SYBASE®

Connecting to Your Database

DataWindow .NET™

2.5

DOCUMENT ID: DC00038-01-0250-01

LAST REVISED: August 2007

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Contents

About This Book

Audience

This book is for anyone who uses the DataWindow Designer plug-in to connect to a database. It assumes that you are familiar with the database you are using and have installed the server and client software required to access the data.

How to use this book

This book describes how to connect to a database in the DataWindow Designer plug-in by using a standard or native database interface. The database interfaces described in this book are installed with both DataWindow Designer and DataWindow .NET™. This book gives procedures for preparing, defining, establishing, maintaining, and troubleshooting your database connections. For an overview of the steps you need to take, see "Basic connection procedure" on page 3.

Related documents

For information about database parameters and preferences, see the *Connection Reference* in the online Help for DataWindow Designer. For more information about database connectivity in DataWindow .NET, see the DataWindow .NET *Programmer's Guide*.

Other sources of information

Use the Sybase Getting Started CD, the SyBooks CD, and the Sybase Product Manuals Web site to learn more about your product:

- The Getting Started CD contains release bulletins and installation guides in PDF format, and may also contain other documents or updated information not included on the SyBooks CD. It is included with your software. To read or print documents on the Getting Started CD, you need Adobe Acrobat Reader, which you can download at no charge from the Adobe Web site using a link provided on the CD.
- The SyBooks CD contains product manuals and is included with your software. The Eclipse-based SyBooks browser allows you to access the manuals in an easy-to-use, HTML-based format.

  Some documentation may be provided in PDF format, which you can access through the PDF directory on the SyBooks CD. To read or print the PDF files, you need Adobe Acrobat Reader.

  Refer to the *SyBooks Installation Guide* on the Getting Started CD, or the *README.txt* file on the SyBooks CD for instructions on installing and starting SyBooks.

- The Sybase Product Manuals Web site is an online version of the SyBooks CD that you can access using a standard Web browser. In addition to product manuals, you will find links to EBFs/Maintenance, Technical Documents, Case Management, Solved Cases, newsgroups, and the Sybase Developer Network.

  To access the Sybase Product Manuals Web site, go to Product Manuals at http://www.sybase.com/support/manuals/.

Conventions

The formatting conventions used in this manual are:

Formatting example	Indicates
Retrieve and Update	When used in descriptive text, this font indicates: • Command, function, and method names • Keywords such as true, false, and null • Datatypes such as integer and char • Database column names such as emp_id and f_name • User-defined objects such as dw_emp or w_main
variable or *file name*	When used in descriptive text and syntax descriptions, oblique font indicates: • Variables, such as *myCounter* • Parts of input text that must be substituted, such as *pblname*.pbd • File and path names
File>Save	Menu names and menu items are displayed in plain text. The greater than symbol (>) shows you how to navigate menu selections. For example, File>Save indicates "select Save from the File menu."
`dw1.Update()`	Monospace font indicates: • Information that you enter in a dialog box or on a command line • Sample script fragments • Sample output fragments

If you need help

Each Sybase installation that has purchased a support contract has one or more designated people who are authorized to contact Sybase Technical Support. If you cannot resolve a problem using the manuals or online help, please have the designated person contact Sybase Technical Support or the Sybase subsidiary in your area.

PART 1

Introduction to Database Connections

This part introduces data connections in the DataWindow Designer plug-in. It helps you understand how to connect to a database in the Database painter.

CHAPTER 1

Understanding Data Connections

About this chapter

This chapter gives an overview of the concepts and procedures for connecting to a database in the DataWindow Designer plug-in.

Contents

How to find the information you need

This book describes how to connect to your database in the DataWindow Designer plug-in.

For information about connecting to a database in Visual Basic or C# code, see the *Programmer's Guide*.

Basic connection procedure

The following table gives an overview of the connection procedure and indicates where you can find detailed information about each step.

Table 1-1: Basic connection procedure

Step	Action	Details	See
1	(Optional) Get an introduction to database connections in DataWindow Designer	If necessary, learn more about how DataWindow Designer connects to a database	Chapter 1 (this chapter)
2	Prepare to use the data source or database before connecting to it for the first time in DataWindow Designer	Install the required network, database server, and database client software and verify that you can connect to the database	*For ODBC data sources*: Chapter 2, "Using the ODBC Interface" *For OLE DB data sources*: Chapter 3, "Using the OLE DB Interface" *For ADO.NET data sources*: Chapter 4, "Using the ADO.NET Interface" *For native database interfaces*: Chapter 5, "Using Native Database Interfaces"
3	Install the driver , database provider, or native database interface required to access your data	Install the ODBC driver, OLE DB data provider, ADO.NET data provider, or native database interface	
4	Define the data source (ODBC connections and some OLE DB drivers)	Create the required configuration for a data source accessed through ODBC	*For ODBC data sources*: Chapter 2, "Using the ODBC Interface"
5	Define the database interface	Create the database profile	*For ODBC data sources*: Chapter 2, "Using the ODBC Interface" *For OLE DB data sources*: Chapter 3, "Using the OLE DB Interface" *For ADO.NET data sources*: Chapter 4, "Using the ADO.NET Interface" *For native database interfaces*: Chapter 5, "Using Native Database Interfaces"
6	Connect to the data source or database	Access the data in DataWindow Designer	Chapter 11, "Managing Database Connections"
7	(Optional) Set additional connection parameters	If necessary, set database parameters and database preferences to fine-tune your database connection and take advantage of DBMS-specific features that your interface supports	*For procedures*: Chapter 12, "Setting Additional Connection Parameters" *For database parameter and preference descriptions*: online Help

Step	Action	Details	See
8	(Optional) Troubleshoot the data connection	If necessary, use the trace tools to troubleshoot problems with your connection	Chapter 13, "Troubleshooting Your Connection"

Accessing data in DataWindow Designer

There are several ways to access data in the DataWindow Designer plug-in:

- Through one of the standard database interfaces such as ODBC, ADO.NET, or OLE DB
- Through one of the native database interfaces

Standard database interfaces

A standard database interface communicates with a database through a standard-compliant driver (in the case of ODBC) or data provider (in the case of OLE DB and ADO.NET). The standard-compliant driver or data provider translates the abstract function calls defined by the standard's API into calls that are understood by a specific database. To use a standard interface, you need to install the standard's API and a suitable driver or data provider. Then, install the standard database interface you want to use to access your DBMS by selecting the interface in the DataWindow Designer Setup program.

DataWindow Designer currently supports the following standard interfaces:

- Open Database Connectivity (ODBC)
- Microsoft's Universal Data Access Component OLE DB
- Microsoft's ADO.NET

Native database interfaces

A native database interface communicates with a database through a direct connection. It communicates to a database using that database's native API.

To access data through one of the native database interfaces, you must first install the appropriate database software on the server and client workstations at your site. Then, install the native database interface that accesses your DBMS by selecting the interface in the DataWindow Designer Setup program.

For example, if you have the appropriate Sybase Adaptive Server® Enterprise server and client software installed, you can access the database by installing the Adaptive Server Enterprise database interface.

Making connections parallel

When you use DataWindow Designer to define or edit a DataWindow® object, make sure you specify the same connection parameters for the data source in both DataWindow Designer and DataWindow .NET. Otherwise, you will be unable to use the DataWindow object in your application.

Accessing the EAS Demo DB

DataWindow Designer includes a standalone SQL Anywhere® database called the EAS Demo DB. Unless you clear this option in the setup program, the database is installed automatically. You access tables in the EAS Demo DB when you use the DataWindow Designer samples.

A SQL Anywhere database is considered an ODBC data source, because you access it with the SQL Anywhere ODBC driver.

Using database profiles

What is a database profile?

A database profile is a named set of parameters stored in your system registry that defines a connection to a particular database in the DataWindow Designer plug-in. You must create a database profile for each data connection.

What you can do

Using database profiles is the easiest way to manage data connections in the DataWindow Designer plug-in. For example, you can:

- Select a database profile to connect to or switch between databases
- Edit a database profile to customize a connection
- Delete a database profile if you no longer need to access that data
- Import and export database profiles to share connection parameters quickly

For more information

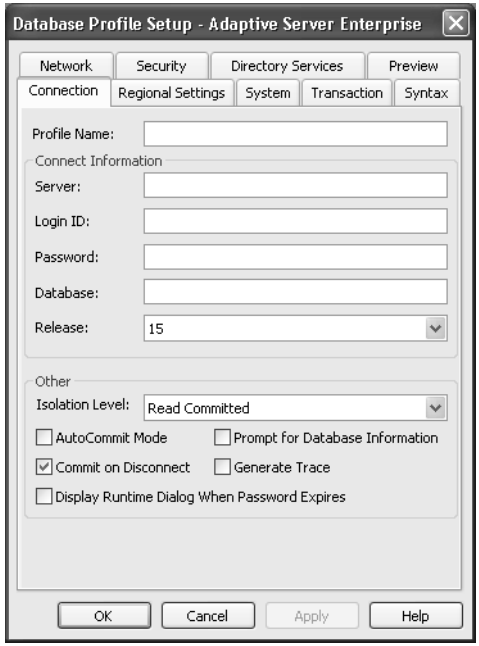
For instructions on using database profiles, see Chapter 11, “Managing Database Connections.”

About creating database profiles

You use the interface-specific Database Profile Setup dialog box, accessible from the Database painter, to create a database profile.

Database Profile Setup dialog box

Each database interface has its own Database Profile Setup dialog box where you can set interface-specific connection parameters. For example, if you install the Adaptive Server® Enterprise ASE interface and right-click the interface name in the Database painter Objects view and select New Profile, the Database Profile Setup - Adaptive Server Enterprise dialog box displays, containing settings for the connection options that apply to this interface.



The Database Profile Setup dialog box groups similar connection parameters on the same tab page and lets you easily set their values by using check boxes, drop-down lists, and text boxes. Basic (required) connection parameters are on the Connection tab page, and additional connection options are on the other tab pages.

Supplying sufficient information in the Database Profile Setup dialog box

For some database interfaces, you might not need to supply values for all boxes in the Database Profile Setup dialog box. If you supply the profile name and click OK, DataWindow Designer displays a series of dialog boxes to prompt you for additional information when you connect to the database.

This information can include:

User ID or login ID
Password or login password
Database name
Server name

For some databases, supplying only the profile name does not give DataWindow Designer enough information to prompt you for additional connection values. For these interfaces, you must supply values for all applicable boxes in the Database Profile Setup dialog box.

For information about the values you should supply for your connection, click Help in the Database Profile Setup dialog box for your interface.

Creating a database profile

To create a new database profile for a database interface, you must complete the Database Profile Setup dialog box for the interface you are using to access the database.

❖ **To create a database profile for a database interface:**

1 Select View>Database Painter from the Visual Studio menu bar.

The Database painter displays. The Objects view lists your installed database interfaces.

Where the interface list comes from
When you run the Setup program, it updates the Vendors list in the registry with the interfaces you install. The Database painter Objects view displays the same interfaces that appear in the Vendors list.

2 Highlight an interface name and select New profile from the pop-up menu.

The Database Profile Setup dialog box for the selected interface displays. For example, if you select the SYC interface, the Database Profile Setup - Adaptive Server Enterprise dialog box displays.

Client software and interface must be installed
To display the Database Profile Setup dialog box for your interface, the required client software and native database interface must be properly installed and configured. For specific instructions for your database interface, see the chapter on using the interface.

3 On the Connection tab page, type the profile name and supply values for any other basic parameters your interface requires to connect.

For information about the basic connection parameters for your interface and the values you should supply, click Help.

About the DBMS identifier
You do *not* need to specify the DBMS identifier in a database profile. When you create a new profile for any installed database interface, DataWindow Designer generates the correct DBMS connection syntax for you.

4 (Optional) On the other tab pages, supply values for any additional connection options to take advantage of DBMS-specific features that your interface supports.

For information about the additional connection parameters for your interface and the values you should supply, click Help.

5 (Optional) Click the Preview tab if you want to see the connection syntax that DataWindow Designer generates for each selected option.

You can copy the connection syntax from the Preview tab directly into your code.

For instructions on using the Preview tab to help you connect in a DataWindow .NET application, see the chapter on using transaction objects in the *Programmer's Guide*.

6 Click OK to save your changes and close the Database Profile Setup dialog box. (To save your changes on a particular tab page *without* closing the dialog box, click Apply.)

The database profile values are saved in the system registry in *HKEY_CURRENT_USER\Software\Sybase\DataWindowDesigner\2.5\DatabaseProfiles\PowerBuilder*.

You can look at the registry entry or export the profile as described in "Importing and exporting database profiles" on page 121 to see the settings you made. The NewLogic parameter is set to True by default. This setting specifies that the password is encrypted using Unicode encoding.

What to do next

For instructions on preparing to use and then defining an ODBC data source, see Chapter 2, “Using the ODBC Interface.”

For instructions on preparing to use and then defining an OLE DB data provider, see Chapter 3, “Using the OLE DB Interface.”

For instructions on preparing to use and then defining an ADO.NET data provider, see Chapter 4, “Using the ADO.NET Interface.”

For instructions on preparing to use and then defining a native database interface, see Chapter 5, “Using Native Database Interfaces.”

PART 2

Working with Standard Database Interfaces

This part describes how to set up and define database connections accessed through one of the standard database interfaces.

CHAPTER 2

Using the ODBC Interface

About this chapter

This chapter gives an introduction to the ODBC interface and then describes how to prepare to use the data source, how to define the data source, and how to define the ODBC database profile. It also describes how to use the Sybase SQL Anywhere ODBC driver.

Contents

For more information

This chapter gives general information about preparing to use and defining each ODBC data source. For more detailed information:

- Use the online Help provided by the driver vendor. This Help provides important details about using the data source.
- Check to see if there is a technical document that describes how to connect to your ODBC data source. Any updated information about connectivity issues is available from the Sybase Customer Service and Support Web site at http://www.sybase.com/suppot.

About the ODBC interface

You can access a wide variety of ODBC data sources in DataWindow Designer. This section describes what you need to know to use ODBC connections to access your data in DataWindow Designer.

What is ODBC?

The ODBC API

Open Database Connectivity (**ODBC**) is a standard application programming interface (API) developed by Microsoft. It allows a single application to access a variety of data sources for which ODBC-compliant drivers exist. The application uses Structured Query Language (SQL) as the standard data access language.

The ODBC API defines the following:

- A library of ODBC function calls that connect to the data source, execute SQL statements, and retrieve results
- A standard way to connect and log in to a DBMS
- SQL syntax based on the X/Open and SQL Access Group (SAG) CAE specification (1992)
- A standard representation for datatypes
- A standard set of error codes

Accessing ODBC data sources

Applications that provide an ODBC interface, like DataWindow Designer, can access data sources for which an ODBC driver exists. An **ODBC data source driver** is a dynamic link library (DLL) that implements ODBC function calls. The application invokes the ODBC driver to access a particular data source.

Accessing Unicode data

Using the ODBC interface, DataWindow Designer can connect, save, and retrieve data in both ANSI/DBCS and Unicode databases but does not convert data between Unicode and ANSI/DBCS. When character data or command text is sent to the database, DataWindow Designer sends a Unicode string. The driver must guarantee that the data is saved as Unicode data correctly. When DataWindow Designer retrieves character data, it assumes the data is Unicode.

A Unicode database is a database whose character set is set to a Unicode format, such as UTF-8, UTF-16, UCS-2, or UCS-4. All data must be in Unicode format, and any data saved to the database must be converted to Unicode data implicitly or explicitly.

A database that uses ANSI (or DBCS) as its character set might use special datatypes to store Unicode data. Columns with these datatypes can store *only* Unicode data. Any data saved into such a column must be converted to Unicode explicitly. This conversion must be handled by the database server or client.

