont.)

Data Store	String Functions	Numeric Functions	Timedate Functions	System Functions
DB2 V6 on OS/390	CHAR_LENGTH	ABS	CURDATE	IFNULL
	CHARACTER_LENGTH	ACOS	CURRENT_DATE	USERNAME
	CONCAT	ASIN	CURTIME	
	INSERT	ATAN	DAYOFMONTH	
	LCASE	ATAN2	DAYOFWEEK	
	LEFT	CEILING	DAYOFYEAR	
	LENGTH	COS	HOUR	
	LOCATE	DEGREES	MINUTE	
	LOCATE_2	EXP	MONTH	
	LTRIM	FLOOR	NOW	
	POSITION	LOG	QUARTER	
	REPEAT	LOG10	SECOND	
	REPLACE	MOD	WEEK	
	RIGHT	PI	YEAR	
	RTRIM	POWER		
	SPACE	RADIANS		
	SUBSTRING	RAND		
	UCASE	ROUND		
		SIGN		
		SIN		
	SQRT			
	TAN			
	TRUNCATE			

Table A-1. Scalar Functions Supported by the SequeLink ODBC Driver and SequeLink JDBC Driver (cont.)

Data Store	String Functions	Numeric Functions	Timedate Functions	System Functions
DB2 V6R1, V7R1 on Windows NT, Windows 2000, and UNIX	ASCII	ABS	CURDATE	DBNAME
	CHAR	ACOS	CURTIME	IFNULL
	DIFFERENCE	ASIN	DAYNAME	USERNAME
	CONCAT	ATAN	DAYOFMONTH	
	INSERT	ATAN2	DAYOFWEEK	
	LCASE	CEILING	DAYOFYEAR	
	LEFT	COS	HOUR	
	LENGTH	COT	MINUTE	
	LOCATE	DEGREES	MONTH	
	LOCATE_2	EXP	MONTHNAME	
	LTRIM	FLOOR	NOW	
	REPEAT	LOG	QUARTER	
	REPLACE	LOG10	SECOND	
	RIGHT	MOD	TIMESTAMPADD	
	RTRIM	PI	TIMESTAMPDIFF	
	SOUNDEX	POWER	WEEK	
	SPACE	RADIANS	YEAR	
	SUBSTRING	RAND		
	UCASE	ROUND		
		SIGN		
	SIN			
	SQRT			
	TAN			
	TRUNCATE			

Table A-1. Scalar Functions Supported by the SequeLink ODBC Driver and SequeLink JDBC Driver (cont.)

Data Store	String Functions	Numeric Functions	Timedate Functions	System Functions
Informix	BIT_LENGTH CHAR_LENGTH CONCAT LENGTH LTRIM RTRIM STR_LENGTH	ABS ACOS ASIN ATAN ATAN2 COS COT EXP FLOOR LOG LOG10 MOD POWER ROUND SIGN SIN SQRT TAN TRUNCATE	CURDATE CURRENT DATE CURTIME DAYOFMONTH DAYOFWEEK MONTH NOW QUARTER YEAR	DBNAME USERNAME

Table A-1. Scalar Functions Supported by the SequeLink ODBC Driver and SequeLink JDBC Driver (cont.)

Data Store	String Functions	Numeric Functions	Timedate Functions	System Functions
Microsoft SQL Server	ASCII BITLENGTH CHAR CONCAT DIFFERENCE INSERT LCASE LEFT LENGTH LOCATE LOCATE2 LTRIM OCTET_LENGTH REPEAT REPLACE RIGHT RTRIM SOUNDEX SPACE SUBSTRING UCASE	ABS ACOS ASIN ATAN ATAN2 CEILING COS COT DEGREES EXP FLOOR LOG LOG10 MOD PI POWER RADIANS RAND ROUND SIGN SIN SQRT TAN TRUNCATE	CURDATE CURRENT DATE CURRENT TIME CURRENT TIMESTAMP CURTIME DAYOFMONTH DAYOFWEEK DAYOFYEAR DAYNAME EXTRACT HOUR MINUTE MONTH MONTHNAME NOW QUARTER SECOND TIMESTAMPADD TIMESTAMPDIFF WEEK YEAR	DBNAME IFNULL USERNAME

Table A-1. Scalar Functions Supported by the SequeLink ODBC Driver and SequeLink JDBC Driver (cont.)

Data Store	String Functions	Numeric Functions	Timedate Functions	System Functions
Oracle	ASCII BIT_LENGTH CHAR CONCAT INSERT LCASE LEFT LENGTH LOCATE LOCATE2 LTRIM OCTET_LENGTH REPEAT REPLACE RIGHT RTRIM SOUNDEX SPACE SUBSTRING UCASE	ABS CEILING COS COT EXP FLOOR LOG LOG10 MOD POWER ROUND SIGN SIN SQRT TAN TRUNCATE	CURDATE CURRENT DATE DAYOFMONTH DAYOFWEEK DAYOFYEAR DAYNAME HOUR MINUTE MONTH MONTHNAME NOW QUARTER SECOND TIMESTAMPADD TIMESTAMPDIFF WEEK YEAR	IFNULL USERNAME

Table A-1. Scalar Functions Supported by the SequeLink ODBC Driver and SequeLink JDBC Driver (cont.)

Data Store	String Functions	Numeric Functions	Timedate Functions	System Functions
Sybase	ASCII CHAR CONCAT DIFFERENCE LCASE LEFT LENGTH LTRIM REPEAT RIGHT RTRIM SOUNDEX SPACE SUBSTRING UCASE	ABS ACOS ASIN ATAN ATAN2 CEILING COS COT EXP FLOOR LOG LOG10 MOD NUM_DEGREES NUM_RADIANS PI POWER RAND ROUND SIGN SIN SQRT TAN TRUNCATE	CURDATE CURRENT DATE DAYOFMONTH DAYOFWEEK DAYOFYEAR DAYNAME HOUR MINUTE MONTH MONTHNAME NOW QUARTER SECOND TIMESTAMPADD TIMESTAMPDIFF WEEK YEAR	DBNAME IFNULL USERNAME

String Functions

Table A-2 lists string functions. The following arguments can be used with these functions:

- *string_exp* can be a column name, a string literal, or the result of another scalar function, where the underlying data type is SQL_CHAR, SQL_VARCHAR, or SQL_LONGVARCHAR.
- *start*, *length*, and *count* can be the result of another scalar function or a literal numeric value, where the underlying data type is SQL_TINYINT, SQL_SMALLINT, or SQL_INTEGER.

The string functions are one-based; that is, the first character in the string is the character 1. Character string literals must be enclosed by single quotation marks.

Table A-2. Scalar String Functions

Function	Returns
ASCII(<i>string_exp</i>)	The ASCII code of the leftmost character of <i>string_exp</i> as an integer.
BIT_LENGTH(<i>string_exp</i>)	The length, in bits, of the string expression.
CHAR(<i>code</i>)	The character with the ASCII code specified by <i>code</i> . <i>code</i> should be between 0 and 255; otherwise, the return value depends on the data source.
CHAR_LENGTH(<i>string_exp</i>)	The length, in characters, of the string expression, when the string expression is a character data type; otherwise, the length, in bytes, of the string expression (the lowest integer that is not less than the number of bits divided by 8). (This function is the same as the CHARACTER_LENGTH function.)
CHARACTER_LENGTH(<i>string_exp</i>)	The length, in characters, of the string expression, when the string expression is a character data type; otherwise, the length, in bytes, of the string expression (the lowest integer that is not less than the number of bits divided by 8). (This function is the same as the CHAR_LENGTH function.)

Table A-2. Scalar String Functions (cont.)

Function	Returns
CONCAT(<i>string_exp1</i> , <i>string_exp2</i>)	The string resulting from concatenating <i>string_exp2</i> and <i>string_exp1</i> . The string is system dependent.
DIFFERENCE(<i>string_exp1</i> , <i>string_exp2</i>)	An integer indicating the difference between the values returned by the SOUNDEX function for <i>string_exp1</i> and <i>string_exp2</i> .
INSERT(<i>string_exp1</i> , <i>start</i> , <i>length</i> , <i>string_exp2</i>)	A string where <i>length</i> characters have been deleted from <i>string_exp1</i> beginning at <i>start</i> and where <i>string_exp2</i> has been inserted into <i>string_exp1</i> beginning at <i>start</i> .
LCASE(<i>string_exp</i>)	Uppercase characters in <i>string_exp</i> converted to lowercase.
LEFT(<i>string_exp</i> , <i>count</i>)	The <i>count</i> of characters of <i>string_exp</i> .
LENGTH(<i>string_exp</i>)	The number of characters in <i>string_exp</i> , excluding trailing blanks and the string termination character.
LOCATE(<i>string_exp1</i> , <i>string_exp2</i> [, <i>start</i>])	The starting position of the first occurrence of <i>string_exp1</i> in <i>string_exp2</i> . If <i>start</i> is not specified, the search begins with the first character position in <i>string_exp2</i> . If <i>start</i> is specified, the search begins with the character position indicated by <i>start</i> . The first character position in <i>string_exp2</i> is indicated by 1. If <i>string_exp1</i> is not found, 0 is returned.
LTRIM(<i>string_exp</i>)	The characters of <i>string_exp</i> , with leading blanks removed.
OCTET_LENGTH(<i>string_exp</i>)	The length, in bytes, of the string expression. The result is the lowest integer that is not less than the number of bits divided by 8.
POSITION(<i>character_exp</i> IN <i>character_exp</i>)	The position of the first character expression in the second character expression. The result is a numeric with an implementation-defined precision and a scale of 0.
REPEAT(<i>string_exp</i> , <i>count</i>)	A string composed of <i>string_exp</i> repeated <i>count</i> times.
REPLACE(<i>string_exp1</i> , <i>string_exp2</i> , <i>string_exp3</i>)	Replaces all occurrences of <i>string_exp2</i> in <i>string_exp1</i> with <i>string_exp3</i> .
RIGHT(<i>string_exp</i> , <i>count</i>)	The rightmost <i>count</i> of characters in <i>string_exp</i> .

Table A-2. Scalar String Functions (cont.)

Function	Returns
RTRIM(<i>string_exp</i>)	The characters of <i>string_exp</i> with trailing blanks removed.
SOUNDEX(<i>string_exp</i>)	A data-source dependent string representing the sound of the words in <i>string_exp</i> .
SPACE(<i>count</i>)	A string consisting of <i>count</i> spaces.
SUBSTRING(<i>string_exp</i> , <i>start</i> , <i>length</i>)	A string derived from <i>string_exp</i> , beginning at the character position <i>start</i> for <i>length</i> characters.
UCASE(<i>string_exp</i>)	Lowercase characters in <i>string_exp</i> converted to uppercase.

Numeric Functions

[Table A-3](#) lists numeric functions. The following arguments can be used with numeric functions:

- *numeric_exp* can be a column name, a numeric literal, or the result of another scalar function, where the underlying data type is SQL_NUMERIC, SQL_DECIMAL, SQL_TINYINT, SQL_SMALLINT, SQL_INTEGER, SQL_BIGINT, SQL_FLOAT, SQL_REAL, or SQL_DOUBLE.
- *float_exp* can be a column name, a numeric literal, or the result of another scalar function, where the underlying data type is SQL_FLOAT.
- *integer_exp* can be a column name, a numeric literal, or the result of another scalar function, where the underlying data type is SQL_TINYINT, SQL_SMALLINT, SQL_INTEGER, or SQL_BIGINT.

Table A-3. Scalar Numeric Functions

Function	Returns
<code>ABS(<i>numeric_exp</i>)</code>	Absolute value of <i>numeric_exp</i> .
<code>ACOS(<i>float_exp</i>)</code>	Arccosine of <i>float_exp</i> as an angle in radians.
<code>ASIN(<i>float_exp</i>)</code>	Arcsine of <i>float_exp</i> as an angle in radians.
<code>ATAN(<i>float_exp</i>)</code>	Arctangent of <i>float_exp</i> as an angle in radians.
<code>ATAN2(<i>float_exp1</i>, <i>float_exp2</i>)</code>	Arctangent of the x and y coordinates, specified by <i>float_exp1</i> and <i>float_exp2</i> as an angle in radians.
<code>CEILING(<i>numeric_exp</i>)</code>	Smallest integer greater than or equal to <i>numeric_exp</i> .
<code>COS(<i>float_exp</i>)</code>	Cosine of <i>float_exp</i> as an angle in radians.
<code>COT(<i>float_exp</i>)</code>	Cotangent of <i>float_exp</i> as an angle in radians.
<code>DEGREES(<i>numeric_exp</i>)</code>	Number if degrees converted from <i>numeric_exp</i> radians.
<code>EXP(<i>float_exp</i>)</code>	Exponential value of <i>float_exp</i> .
<code>FLOOR(<i>numeric_exp</i>)</code>	Largest integer less than or equal to <i>numeric_exp</i> .
<code>LOG(<i>float_exp</i>)</code>	Natural log of <i>float_exp</i> .
<code>LOG10(<i>float_exp</i>)</code>	Base 10 log of <i>float_exp</i> .
<code>MOD(<i>integer_exp1</i>, <i>integer_exp2</i>)</code>	Remainder of <i>integer_exp1</i> divided by <i>integer_exp2</i> .
<code>PI()</code>	Constant value of pi as a floating-point number.
<code>POWER(<i>numeric_exp</i>, <i>integer_exp</i>)</code>	Value of <i>numeric_exp</i> to the power of <i>integer_exp</i> .
<code>RADIANS(<i>numeric_exp</i>)</code>	Number of radians converted from <i>numeric_exp</i> degrees.
<code>RAND([<i>integer_exp</i>])</code>	Random floating-point value using <i>integer_exp</i> as the optional seed value.
<code>ROUND(<i>numeric_exp</i>, <i>integer_exp</i>)</code>	<i>numeric_exp</i> rounded to <i>integer_exp</i> places right of the decimal (left of the decimal if <i>integer_exp</i> is negative).
<code>SIGN(<i>numeric_exp</i>)</code>	Indicator of the sign of <i>numeric_exp</i> . If <i>numeric_exp</i> < 0, -1 is returned. If <i>numeric_exp</i> = 0, 0 is returned. If <i>numeric_exp</i> > 0, 1 is returned.

Table A-3. Scalar Numeric Functions (cont.)

Function	Returns
SIN(<i>float_exp</i>)	Sine of <i>float_exp</i> , where <i>float_exp</i> is an angle in radians.
SQRT(<i>float_exp</i>)	Square root of <i>float_exp</i> .
TAN(<i>float_exp</i>)	Tangent of <i>float_exp</i> , where <i>float_exp</i> is an angle in radians.
TRUNCATE(<i>numeric_exp</i> , <i>integer_exp</i>)	<i>numeric_exp</i> truncated to <i>integer_exp</i> places right of the decimal. (If <i>integer_exp</i> is negative, truncation is to the left of the decimal.)

Date and Time Functions

[Table A-4](#) lists date and time functions. The following arguments can be used with the date and time functions:

- *date_exp* can be a column name, a date or timestamp literal, or the result of another scalar function, where the underlying data type can be represented as SQL_CHAR, SQL_VARCHAR, SQL_DATE, or SQL_TIMESTAMP.
- *time_exp* can be a column name, a timestamp or timestamp literal, or the result of another scalar function, where the underlying data type can be represented as SQL_CHAR, SQL_VARCHAR, SQL_TIME, or SQL_TIMESTAMP.
- *timestamp_exp* can be a column name; a time, date, or timestamp literal; or the result of another scalar function, where the underlying data type can be represented as SQL_CHAR, SQL_VARCHAR, SQL_TIME, SQL_DATE, or SQL_TIMESTAMP.

Table A-4. Scalar Time and Date Functions

Function	Returns
CURDATE()	Current date as a date value.
CURRENT_DATE()	Current date.
CURRENT_TIME[(<i>time-precision</i>)]	Current local time. The <i>time-precision</i> argument determines the seconds precision of the returned value.
CURRENT_TIMESTAMP[(<i>timestamp-precision</i>)]	Current local date and local time as a timestamp value. The <i>timestamp-precision</i> argument determines the seconds precision of the returned timestamp.
CURTIME()	Current local time as a time value.
DAYNAME(<i>date_exp</i>)	Character string containing a data-source-specific name of the day for the day portion of <i>date_exp</i> .
DAYOFMONTH(<i>date_exp</i>)	Day of the month in <i>date_exp</i> as an integer value (1-31).
DAYOFWEEK(<i>date_exp</i>)	Day of the week in <i>date_exp</i> as an integer value (1-7).
DAYOFYEAR(<i>date_exp</i>)	Day of the year in <i>date_exp</i> as an integer value (1-366).
EXTRACT(<i>extract-field</i> FROM <i>extract-source</i>)	<i>Extract-field</i> portion of the <i>extract-source</i> . The <i>extract-source</i> argument is a datetime or interval expression. The <i>extract-field</i> argument can be one of the following keywords: YEAR MONTH DAY HOUR MINUTE SECOND The precision scale of the returned value is 0 unless SECOND is specified, in which case, the scale is not less than the fractional seconds precision of the <i>extract-source</i> field.
HOUR(<i>time_exp</i>)	Hour in <i>time_exp</i> as an integer value (0-23).