Using ODBC in DataWindow Designer

What you can do

The following ODBC connectivity features are available in DataWindow Designer:

- Connect to a SQL Anywhere standalone database (including the EAS Demo DB) using the SQL Anywhere ODBC driver and the ODBC interface
- Create and delete local SQL Anywhere databases

  For instructions, see the *User's Guide*.
- Use Level 1 or later ODBC-compliant drivers to access your data

  See "Obtaining ODBC drivers" on page 20.
- Use Microsoft's ODBC Data Source Administrator to define ODBC data sources

  See "Defining ODBC data sources" on page 22.

Components of an ODBC connection

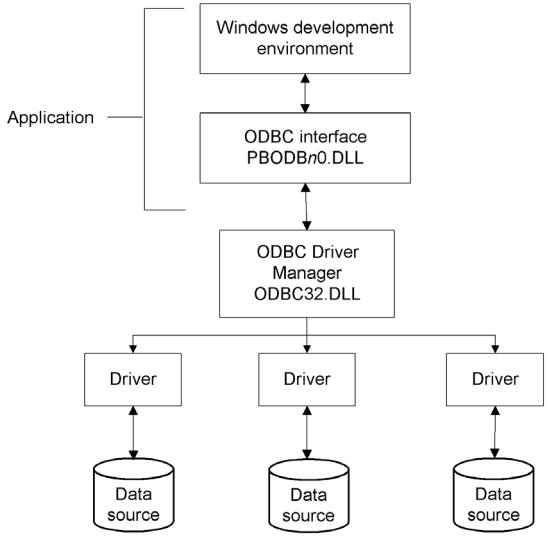
How an ODBC connection is made

When you access an ODBC data source in DataWindow Designer, your connection goes through several layers before reaching the data source. It is important to understand that each layer represents a separate component of the connection, and that each component might come from a different vendor.

Because ODBC is a standard API, DataWindow Designer uses the same interface to access every ODBC data source. As long as a driver is ODBC compliant, DataWindow Designer can access it through the ODBC interface to the ODBC Driver Manager. The plug-in and the ODBC interface work together as the application component.

Figure 2-1 shows the general components of an ODBC connection.

Figure 2-1: Components of an ODBC connection



Component descriptions

Table 2-1 gives the provider and a brief description of each ODBC component shown in the diagram.

Table 2-1: Provider and function of ODBC connection components

Component	Provider	What it does
Application	Sybase	Calls ODBC functions to submit SQL statements, catalog requests, and retrieve results from a data source. DataWindow Designer uses the same ODBC interface to access all ODBC data sources.
ODBC Driver Manager	Microsoft	Installs, loads, and unloads drivers for an application.
Driver	Driver vendor	Processes ODBC function calls, submits SQL requests to a particular data source, and returns results to an application. If necessary, translates an application's request so that it conforms to the SQL syntax supported by the back-end database. See "Types of ODBC drivers" next.
Data source	DBMS or database vendor	Stores and manages data for an application. Consists of the data to be accessed and its associated DBMS, operating system, and (if present) network software that accesses the DBMS.

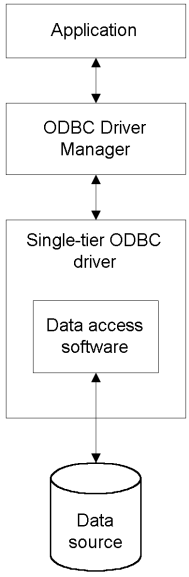
Types of ODBC drivers

When DataWindow Designer is connected to an ODBC data source, you might see messages from the ODBC driver that include the words *single-tier* or *multiple-tier*. These terms refer to the two types of drivers defined by the ODBC standard.

Single-tier driver

A single-tier ODBC driver processes both ODBC functions and SQL statements. In other words, a single-tier driver includes the data access software required to manage the data source file and catalog tables. An example of a single-tier ODBC driver is the Microsoft Access driver.

Figure 2-2: Single-tier ODBC driver

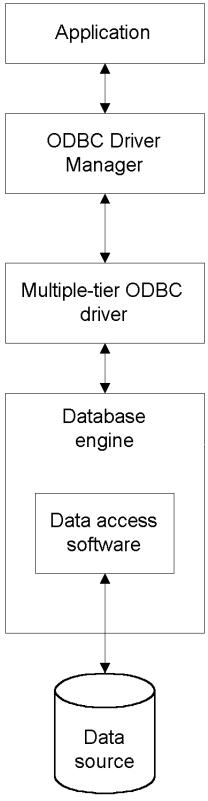


Multiple-tier driver

A multiple-tier ODBC driver processes ODBC functions, but sends SQL statements to the database engine for processing. Unlike the single-tier driver, a multiple-tier driver does not include the data access software required to manage the data directly.

An example of a multiple-tier ODBC driver is the Sybase SQL Anywhere driver.

Figure 2-3: Multi-tier ODBC driver



Ensuring the proper ODBC driver conformance levels

You can access data in DataWindow Designer with ODBC drivers obtained from vendors other than Sybase, such as DBMS vendors.

An ODBC driver obtained from another vendor must meet certain conformance requirements to ensure that it works properly with DataWindow Designer. This section describes how to make sure your driver meets these requirements.

What are ODBC conformance levels?

DataWindow Designer can access many data sources for which ODBC-compliant drivers exist. However, ODBC drivers manufactured by different vendors might vary widely in the functions they provide.

To ensure a standard level of compliance with the ODBC interface, and to provide a means by which application vendors can determine whether a specific driver provides the functions they need, ODBC defines conformance levels for drivers in two areas:

- **API** Deals with supported ODBC function calls
- **SQL grammar** Deals with supported SQL statements and SQL datatypes

API conformance levels

ODBC defines three API conformance levels, in order of increasing functionality:

- **Core** A set of core API functions that corresponds to the functions in the ISO Call Level Interface (CLI) and X/Open CLI specification
- **Level 1** Includes all Core API functions and several extended functions usually available in an OLTP relational DBMS
- **Level 2** Includes all Core and Level 1 API functions and additional extended functions

❖ **To ensure the proper ODBC driver API conformance level:**

- Sybase recommends that the ODBC drivers you use with DataWindow Designer meet *Level 1 or higher* API conformance requirements. However, DataWindow Designer might also work with drivers that meet Core level API conformance requirements.

SQL conformance levels

ODBC defines three SQL grammar conformance levels, in order of increasing functionality:

- **Minimum** A set of SQL statements and datatypes that meets a basic level of ODBC conformance
- **Core** Includes all Minimum SQL grammar and additional statements and datatypes that roughly correspond to the X/Open and SAG CAE specification (1992)
- **Extended** Includes all Minimum and Core SQL grammar and an extended set of statements and datatypes that support common DBMS extensions to SQL

❖ **To ensure the proper ODBC driver SQL conformance level:**

- Sybase recommends that the ODBC drivers you use with DataWindow Designer meet *Core or higher* SQL conformance requirements. However, DataWindow Designer might also work with drivers that meet Minimum level SQL conformance requirements.

Obtaining ODBC drivers

DataWindow Designer lets you access data with *any* Level 1 or higher ODBC-compliant drivers obtained from a vendor other than Sybase. In most cases, these drivers will work with DataWindow Designer.

Getting help with ODBC drivers

To ensure that you have up-to-date and accurate information about using your ODBC driver with DataWindow Designer, get help as needed by doing one or more of the following:

To get help on	Do this
Using the ODBC Data Source Administrator	Click the Help button on each tab.
Completing the ODBC setup dialog box for your driver	Click the Help button (if present) in the ODBC setup dialog box for your driver.
Using SQL Anywhere	See the SQL Anywhere documentation.
Using an ODBC driver obtained from a vendor other than Sybase	See the vendor's documentation for that driver.
Troubleshooting your ODBC connection	Check for a technical document that describes how to connect to your ODBC data source. Updated information about connectivity issues is available on the Sybase Customer Service and Support Web site at http://www.sybase.com/suppot.

Preparing ODBC data sources

The first step in connecting to an ODBC data source is preparing the data source. This ensures that you are able to connect to the data source and use your data in DataWindow Designer.

You prepare to use a data source *outside* DataWindow Designer *before* you start the product, define the data source, and connect to it. The requirements differ for each data source, but in general, preparing to use a data source involves the following steps.

❖ **To prepare to use an ODBC data source with DataWindow Designer:**

1. If network software is required to access the data source, make sure it is properly installed and configured at your site and on the client workstation.
2. If database software is required, make sure it is properly installed and configured on your computer or network server.
3. Make sure the required data files are present on your computer or network server.
4. Make sure the names of tables and columns you want to access follow standard SQL naming conventions.

   Avoid using blank spaces or database-specific reserved words in table and column names. Be aware of the case-sensitivity options of the DBMS. It is safest to use all uppercase characters when naming tables and columns that you want to access in DataWindow Designer.
5. If your database requires it, make sure the tables you want to access have unique indexes.
6. Install both of the following using the DataWindow Designer Setup program:
   - The ODBC driver that accesses your data source
   - The ODBC interface

Defining ODBC data sources

Each ODBC data source requires a corresponding ODBC driver to access it. When you define an ODBC data source, you provide information about the data source that the driver requires in order to connect to it. Defining an ODBC data source is often called configuring the data source.

After you prepare to use the data source, you must define it using Microsoft's ODBC Data Source Administrator utility. This utility can be accessed from the Control Panel in Windows or DataWindow Designer's Database painter.

The rest of this section describes what you need to know to define an ODBC data source in order to access it in DataWindow Designer.

Making connections parallel

When you use DataWindow Designer to define or edit a DataWindow object, make sure you specify the same connection parameters for the data source in both DataWindow Designer and DataWindow .NET. Otherwise, you will be unable to use the DataWindow object in your application.

How DataWindow Designer accesses the data source

When you access an ODBC data source in DataWindow Designer, there are several initialization files and registry entries on your computer that work with the ODBC interface and driver to make the connection.

PBODB110 initialization file

Contents

PBODB110.INI is installed in both the DataWindow .NET and DataWindow Designer directories. DataWindow Designer and DataWindow .NET use *PBODB110.INI* to maintain access to extended functionality in the back-end DBMS, for which ODBC does not provide an API call. Examples of extended functionality are SQL syntax or DBMS-specific function calls.

Editing

In most cases, you do not need to edit *PBODB110.INI*. In certain situations, however, you might need to add functions to *PBODB110.INI* for your back-end DBMS.

For instructions, see the Appendix, "Adding Functions to the PBODB110 Initialization File."

ODBCINST registry entries

Contents

The ODBCINST initialization information is located in the *HKEY_LOCAL_MACHINE\SOFTWARE\ODBC\ODBCINST.INI* registry key. When you install an ODBC-compliant driver, *ODBCINST.INI* is automatically updated with a description of the driver.

This description includes:

- The DBMS or data source associated with the driver
- The drive and directory of the driver and setup DLLs (for some data sources, the driver and setup DLLs are the same)
- Other driver-specific connection parameters

Editing

You do *not* need to edit the registry key directly to modify connection information. If your driver uses the information in the *ODBCINST.INI* registry key, the key is automatically updated when you install the driver. This is true whether the driver is supplied by Sybase or another vendor.

ODBC registry entries

Contents

ODBC initialization information is located in the *HKEY_CURRENT_USER\SOFTWARE\ODBC\ODBC.INI* registry key. When you define a data source for a particular ODBC driver, the driver writes the values you specify in the ODBC setup dialog box to the *ODBC.INI* registry key.

The *ODBC.INI* key contains subkeys named for each defined data source. Each subkey contains the values specified for that data source in the ODBC setup dialog box. The values might vary for each data source but generally include the following:

- Database
- Driver
- Optional description
- DBMS-specific connection parameters

Editing

Do *not* edit the *ODBC* subkey directly to modify connection information. Instead, use a tool designed to define ODBC data sources and the ODBC configuration automatically, such as the ODBC Data Source Administrator.

Database profiles registry entry

Contents

Database profiles for all data sources are stored in the registry in *HKEY_CURRENT_USER\SOFTWARE\Sybase\DataWindowDesigner\2.5\DatabaseProfiles*.

Editing

You should *not* need to edit the profiles directly to modify connection information. These files are updated automatically when DataWindow Designer creates the database profile as part of the ODBC data source definition.

You can also edit the profile in the Database Profile Setup dialog box or complete the Database Preferences dialog box in DataWindow Designer to specify other connection parameters stored in the registry. (For instructions, see Chapter 12, "Setting Additional Connection Parameters.")

Example

The following example shows a portion of the database profile for an EAS Demo DB data source:

```
DBMS=ODBC
DbParameter=ConnectString='DSN=EAS Demo
DB;UID=dba;PWD=00c61737'
Prompt=0
```

This registry entry example shows the two most important values in a database profile for an ODBC data source:

- **DBMS** The DBMS value (ODBC) indicates that you are using the ODBC interface to connect to the data source.
- **DbParameter** The ConnectString DbParameter parameter controls your ODBC data source connection. The connect string *must* specify the DSN (data source name) value, which tells ODBC which data source you want to access. When you select a database profile to connect to a data source, ODBC looks in the ODBC.INI registry key for a subkey that corresponds to the data source name in your profile. ODBC then uses the information in the subkey to load the required libraries to connect to the data source. The connect string can also contain the UID (user ID) and PWD (password) values needed to access the data source.

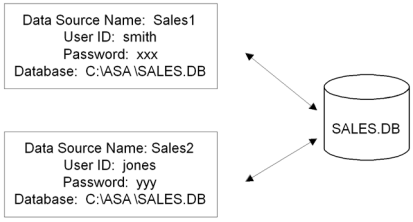
Defining multiple data sources for the same data

When you define an ODBC data source in DataWindow Designer, each data source name must be unique. You can, however, define multiple data sources that access the same data, as long as the data sources have unique names.

For example, assume that your data source is a SQL Anywhere database located in *C:\SQL Anywhere\SALES.DB*. Depending on your application, you might want to specify different sets of connection parameters for accessing the database, such as different passwords and user IDs.

To do this, you can define two ODBC data sources named Sales1 and Sales2 that specify the same database (*C:\SQL Anywhere\SALES.DB*) but use different user IDs and passwords. When you connect to the data source using a profile created for either of these data sources, you are using different connection parameters to access the same data.

Figure 2-4: Using two data sources to access a database



Displaying Help for ODBC drivers

The online Help for ODBC drivers in DataWindow Designer is provided by the driver vendors. It gives help on:

- Completing the ODBC setup dialog box to define the data source
- Using the ODBC driver to access the data source

Help for any ODBC driver

Use the following procedure to display vendor-supplied Help when you are in the ODBC setup dialog box for ODBC drivers.

❖ **To display Help for any ODBC driver:**

- Click the Help button in the ODBC setup dialog box for your driver.

  A Help window displays, describing features in the setup dialog box.

Selecting an ODBC translator

What is an ODBC translator?

Some ODBC drivers allow you to specify a translator when you define the data source. An **ODBC translator** is a DLL that translates data passing between an application and a data source. Typically, translators are used to translate data from one character set to another.

What you do

Follow these steps to select a translator for your ODBC driver.

❖ **To select a translator when using an ODBC driver:**

1 In the ODBC setup dialog box for your driver, display the Select Translator dialog box.

The way you display the Select Translator dialog box depends on the driver and Windows platform you are using. Click Help in your driver's setup dialog box for instructions on displaying the Select Translator dialog box.

In the Select Translator dialog box, the translators listed are determined by the values in your *ODBCINST.INI* registry key.

2 From the Installed Translators list, select a translator to use.

If you need help using the Select Translator dialog box, click Help.

3 Click OK.

The Select Translator dialog box closes and the driver performs the translation.

Defining the ODBC interface

To define a connection through the ODBC interface, you must create a database profile by supplying values for at least the basic connection parameters in the Database Profile Setup - ODBC dialog box. You can then select this profile at any time to connect to your data source in the plug-in.

For information on how to define a database profile, see "Using database profiles" on page 6.

Sybase SQL Anywhere

This section describes how to prepare and define a Sybase SQL Anywhere data source in order to connect to it using the SQL Anywhere ODBC driver.

Name change
For versions 6 through 9, the SQL Anywhere database server was called Adaptive Server® Anywhere (ASA).

SQL Anywhere includes two database servers—a personal database server and a network database server. For information about using Sybase SQL Anywhere, see the SQL Anywhere documentation.