Table A-4. Scalar Time and Date Functions (cont.)

Function	Returns
MINUTE(<i>time_exp</i>)	Minute in <i>time_exp</i> as an integer value (0–59).
MONTH(<i>date_exp</i>)	Month in <i>date_exp</i> as an integer value (1–12).
MONTHNAME(<i>date_exp</i>)	Character string containing the data source–specific name of the month.
NOW()	Current date and time as a timestamp value.
QUARTER(<i>date_exp</i>)	Quarter in <i>date_exp</i> as an integer value (1–4).
SECOND(<i>time_exp</i>)	Second in <i>time_exp</i> as an integer value (0–59).
TIMESTAMPADD(<i>interval</i> , <i>integer_exp</i> , <i>time_exp</i>)	Timestamp calculated by adding <i>integer_exp</i> intervals of type <i>interval</i> to <i>time_exp</i> . <i>interval</i> can be one of the following values: SQL_TSI_FRAC_SECOND SQL_TSI_SECOND SQL_TSI_MINUTE SQL_TSI_HOUR SQL_TSI_DAY SQL_TSI_WEEK SQL_TSI_MONTH SQL_TSI_QUARTER SQL_TSI_YEAR Fractional seconds are expressed in billionths of a second.
TIMESTAMPDIFF(<i>interval</i> , <i>time_exp1</i> , <i>time_exp2</i>)	Integer number of intervals of type <i>interval</i> by which <i>time_exp2</i> is greater than <i>time_exp1</i> . <i>interval</i> has the same value as <i>TIMESTAMPADD</i> . Fractional seconds are expressed in billionths of a second.
WEEK(<i>date_exp</i>)	Week of the year in <i>date_exp</i> as an integer value (1–53).
YEAR(<i>date_exp</i>)	Year in <i>date_exp</i> . The range is data-source dependent.

System Functions

Table A-5 lists system functions.

Table A-5. Scalar System Functions

Function	Returns
DATABASE()	Name of the database, corresponding to the connection handle (<code>hdbc</code>).
IFNULL(<i>exp,value</i>)	<i>value</i> , if <i>exp</i> is null.
USER()	Authorization name of the user.

Like Predicate Escape Characters

In a LIKE predicate, the percent sign (%) matches zero or more of any character and the underscore(_) matches any one character. To match an actual percent sign or underscore in a LIKE predicate, an escape character must precede the % or _. The escape sequence that defines the LIKE predicate escape character is:

```
{escape 'escape-character'}
```

where *escape-character* is any character supported by the data source.

Example:

```
SELECT Name FROM Customers
WHERE Name LIKE '\%AAA%' {escape '\'}
```

returns all the customers for which the name start with "%AAA".

Outer Join Escape Sequences

ODBC and JDBC support the SQL92 left, right, and full outer join syntax. The escape sequence for outer joins is:

```
{oj outer-join}
```

where *outer-join* is:

```
table-reference {LEFT | RIGHT | FULL} OUTER JOIN  
{table-reference | outer-join} ON search-condition
```

where *table-reference* is a table name, and *search-condition* is the join condition you want to use for the tables.

Example:

```
SELECT Customers.CustID, Customers.Name, Orders.OrderID,  
Orders.Status  
FROM {oj Customers LEFT OUTER JOIN  
Orders ON Customers.CustID=Orders.CustID}  
WHERE Orders.Status='OPEN'
```

[Table A-6](#) lists the outer join escape sequences supported by SequeLink for each data store.

Table A-6. Outer Join Escape Sequences Supported by the SequeLink ODBC Driver and SequeLink JDBC Driver

Data Store	Outer Join Escape Sequences
DB2 V4, V5, V6R1 on OS/390	Left outer joins Right outer joins Nested outer joins
DB2 V6R1, V7R1 on Windows NT and Windows 2000	Left outer joins Right outer joins Full outer joins Nested outer joins Unordered outer joins Inner outer joins

Table A-6. Outer Join Escape Sequences Supported by the SequeLink ODBC Driver and SequeLink JDBC Driver (cont.)

Data Store	Outer Join Escape Sequences
DB2 V6R1, V7R1 on UNIX	Left outer joins Right outer joins Full outer joins Nested outer joins Unordered outer joins Inner outer joins
Informix	Left outer joins Unordered outer joins
Microsoft SQL Server	Left outer joins Right outer joins Full outer joins Nested outer joins
Oracle	Left outer joins Right outer joins Nested outer joins
Sybase	Inner joins Left outer joins Nested outer joins Unordered outer joins

Procedure Call Escape Sequences

A procedure is an executable object stored in the data store. Generally, it is one or more SQL statements that have been precompiled. The escape sequence for calling a procedure is:

```
{[?]=call procedure-name([parameter] [, [parameter]]...)}
```

where:

procedure-name is the name of a stored procedure.

parameter is a stored procedure parameter.

NOTE: For DB2 V6, stored procedures may be qualified with a schema name. You can call stored procedures with or without the schema name qualification.

B Data Types and Isolation Levels

This appendix lists the data types and isolation levels supported for each data store supported by SequeLink.

Supported Data Types

This section lists the ODBC, ADO/OLE DB, and JDBC data types that SequeLink supports for each data store.

DB2 V4, V5

[Table B-1](#) lists the data types supported by the SequeLink ODBC Driver, SequeLink ADO Provider, and SequeLink JDBC Driver for DB2 V4 and V5.

Table B-1. Data Types (DB2 V4, V5)

DB2 Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Char	SQL_CHAR	DBTYPE_STR	CHAR
Char() for Bit Data	SQL_BINARY	DBTYPE_BYTES	BINARY
Date	SQL__TYPE_DATE	DBTYPE_DBDATE	DATE
Decimal	SQL_DECIMAL	DBTYPE_NUMERIC	DECIMAL
Float	SQL_FLOAT	DBTYPE_R8	FLOAT
Integer	SQL_INTEGER	DBTYPE_I4	INTEGER
Long Varchar	SQL_LONGVARCHAR	DBTYPE_STR	LONGVARCHAR

Table B-1. Data Types (DB2 V4, V5) (cont.)

DB2 Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Long Varchar for Bit Data	SQL_LONGVARBINARY	DBTYPE_BYTES	LONGVARBINARY
Real	SQL_REAL	DBTYPE_R4	REAL
Smallint	SQL_SMALLINT	DBTYPE_I2	SMALLINT
Time	SQL_TYPE_TIME	DBTYPE_DBTIME	TIME
Timestamp	SQL_TYPE_TIMESTAMP	DBTYPE_DBTIMESTAMP	TIMESTAMP
Varchar	SQL_VARCHAR	DBTYPE_STR	VARCHAR
Varchar() for Bit Data	SQL_VARBINARY	DBTYPE_BYTES	VARBINARY

DB2 V6, V7

[Table B-2](#) lists the data types supported by the SequeLink ODBC Driver, SequeLink ADO Provider, and SequeLink JDBC Driver for DB2 V6 and V7.

Table B-2. Data Types (DB2 V6, V7)

DB2 Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Blob	SQL_LONGVARBINARY	DBTYPE_BYTES	LONGVARBINARY
Char	SQL_CHAR	DBTYPE_STR	CHAR
Char() for Bit Data	SQL_BINARY	DBTYPE_BYTES	BINARY
Clob	SQL_LONGVARCHAR	DBTYPE_STR	LONGVARCHAR
Date	SQL__TYPE_DATE	DBTYPE_DBDATE	DATE
Decimal	SQL_DECIMAL	DBTYPE_NUMERIC	DECIMAL
Float	SQL_FLOAT	DBTYPE_R8	FLOAT
Integer	SQL_INTEGER	DBTYPE_I4	INTEGER
Long Varchar	SQL_LONGVARCHAR	DBTYPE_STR	LONGVARCHAR

Table B-2. Data Types (DB2 V6, V7) (cont.)

DB2 Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Long Varchar for Bit Data	SQL_LONGVARBINARY	DBTYPE_BYTES	LONGVARBINARY
Real	SQL_REAL	DBTYPE_R4	REAL
Rowid	SQL_VARBINARY	DBTYPE_BYTES	VARBINARY
Smallint	SQL_SMALLINT	DBTYPE_I2	SMALLINT
Time	SQL_TYPE_TIME	DBTYPE_DBTIME	TIME
Timestamp	SQL_TYPE_TIMESTAMP	DBTYPE_DBTIMESTAMP	TIMESTAMP
Varchar	SQL_VARCHAR	DBTYPE_STR	VARCHAR
Varchar() for Bit Data	SQL_VARBINARY	DBTYPE_BYTES	VARBINARY

Informix 7

[Table B-3](#) lists the data types supported by the SequeLink ODBC Driver, SequeLink ADO Provider, and SequeLink JDBC Driver for Informix 7.

Table B-3. Data Types (Informix 7)

Informix Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Byte	SQL_LONGVARBINARY	DBTYPE_BYTES	LONGVARBINARY
Char	SQL_CHAR	DBTYPE_STR	CHAR
Date	SQL__TYPE_DATE	DBTYPE_DBDATE	DATE
Datetime hour to second	SQL__TYPE_TIME	DBTYPE_DBTIME	TIME
Datetime year to fraction(5)	SQL__TYPE_TIMESTAMP	DBTYPE_DBTIMESTAMP	TIMESTAMP
Decimal	SQL_DECIMAL	DBTYPE_NUMERIC	DECIMAL

Table B-3. Data Types (Informix 7) (cont.)

Informix Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Float	SQL_FLOAT	DBTYPE_R8	FLOAT
Integer	SQL_INTEGER	DBTYPE_I4	INTEGER
Money	SQL_DECIMAL	DBTYPE_NUMERIC	DECIMAL
Serial	SQL_INTEGER	DBTYPE_I4	INTEGER
Smallfloat	SQL_REAL	DBTYPE_R4	REAL
Smallint	SQL_SMALLINT	DBTYPE_I2	SMALLINT
Text	SQL_LONGVARCHAR	DBTYPE_STR	LONGVARCHAR
Varchar	SQL_VARCHAR	DBTYPE_STR	VARCHAR

Informix 9

[Table B-4](#) lists the data types supported by the SequeLink ODBC Driver, SequeLink ADO Provider, and SequeLink JDBC Driver for Informix 9.

Table B-4. Data Types (Informix 9)

Informix Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Blob	SQL_LONGVARBINARY	DBTYPE_BYTES	LONGVARBINARY
Boolean	SQL_BIT	DBTYPE_BOOL	BIT
Byte	SQL_LONGVARBINARY	DBTYPE_BYTES	LONGVARBINARY
Char	SQL_CHAR	DBTYPE_STR	CHAR
Clob	SQL_LONGVARCHAR	DBTYPE_STR	LONGVARCHAR
Date	SQL__TYPE_DATE	DBTYPE_DBDATE	DATE
Datetime hour to second	SQL__TYPE_TIME	DBTYPE_DBTIME	TIME

Table B-4. Data Types (Informix 9) (cont.)

Informix Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Datetime year to fraction(5)	SQL__TYPE_TIMESTAMP	DBTYPE_DBTIMESTAMP	TIMESTAMP
Decimal	SQL_DECIMAL	DBTYPE_NUMERIC	DECIMAL
Float	SQL_FLOAT	DBTYPE_R8	FLOAT
Integer	SQL_INTEGER	DBTYPE_I4	INTEGER
Int8	SQL_BIGINT	DBTYPE_I8	BIGINT
Lvarchar	SQL_LONGVARCHAR	DBTYPE_STR	LONGVARCHAR
Money	SQL_DECIMAL	DBTYPE_NUMERIC	DECIMAL
Numeric	SQL_NUMERIC	DBTYPE_NUMERIC	NUMERIC
Serial	SQL_INTEGER	DBTYPE_I4	INTEGER
Serial8	SQL_BIGINT	DBTYPE_I8	BIGINT
Smallfloat	SQL_REAL	DBTYPE_R4	REAL
Smallint	SQL_SMALLINT	DBTYPE_I2	SMALLINT
Text	SQL_LONGVARCHAR	DBTYPE_STR	LONGVARCHAR
Varchar	SQL_VARCHAR	DBTYPE_STR	VARCHAR

Microsoft SQL Server 7

Table B-5 lists the data types supported by the SequeLink ODBC Driver, SequeLink ADO Provider, and SequeLink JDBC Driver for Microsoft SQL Server7.

Table B-5. Data Types (Microsoft SQL Server 7)

Microsoft SQL Server Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Binary	SQL_BINARY	DBTYPE_BYTES	BINARY
Bit	SQL_BIT	DBTYPE_BOOL	BIT
Char	SQL_CHAR	DBTYPE_STR	CHAR
Datetime	SQL_DATETIME	DBTYPE_DBTIMESTAMP	TIMESTAMP
Decimal	SQL_DECIMAL	DBTYPE_NUMERIC	DECIMAL
Decimal() identity	SQL_DECIMAL	DBTYPE_NUMERIC	DECIMAL
Float	SQL_FLOAT	DBTYPE_R8	FLOAT
Image	SQL_LONGVARBINARY	DBTYPE_BYTES	LONGVARBINARY
Int	SQL_INTEGER	DBTYPE_I4	INTEGER
Int identity	SQL_INTEGER	DBTYPE_I4	INTEGER
Money	SQL_DECIMAL	DBTYPE_CY	DECIMAL
Nchar	SQL_CHAR	DBTYPE_WSTR	CHAR
Ntext	SQL_LONGVARCHAR	DBTYPE_WSTR	LONGVARCHAR
Numeric	SQL_NUMERIC	DBTYPE_NUMERIC	NUMERIC
Numeric() identity	SQL_NUMERIC	DBTYPE_NUMERIC	NUMERIC
Nvarchar	SQL_VARCHAR	DBTYPE_WSTR	VARCHAR
Real	SQL_REAL	DBTYPE_R4	REAL
Smalldatetime	SQL_DATETIME	DBTYPE_DBTIMESTAMP	TIMESTAMP
Smallint	SQL_SMALLINT	DBTYPE_I2	SMALLINT
Smallint identity	SQL_SMALLINT	DBTYPE_I2	SMALLINT
Smallmoney	SQL_DECIMAL	DBTYPE_CY	DECIMAL
Sysname	SQL_VARCHAR	DBTYPE_WSTR	VARCHAR
Text	SQL_LONGVARCHAR	DBTYPE_STR	LONGVARCHAR

Table B-5. Data Types (Microsoft SQL Server 7) (cont.)

Microsoft SQL Server Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Timestamp	SQL_BINARY	DBTYPE_BYTES	BINARY
Tinyint	SQL_TINYINT	DBTYPE_UI1	TINYINT
Tinyint identity	SQL_TINYINT	DBTYPE_UI1	TINYINT
Uniqueidentifier	SQL_BINARY	DBTYPE_BINARY	BINARY
Varbinary	SQL_VARBINARY	DBTYPE_BYTES	VARBINARY
Varchar	SQL_VARCHAR	DBTYPE_STR	VARCHAR

Microsoft SQL Server2000

[Table B-6](#) lists the data types supported by the SequeLink ODBC Driver, SequeLink ADO Provider, and SequeLink JDBC Driver for Microsoft SQL Server2000.

Table B-6. Data Types (Microsoft SQL Server2000)

Microsoft SQL Server Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Bigint	SQL_BIGINT	DBTYPE_I8	BIGINT
Bigint identity	SQL_BIGINT	DBTYPE_I8	BIGINT
Binary	SQL_BINARY	DBTYPE_BYTES	BINARY
Bit	SQL_BIT	DBTYPE_BOOL	BIT
Char	SQL_CHAR	DBTYPE_STR	CHAR
Datetime	SQL_DATETIME	DBTYPE_DBTIMESTAMP	TIMESTAMP
Decimal	SQL_DECIMAL	DBTYPE_NUMERIC	DECIMAL
Decimal() identity	SQL_DECIMAL	DBTYPE_NUMERIC	DECIMAL
Float	SQL_FLOAT	DBTYPE_R8	FLOAT

Table B-6. Data Types (Microsoft SQL Server2000) (cont.)