Supported versions for SQL Anywhere

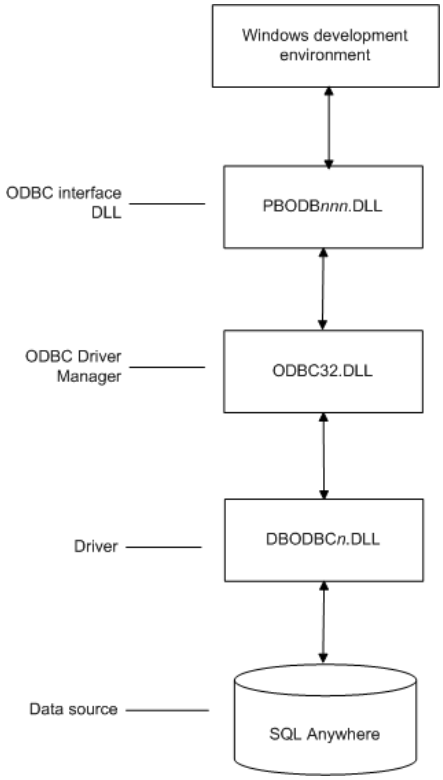
The SQL Anywhere ODBC driver supports connection to local and remote databases created with the following:

- DataWindow Designer running on your computer
- SQL Anywhere version 10.x
- ASA version 9.x
- ASA version 8.x
- ASA version 7.x
- ASA version 6.x
- SQL Anywhere version 5.x

Basic software components for SQL Anywhere

Figure 2-5 shows the basic software components required to connect to a SQL Anywhere data source in DataWindow Designer.

Figure 2-5: Components of a SQL Anywhere connection



Preparing to use the SQL Anywhere data source

Before you define and connect to a SQL Anywhere data source in DataWindow Designer, follow these steps to prepare the data source.

❖ **To prepare a SQL Anywhere data source:**

1 Make sure the database file for the SQL Anywhere data source already exists. You can create a new database by:

- Launching the Create SQL Anywhere Database utility. You can access this utility from the Utilities folder for the ODBC interface in the Database profile or Database painter when DataWindow Designer is running on your computer.

This method creates a local SQL Anywhere database on your computer, and also creates the data source definition and database profile for this connection. (For instructions, see the *User's Guide*.)

- Creating the database some other way, such as with DataWindow Designer running on another user's computer or by using SQL Anywhere outside DataWindow Designer. (For instructions, see the SQL Anywhere documentation.)

2 Make sure you have the log file associated with the SQL Anywhere database so that you can fully recover the database if it becomes corrupted.

If the log file for the SQL Anywhere database does not exist, the SQL Anywhere database engine creates it. However, if you are copying or moving a database from another computer or directory, you should copy or move the log file with it.

Defining the SQL Anywhere data source

When you create a local SQL Anywhere database, DataWindow Designer automatically creates the data source definition and database profile for you. Therefore, you need only use the following procedure to define a SQL Anywhere data source when you want to access a SQL Anywhere database not created using DataWindow Designer on your computer.

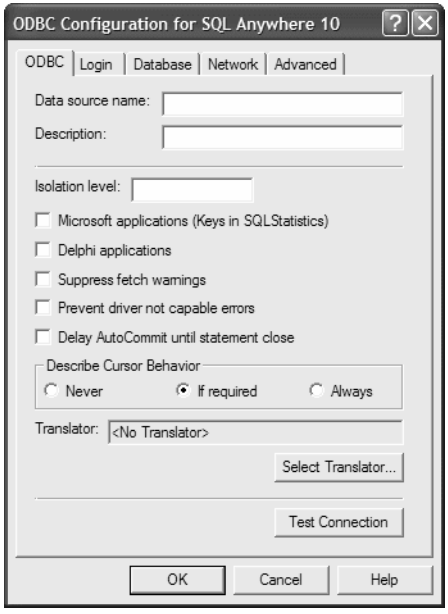
❖ **To define a SQL Anywhere data source for the SQL Anywhere driver:**

1 Select Create ODBC Data Source from the list of ODBC utilities in the Database Profiles dialog box or the Database painter.

2 Select User Data Source and click Next.

3 On the Create New Data Source page, select the SQL Anywhere driver and click Finish.

The ODBC Configuration for SQL Anywhere dialog box displays:



4 You must supply the following values:

- Data source name on the ODBC tab page
- User ID and password on the Login tab page
- Database file on the Database tab page

Use the Help button to get information about fields in the dialog box.

5 (Optional) To select an ODBC translator to translate your data from one character set to another, click the Select button on the ODBC tab.

See “Selecting an ODBC translator” on page 26.

6 Click OK to save the data source definition.

Specifying a Start Line value

When the SQL Anywhere ODBC driver cannot find a running personal or network database server using the PATH variable and Database Name setting, it uses the commands specified in the Start Line field to start the database servers.

Specify one of the following commands in the Start Line field on the Database tab, where *n* is the version of SQL Anywhere you are using.

Specify this command	To
dbeng*n*.exe	Start the personal database server and the database specified in the Database File box
rteng*n*.exe	Start the restricted runtime database server and the database specified in the Database File box

For information on completing the ODBC Configuration For SQL Anywhere dialog box, see the SQL Anywhere documentation.

Support for Transact-SQL special timestamp columns

When you work with a SQL Anywhere table in the DataWindow or Database painter, the default behavior is to treat any column named timestamp as a SQL Anywhere Transact-SQL special timestamp column.

Creating special timestamp columns

You can create a Transact-SQL special timestamp column in a SQL Anywhere table.

❖ **To create a Transact-SQL special timestamp column in a SQL Anywhere table in DataWindow Designer:**

1. Give the name timestamp to any column having a timestamp datatype that you want treated as a Transact-SQL special timestamp column. Do this in one of the following ways:
   - In the painter – Select timestamp as the column name. (For instructions, see the *User's Guide*.)
   - In a SQL CREATE TABLE statement – Follow the "CREATE TABLE example" next.
2. Specify *timestamp* as the default value for the column. Do this in one of the following ways:
   - In the painter – Select timestamp as the default value for the column. (For instructions, see the *User's Guide*.)
   - In a SQL CREATE TABLE statement – Follow the "CREATE TABLE example" next.

3 If you are working with the table in the Data Pipeline painter, select the initial value exclude for the special timestamp column from the drop-down list in the Initial Value column of the workspace.

You must select exclude as the initial value to exclude the special timestamp column from INSERT or UPDATE statements.

For instructions, see the *User's Guide*.

CREATE TABLE example

The following CREATE TABLE statement defines a SQL Anywhere table named timesheet containing three columns: employee_ID (integer datatype), hours (decimal datatype), and timestamp (timestamp datatype and timestamp default value):

```
CREATE TABLE timesheet (
   employee_ID INTEGER,
   hours DECIMAL,
   timestamp TIMESTAMP default timestamp )
```

Not using special timestamp columns

If you want to change the default behavior, you can specify that DataWindow Designer *not* treat SQL Anywhere columns named *timestamp* as Transact-SQL special timestamp columns.

❖ **To specify that DataWindow Designer *not* treat columns named *timestamp* as a Transact-SQL special timestamp column:**

- Edit the Sybase SQL Anywhere section of the PBODB110 initialization file to change the value of SQLSrvrTSName from `'Yes'` to `'No'`.

  After making changes in the initialization file, you must reconnect to the database to have them take effect. See the Appendix, "Adding Functions to the PBODB110 Initialization File."

What to do next

For instructions on connecting to the ODBC data source, see "Connecting to a database" on page 119.

CHAPTER 3

Using the OLE DB Interface

About this chapter

This chapter describes the OLE DB interface and explains how to prepare to use this interface and how to define the OLE DB database profile.

Contents

For more information

This chapter gives general information about using the OLE DB interface. For more detailed information:

- See the Data Access section in the Microsoft MSDN library at http://msdn2.microsoft.com/en-us/library/default.aspx.
- Use the online Help provided by the data provider vendor.
- Check to see if there is a technical document that describes how to connect to your OLE DB data provider. Any updated information about connectivity issues is available from the Sybase Customer Service and Support Web site at http://www.sybase.com/support.

About the OLE DB interface

You can access a wide variety of data through OLE DB data providers in DataWindow Designer. This section describes what you need to know to use OLE DB connections to access your data in DataWindow Designer.

What is OLE DB?

OLE DB API

OLE DB is a standard application programming interface (API) developed by Microsoft. It is a component of Microsoft's Data Access Components software. OLE DB allows an application to access a variety of data for which OLE DB data providers exist. It provides an application with uniform access to data stored in diverse formats, such as indexed-sequential files like Btrieve, personal databases like Paradox, productivity tools such as spreadsheets and electronic mail, and SQL-based DBMSs.

The OLE DB interface supports direct connections to SQL-based databases.

Accessing data through OLE DB

Applications like DataWindow Designer that provide an OLE DB interface can access data for which an OLE DB data provider exists. An **OLE DB data provider** is a dynamic link library (DLL) that implements OLE DB function calls to access a particular data source.

The DataWindow Designer OLE DB interface can connect to any OLE DB data provider that supports the OLE DB object interfaces listed in Table 3-1. An OLE DB data provider must support these interfaces in order to adhere to the Microsoft OLE DB 2.0 specification.

Table 3-1: Required OLE DB interfaces

IAccessor	IDBInitialize
IColumnsInfo	IDBProperties
ICommand	IOpenRowset
ICommandProperties	IRowset
ICommandText	IRowsetInfo
IDBCreateCommand	IDBSchemaRowset
IDBCreateSession	ISourcesRowset

In addition to the required OLE DB interfaces, DataWindow Designer also uses the OLE DB interfaces listed in Table 3-2 to provide further functionality.

Table 3-2: Additional OLE DB interfaces

OLE DB interface	Use in DataWindow Designer
ICommandPrepare	Preparing commands and retrieving column information.
IDBInfo	Querying the data provider for its properties. If this interface is not supported, database connections might fail.
IDBCommandWithParameters	Querying the data provider for parameters.
IErrorInfo	Providing error information.
IErrorRecords	Providing error information.
IIndexDefinition	Creating indexes for the extended attribute system tables. Also creating indexes in the Database painter. If this interface is not supported, DataWindow Designer looks for index definition syntax in the *pbodb110.ini* file.
IMultipleResults	Providing information.
IRowsetChange	Populating the extended attribute system tables when they are created. Also, for updating blobs.
IRowsetUpdate	Creating the extended attribute system tables.
ISQLErrorInfo	Providing error information.
ISupportErrorInfo	Providing error information.
ITableDefinition	Creating the extended attribute system tables and also for creating tables in the Database painter. If this interface is not supported, the following behavior results: • DataWindow Designer looks for table definition syntax in the *pbodb110.ini* file • DataWindow Designer catalog tables cannot be used • DDL and DML operations, like modifying columns or editing data in the database painter, do not function properly
ITransactionLocal	Supporting transactions. If this interface is not supported, DataWindow Designer defaults to AutoCommit mode.

Accessing Unicode data

Using the OLE DB interface, DataWindow Designer can connect, save, and retrieve data in both ANSI/DBCS and Unicode databases but does not convert data between Unicode and ANSI/DBCS. When character data or command text is sent to the database, DataWindow Designer sends a Unicode string. The data provider must guarantee that the data is saved as Unicode data correctly. When DataWindow Designer retrieves character data, it assumes the data is Unicode.

A Unicode database is a database whose character set is set to a Unicode format, such as UTF-8, UTF-16, UCS-2, or UCS-4. All data must be in Unicode format, and any data saved to the database must be converted to Unicode data implicitly or explicitly.

A database that uses ANSI (or DBCS) as its character set might use special datatypes to store Unicode data. Columns with these datatypes can store *only* Unicode data. Any data saved into such a column must be converted to Unicode explicitly. This conversion must be handled by the database server or client.

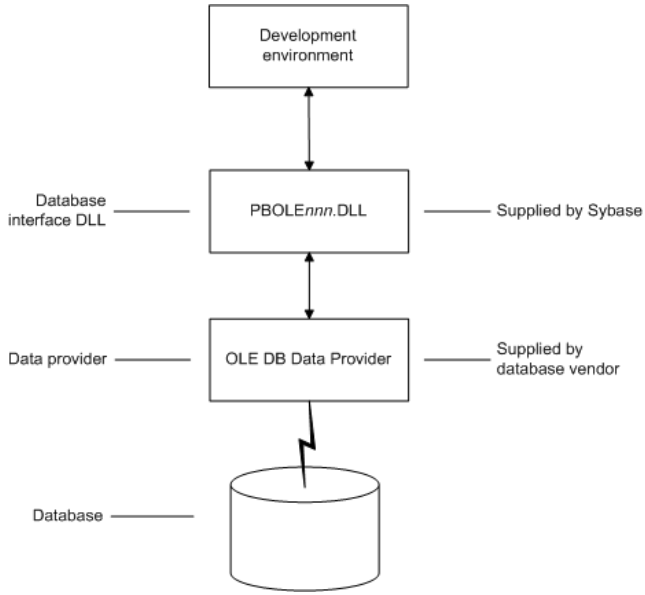
Components of an OLE DB connection

When you access an OLE DB data provider in DataWindow Designer, your connection goes through several layers before reaching the data provider. It is important to understand that each layer represents a separate component of the connection, and that each component might come from a different vendor.

Because OLE DB is a standard API, DataWindow Designer uses the same interface to access every OLE DB data provider. As long as an OLE DB data provider supports the object interfaces required by DataWindow Designer, DataWindow Designer can access it through the OLE DB interface.

Figure 3-1 shows the general components of a OLE DB connection.

Figure 3-1: Components of an OLE DB connection



Obtaining OLE DB data providers

DataWindow Designer lets you access data with *any* OLE DB data provider obtained from a vendor other than Sybase if that data provider supports the OLE DB object interfaces required by DataWindow Designer. In most cases, these drivers work with DataWindow Designer. However, Sybase might not have tested the drivers to verify this.

Supported versions for OLE DB

The OLE DB interface uses a DLL named *PBOLE110.DLL* to access a database through an OLE DB data provider.

Required OLE DB version
To use the OLE DB interface to access an OLE DB database, you must connect through an OLE DB data provider that supports OLE DB version 2.0 or later. For information on OLE DB specifications, see the Microsoft documentation at http://msdn2.microsoft.com/en-us/library/default.aspx.

Preparing to use the OLE DB interface

Before you define the interface and connect to a data provider through OLE DB:

1 Install and configure the database server, network, and client software.

2 Install the OLE DB interface and the OLE DB data provider that accesses your data source.

3 Install Microsoft's Data Access Components software on your machine.

4 If required, define the OLE DB data source.

Step 1: Install and configure the data server

You must install and configure the database server and install the network software and client software.

❖ **To install and configure the database server, network, and client software:**

1 Make sure the appropriate database software is installed and running on its server.

You must obtain the database server software from your database vendor. For installation instructions, see your database vendor's documentation.

2 Make sure the required network software (such as TCP/IP) is installed and running on your computer and is properly configured so that you can connect to the data server at your site. You must install the network communication driver that supports the network protocol and operating system platform you are using.

For installation and configuration instructions, see your network or data source administrator.

3 If required, install the appropriate client software on each client computer on which DataWindow Designer is installed.

Client software requirements
To determine client software requirements, see your database vendor's documentation.

Step 2: Install the OLE DB interface and data provider

In the DataWindow Designer Setup program, select the Custom install and select the OLE DB provider that accesses your database. You can install one or more of the OLE DB data providers shipped with DataWindow Designer, or you can install data providers from another vendor later.

Step 3: Install the Microsoft Data Access Components software

The DataWindow Designer OLE DB interface requires the functionality of the Microsoft Data Access Components (MDAC) version 2.8 or higher software. Version 2.8 is distributed with Windows XP Service Pack 2 and Windows Server 2003.

To check the version of MDAC on your computer, you can download and run the MDAC Component Checker utility from the MDAC Downloads page at http://msdn2.microsoft.com/en-us/data/aa937730.aspx.