Microsoft SQL Server Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Image	SQL_LONGVARBINARY	DBTYPE_BYTES	LONGVARBINARY
Int	SQL_INTEGER	DBTYPE_I4	INTEGER
Int identity	SQL_INTEGER	DBTYPE_I4	INTEGER
Money	SQL_DECIMAL	DBTYPE_CY	DECIMAL
Nchar	SQL_CHAR	DBTYPE_WSTR	CHAR
Ntext	SQL_LONGVARCHAR	DBTYPE_WSTR	LONGVARCHAR
Numeric	SQL_NUMERIC	DBTYPE_NUMERIC	NUMERIC
Numeric() identity	SQL_NUMERIC	DBTYPE_NUMERIC	NUMERIC
Nvarchar	SQL_VARCHAR	DBTYPE_WSTR	VARCHAR
Real	SQL_REAL	DBTYPE_R4	REAL
Smalldatetime	SQL_DATETIME	DBTYPE_DBTIMESTAMP	TIMESTAMP
Smallint	SQL_SMALLINT	DBTYPE_I2	SMALLINT
Smallint identity	SQL_SMALLINT	DBTYPE_I2	SMALLINT
Smallmoney	SQL_DECIMAL	DBTYPE_CY	DECIMAL
Sysname	SQL_VARCHAR	DBTYPE_WSTR	VARCHAR
Text	SQL_LONGVARCHAR	DBTYPE_STR	LONGVARCHAR
Timestamp	SQL_BINARY	DBTYPE_BYTES	BINARY
Tinyint	SQL_TINYINT	DBTYPE_UI1	TINYINT
Tinyint identity	SQL_TINYINT	DBTYPE_UI1	TINYINT
Uniqueidentifier	SQL_BINARY	DBTYPE_BINARY	BINARY
Varbinary	SQL_VARBINARY	DBTYPE_BYTES	VARBINARY
Varchar	SQL_VARCHAR	DBTYPE_STR	VARCHAR

Oracle7

[Table B-7](#) lists the data types supported by the SequeLink ODBC Driver, SequeLink ADO Provider, and SequeLink JDBC Driver for Oracle7.

Table B-7. Data Types (Oracle7)

Oracle7 Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Char	SQL_CHAR	DBTYPE_STR	CHAR
Date	SQL_TYPE_TIMESTAMP	DBTYPE_DBTIMESTAMP	TIMESTAMP
Long	SQL_LONGVARCHAR	DBTYPE_STR	LONGVARCHAR
Long Raw	SQL_LONGVARBINARY	DBTYPE_BYTES	LONGVARBINARY
Number	SQL_FLOAT	DBTYPE_R8	FLOAT
Number(p,s)	SQL_DECIMAL	DBTYPE_NUMERIC	DECIMAL
Raw	SQL_VARBINARY	DBTYPE_BYTES	VARBINARY
Varchar2	SQL_VARCHAR	DBTYPE_STR	VARCHAR

Oracle8

[Table B-8](#) lists the data types supported by the SequeLink ODBC Driver, SequeLink ADO Provider, and SequeLink JDBC Driver for Oracle8.

Table B-8. Data Types (Oracle8)

Oracle8 Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Bfile	SQL_LONGVARBINARY	DBTYPE_BYTES	LONGVARBINARY
Blob	SQL_LONGVARBINARY	DBTYPE_BYTES	LONGVARBINARY
Char	SQL_CHAR	DBTYPE_STR	CHAR

Table B-8. Data Types (Oracle8) (cont.)

Oracle8 Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Clob	SQL_LONGVARCHAR	DBTYPE_STR	LONGVARCHAR
Date	SQL_TYPE_TIMESTAMP	DBTYPE_DBTIMESTAMP	TIMESTAMP
Long	SQL_LONGVARCHAR	DBTYPE_STR	LONGVARCHAR
Long Raw	SQL_LONGVARBINARY	DBTYPE_BYTES	LONGVARBINARY
Nchar	SQL_CHAR	*	CHAR
Nclob	SQL_LONGVARCHAR	*	LONGVARCHAR
Number	SQL_FLOAT	DBTYPE_R8	FLOAT
Number(p,s)	SQL_DECIMAL	DBTYPE_NUMERIC	DECIMAL
NVarchar2	SQL_VARCHAR	*	VARCHAR
Raw	SQL_VARBINARY	DBTYPE_BYTES	VARBINARY
Rowid	SQL_VARCHAR	DBTYPE_STR	VARCHAR
Varchar2	SQL_VARCHAR	DBTYPE_STR	VARCHAR

* No corresponding data type.

Sybase

Table B-9 lists the data types supported by the SequeLink ODBC Driver, SequeLink ADO Provider, and SequeLink JDBC Driver for Sybase 11 and 12.

Table B-9. Data Types (Sybase 11 and 12)

Sybase Data Type	ODBC Data Type	OLE DB Data Type	JDBC Data Type
Binary	SQL_BINARY	DBTYPE_BYTES	BINARY
Bit	SQL_BIT	DBTYPE_BOOL	BIT
Char	SQL_CHAR	DBTYPE_STR	CHAR
Datetime	SQL__TYPE_DATETIME	DBTYPE_DBTIMESTAMP	TIMESTAMP
Decimal	SQL_DECIMAL	DBTYPE_NUMERIC	DECIMAL
Float	SQL_FLOAT	DBTYPE_R8	FLOAT
Image	SQL_LONGVARBINARY	DBTYPE_BYTES	LONGVARBINARY
Int	SQL_INTEGER	DBTYPE_I4	INTEGER
Money	SQL_DECIMAL	DBTYPE_CY	DECIMAL
Numeric	SQL_NUMERIC	DBTYPE_NUMERIC	NUMERIC
Real	SQL_REAL	DBTYPE_R4	REAL
Smalldatetime	SQL__TYPE_DATETIME	DBTYPE_DBTIMESTAMP	TIMESTAMP
Smallint	SQL_SMALLINT	DBTYPE_I2	SMALLINT
Smallmoney	SQL_DECIMAL	DBTYPE_CY	DECIMAL
Sysname	SQL_VARCHAR	DBTYPE_STR	VARCHAR
Text	SQL_LONGVARCHAR	DBTYPE_STR	LONGVARCHAR
Timestamp	SQL_BINARY	DBTYPE_BYTES	BINARY
Tinyint	SQL_TINYINT	DBTYPE_UI1	TINYINT
Varbinary	SQL_VARBINARY	DBTYPE_BYTES	VARBINARY
Varchar	SQL_VARCHAR	DBTYPE_STR	VARCHAR

Isolation Levels

This section lists the isolation levels supported by Sequelink for each data store, including their default isolation level.

Table B-10. Isolation Levels

Database	Isolation Levels	Default
DB2 V4	Read uncommitted Read committed Serializable	Read committed
DB2 V5, V6, V7	Read uncommitted Read committed Repeatable read Serializable	Read committed
Informix	Read uncommitted Read committed Repeatable read	Read committed
Microsoft SQL Server	Read uncommitted Read committed Repeatable read Serializable	Read committed
Oracle	Read committed Serializable	Read committed
Sybase	Read uncommitted Read committed Repeatable read Serializable	Read committed

Index

A

- ABS function 301
- Access Order 165, 191
- accessor support 136, 179
- ACOS function 301
- Active Sessions 173
- ADO
 - Command object 164
 - connection attributes 118
 - Connection object 172
 - data shaping 197
 - Field object 184
 - mapping OLE DB methods 164
 - OLE DB interfaces supported 162
 - Parameter object 185, 186
 - Recordset object 188
- ADO client data sources. *See* client data sources
- API functions 60
- application developers
 - ADO 123
 - ODBC 58
- application ID
 - generating automatically 73, 200
 - specifying explicitly 72, 118, 200
 - specifying for ADO Provider 200
 - specifying for JDBC Driver 280
- ApplicationID ODBC connection attribute 47
- arguments, null 78
- ASCII
 - converting to character 298
 - function 298
- ASIN function 301
- Asynchable
 - Abort 173
 - Commit 173

- ATAN function 301
- ATAN2 function 301
- attribute=value pairs 117
- attributes
 - ADO 118
 - DistinguishedName 121
 - ODBC 47
- Autocommit Isolation Levels 174
- AutomaticApplicationID ODBC connection attribute 47

B

- backward scrolling 170
- batch execution on a prepared statement 251
- BigDecimal objects 286
- blanks, generating 300
- BlockFetchForUpdate 47
- Blocking Storage Objects 165, 191
- Bookmark Information 191
- Bookmark Type 191
- bookmarks 155, 196
- Bookmarks Ordered 191
- books, ordering printed 21
- bound columns 85
- built-in properties
 - Command object 164
 - Connection object 172
 - Field object 184
 - Parameter object 185
 - Recordset object 189