On the Windows Vista operating system, Windows Data Access Components (DAC) 6.0 includes some changes to work with Vista but is otherwise functionally equivalent to MDAC 2.8.

OLE DB data providers installed with MDAC
Several Microsoft OLE DB data providers are automatically installed with MDAC, including the providers for SQL Server (SQLOLEDB) and ODBC (MSDASQL).

Step 4: Define the OLE DB data source

Once the OLE DB data provider is installed, you might have to define the OLE DB data source the data provider will access. How you define the data source depends on the OLE DB data provider you are using and the vendor who provided it.

If you are connecting to an ODBC data provider (such as Microsoft's OLE DB Provider for ODBC), you must define the ODBC data source as you would if you were using a direct ODBC connection. To define an ODBC data source, use Microsoft's ODBC Data Source Administrator. You can access this utility from the Control Panel in Windows or from the Database painter in DataWindow Designer.

Defining the OLE DB interface

Using the OLE DB Database Profile Setup

To define a connection through the OLE DB interface, you must create a database profile by supplying values for at least the basic connection parameters in the Database Profile Setup – OLE DB dialog box. You can then select this profile anytime to connect to your data in the plug-in.

For information on how to define a database profile, see "Using database profiles" on page 6.

Specifying connection parameters

You must supply values for the Provider and Data Source connection parameters. Select a data provider from the list of installed data providers in the Provider drop-down list. The Data Source value varies depending on the type of data source connection you are making. For example:

- If you are using Microsoft's OLE DB Provider for ODBC to connect to the EAS Demo DB, you select MSDASQL as the Provider value and enter the actual ODBC data source name (for example, EAS Demo DB) as the Data Source value.
- If you are using Microsoft's OLE DB Provider for SQL Server, you select SQLOLEDB as the Provider value and enter the actual server name as the Data Source value. You must also use the Extended Properties field to provide the database name (for example, Database=Pubs) since you can have multiple instances of a database.

Using the Data Link API

The Data Link option allows you to access Microsoft's Data Link API, which allows you to define a file or use an existing file that contains your OLE DB connection information. A Data Link file is identified with the suffix *.udl*. If you use a Data Link file to connect to your data source, all other settings you make in the OLE DB Database Profile Setup dialog box are ignored.

To launch this option, select the File Name check box on the Connection tab and double-click on the button next to the File Name box. (You can also launch the Data Link API in the Database painter by double-clicking on the Manage Data Links utility included with the OLE DB interface in the list of Installed Database Interfaces.)

For more information on using the Data Link API, see Microsoft's Universal Data Access Web site at http://msdn2.microsoft.com/en-us/data/default.aspx.

CHAPTER 4

Using the ADO.NET Interface

About this chapter

This chapter describes the ADO.NET interface and explains how to prepare to use this interface and how to define an ADO.NET database profile.

Contents

For more information

This chapter gives general information about using the ADO.NET interface. For more detailed information:

- See the Data Access and .NET development sections in the Microsoft MSDN library at http://msdn2.microsoft.com/en-us/data/default.aspx.
- Use the online Help provided by the data provider vendor.
- Check to see if there is a technical document that describes how to connect to your ADO.NET data provider. Any updated information about connectivity issues is available from the Sybase Customer Service and Support Web site at http://www.sybase.com/support.

About ADO.NET

ADO.NET is a set of technologies that provides native access to data in the Microsoft .NET Framework. It is designed to support an n-tier programming environment and to handle a disconnected data architecture. ADO.NET is tightly integrated with XML and uses a common data representation that can combine data from disparate sources, including XML.

One of the major components of ADO.NET is the .NET Framework data provider, which connects to a database, executes commands, and retrieves results.

Microsoft provides .NET Framework data providers for SQL Server and OLE DB with the .NET Framework, and data providers for ODBC and Oracle can be downloaded from the Microsoft Web site. You can also obtain .NET Framework data providers from other vendors, such as the .NET Framework Data Provider for Adaptive Server Enterprise from Sybase.

To connect to a database using the DataWindow Designer ADO.NET database interface, you must use a .NET Framework data provider.

Accessing Unicode data

Using the ADO.NET interface, DataWindow Designer can connect, save, and retrieve data in both ANSI/DBCS and Unicode databases but does not convert data between Unicode and ANSI/DBCS. When character data or command text is sent to the database, DataWindow Designer sends a Unicode string. The data provider must guarantee that the data is saved as Unicode data correctly. When DataWindow Designer retrieves character data, it assumes the data is Unicode.

A Unicode database is a database whose character set is set to a Unicode format, such as UTF-8, UTF-16, UCS-2, or UCS-4. All data must be in Unicode format, and any data saved to the database must be converted to Unicode data implicitly or explicitly.

A database that uses ANSI (or DBCS) as its character set might use special datatypes to store Unicode data. Columns with these datatypes can store *only* Unicode data. Any data saved into such a column must be converted to Unicode explicitly. This conversion must be handled by the database server or client.

About the DataWindow Designer ADO.NET database interface

You can use the DataWindow Designer ADO.NET database interface to connect to a data source such as Adaptive Server® Enterprise, Oracle, and Microsoft SQL Server, as well as to data sources exposed through OLE DB and XML, in much the same way as you use the DataWindow Designer ODBC and OLE DB database interfaces.

Performance
You might experience better performance if you use a native database interface. The primary purpose of the ADO.NET interface is to support shared connections with other database constructs such as the .NET DataGrid in Sybase DataWindow .NET.

Components of an ADO.NET connection

When you access a database using ADO.NET in DataWindow Designer, your connection goes through several layers before reaching the database. It is important to understand that each layer represents a separate component of the connection, and that components might come from different vendors.

The DataWindow Designer ADO.NET interface consists of a driver (*pbado110.dll*) and a server (either *Sybase.DataWindow.Db.dll* or *Sybase.DataWindow.DbExt.dll*). The server has dependencies on a file called *pbrth110.dll*. These DLLs must be deployed with an application that connects to a database using ADO.NET. For Oracle 10*g* or Adaptive Server 15 or later, use *Sybase.DataWindow.DbExt.dll*. For earlier versions and other DBMSs, use *Sybase.DataWindow.Db.dll*.

The DataWindow .NET database interface for ADO.NET supports the ADO.NET data providers listed in Table 4-1.

Table 4-1: Supported ADO.NET data providers

Data Provider	Namespace
.NET Framework Data Provider for OLE DB	System.Data.OleDb
.NET Framework Data Provider for SQL Server	System.Data.SqlClient
Oracle Data Provider for .NET (ODP.NET)	Oracle.DataAccess.Client
Sybase ADO.NET Data Provider for Adaptive Server Enterprise (ASE)	Sybase.Data.AseClient

Additional .NET Framework data providers may be supported in future releases. Please see the release bulletin for the latest information.

Figure 4-1 shows the general components of an ADO.NET connection using the OLE DB .NET Framework data provider.

Figure 4-1: Components of an ADO.NET OLE DB connection

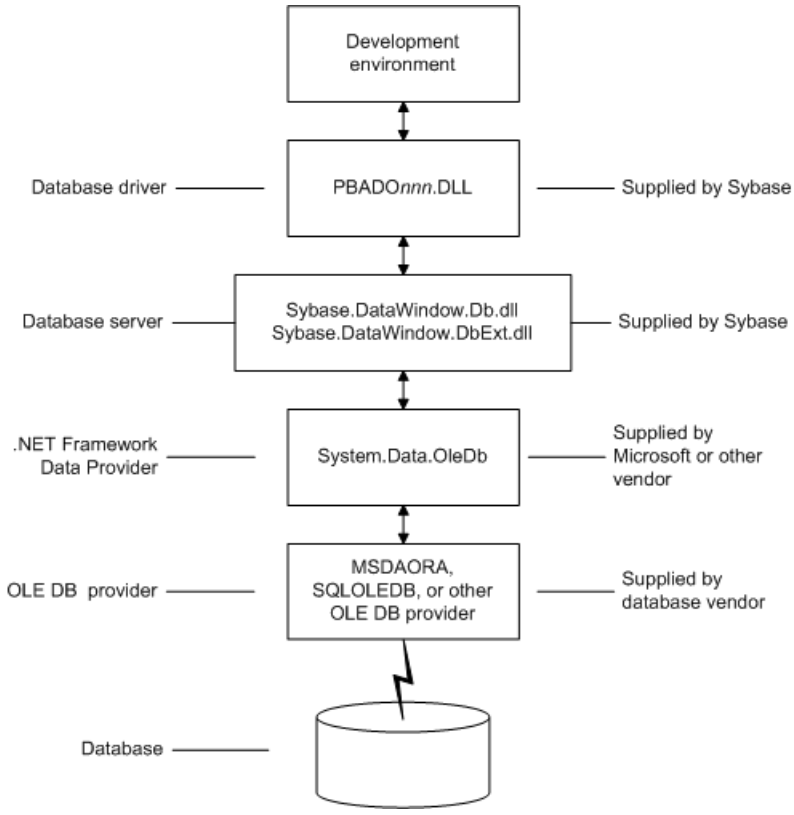
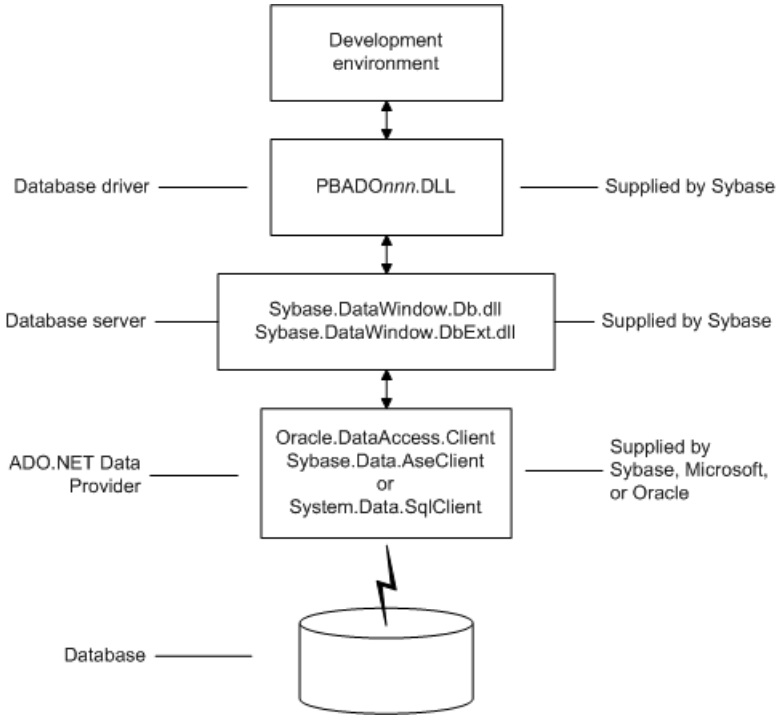


Figure 4-2 shows the general components of an ADO.NET connection using a native ADO.NET data provider.

Figure 4-2: Components of a native ADO.NET connection



OLE DB data providers

When you use the .NET Framework data provider for OLE DB, you connect to a database through an OLE DB data provider, such as Microsoft's SQLOLEDB or MSDAORA or a data provider from another vendor.

The .NET Framework Data Provider for OLE DB does not work with the MSDASQL provider for ODBC, and it does not support OLE DB version 2.5 interfaces.

You can use any OLE DB data provider that supports the OLE DB interfaces listed in Table 4-2 with the OLE DB .NET Framework data provider. For more information about supported providers, see the topic on .NET Framework data providers in the Microsoft *.NET Framework Developer's Guide*.

The DataWindow Designer ADO.NET interface supports connection to SQL Anywhere, Adaptive Server Enterprise, Microsoft SQL Server, Oracle, Informix, and Microsoft Access with the OLE DB .NET Framework data provider.

After you install the data provider, you might need to define a data source for it.

Table 4-2: Required interface support for OLE DB data providers

OLE DB object	Required interfaces
OLE DB Services	IDataInitialize
DataSource	IDBInitialize IDBCreateSession IDBProperties IPersist
Session	ISessionProperties IOpenRowset
Command	ICommandText ICommandProperties
MultipleResults	IMultipleResults
RowSet	IRowset IAccessor IColumnsInfo IRowsetInfo (only required if DBTYPE_HCHAPTER is supported)
Error	IErrorInfo IErrorRecords

Preparing to use the ADO.NET interface

Before you define the interface and connect to a database using ADO.NET:

1 Install and configure the database server, network, and client software.

2 Install the ADO.NET interface.

3 Install Microsoft's Data Access Components version 2.6 or higher software on your machine.

Step 1: Install and configure the data server

You must install and configure the database server and install the network software and client software.

❖ **To install and configure the database server, network, and client software:**

1 Make sure the appropriate database software is installed and running on its server.

You must obtain the database server software from your database vendor. For installation instructions, see your database vendor's documentation.

2 Make sure the required network software (such as TCP/IP) is installed and running on your computer and is properly configured so that you can connect to the data server at your site. You must install the network communication driver that supports the network protocol and operating system platform you are using.

For installation and configuration instructions, see your network or data source administrator.

3 If required, install the appropriate client software on each client computer on which DataWindow Designer is installed.

Client software requirements
To determine client software requirements, see your database vendor's documentation.

Step 2: Install the ADO.NET interface

In the DataWindow Designer Setup program, select the Custom install and select the ADO.NET database interface.

Step 3: Install the Microsoft Data Access Components software

The DataWindow Designer ADO.NET interface requires the functionality of the Microsoft Data Access Components (MDAC) version 2.8 or higher software. Version 2.8 is distributed with Windows XP Service Pack 2 and Windows Server 2003.

To check the version of MDAC on your computer, you can download and run the MDAC Component Checker utility from the MDAC Downloads page at http://msdn2.microsoft.com/en-us/data/aa937730.aspx.

On the Windows Vista operating system, Windows Data Access Components (DAC) 6.0 includes some changes to work with Vista but is otherwise functionally equivalent to MDAC 2.8.

OLE DB data providers installed with MDAC
Several Microsoft OLE DB data providers are automatically installed with MDAC, including the providers for SQL Server (SQLOLEDB) and ODBC (MSDASQL).

Defining the ADO.NET interface

Using the ADO.NET Database Profile Setup

To define a connection using the ADO.NET interface, you must create a database profile by supplying values for at least the basic connection parameters in the Database Profile Setup – ADO.NET dialog box. You can then select this profile at any time to connect to your data in DataWindow Designer.

For information on how to define a database profile, see "Using database profiles" on page 6.

Specifying connection parameters

You must supply a value for the Namespace and DataSource connection parameters and for the User ID and Password. When you use the System.Data.OleDb namespace, you must also select a data provider from the list of installed data providers in the Provider drop-down list.

The Data Source value varies depending on the type of data source connection you are making. For example, if you are using Microsoft's OLE DB Provider for SQL Server, you select SQLOLEDB as the Provider value and enter the actual server name as the Data Source value. In the case of Microsoft SQL Server, you must also use the Extended Properties field to provide the database name (for example, Database=Pubs) since you can have multiple instances of a database.

Using the Data Link API with OLE DB

The Data Link option allows you to access Microsoft's Data Link API, which allows you to define a file or use an existing file that contains your OLE DB connection information. A Data Link file is identified with the suffix *.udl*.

To launch this option, select the File Name check box on the Connection page and double-click the button next to the File Name box. (You can also launch the Data Link API in the Database painter by double-clicking the Manage Data Links utility included with the OLE DB interface in the list of Installed Database Interfaces.)

For more information on using the Data Link API, see Microsoft's Universal Data Access Web site at http://msdn2.microsoft.com/en-us/data/default.aspx.

Using a Data Link file versus setting the database parameters
If you use a Data Link file to connect to your data source, all other database-specific settings you make in the ADO.NET Database Profile Setup dialog box are ignored.

Getting identity column values

You can use the standard `select @@identity` syntax to obtain the value of an identity column. You can also use an alternative syntax, such as `select scope_identity()`, by adding sections to a .NET configuration file for your application.