C

- catalog functions 77
- Catalog Location 174
- Catalog Term 174
- Catalog Usage 174
- CEILING function 301
- centralized odbc.ini files 45
- Change Inserted Rows 165, 191
- changing
 - directories for ADO client data sources 107
- CHAR function 298
- character set conversion 285
- Client
 - SequeLink ADO 97
 - SequeLink Java 207
 - SequeLink ODBC 27
- client data sources
 - creating
 - ADO 103
 - creating ADO 107
 - modifying ADO 106
- Column Definition 174
- Column Privileges 165, 191
- COM Object Support 175
- Command object
 - built-in properties 164
 - dynamic properties 165
 - methods 164
- Command object (OLE DB) 126
- Command Time Out 165
- Command Timeout 192
- committing data 91
- compatibility of JDBC versions 274
- compiler requirements, UNIX 59
- CONCAT function 299
- concurrency types for result sets 278
- Configuration Manager
 - menu bar 100
 - overview 98
- configuring
 - client data sources
 - ADO 103
 - file client data sources 32
 - JDBC data sources 217
 - JDBCTest 236
 - ODBC client data source for UNIX 43
- connecting
 - to data source, logon dialog box 109
 - using URLs (JDBC) 215
 - with a provider string 117
 - with JDBC data sources 223
- connection
 - attributes in ADO 118
 - attributes in ODBC 47
 - handles 91
 - JDBC options 207
 - ODBC options 27
 - precedence for JDBC Driver 223
 - properties for JDBC Driver 223
 - testing ADO 109
 - testing ODBC on Windows 42
 - testing with JDBCTest 236
- Connection dialog box 109
- Connection object
 - built-in properties 172
 - dynamic properties 173
 - methods supported 172
- connection pooling 219
- Connection Status 175
- contacting Technical Support 23
- conventions used in this book 18
- converting variant types to date/time types
 - 199
- copying, ADO client data sources 108
- COS function 301
- COT function 301
- creating
 - ADO client data sources 103
- CURDATE function 303
- Current Catalog 175
- cursors
 - insensitive scrollable 230
 - keyset-driven 66

- scrollable 66
- static 66
- support for 66, 278
- using scrollable instead of cursor library 89

CURTIME function 303

D

- data shaping 197
- data source
 - configuring JDBC 217
 - configuring User and System (ODBC) 28
 - configuring, values overridden 198
 - connecting with JDBC 223
 - creating file client (ODBC) 32
 - creating JDBC 218
 - displaying properties 101
 - examples of JDBC 218
 - System 28
 - User 28
- data source information property group 135
- Data Source Name 175
- Data Source object 127
- Data Source Object Threading Model 175
- Data Source property 175
- data source property group 135
- data types
 - DB2 V4, V5 309
 - DB2 V6, V7 310
 - Informix 7 311
 - Informix 9 312
 - mapping year format 199
 - Microsoft SQL Server 314
 - Microsoft SQL Server2000 315
 - Oracle7 317
 - Oracle8 317
 - Sybase 319
- database
 - data dictionary 81
 - meta-information, retrieving 81
 - naming using JNDI 219
- Database connection attribute 47, 118

- Database Data Dictionary
 - filters 82
 - views (DB2 for OS/390) 82
- DATABASE function 305
- database-specific information 309
- date
 - returning current 303
 - returning month 304
 - returning year 304
 - scalar functions 302
- DAYNAME function 303
- DAYOFMONTH function 303
- DAYOFWEEK function 303
- DAYOFYEAR function 303
- DB2 data types 309, 310, 314
- DB2 for OS/390
 - Database Data Dictionary views 82
 - Distributed Transaction Management
 - support 221
 - support for scrollable cursors (JDBC) 278
 - support for scrollable cursors (ODBC) 66
- DBMS Name property 175
- DBMS Version property 175
- DBPassword 48
- DBPROP_
 - ROWSETCONVERSIONSONCOMMAND 144
- DBPROP_ SUBQUERIES 146
- DBPROP_ SUPPORTEDTXNDDL 147
- DBPROP_ SUPPORTEDTXNISOLEVELS 147
- DBPROP_ SUPPORTEDTXNISORETAIN 147
- DBPROP_ABORTPRESERVE 153
- DBPROP_ACCESSORDER 154
- DBPROP_ASYNCCTXNABORT 136
- DBPROP_ASYNCCTXNCOMMIT 136
- DBPROP_AUTH_ PASSWORD 148
- DBPROP_AUTH_ USERID 149
- DBPROP_AUTH_ PASSWORD 148
- DBPROP_AUTH_PERSIST_SENSITIVE_AUTHIN
 - FO 149
- DBPROP_AUTH_USERID 148
- DBPROP_BLOCKINGSTORAGEOBJECTS 154
- DBPROP_BOOKMARKINFO 155
- DBPROP_BOOKMARKS 155
- DBPROP_BOOKMARKSKIPPED 155

- DBPROP_BOOKMARKTYPE 155
- DBPROP_CANFETCHBACKWARDS 156
- DBPROP_CANSROLLBACKWARDS 156
- DBPROP_CATALOGLOCATION 136
- DBPROP_CATALOGTERM 136
- DBPROP_CATALOGUSAGE 137
- DBPROP_COLUMNDEFINITION 137
- DBPROP_COMMITPRESERVE 157
- DBPROP_CONCATNULLBEHAVIOR 137
- DBPROP_CONNECTIONSTATUS 138
- DBPROP_CURRENTCATALOG 135
- DBPROP_DATASOURCENAME 138
- DBPROP_DATASOURCEREADONLY 138
- DBPROP_DBMSNAME 138
- DBPROP_DBMSVER 138
- DBPROP_DELAYSTORAGEOBJECTS 157
- DBPROP_DSOTHREADMODEL 138
- DBPROP_HETEROGENEOUSTABLES 139
- DBPROP_IDENTIFIERCASE 140
- DBPROP_IMMOBILEROWS 157
- DBPROP_INIT_DATASOURCE 149
- DBPROP_INIT_HWND 149
- DBPROP_INIT_MODE 149
- DBPROP_INIT_PROMPT 150
- DBPROP_INIT_PROVIDERSTRING 151
- DBPROP_INIT_CATALOG 149
- DBPROP_INIT_DATASOURCE 148
- DBPROP_INIT_HWND 148
- DBPROP_INIT_LCID 149
- DBPROP_INIT_MODE 148
- DBPROP_INIT_OLEDBSERVICES 150
- DBPROP_INIT_PROMPT 148
- DBPROP_INIT_PROVIDERSTRING 148
- DBPROP_LITERALBOOKMARKS 158
- DBPROP_LITERALIDENTITY 158
- DBPROP_LOCKMODE 158
- DBPROP_MAXINDEXSIZE 140
- DBPROP_MAXOPENROWS 158
- DBPROP_MAXPENDINGROWS 158
- DBPROP_MAXROWS 158
- DBPROP_MAXROWSIZE 140
- DBPROP_MAXROWSIZEINCLUDESBLOB 140
- DBPROP_MAXTABLEINSELECT 141
- DBPROP_MEMORYUSAGE 158
- DBPROP_MULTIPLEPARAMSETS 141
- DBPROP_MULTIPLERESULTS 141
- DBPROP_MULTIPLESTORAGEOBJECTS 141
- DBPROP_MULTITABLEUPDATE 141
- DBPROP_NULLCOLLATION 141
- DBPROP_OLEOBJECTS 142
- DBPROP_OPENROWSETSUPPORT 142
- DBPROP_ORDERBYCOLUMNSINSELECT 142
- DBPROP_OTHERINSERT 158
- DBPROP_OTHERUPDETEDELETE 159
- DBPROP_OUTPUTPARAMETERAVAILABILITY 142
- DBPROP_OWNINSERT 159
- DBPROP_OWNUPDETEDELETE 159
- DBPROP_PERSISTENTIDTYPE 143
- DBPROP_PREPAREABORTBEHAVIOR 143
- DBPROP_PREPARECOMMITBEHAVIOR 143
- DBPROP_PROCEDURETERM 143
- DBPROP_PROVIDERFRIENDLYNAME 143
- DBPROP_PROVIDERNAME 143
- DBPROP_PROVIDEROLEDBVER 144
- DBPROP_PROVIDERVER 144
- DBPROP_QUOTEDIDENTIFIERCASE 144
- DBPROP_REENTRANTEVENTS 159
- DBPROP_REMOVEDELETED 159
- DBPROP_REPORTMULTIPLECHANGES 159
- DBPROP_RETURNPENDINGINSERTS 159
- DBPROP_ROWRESTRICT 159
- DBPROP_ROWTHREADMODEL 159
- DBPROP_SCHEMATERM 144
- DBPROP_SCHEMAUSAGE 145
- DBPROP_SERVERCURSOR 160
- DBPROP_SERVERNAME 145
- DBPROP_SESS_AUTOCOMMITISOLEVELS 162
- DBPROP_SQLSUPPORT 146
- DBPROP_STRONGIDENTITY 160
- DBPROP_STRUCTUREDSTORAGE 146
- DBPROP_TABLETERM 148
- DBPROP_TRANSACTEDOBJECT 160
- DBPROP_UNIQUEROWS 160
- DBPROP_UPDATABILITY 161
- DBPROP_USERNAME 148
- Default Length for Long Data 119

- Defer Column 192
- defining a data source
 - using the ODBC Administrator 28
- DEGREES function 301
- Delay Storage Object Updates 192
- deletes, positional 92
- deleting
 - ADO client data sources 107
- developing ODBC applications 58
- DIFFERENCE function 299
- directory structure for SequeLink Java Client 210
- Distinguished Name identifier 119
- DistinguishedName attribute
 - ADO 121
 - ODBC 48
- Distributed Transaction Management 221
- documentation
 - order form 22
 - ordering printed books 21
- downloading applets, tips 283
- DPROP_GROUPBY 139
- DSN connection attribute 48
- dynamic properties
 - Command object 165
 - Connection object 173
 - Recordset object 191

E

- EnableDescribeParam 48
- enumerator object 128
- environment variables (UNIX) 44
- error handling
 - ADO Provider 201
 - JDBC Driver 281
 - ODBC Driver 74
- ErrorLookup object 126
- examples
 - connection pooling 219
 - creating and using JDBC data sources 218
 - odbc.ini file configured for Solaris 43

- Spy log 266
- URL (Spy) 265
- EXP function 301
- extended transaction functionality 132
- EXTRACT function 303

F

- Fetch Backward property for Recordset object 192
- fetching backward 156, 166
- fetching BigDecimal objects 286
- FetchNextOnly 49
- Field object in ADO 184
- file client data sources, configuring 32
- files in the SequeLink Java Client directory 210
- filters, Database Data Dictionary 82
- FixCharTrim 49
- FLOOR function 301
- forward-only result sets 277
- functions
 - cancelling in multithreaded applications 64, 276
 - date and time 302
 - numeric 300
 - scalar 290
 - string 298

G

- generating ODBC application IDs
 - automatically 73
- GetOutputParams 50
- GROUP BY Support 176

H

- header files and libraries, platform-specific 58
- Heterogeneous Table Support 176
- HLogonID 50
- Hold Rows 166, 192
- Host attribute 50, 119
- host password 50
- Host Password attribute 119
- HOURL function 303
- HPassword 50

I

- Identifier Case Sensitivity 176
- IFNULL function 305
- Immobile Rows 193
- improving character set conversion performance 285
- Informix 7 data types 311
- Informix 9 data types 312
- Initial Catalog 176
- initialization properties 148
- insensitive scrollable cursors 230
- INSERT function 299
- interfaces supported by the ADO Provider 124
- IOpenRowset 198
- IPersistFile 198
- IRowset interface 124
- IRowsetIdentity 151
- isolation levels 176, 320
- Isolation Retention 176

J

- JAR files 283
- Java 2 Platform 229
- Java Virtual Machine versions 274
- JDBC
 - compatibility 274
 - connection options 207
 - data types supported by Driver 309, 311, 312, 314, 317, 319
 - DB2 V6, V7 310
 - Microsoft SQL Server2000 315
 - function calls to execute store procedure 256
 - Optional Package functionality supported 273
 - support for 1.22 functionality 270
 - support for 2.0 functionality 271
- JDBC Driver
 - connection properties 224
 - error handling 281
 - registering with the JDBC Driver Manager 214
 - specifying application IDs 280
 - specifying connection URLs 215
 - threading for Driver 275
 - tracking calls 209
 - using data sources 223
 - using Distributed Transaction Management 221
 - using Spy with data sources 264
- JDBCSpy
 - log example 266
 - overview 209
 - registering the JDBC Driver 262
 - URL syntax 263
 - using with JDBC data sources 264
- JDBCSpy attributes 263
- JDBCTest
 - about 233
 - configuring 236
 - connecting with 236
 - executing

- batch on a prepared statement 251
- prepared statement 241
- simple Select statement 239
- making a JDBCTest connection 236
- retrieving database metadata 245
- scrolling through a result set 248
- starting 234
- starting using a Java Virtual Machine
 - other than the JDK 234
- tutorial 234
- using stored procedures with Oracle 254
- JNDI, using for naming databases 219

K

- keyset-driven cursors 66, 67

L

- LCASE function 299
- LDAP
 - property in ADO 121
 - specifying port for the listener 120
 - specifying the port for the listener 51
 - TCP/IP address of the LDAP server 50
 - UseLDAP attribute 52
- LEFT function 299
- LENGTH function 299
- libraries and header files, platform-specific 58
- Literal Bookmarks 193
- Literal Row Identity 167, 193
- loading the SequeLink JDBC Driver 214
- Locale Identifier 177
- LOCATE function 299
- Lock Mode 167, 193
- log example, Spy 266
- LOG function 301
- LOG10 function 301
- logging JDBC calls 261

- LogonID 51
- LTRIM function 299

M

- managing
 - connections to improve driver performance 90
 - JDBC data sources 218
 - retrieval of database meta-information 81
- mapping data types
 - ADO Provider 199
 - supported by DB2 V4, V5 309
 - supported by DB2 V6, V7 310
 - supported by Informix 7 311
 - supported by Informix 9 312
 - supported by Microsoft SQL Server 314
 - supported by Microsoft SQL Server2000 315
 - supported by Oracle7 317
 - supported by Oracle8 317
 - supported by Sybase 319
- Maximum Index Size 177
- Maximum Open Chapters 177
- Maximum Open Rows 167, 193
- Maximum Pending Rows 167, 194
- Maximum Row Size 177
- Maximum Row Size Includes BLOB 177
- Maximum Rows 167, 194
- Maximum Tables in SELECT 177
- Memory Usage 167, 194
- MERANT Configuration Manager
 - menu bar 100
 - overview 98
- meta-information
 - limiting amount to be retrieved 82
 - managing retrieval of 81
- methods
 - ADO Command object 164
 - ADO Recordset object 188
 - supported by ADO Connection object 172
 - supported by the Parameter object 185

- Microsoft SQL Server2000 data types 315
- MINUTE function 304
- MOD function 301
- Mode property 177
- modifying, ADO client data sources 106
- MONTH function 304
- MONTHNAME function 304
- Multiple Connections property 177
- Multiple Parameter Sets 177
- Multiple Results 178
- Multiple Results object 129
- Multiple Storage Objects 178
- Multi-Table Update 177
- Multi-table Update 177
- multithreaded applications, cancelling
 - functions 64, 276

N

- naming databases with JNDI 219
- NewPassword 51
- NOW function 304
- null arguments, impact on performance 78
- NULL Collation Order 178
- NULL Concatenation Behavior 178
- Number
 - Raising to power 301
 - Rounding 301
- numeric functions 300

O

- objects supported by the SequeLink ADO
 - Provider 124
- Objects Transacted 167, 194
- ODBC
 - connecting to a data source
 - using a connection string 45
 - connection string 45
 - functions that improve Driver
 - performance 87
 - Level 2.x API functions 60
 - Level 3.x API functions 62
 - optimizing performance 77
 - translators 31
 - ODBC Administrator
 - configuring file client data sources 32
 - configuring User and System client data
 - sources 28
 - starting 28
 - ODBC Driver
 - application IDs 72
 - cancelling functions in multithreaded
 - applications 64, 276
 - configuring file client data sources 32
 - connecting to a data source
 - using a logon dialog box 110
 - connecting to a data source using a logon
 - dialog box 36
 - connection attributes 47
 - connection options 27
 - error handling 74
 - ODBC API functions supported 60
 - overview 27
 - required libraries and header files 58
 - specification supported 27
 - threading 63
 - odbc.ini (UNIX)
 - sample odbc.ini file 43
 - using a centralized odbc.ini file 45
 - ODBCINI environment variable (UNIX) 44
 - ODBCTest 71
 - OLE DB
 - mapping methods to ADO 164
 - objects supported by the SequeLink ADO
 - Provider 124
 - property groups supported 134
 - schema rowsets supported 133
 - OLE DB Services 178
 - OLE DB Version 178
 - online manuals
 - order form 22
 - ordering hard-copy manuals 21

- Open Rowset Support 178
- Oracle7
 - data types 317
 - support for scrollable cursors (ODBC) 66
 - supports for scrollable cursors (JDBC) 278
 - using stored procedures with 68
- Oracle8
 - data types 317
 - support for scrollable cursors (ODBC) 66
 - supports for scrollable cursors (JDBC) 278
 - using stored procedures with 68
- ORDER BY Columns In Select List 178
- ordering printed books 21
- Others' Changes Visible 168, 194
- Others' Inserts Visible 168
- Others' Inserts Visible 194
- Output Parameter Availability 179
- Own Changes Visible 168, 194
- Own Inserts Visible 168, 194