Setting up a dbConfiguration section in a configuration file

The following example shows the general structure of a configuration file with a database configuration section and one custom configuration section:

```
<configuration>
   <configSections>
     <sectionGroup name="dbConfiguration">
        <section name="mycustomconfig"
         type="Sybase.DataWindow.Db.DbConfiguration,
         Sybase.DataWindow.Db"
        />
      </sectionGroup>
   </configSections>

   <dbConfiguration>
      <mycustomconfig dbParm="optional_value"
       getIdentity="optional_syntax"
      />
   </dbConfiguration>
</configuration>
```

❖ **To add a database configuration section to a .NET configuration file:**

1 In the <configSections> section of the configuration file, add a <sectionGroup> element with the name “dbConfiguration”. This name is case sensitive.

<configSections> must appear at the beginning of the configuration file, before the <runtime> section if any.

2 In the dbConfiguration <sectionGroup> element, add one of more <section> elements.

   For each section, specify a name of your choice and a type. The type is the strong name of the assembly used to parse this section of the configuration file.

3 Close the <section> and <configSections> elements and add a <dbConfiguration> element.

4 For each section you defined in step 2, add a new element to the <dbConfiguration> element.

   For example, if you defined a section called `config1`, add a `config1` element. Each element has two attributes: dbParm and getIdentity. You can set either or both of these attributes.

   The dbParm value sets the value of the DbParameter parameter of the transaction object. It has a maximum length of 1000 characters. If you set a value for a parameter in the configuration file, any value that you set in code or in the Database Profile Setup dialog box is overridden.

   The getIdentity value specifies the syntax used to retrieve the value of an identity column. It has a maximum length of 100 characters. If you do not specify a value for getIdentity, the `select @@identity` syntax is used.

Sample configuration file

This sample configuration file contains three custom configurations. The <myconfig> element sets both the dbParm and getIdentity attributes. <myconfig1> sets getIdentity only, and <myconfig2> sets dbParm only.

```
<configuration>
   <configSections>
     <sectionGroup name="dbConfiguration">
        <section name="myconfig"
         type="Sybase.DataWindow.Db.DbConfiguration,
         Sybase.DataWindow.Db"
        />
        <section name="myconfig1"
         type="Sybase.DataWindow.Db.DbConfiguration,
         Sybase.DataWindow.Db"
        />
        <section name="myconfig2"
         type="Sybase.DataWindow.Db.DbConfiguration,
         Sybase.DataWindow.Db"
        />
      </sectionGroup>
   </configSections>
```

```
    <dbConfiguration>
       <myconfig dbParm="disablebind=1"
        getIdentity="select scope_identity()"
       />
       <myconfig1 getIdentity="select scope_identity()"
       />
       <myconfig2 dbParm=
        "Namespace='Oracle.DataAccess.Client',
        DataSource='ora10gen',DisableBind=1,
        NCharBind=1,ADORelease='10.1.0.301'"
       />
    </dbConfiguration>
</configuration>
```

Specifying the custom configuration to be used

On the System tab page in the Database Profile Setup dialog box for ADO.NET or in code, specify the name of the custom configuration section you want to use as the value of the DbConfigSection parameter. For example:

```
Sqlca.DbParameter="DbConfigSection='myconfig'"
```

If you set any parameters in the profile or in code that are also set in the configuration file, the value specified in the configuration file takes precedence.

The configuration file must be present in the same directory as the executable file and must have the same name with the extension *.config*.

PART 3

Working with Native Database Interfaces

This part describes how to set up and define database connections accessed through one of the native database interfaces.

CHAPTER 5

Using Native Database Interfaces

About this chapter

This chapter describes native database interfaces. The following chapters explain how to prepare to use the database and define any unique database interface parameters so that you can access your data.

Contents

About native database interfaces

The native database interfaces provide native connections to many databases and DBMSs. This chapter describes how the native database interfaces access these databases.

A native database interface is a direct connection to your data in DataWindow Designer.

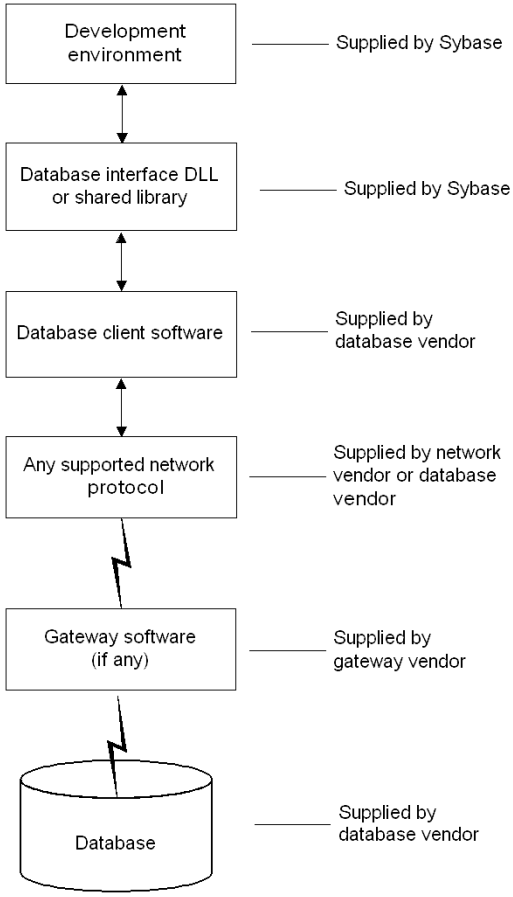
Each native database interface uses its own interface DLL to communicate with a specified database through a vendor-specific database API. For example, the SQL Native Client interface for Microsoft SQL Server uses a DLL named *PBSNC110.DLL* to access the database, whereas the Oracle 10*g* database interface accesses the database through *PBO10110.DLL*.

In contrast, a standard database interface uses a standard API to communicate with the database. For example, DataWindow Designer can use a single-interface DLL (*PBODB110.DLL*) to communicate with the ODBC Driver Manager and corresponding driver to access any ODBC data source.

Components of a database interface connection

When you use a native database interface to access a database, your connection goes through several layers before reaching the data. Each layer is a separate component of the connection and each component might come from a different vendor.

Figure 5-1: Components of a database connection



For diagrams showing the specific components of your connection, see “Basic software components” in the chapter for your native database interface.

Using a native database interface

You perform several basic steps to use a native database interface to access a database.

About preparing to use the database

The first step in connecting to a database through a native database interface is to prepare to use the database. Preparing the database ensures that you will be able to access and use your data in DataWindow Designer.

You must prepare the database *outside* DataWindow Designer *before* you start the product, then define the database interface and connect to it. The requirements differ for each database—but in general, preparing a database involves four basic steps.

❖ **To prepare to use your database with DataWindow Designer:**

1 Make sure the required database server software is properly installed and configured at your site.

2 If network software is required, make sure it is properly installed and configured at your site and on the client computer so that you can connect to the database server.

3 Make sure the required database client software is properly installed and configured on the client computer. (Typically, the client computer is the one running DataWindow Designer.)

You must obtain the client software from your database vendor and make sure that the version you install supports *all* of the following:

The operating system running on the client computer
The version of the database that you want to access
The version of DataWindow Designer that you are running

4 Verify that you can connect to the server and database you want to access outside DataWindow Designer.

For specific instructions to use with your database, see “Preparing to use the database” in the chapter for your native database interface.

About installing native database interfaces

After you prepare to use the database, you must install the native database interface that accesses the database.

About defining native database interfaces

Once you are ready to access the database, you start DataWindow Designer and define the database interface. To define a database interface, you must create a database profile by completing the Database Profile Setup dialog box for that interface.

For general instructions, see "About creating database profiles" on page 7. For instructions about defining database interface parameters unique to a particular database, see "Preparing to use the database" in the chapter for your database interface.

Making connections parallel

When you use DataWindow Designer to define or edit a DataWindow object, make sure you specify the same connection parameters for the database in both DataWindow Designer and DataWindow .NET. Otherwise, you will be unable to use the DataWindow object in your application.

For more information

The following chapters give general information about using each native database interface. For more detailed information:

- Check to see if there is a technical document that describes how to connect to your database. Any updated information about connectivity issues is available from the Sybase Support Web site at http://www.sybase.com/support.
- Ask your network or system administrator for assistance when installing and setting up the database server and client software at your site.

CHAPTER 6

Using Adaptive Server Enterprise

About this chapter

This section describes how to use the Adaptive Server Enterprise database interfaces in DataWindow Designer.

Contents

Supported versions for Adaptive Server

You can access Adaptive Server versions 11.x, 12.x, and 15.x using the SYC Adaptive Server database interface. Use of this interface to access other Open Server™ programs is not supported. The SYC database interface uses a DLL named *PBSYC110.DLL* to access the database through the Open Client CT-Lib API.

You can also access Adaptive Server version 15.x using the ASE Adaptive Server database interface. Use of this interface to access other Open Server™ programs is not supported. The Adaptive Server database interface uses a DLL named *PBASE110.DLL* to access the database through the Open Client CT-Lib API. To use this interface, the Adaptive Server 15 client must be installed on the client computer. The ASE interface supports large identifiers with up to 128 characters.

Client Library API
The Adaptive Server database interfaces use the Open Client™ CT-Library (CT-Lib) application programming interface (API) to access the database.

When you connect to an Adaptive Server database, DataWindow Designer makes the required calls to the API. Therefore, you do not need to know anything about CT-Lib to use the database interface.

Supported Adaptive Server datatypes

The Adaptive Server interface supports the Sybase datatypes listed in Table 6-1 in DataWindow objects.

Table 6-1: Supported datatypes for Adaptive Server Enterprise

Binary	NVarChar
BigInt (15.x and later)	Real
Bit	SmallDateTime
Char (see “Column-length limits” on page 61)	SmallInt
DateTime	SmallMoney
Decimal	Text
Double precision	Timestamp
Float	TinyInt
Identity	UniChar
Image	UniText (15.x and later)
Int	UniVarChar
Money	VarBinary
NChar	VarChar
Numeric	

In Adaptive Server 15.0 and later, DataWindow Designer supports unsigned as well as signed bigint, int, and smallint datatypes.

Accessing Unicode data

DataWindow Designer can connect, save, and retrieve data in both ANSI/DBCS and Unicode databases. When character data or command text is sent to the database, DataWindow Designer sends a DBCS string if the UTF8 database parameter is set to 0 (the default). If UTF8 is set to 1, DataWindow Designer sends a UTF-8 string. The database server must be configured correctly to accept UTF-8 strings. See the description of the UTF8 database parameter in the online Help for more information.

The character set used by an Adaptive Server database server applies to all databases on that server. The nchar and nvarchar datatypes can store UTF-8 data if the server character set is UTF-8. The Unicode datatypes unichar and univarchar were introduced in Adaptive Server 12.5 to support Unicode data. Columns with these datatypes can store *only* Unicode data. Any data saved into such a column must be converted to Unicode explicitly. This conversion must be handled by the database server or client.

In Adaptive Server 12.5.1 and later, additional support for Unicode data has been added. For more information, see the documentation for your version of Adaptive Server.

Different display values in painters

The unichar and univarchar datatypes support UTF-16 encoding, therefore each unichar or univarchar character requires two bytes of storage. The following example creates a table with one unichar column holding 10 Unicode characters:

```
create table unitbl (unicol unichar(10))
```

In the Database painter, the column displays as `unichar(20)` because the column requires 20 bytes of storage. This is consistent with the way the column displays in Sybase Central.

However, the mapping between the Type in the Column Specifications view in the DataWindow® painter and the column datatype of a table in the database is not one-to-one. The Type in the Column Specifications view shows the DataWindow column datatype and DataWindow column length. The column length is the number of characters, therefore an Adaptive Server `unichar(20)` column displays as `char(10)` in the Column Specifications view.

Column-length limits

Adaptive Server 12.0 and earlier have a column-length limit of 255 bytes. Adaptive Server 12.5.x and later support wider columns for Char, VarChar, Binary, and VarBinary datatypes, depending on the logical page size and the locking scheme used by the server.

In DataWindow Designer, you can use these wider columns for Char and VarChar datatypes with Adaptive Server 12.5.x when the following conditions apply:

- The Release database parameter is set to 12.5 or higher.
- You are accessing the database using Open Client 12.5.x or later.

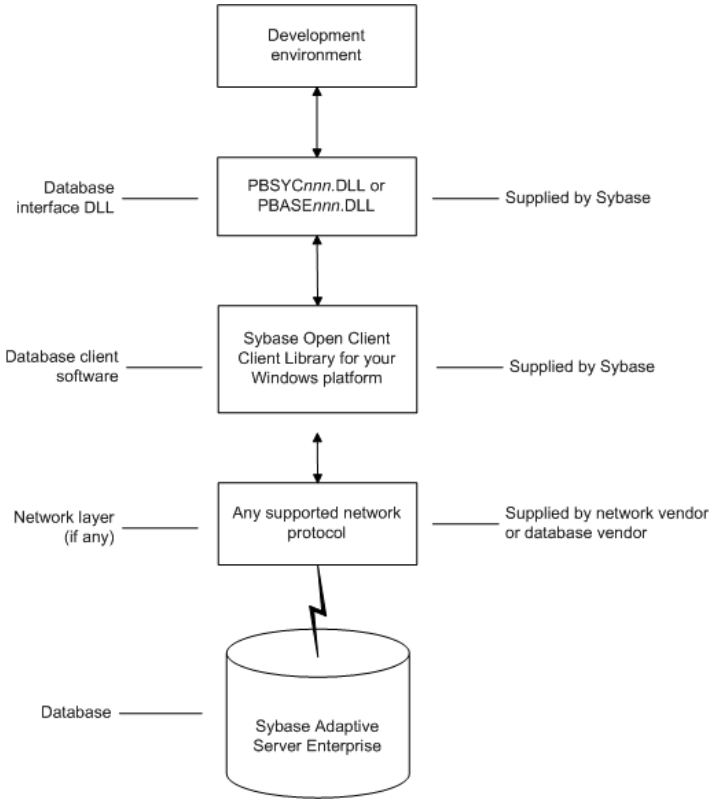
The database must be configured to use a larger page size to take full advantage of the widest limits.

For detailed information about wide columns and configuration issues, see the Adaptive Server documentation on the Product Manuals Web site at http://www.sybase.com/support/manuals/. For more information about the Release database parameter, see the online Help.

Basic software components for Adaptive Server

You must install the software components in Figure 6-1 to access an Adaptive Server database in DataWindow Designer.

Figure 6-1: Components of an Adaptive Server Enterprise connection



Preparing to use the Adaptive Server database

Before you define the interface and connect to an Adaptive Server database in DataWindow Designer, follow these steps to prepare the database for use:

1 Install and configure the required database server, network, and client software.

2 Install the Adaptive Server database interface.

3 Verify that you can connect to Adaptive Server outside DataWindow Designer.

4 Install the required DataWindow Designer stored procedures in the sybsystemprocs database.

Preparing an Adaptive Server database for use with DataWindow Designer involves these four basic tasks.

Step 1: Install and configure the database server

You must install and configure the database server, network, and client software for Adaptive Server.

❖ **To install and configure the database server, network, and client software:**

1 Make sure the Adaptive Server database software is installed on the server specified in your database profile.

You must obtain the database server software from Sybase.

For installation instructions, see your Adaptive Server documentation.

2 Make sure the supported network software (for example, TCP/IP) is installed and running on your computer and is properly configured so that you can connect to the database server at your site.

You must install the network communication driver that supports the network protocol and operating system platform you are using. The driver is installed as part of the Net-Library client software.

For installation and configuration instructions, see your network or database administrator.

3 Install the required Open Client CT-Library (CT-Lib) software on each client computer on which DataWindow Designer is installed.

You must obtain the Open Client software from Sybase. Make sure the version of Open Client you install supports *all* of the following:

The operating system running on the client computer
The version of Adaptive Server that you want to access
The version of DataWindow Designer that you are running

Required client software versions
To use the ASE Adaptive Server interface, you must install Open Client version 15.x or later. To use the SYC Adaptive Server interface, you must install Open Client version 11.x or later.