P

- Parameter object in ADO 185, 186
- Pass By Ref Accessors 179
- Password
 - ODBC connection attribute 51
 - property for Connection object 179
- performance hints
 - avoiding the cursor library 89
 - committing data 91
- JDBC
 - fetching BigDecimal objects 286
 - improving character set conversion 285
 - reducing download time 283
- managing ODBC driver connections 90
- ODBC Driver
 - catalog functions 77
 - locking a row when isolation level is
 - Read committed 47
 - managing retrieval of database
 - meta-information 81
 - null arguments 78
 - SQLColumns 80
 - updating data in databases 92
 - reducing the size of retrieved data 83
 - using bound columns 85
 - using SQLExtendedFetch 86
 - persisting information 198
 - PI function 301
 - platform-specific header files and libraries 58
 - Port (ODBC connection attribute) 51
 - Port attribute 120
 - positional updates and deletes 92
 - POWER function 301
 - precedence of JDBC connection properties
 - 223
 - Prepare Abort Behavior 179
 - Prepare Commit Behavior 179
 - Preserve on Abort 168, 195
 - Preserve on Commit 169, 195
 - printed manuals
 - order form 22
 - ordering 21
 - Procedure Term 179
 - Prompt property for the Connection object
 - 179
 - properties
 - ADO Command object 165
 - ADO Connection object 172, 173
 - ADO Recordset object 189, 191
 - Data Source Information property group
 - 135
 - Data Source property group 135
 - displaying ADO Provider data source 101
 - Initialization property group 148
 - JDBC connection 223, 224
 - Rowset property group 151
 - Session property group 162
 - property groups supported 134
 - Provider Friendly Name 179, 180
 - Provider Name 180
 - provider string 117
 - Provider Version 180
 - pseudo-columns 92

Q

QUARTER function 304
 Quick Restart 169, 195
 Quoted Identifier Sensitivity 180

R

RADIANS function 301
 RAND function 301
 Read Only Data Source 180
 read-only result set 278
 Recordset object

- dynamic properties 191
- methods 188
- overview 188

 registering the JDBC Driver 214
 Remove Deleted Rows 169, 195
 renaming

- ADO client data sources 106

 REPEAT function 299
 REPLACE function 299
 Report Multiple Changes 169, 195
 result sets

- concurrency types 278
- types 277
- updatable not supported 279

 retrieving database meta-information 81
 Return Pending Inserts 169, 195
 reusing connections with connection

- pooling 219

 reverse scrolling 196
 RIGHT function 299
 ROUND function 301
 Row Privileges 169, 195
 Row Threading Model 169, 196
 rowset

- characteristics 151
- properties 151
- using 198

 Rowset Conversions on Command 180

Rowset object 129
 RTRIM function 300

S

saving current connection information 198
 scalar functions 290
 schema rowsets supported 133
 Schema Term 180
 Schema Usage 181
 Scroll Backward 170, 196
 scrollable cursors

- concurrency types 278
- insensitive 230
- limitations for ODBC Driver 67
- result set types 277
- using 67
- using with ODBC Driver 66
- using with the SequeLink JDBC Driver 279

 scrolling backward 156, 170
 scroll-insensitive result sets 277
 scroll-sensitive result sets 277
 SECOND function 304
 SequeLink ADO Provider

- ADO Command object 164
- data shaping 197
- data source information properties 135
- data source property group 135
- data types 314, 317
 - DB2 V4, V5 309
 - DB2 V6, V7 310
 - Informix 7 311
 - Informix 9 312
 - Microsoft SQL Server2000 315
 - Sybase 319
- error handling 201
- initialization properties 148
- objects and interfaces 124
- overview 97
- rowset properties 151
- rowset property group 151
- session property group 162

- SQL grammar supported 182
- Sequelink Java Client 210
- Sequelink JDBC Driver
 - configuring data sources 217
 - connection properties 223
 - creating data sources 218
 - JDBC 2.0 functionality 271
 - loading 214
 - on Java 2 platform 229
 - Optional Package functionality supported 273
 - overview 208
 - registering with JDBC Driver Manager 214
 - specification supported 208
 - specifying application IDs 280
 - specifying connection URLs 215
 - supported JDBC connection properties 224
 - using connection pooling 219
 - using Distributed Transaction Management 221
- Sequelink ODBC Driver
 - configuring file client data sources 32
 - connection dialogs 36
 - data types 314, 317
 - DB2 V4, V5 309
 - DB2 V6, V7 310
 - Informix 7 311
 - Informix 9 312
 - Microsoft SQL Server2000 315
 - Sybase 319
 - functions that improve performance 87
 - isolation levels 320
 - overview 27
 - required libraries and header files 58
 - threading 63
- Sequelink Proxy Server
 - about 208
- Server Cursor 170, 196
- Server Name 181
- Session object 130
- session property group 162
- sessions
 - maximum number supported 136
 - setting maximum number supported 173
- setting
 - maximum number of sessions 173
 - ODBCINI environment variable (UNIX) 44
- shell script, using to set environmental variables 44
- SIGN function 301
- SIN function 302
- Skip Deleted Bookmarks 196
- SLKStaticCursorLongColBuffLen 51
- SOUNDEX function 300
- SPACE function 300
- specification supported
 - JDBC 208, 270, 271
 - ODBC Driver 27
 - OLE DB 144
- specifying provider-specific logon information 175
- Spy
 - about 261
 - attributes 263
 - log example 266
 - options 263
 - registering the JDBC Driver 262
 - URL examples 265
 - URL syntax 263
 - using with JDBC data sources 264
- SQL Support 182
- SQLCancel, effect of threading 64, 276
- SQLColumns, performance implications 80
- SQLExtendedFetch 86
- SQLSetConnectAttr 74
- SQLSpecialColumns 92
- SQRT function 302
- SSL encryption
 - connection URL format 215
 - permissions required 230
- starting
 - Configuration Manager 101
 - JDBCTest 234
 - ODBC Administrator 28
- statement
 - executing prepared 241
 - executing Select with JDBCTest 239
- statement handles 91

- static cursors 66, 67
- Status bar (Configuration Manager) 101
- stored procedure
 - creating for Oracle 255
 - executing with JDBC function call 256
 - using with Oracle 68
- string
 - changing case of 299
 - functions 298
 - length of 299
 - removing blanks from 299, 300
 - returning substring of 300
- Strong Row Identity 170, 196
- Structured Storage 182
- Subquery Support 182
- SUBSTRING function 300
- SupportNet 23
- Sybase data types 319
- syntax for URLs for Spy 263

T

- table characteristics, determining 80
- Table Term 182
- TAN function 302
- TCommand 126
- TCP port 215
- TCP/IP port for Sequelink listener 51
- TDataSource 127
- Technical Support, contacting 23
- TEnumerator 128
- testing
 - ADO connections 109
 - JDBC connections 236
 - ODBC connections on Windows 42
- threading
 - ADO Provider 175
 - JDBC Driver 275
 - model 138
 - ODBC driver use 63
- time functions 302
- TIMESTAMPADD function 304

- TIMESTAMPDIFF function 304
- TMultipleResults 129
- tracking JDBC calls 209, 264
- Transaction DDL 183
- transaction isolation levels 176
- Transaction object 132
- Transaction Options object 132
- Translate button 31
- transliteration
 - Class 285
 - improving character set conversion performance 285
- TRowset 129
- TRUNCATE function 302
- TSession 130
- TTransaction 132
- TTransactionOptions 132
- tutorial, JDBC 234

U

- UCASE function 300
- Unicode, support for 199
- Unique Rows 170, 196
- UNIX
 - compiler requirements 59
 - configuring an ODBC data source 43
 - setting environmental variables 44
 - using a centralized odbc.ini file 45
- Updatability 170, 196
- updatable result set 278
- updates, positional 92
- updating opened rowset 151
- URL
 - examples (Spy) 265
- URL, format for connections (JDBC) 215
- Use Bookmarks 170, 196
- Use LDAP attribute (ADO) 121
- UseLDAP attribute (ODBC) 52
- USER function 305
- User Name property of Connection object 183

using

- centralized odbc.ini files (UNIX) 45
- keyset-driven cursors 66
- scrollable cursors 66
- scrollable cursors with the JDBC Driver 279
- SequeLink Java Client on a Java 2 Platform
229
- static cursors 66
- stored procedures with Oracle 68

W

- WEEK function 304
- window handle 149
- Window Handle (ADO Connection object
property) 183
- WorkArounds attribute 52

Y

- year format for conversions 199
- YEAR function 304