4 Make sure the Open Client software is properly configured so that you can connect to the database at your site.

Installing the Open Client software places the *SQL.INI* configuration file in the Adaptive Server directory on your computer.

SQL.INI provides information that Adaptive Server needs to find and connect to the database server at your site. You can enter and modify information in *SQL.INI* by using the configuration utility that comes with the Open Client software.

For information about setting up the *SQL.INI* or other required configuration file, see your Adaptive Server documentation.

5 If required by your operating system, make sure the directory containing the Open Client software is in your system path.

6 Make sure only one copy of each of the following files is installed on your client computer:

- Adaptive Server interface DLL
- Network communication DLL (for example, *NLWNSCK.DLL* for Windows Sockets-compliant TCP/IP)
- Database vendor DLL (for example, *LIBCT.DLL*)

Step 2: Install the database interface

In the DataWindow Designer Setup program, select the Typical install, or select the Custom install and select the Adaptive Server Enterprise (ASE or SYC) database interface.

Step 3: Verify the connection

Make sure you can connect to the Adaptive Server database server and log in to the database you want to access from outside DataWindow Designer.

Some possible ways to verify the connection are by running the following tools:

- **Accessing the database server** Tools such as the Open Client/Open Server Configuration utility (or any Ping utility) check whether you can reach the database server from your computer.
- **Accessing the database** Tools such as ISQL (interactive SQL utility) check whether you can log in to the database and perform database operations. It is a good idea to specify the same connection parameters you plan to use in your DataWindow Designer database profile to access the database.

Defining the Adaptive Server database interface

To define a connection through the Adaptive Server interface, you must create a database profile by supplying values for at least the basic connection parameters in the Database Profile Setup - Adaptive Server Enterprise dialog box. You can then select this profile anytime to connect to your database in the plug-in.

For information on how to define a database profile, see "Using database profiles" on page 6.

Using Open Client security services

The Adaptive Server interface provides several DBParm parameters that support Open Client 11.1.x or later network-based security services in your application. If you are using the required database, security, and DataWindow Designer software, you can build applications that take advantage of Open Client security services.

What are Open Client security services?

Open Client 11.1.x or later **security services** allow you to use a supported third-party security mechanism (such as CyberSafe Kerberos) to provide login authentication and per-packet security for your application. Login authentication establishes a secure connection, and per-packet security protects the data you transmit across the network.

Requirements for using Open Client security services

For you to use Open Client security services in your application, *all of the following must be true*:

- You are accessing an Adaptive Server database server using Open Client Client-Library (CT-Lib) 11.1.x or later software.
- You have the required network security mechanism and driver.

You have the required Sybase-supported network security mechanism and Sybase-supplied security driver properly installed and configured for your environment. Depending on your operating system platform, examples of supported security mechanisms include: Distributed Computing Environment (DCE) security servers and clients, CyberSafe Kerberos, and Windows NT LAN Manager Security Services Provider Interface (SSPI).

- You can access the secure server outside DataWindow Designer.

  You must be able to access a secure Adaptive Server server using Open Client 11.1.x or later software from outside DataWindow Designer.

  To verify the connection, use a tool such as ISQL or SQL Advantage to make sure you can connect to the server and log in to the database with the same connection parameters and security options you plan to use in your DataWindow Designer application.

- You are using a DataWindow DesignerDataWindow .NET database interface.

  You are using the ASE or SYC Adaptive Server interface to access the database.

- The Release DbParameter parameter is set to the appropriate value for your database.

  You have set the Release DbParameter parameter to 11or higher to specify that your application should use the appropriate version of the Open Client CT-Lib software.

  For instructions, see Release in the online Help.

- Your security mechanism and driver support the requested service.

  The security mechanism and driver you are using must support the service requested by the DbParameter parameter.

Security services DBParm parameters

If you have met the requirements described in “Requirements for using Open Client security services” on page 66, you can set the security services DbParameter parameters in the Database Profile Setup dialog box for your connection.

There are two types of DbParameter parameters that you can set to support Open Client security services: login authentication and per-packet security.

Login authentication DBParms

The following login authentication DbParameter parameters correspond to Open Client 11.1.x or later connection properties that allow an application to establish a secure connection.

Sec_Channel_Bind
Sec_Cred_Timeout
Sec_Delegation
Sec_Keytab_File
Sec_Mechanism
Sec_Mutual_Auth
Sec_Network_Auth
Sec_Server_Principal
Sec_Sess_Timeout

For instructions on setting these DbParameter parameters, see their descriptions in online Help.

Per-packet security DBParms

The following per-packet security DbParameter parameters correspond to Open Client 11.1.x or later connection properties that protect each packet of data transmitted across a network. Using per-packet security services might create extra overhead for communications between the client and server.

Sec_Confidential
Sec_Data_Integrity
Sec_Data_Origin
Sec_Replay_Detection
Sec_Seq_Detection

For instructions on setting these DbParameter parameters, see their descriptions in online Help.

Using Open Client directory services

The Adaptive Server interface provides several DbParameter parameters that support Open Client 11.1.x or later network-based directory services in your application. If you are using the required database, directory services, and DataWindow Designer software, you can build applications that take advantage of Open Client directory services.

What are Open Client directory services?

Open Client 11.1.x or later **directory services** allow you to use a supported third-party directory services product (such as the Windows Registry) as your directory service provider. Directory services provide centralized control and administration of the network entities (such as users, servers, and printers) in your environment.

Requirements for using Open Client directory services

For you to use Open Client directory services in your application, *all of the following must be true*:

- You are accessing an Adaptive Server database server using Open Client Client-Library (CT-Lib) 11.x or later software
- You have the required Sybase-supported directory service provider software and Sybase-supplied directory driver properly installed and configured for your environment. Depending on your operating system platform, examples of supported security mechanisms include: the Windows Registry, Distributed Computing Environment Cell Directory Services (DCE/CDS), Banyan StreetTalk Directory Assistance (STDA), and Novell NetWare Directory Services (NDS).
- You must be able to access a secure Adaptive Server server using Open Client 11.1.x or later software from outside DataWindow Designer.

  To verify the connection, use a tool such as ISQL or SQL Advantage to make sure you can connect to the server and log in to the database with the same connection parameters and directory service options you plan to use in your DataWindow Designer application.
- You are using the ASE or SYC Adaptive Server interface to access the database.
- You must use the correct syntax as required by your directory service provider when specifying the server name in a database profile. Different providers require different syntax based on their format for specifying directory entry names.

  For information and examples for different directory service providers, see “Specifying the server name with Open Client directory services” next.
- You have set the Release DbParameter parameter to 11 or higher to specify that your application should use the behavior of the appropriate version of the Open Client CT-Lib software.

For instructions, see Release (Adaptive Server Enterprise) in the online Help.

- The directory service provider and driver you are using must support the service requested by the DbParameter parameter.

Specifying the server name with Open Client directory services

When you are using Open Client directory services in a DataWindow Designer application, you must use the syntax required by your directory service provider when specifying the server name in a database profile to access the database.

Different directory service providers require different syntax based on the format they use for specifying directory entry names. Directory entry names can be fully qualified or relative to the default (active) Directory Information Tree base (DIT base) specified in the Open Client/Server™ configuration utility.

The **DIT base** is the starting node for directory searches. Specifying a DIT base is analogous to setting a current working directory for UNIX or MS-DOS file systems. (You can specify a nondefault DIT base with the DS_DitBase DbParameter parameter. For information, see DS_DitBase in the online Help.)

Windows registry server name example

This example shows typical server name syntax if your directory service provider is the Windows registry.

```
Node name: SALES:software\sybase\server\SYS12
DIT base: SALES:software\sybase\server
Server name: SYS12
```

❖ **To specify the server name in a database profile:**

- Type the following in the Server box on the Connection tab in the Database Profile Setup dialog box. Do *not* start the server name with a backslash (\).

```
SYS12
```

DCE/DCS server name example

This example shows typical server name syntax if your directory service provider is Distributed Computing Environment Cell Directory Services (DCE/CDS).

```
Node name: /.../boston.sales/dataservers/sybase/SYS12
DIT base: /../boston.sales/dataservers
Server name: sybase/SYS12
```

❖ **To specify the server name in a database profile:**

- Type the following in the Server box on the Connection tab in the Database Profile Setup dialog box. Do *not* start the server name with a slash (/).

```
sybase/SYS12
```

Banyan STDA server name example

This example shows typical server name syntax if your directory service provider is Banyan StreetTalk Directory Assistance (STDA).

```
Node name: SYS12@sales@chicago
DIT base: chicago
Server name: SYS12@sales
```

❖ **To specify the server name in a database profile:**

- Type the following in the Server box on the Connection tab in the Database Profile Setup dialog box. Do *not* end the server name with @.

```
SYS12@sales
```

Novell NDS server name example

This example shows typical server name syntax if your directory service provider is Novell NetWare Directory Services (NDS).

```
Node name: CN=SYS12.OU=miami.OU=sales.O=sybase
DIT base: OU=miami.OU=sales.O=sybase
Server name: SYS12
```

❖ **To specify the server name in a database profile:**

- Type the following in the Server box on the Connection tab in the Database Profile Setup dialog box. Do *not* start the server name with CN=.

```
SYS12
```

Directory services DbParameter parameters

If you have met the requirements described in “Requirements for using Open Client directory services” on page 69, you can set the directory services DbParameter parameters in a database profile for your connection.

The following DbParameter parameters correspond to Open Client 11.1.x or later directory services connection parameters:

DS_Alias
DS_Copy
DS_DitBase
DS_Failover
DS_Password (Open Client 12.5 or later)

DS_Principal
DS_Provider
DS_TimeLimit

For instructions on setting these DbParameter parameters, see their descriptions in the online Help.

Using PRINT statements in Adaptive Server stored procedures

The ASE or SYC Adaptive Server database interface allows you to use PRINT statements in your stored procedures for debugging purposes.

This means, for example, that if you turn on Database Trace when accessing the database through the ASE or SYC interface, PRINT messages appear in the trace log but they do not return errors or cancel the rest of the stored procedure.

Creating a DataWindow based on a cross-database join

The ability to create a DataWindow based on a heterogeneous cross-database join is available through the use of Adaptive Server's Component Integration Services. Component Integration Services allow you to connect to multiple remote heterogeneous database servers and define multiple proxy tables that reference the tables residing on those servers.

For information on how to create proxy tables, see the Adaptive Server documentation. For information on identifying identity columns in the underlying database tables referenced by proxy tables, see the technical note "Techniques for Working with Identity Columns in SQL Anywhere Proxy Tables" on the Sybase Web site at http://www.sybase.com/detail?id=1035056.

CHAPTER 7

Using Informix

About this chapter

This chapter describes how to use the native IBM Informix database interface in DataWindow Designer.

Contents

Supported versions for Informix

You can access the following Informix databases using the native Informix database interface:

- Informix Dynamic Server
- Informix-OnLine and Informix-SE version 9.x

DataWindow Designer provides the IN9 interface in the *PBIN9110.DLL* to connect through Informix-Connect version 9.x client software.

Accessing Unicode data

DataWindow Designer can connect, save, and retrieve data in ANSI/DBCS databases. The Informix native driver does not currently support access to Unicode databases.

Supported Informix datatypes

The Informix database interface supports the Informix datatypes listed in Table 7-1 in DataWindow objects.

Table 7-1: Supported datatypes for Informix

Byte (a maximum of 231 bytes)	Integer (4 bytes)
Character (1 to 32,511 bytes)	Money
Date	Real
DateTime	Serial
Decimal	SmallInt (2 bytes)
Float	Text (a maximum of 231 bytes)
Interval	VarChar (1 to 255 bytes)

Exceptions
Byte, Text, and VarChar datatypes are not supported in Informix SE.

Informix DateTime datatype

The DateTime datatype is a contiguous sequence of boxes. Each box represents a component of time that you want to record. The syntax is:

DATETIME *largest_qualifier* **TO** *smallest_qualifier*

DataWindow Designer defaults to Year TO Fraction(5).

For a list of qualifiers, see your Informix documentation.

❖ **To create your own variation of the DateTime datatype:**

1 In the Database painter, create a table with a DateTime column.

For instructions on creating a table, see the *User's Guide*.

2 In the Columns view, select Pending Syntax from the Objects or pop-up menu.

The Columns view displays the pending changes to the table definition. These changes execute only when you click the Save button to save the table definition.

3 Select Copy from the Edit or pop-up menu
or
Click the Copy button.

The SQL syntax (or the portion you selected) is copied to the clipboard.

4 In the ISQL view, modify the DateTime syntax and execute the CREATE TABLE statement.

For instructions on using the ISQL view, see the *User's Guide*.

Informix Time datatype

The Informix database interfaces also support a time datatype. The time datatype is a subset of the DateTime datatype. The time datatype uses only the time qualifier boxes.

Informix Interval datatype

The interval datatype is one value or a sequence of values that represent a component of time. The syntax is:

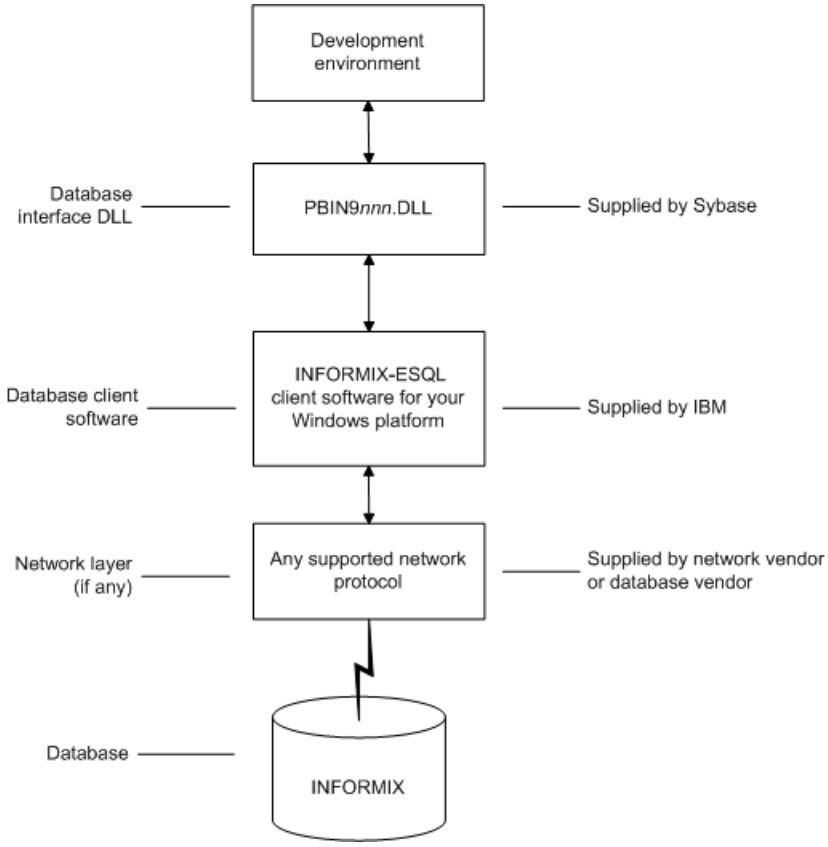
INTERVAL *largest_qualifier* **TO** *smallest_qualifier*

DataWindow Designer defaults to `Day(3) TO Day`. For more about interval datatypes, see your Informix documentation.

Basic software components for Informix

Figure 7-1 shows the basic software components required to access an Informix database using the native Informix database interfaces.

Figure 7-1: Components of an Informix connection



Preparing to use the Informix database

Before you define the database interface and connect to an Informix database in DataWindow Designer, follow these steps to prepare the database for use:

1 Install and configure the required database server, network, and client software.

2 Install the native Informix IN9 database interface.

3 Verify that you can connect to the Informix server and database outside DataWindow Designer.

Step 1: Install and configure the database server

You must install and configure the required database server, network, and client software for Informix.

❖ **To install and configure the required database server, network, and client software:**

1 Make sure the Informix database server software and database network software is installed and running on the server specified in your database profile.

You must obtain the database server and database network software from Informix.

For installation instructions, see your Informix documentation.

2 Install the required Informix client software on each client computer on which DataWindow Designer is installed.

Install Informix Connect or the Informix Client SDK (which includes Informix Connect) and run the SetNet32 utility to configure the client registry settings.

You must obtain the Informix client software from IBM. Make sure the version of the client software you install supports *all* of the following:

The operating system running on the client computer
The version of the database that you want to access
The version of DataWindow Designer that you are running

For installation instructions, see your Informix documentation.

3 Make sure the Informix client software is properly configured so that you can connect to the Informix database server at your site.

For example, when you install Informix-Connect client software, it automatically creates the correct configuration file on your computer.

The configuration file contains default parameters that define your network configuration, network protocol, and environment variables. If you omit these values from the database profile when you define the native Informix database interface, they default to the values specified in your configuration file.

For instructions on setting up the Informix configuration file, see your Informix documentation.

4 If required by your operating system, make sure the directory containing the Informix client software is in your system path.

Step 2: Install the database interface

In the DataWindow Designer Setup program, select the Typical install, or select the native Informix database interface in the Custom install.

Step 3: Verify the connection

Make sure you can connect to the Informix server and database you want to access from outside DataWindow Designer.

To verify the connection, use any Windows-based utility (such as the Informix *ILOGIN.EXE* program) that connects to the database. When connecting, be sure to specify the same parameters you plan to use in your DataWindow Designer database profile to access the database.

For instructions on using *ILOGIN.EXE*, see your Informix documentation.

Defining the Informix database interface

To define a connection through an Informix database interface, you must create a database profile by supplying values for at least the basic connection parameters in the Database Profile Setup - Informix IN9 dialog box. You can then select this profile at any time to connect to your database in the plug-in.

For information on how to define a database profile, see “Using database profiles” on page 6.

Specifying the server name

When you specify the server name value, you *must* use the following format to connect to the database through the Informix interface:

host_name@server_name

Parameter	Description
host_name	The name of the host computer running the Informix database server. This corresponds to the Informix HOSTNAME environment variable.
server_name	The name of the server containing the Informix database. This corresponds to the Informix SERVER environment variable.

For example, to use the IN9 interface to connect to an Informix database server named `server01` running on a host machine named `sales`, type the host name (`sales`) in the Host Name box and the server name (`server01`) in the Server box on the Connection tab in the Database Profile Setup - Informix IN9 dialog box. DataWindow Designer saves this server name as `sales@server01` in the database profile entry in the system registry.

CHAPTER 8

Using Microsoft SQL Server

About this chapter

This chapter describes how to use the Microsoft SQL Server Native Client database interface in DataWindow Designer.

Contents

Supported versions for SQL Server

You can access Microsoft SQL Server 2000 and 2005 databases using the SQL Native Client interface. The SQL Native Client interface uses a DLL named *PBSNC110.DLL* to access the database. The interface uses the SQL Server 2005 Native Client (*sqlncli.h* and *sqlncli.dll*) on the client side and connects using OLE DB.

For SQL Server 2000, the SQL client SDK was provided with the Microsoft Database Access Components (MDAC). MDAC does not support new features in SQL Server 2005. To take advantage of these features, you need to use the SNC interface. The SQL Server 2005 SQL Native Client software must be installed on the client computer.

PBODB initialization file not used
Connections made directly through OLE DB use the PBODB initialization file to set some parameters, but connections made using the SNC interface do not depend on the PBODB initialization file.

Supported SQL Server datatypes

The SQL Native Client database interface supports the datatypes listed in Table 8-1.

Table 8-1: Supported datatypes for Microsoft SQL Server 2005

Binary	Real
Bit	SmallDateTime
Character (fewer than 255 characters)	SmallInt
DateTime	SmallMoney
Decimal	Text
Float	Timestamp
Identity	TinyInt
Image	VarBinary(*max*)
Int	VarBinary(*n*)
Money	VarChar(*max*)
Numeric	VarChar(*n*)
NVarChar(*max*)	XML
NVarChar(*n*)	

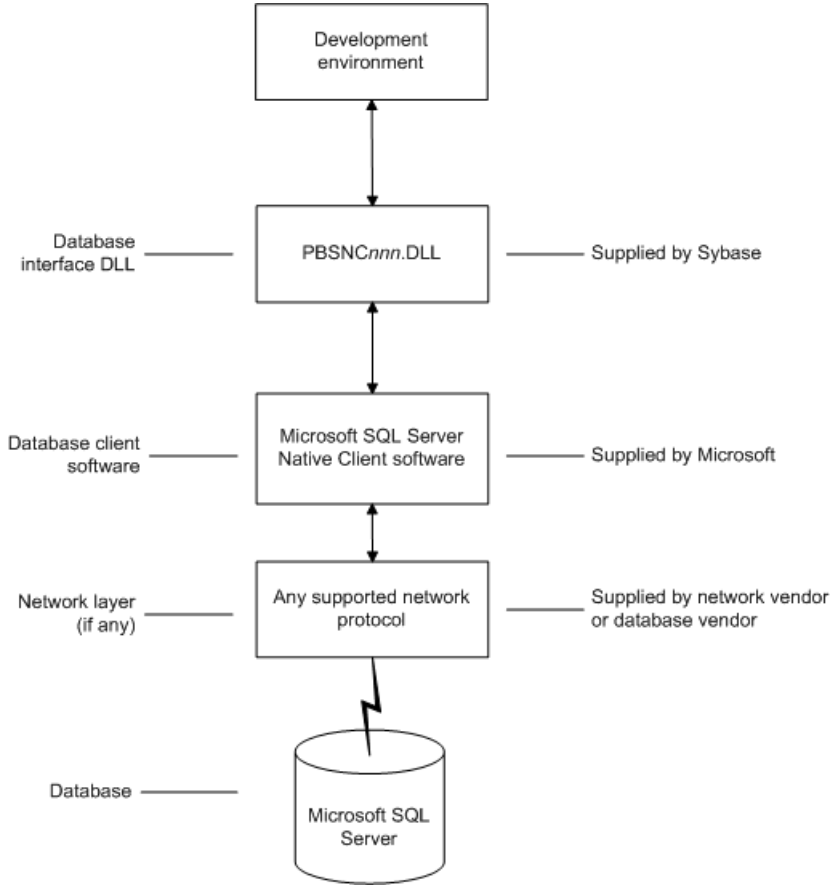
The XML datatype is a built-in datatype in SQL Server 2005 that enables you to store XML documents and fragments in a SQL Server database. You can use this datatype as a column type when you create a table.

In SQL Server 2005, the VarChar(*max*), NVarChar(*max*), and VarBinary(*max*) datatypes store very large values (up to 2^31 bytes). You can use these datatypes to obtain metadata, define new columns, and query data from the columns.

Basic software components for Microsoft SQL Server

You must install the software components in Figure 8-1 to access a database with the SQL Native Client interface. Microsoft SQL Server Native Client software contains a SQL OLE DB provider and ODBC driver in a single DLL.

Figure 8-1: Components of a Microsoft SQL Server connection



Preparing to use the SQL Server database

Before you define the database interface and connect to a Microsoft SQL Server database in DataWindow Designer, follow these steps to prepare the database for use:

1 Install and configure the required database server, network, and client software.

2 Install the SQL Native Client database interface.

3 Verify that you can connect to the Microsoft SQL Server server and database outside DataWindow Designer.

Step 1: Install and configure the database server

You must install and configure the database server, network, and client software for SQL Server.

❖ **To install and configure the database server, network, and client software:**

1 Make sure the Microsoft SQL Server database software is installed and running on the server specified in your database profile.

You must obtain the database server software and required licenses from Microsoft Corporation. For installation instructions, see your Microsoft SQL Server documentation.

Upgrading from an earlier version of SQL Server
For instructions on upgrading to a later version of SQL Server or installing it alongside an earlier version, see your Microsoft SQL Server documentation.

2 If you are accessing a remote SQL Server database, make sure the required network software (for example, TCP/IP) is installed and running on your computer and is properly configured so that you can connect to the SQL Server database server at your site.

For installation and configuration instructions, see your network or database administrator.

3 Install the required Microsoft SQL Native Client software on each client computer on which DataWindow Designer is installed.

You must obtain the SQL Native Client software from Microsoft. Make sure the version of the client software you install supports *all* of the following:

The operating system running on the client computer
The version of the database that you want to access
The version of DataWindow Designer that you are running

For installation instructions, see your Microsoft SQL Server 2005 documentation.

4 Make sure the SQL Native Client client software is properly configured so that you can connect to the SQL Server database server at your site.

For configuration instructions, see your Microsoft SQL Server 2005 documentation.

5 Make sure the directory containing the SQL Native Client software is in your system path.

6 Make sure only one copy of the *Sqlncli.dll* file is installed on your computer.

Step 2: Install the database interface

In the DataWindow Designer Setup program, select the Custom install and select the SQL Native Client database interface.

Step 3: Verify the connection

Make sure you can connect to the SQL Server server and database you want to access from outside DataWindow Designer.

To verify the connection, use any Windows-based utility that connects to the database. When connecting, be sure to specify the same parameters you plan to use in your DataWindow Designer database profile to access the database.

Defining the SQL Server database interface

To define a connection through the SQL Native Client interface, you must create a database profile by supplying values for at least the basic connection parameters in the Database Profile Setup - SQL Native Client dialog box. You can then select this profile at any time to connect to your database in the plug-in.

For information on how to define a database profile, see “Creating a database profile” on page 8. For new features that require special settings in the database profile, see “SQL Server 2005 features” on page 87. For a comparison of the database parameters you might have used with existing applications and those used with the SNC database interface, see “Migrating from the OLE DB database interface” next.

Migrating from the OLE DB database interface

Prior to the introduction of SQL Server 2005 and SQL Native Client, Microsoft recommended using the OLE DB database interface and MDAC to connect to SQL Server. You can continue to use this solution if you do not need to take advantage of new features in SQL Server 2005.

This section provides a comparison between database parameters you might have used in existing applications with the parameters you can use with the SNC database interface.

OLE DB database parameters supported by SNC

Table 8-2 shows the database parameters and preferences that can be set in the Database Profile Setup dialog box for the OLE DB standard interface for Microsoft SQL Server, and indicates whether they are supported by the new SNC interface.

The shows the tab page in the Database Profile Setup dialog box for OLE DB. The parameters and preferences may be on different tab pages in the SNC profile.

Table 8-2: OLE DB parameters supported by SNC

OLE DB	SNC
Connection tab:	
Provider	Not supported
DataSource	Supported at runtime (as SQLCA.ServerName)
DataLink	Supported
Location	Not supported
ProviderString	Supported
System tab:	
PBCatalogOwner	Supported
ServiceComponents	Not supported
AutoCommit	Supported (General tab)
CommitOnDisconnect	Supported (General tab)
StaticBind	Supported (Transaction tab)
DisableBind	Supported (Transaction tab)
Init_Prompt	Not supported
TimeOut	Supported
LCID	Not supported
Transaction tab:	
Block	Supported
PBMaxBlobSize	Supported
Mode	Not supported
Lock	Supported
Syntax tab:	
DelimitIdentifier	Supported
IdentifierQuoteChar	Not supported
DateFormat	Supported

OLE DB	SNC
TimeFormat	Supported
DecimalSeparator	Supported
OJSyntax	Supported
Security tab:	
EncryptPassword	Not supported
CacheAuthentication	Not supported
PersistSensitive	Not supported
MaskPassword	Not supported
PersistEncrypted	Not supported
IntegratedSecurity	Supported (TrustedConnection on General tab)
ImpersonationLevel	Not supported
ProtectionLevel	Not supported

Additional database parameters

The SNC interface also supports the ReCheckRows and BinTxtBlob runtime-only parameters, the Encrypt, TrustServerCertificate, and SPCache parameters (on the System tab page), and the Identity parameter (on the Syntax tab page).

SPCache database parameter

You can control how many stored procedures are cached with parameter information by modifying the setting of the SPCache database parameter. The default is 100 procedures. To turn off caching of stored procedures, set SPCache to 0.

For more information about database parameters supported by the SNC interface, see the *Connection Reference* in the online Help.

SQL Server 2005 features

The SNC database interface supports several features that were introduced in SQL Server 2005. For more information about using these features, see the Microsoft SQL Server 2005 documentation.

Multiple Active Result Sets

The SNC interface supports Multiple Active Result Sets (MARS), which enable applications to have multiple default result sets open and to interleave reading from them. Applications can also execute statements such as INSERT, UPDATE, and DELETE and stored procedure calls while default result sets are open.

Encryption without validation

SQL Server 2005 always encrypts network packets associated with logging. If no certificate is provided on the server when it starts up, SQL Server generates a self-signed certificate that is used to encrypt login packets.

The SQL Native Client supports encrypting data sent to the server without validating the certificate. The TrustServerCertificate database parameter, available on the System page of the database connection profile dialog box, allows you to control this feature.

Snapshot isolation

The snapshot isolation level is designed to enhance concurrency for online transaction processing applications. Transactions that start under snapshot isolation read a database snapshot taken at start up time. Keyset, dynamic, and static server cursors in this context behave like static cursors opened within serializable transactions, but locks are not taken, which can reduce blocking on the server. The SQLCA.Lock value for snapshot isolation is SS. You can set this value in the Isolation Level field on the Transaction page of the database connection profile dialog box.

Notes on using the SNC interface

SQL batch statements

The SNC interface supports SQL batch statements. However, they must be enclosed in a BEGIN...END block or start with the keyword DECLARE:

- Enclosed in a BEGIN...END block:

```
BEGIN
INSERT INTO t_1 values(1, 'sfdfs')
INSERT INTO t_2 values(1, 'sfdfs')
SELECT * FROM t_1
SELECT * FROM t_2
END
```

- Starting with the keyword DECLARE:

```
DECLARE @p1 int, @p2 varchar(50)
SELECT  @p1 = 1
EXECUTE  sp_4 @p1, @p2 OUTPUT
SELECT @p2  AS  'output'
```

You can run the batch of SQL statements in the Database painter. For example:

```
String batchSQL //contains a batch of SQL statements
DECLARE my_cursor DYNAMIC CURSOR FOR SQLSA ;
PREPARE SQLSA FROM :batchSQL ;
OPEN DYNAMIC my_cursor ;
```

```
//first result set
FETCH my_cursor INTO . . .

//second result set
FETCH my_cursor INTO . .
. . .
CLOSE my_cursor ;
```

Connection pooling

The SNC interface pools connections automatically using OLE DB pooling. To disable OLE DB pooling, type the following in the Extended Properties box on the Connection tab page in the Database Profile Setup dialog box:

```
OLE DB Services=-4
```

You can also type the following statement in code:

```
ProviderString='OLE DB Services=-4')
```

Triggers and synonyms in the Database painter

In the Objects view for SNC profiles in the Database painter, triggers display for tables in the Tables folder and Microsoft SQL Server 2005 synonyms display for tables and views.

CHAPTER 9

Using Oracle

About this chapter

This chapter describes how to use the native Oracle database interfaces in DataWindow Designer.

Contents

Supported versions for Oracle

DataWindow Designer provides three Oracle database interfaces. These interfaces use different DLLs and access different versions of Oracle.

Table 9-1: Supported native database interfaces for Oracle

Oracle interface	DLL
O84 Oracle8*i*	*PBO84110.DLL*
O90 Oracle9*i*	*PBO90110.DLL*
O10 Oracle 10*g*	*PBO10110.DLL*

For more information

Updated information about supported versions of databases might be available electronically on the Sybase Customer Service and Support Web site at http://www.sybase.com/support or in the DataWindow .NET Release Bulletin.

The Oracle 10*g* database interface allows you to connect to Oracle 10*g* servers using Oracle 10*g* Database Client or Oracle 10*g* Instant Client. It supports BINARY_FLOAT and BINARY_DOUBLE datatypes and increased size limits for CLOB and NCLOB datatypes. Oracle 10*g* clients can connect to Oracle9*i* or Oracle 10*g* servers; they cannot connect to Oracle8*i* or earlier servers.

Supported Oracle datatypes

The Oracle database interfaces support the Oracle datatypes listed in Table 9-2 in DataWindow objects:

Table 9-2: Supported datatypes for Oracle

Bfile	NChar (Oracle9*i* and later only)
Blob	Number
Char	NVarChar2 (Oracle9*i* and later only)
Clob	Raw
Date	TimeStamp (Oracle9*i* and later only)
Float	VarChar
Long	VarChar2
LongRaw	

The Oracle 10*g* interface also supports BINARY_FLOAT and BINARY_DOUBLE datatypes. These are IEEE floating-point types that pass the work of performing floating-point computations to the operating system, providing greater efficiency for large computations.

Accessing Unicode data

Using the O90 or O10 database interface, DataWindow Designer can connect, save, and retrieve data in both ANSI/DBCS and Unicode databases, but it does not convert data between Unicode and ANSI/DBCS. When character data or command text is sent to the database, DataWindow Designer sends a Unicode string. The driver must guarantee that the data is saved as Unicode data correctly. When DataWindow Designer retrieves character data, it assumes the data is Unicode.

Using the O84 database interface, DataWindow Designer detects whether the Oracle client variable NLS_LANG is set. If the variable is set to a value that requires UTF-8 or DBCS characters, DataWindow Designer converts command text (such as `SELECT * FROM emp`) to the appropriate character set before sending the command to the database. However, if DisableBind is set to 0 (the default), DataWindow Designer always binds string data as Unicode data. Using O84, you can set the DisableUnicode database parameter to 1 to retrieve data as an ANSI string.

A Unicode database is a database whose character set is set to a Unicode format, such as UTF-8, UTF-16, UCS-2, or UCS-4. All data must be in Unicode format, and any data saved to the database must be converted to Unicode data implicitly or explicitly.

A database that uses ANSI (or DBCS) as its character set might use special datatypes to store Unicode data. These datatypes are NCHAR and NVARCHAR2. Columns with this datatype can store *only* Unicode data. Any data saved into such a column must be converted to Unicode explicitly. This conversion must be handled by the database server or client.

A constant string is regarded as a char type by Oracle and its character set is NLS_CHARACTERSET. However, if the datatype in the database is NCHAR and its character set is NLS_NCHAR_CHARACTERSET, Oracle performs a conversion from NLS_CHARACTERSET to NLS_NCHAR_CHARACTERSET. This can cause loss of data. For example, if NLS_CHARACTERSET is WE8ISO8859P1 and NLS_NCHAR_CHARACTERSET is UTF8, when the Unicode data is mapped to WE8ISO8859P1, the Unicode data is corrupted.

By default, the O90 and O10 database interfaces bind all string data to internal variables as the Oracle CHAR datatype to avoid downgrading performance. To ensure that NCHAR and NVARCHAR2 columns are handled as such on the server, set the NCharBind database parameter to 1 to have the O90 and O10 drivers bind string data as the Oracle NCHAR datatype.

If an Oracle stored procedure has an NCHAR or NVARCHAR2 input parameter and the input data is a Unicode string, set the BindSPInput database parameter to 1 to force the Oracle database to bind the input data. The O90 and O10 database interfaces are able to describe the procedure to determine its parameters, therefore you do not need to set the NCharBind database parameter.

For a DataWindow object to access NCHAR and NVARCHAR2 columns and retrieve data correctly, set both DisableBind and StaticBind to 0. Setting StaticBind to 0 ensures that DataWindow Designer gets an accurate datatype before retrieving.

TimeStamp datatype

The TimeStamp datatype in Oracle9*i* and later is an extension of the Date datatype. It stores the year, month, and day of the Date value plus hours, minutes, and seconds:

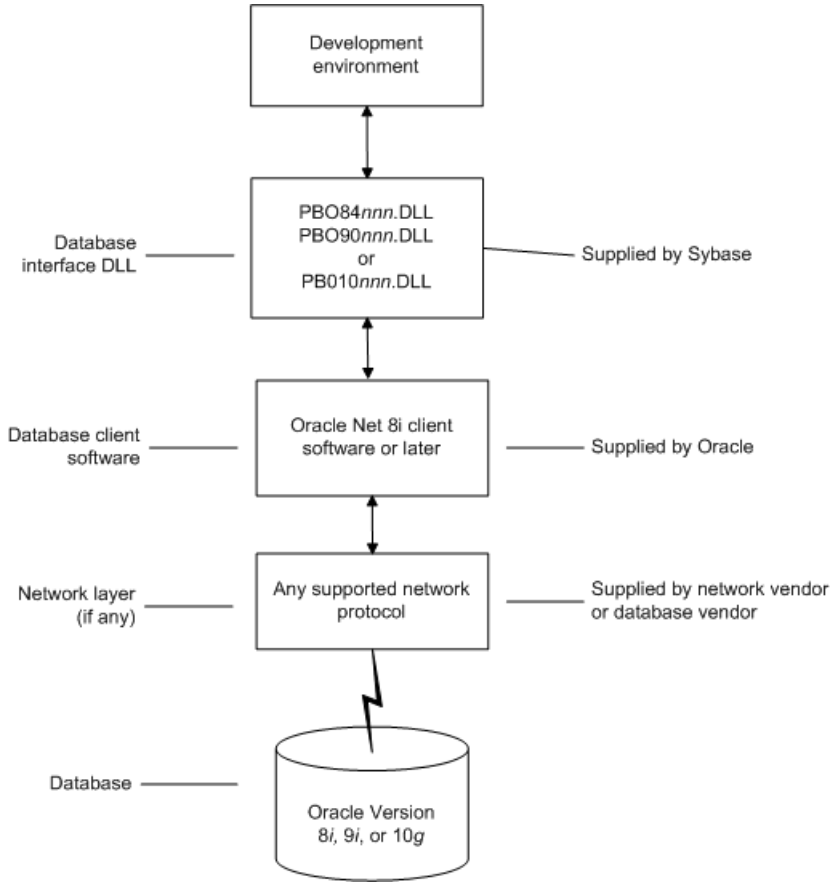
Timestamp[*fractional_seconds_precision*]

The *fractional_seconds_precision* value is optional and provides the number of digits for indicating seconds. The range of valid values for use with DataWindow Designer is 0-6.

Basic software components for Oracle

You must install the software components in Figure 9-1 to access an Oracle database in DataWindow Designer.

Figure 9-1: Components of an Oracle connection



Preparing to use the Oracle database

Before you define the database interface and connect to an Oracle database in DataWindow Designer, follow these steps to prepare the database for use:

1 Install and configure the required database server, network, and client software.

2 Install the native Oracle database interface for the version of Oracle you want to access.

3 Verify that you can connect to the Oracle server and database outside DataWindow Designer.

Preparing an Oracle database for use with DataWindow Designer involves these three basic tasks.

Step 1: Install and configure the database server

You must install and configure the database server, network, and client software for Oracle.

❖ **To install and configure the database server, network, and client software:**

1 Make sure the Oracle database software is installed on your computer or on the server specified in your database profile.

For example, with the Oracle O90 interface you can access an Oracle9*i* or Oracle 10*g* database server.

You must obtain the database server software from Oracle Corporation.

For installation instructions, see your Oracle documentation.

2 Make sure the supported network software (such as TCP/IP) is installed and running on your computer and is properly configured so that you can connect to the Oracle database server at your site.

The Hosts and Services files must be present on your computer and properly configured for your environment.

You must obtain the network software from your network vendor or database vendor.

For installation and configuration instructions, see your network or database administrator.

3 Install the required Oracle client software on each client computer on which DataWindow Designer is installed.

You must obtain the client software from Oracle Corporation. Make sure the client software version you install supports *all* of the following:

The operating system running on the client computer
The version of the database that you want to access
The version of DataWindow Designer that you are running

Oracle 10*g* Instant Client is free client software that lets you run applications without installing the standard Oracle client software. It has a small footprint and can be freely redistributed.

4 Make sure the Oracle client software is properly configured so that you can connect to the Oracle database server at your site.

For information about setting up Oracle configuration files, see your Oracle Net documentation.

5 If required by your operating system, make sure the directory containing the Oracle client software is in your system path.

Step 2: Install the database interface

In the DataWindow Designer Setup program, select the Typical install or select the Custom install and select the Oracle database interfaces you require.

For a list of the Oracle database interfaces available, see "Supported versions for Oracle" on page 91.

Step 3: Verify the connection

Make sure you can connect to the Oracle database server and log in to the database you want to access from outside DataWindow Designer.

Some possible ways to verify the connection are by running the following Oracle tools:

- **Accessing the database server** Tools such as Oracle TNSPING (or any other ping utility) check whether you can reach the database server from your computer.
- **Accessing the database** Tools such as Oracle SQL*Plus check whether you can log in to the Oracle database you want to access and perform database operations. It is a good idea to specify the same connection parameters you plan to use in your DataWindow Designer database profile to access the database.

Defining the Oracle database interface

To define a connection through an Oracle database interface, you must create a database profile by supplying values for at least the basic connection parameters in the Database Profile Setup dialog box for your Oracle interface. You can then select this profile at any time to connect to your database in the plug-in.

For information on how to define a database profile, see "Using database profiles" on page 6.

Specifying the Oracle server connect descriptor

To connect to an Oracle database server that resides on a network, you must specify the proper connect descriptor in the Server box on the Connection tab of the Database Profile Setup dialog box for your Oracle interface. The connect descriptor specifies the connection parameters that Oracle uses to access the database.

For help determining the proper connect descriptor for your environment, see your Oracle documentation or system administrator.

Specifying a connect descriptor

The syntax of the connect descriptor depends on the Oracle client software you are using.

If you are using Net8 or later, the syntax is:

OracleServiceName

If you are using SQL*Net version 2.x, the syntax is:

@ **TNS:** *OracleServiceName*

Parameter	Description
@	The at (@) sign is required
TNS	The identifier for the Oracle Transparent Network Substrate (TNS) technology
:	The colon (:) is required
OracleServiceName	The service name assigned to your server in the Oracle configuration file for your platform

Net9 example To use Net9 client software to connect to the service named ORA9, type the following connect descriptor in the Server box on the Connection tab of the Database Profile Setup dialog box for Oracle9*i* and later:

```
ORA9
```

Using Oracle stored procedures as a data source

This section describes how you can use Oracle stored procedures.

What is an Oracle stored procedure?

Oracle defines a **stored procedure** (or function) as a named PL/SQL program unit that logically groups a set of SQL and other PL/SQL programming language statements together to perform a specific task.

Stored procedures can take parameters and return one or more result sets (also called cursor variables). You create stored procedures in your schema and store them in the data dictionary for use by multiple users.

What you can do with Oracle stored procedures

Ways to use Oracle stored procedures

In DataWindow Designer, you can use an Oracle stored procedure as a data source for DataWindow objects.

Procedures with a single result set You can use stored procedures that return a single result set in DataWindow objects, but *not* when using the RPCFUNC keyword to declare a stored procedure as an external function or subroutine.

Using Oracle stored procedures with result sets

Overview of basic steps

The following procedure assumes you are creating the stored procedure in the ISQL view of the Database painter in DataWindow Designer.

❖ **To use an Oracle stored procedure with a result set:**

1. Set up the ISQL view of the Database painter to create the stored procedure.
2. Create the stored procedure with a result set as an IN OUT (reference) parameter.
3. Create DataWindow objects that use the stored procedure as a data source.

Setting up the Database painter

When you create a stored procedure in the ISQL view of the Database painter, you must change the default SQL statement terminator character to one that you do not plan to use in your stored procedure syntax.

The default SQL terminator character for the Database painter is a semicolon (;). If you plan to use a semicolon in your Oracle stored procedure syntax, you must change the painter's terminator character to something other than a semicolon to avoid conflicts. A good choice is the backquote (`) character.

❖ **To change the default SQL terminator character in the Database painter:**

1 Connect to your Oracle database in DataWindow Designer as the System user.

For instructions, see "Defining the Oracle database interface" on page 98.

2 Open the Database painter.

3 Select Design>Options from the menu bar.

The Database Preferences dialog box displays. If necessary, click the General tab to display the General property page.

4 Type the character you want (for example, a backquote) in the SQL Terminator Character box.

5 Click Apply or OK.

The SQL Terminator Character setting is applied to the current connection and all future connections (until you change it).

Creating the stored procedure

After setting up the Database painter, you can create an Oracle stored procedure that has a result set as an IN OUT (reference) parameter. DataWindow Designer retrieves the result set to populate a DataWindow object.

There are many ways to create stored procedures with result sets. The following procedure describes one possible method that you can use.

For information about when you can use stored procedures with single and multiple result sets, see "What you can do with Oracle stored procedures" on page 99.

❖ **To create Oracle stored procedures with result sets:**

1 Make sure your Oracle user account has the necessary database access and privileges to access Oracle objects (such as tables and procedures).

Without the appropriate access and privileges, you will be unable to create Oracle stored procedures.

2 Assume the following table named tt exists in your Oracle database:

a	b	c
1	Newman	sysdate
2	Everett	sysdate

3 Create an Oracle package that holds the result set type and stored procedure. The result type must match your table definition.

For example, the following statement creates an Oracle package named spm that holds a result set type named rctl and a stored procedure named proc1. The tt%ROWTYPE attribute defines rctl to contain all of the columns in table tt. The procedure proc1 takes one parameter, a cursor variable named rc1 that is an IN OUT parameter of type rctl.

```
CREATE OR REPLACE PACKAGE spm
   IS TYPE rctl IS REF CURSOR
   RETURN tt%ROWTYPE;
   PROCEDURE proc1(rc1 IN OUT rctl);END;`
```

4 Create the Oracle stored procedure separately from the package you defined.

The following example shows how to create a stored procedure named spm_proc 1 that returns a single result set.

The IN OUT specification means that DataWindow Designer passes the cursor variable (rc1 or rc2) by reference to the Oracle procedure and expects the procedure to open the cursor. After the procedure call, DataWindow Designer fetches the result set from the cursor and then closes the cursor.

spm_proc1 example for DataWindow objects The following statements create spm_proc1 which returns one result set. You can use this procedure as the data source for a DataWindow object in DataWindow Designer.

```
CREATE OR REPLACE PROCEDURE spm_proc1(rc1 IN OUT
spm.rctl)
AS
BEGIN
   OPEN rc1 FOR SELECT * FROM tt;
END;`
```

Error checking
If necessary, check the Oracle system table public.user_errors for a list of errors.

Creating the DataWindow object

After you create the stored procedure, you can define the DataWindow object that uses the stored procedure as a data source.

You can use Oracle stored procedures that return a single result set in a DataWindow object.

The following procedure assumes that your Oracle stored procedure returns only a single result set.

❖ **To create a DataWindow object using an Oracle stored procedure with a result set:**

1 Select a presentation style on the DataWindow page of the New dialog box and click OK.

2 Select the Stored Procedure icon and click OK.

The Select Stored Procedure wizard page displays, listing the stored procedures available in your database.

3 Select the stored procedure you want to use as a data source, and click Next.

4 Complete the wizard to define the DataWindow object.

When you preview the DataWindow object or call Retrieve, DataWindow Designer fetches the result set from the cursor in order to populate the DataWindow object. If you selected Retrieve on Preview on the Choose Data Source page in the wizard, the result set displays in the Preview view when the DataWindow opens.

Using a large-object output parameter

You can define a large object (LOB) as an output parameter for an Oracle stored procedure or function to retrieve large-object data. There is no limit on the number of LOB output arguments that can be defined for each stored procedure or function.

In Oracle 10*g*, the maximum size of LOB datatypes has been increased from 4 gigabytes minus 1 to 4 gigabytes minus 1 multiplied by the block size of the database. For a database with a block size of 32K, the maximum size is 128 terabytes.

Using Oracle user-defined types

DataWindow Designer supports SQL CREATE TYPE and CREATE TABLE statements for Oracle user-defined types (objects) in the ISQL view of the Database painter. It correctly handles SQL SELECT, INSERT, UPDATE, and DELETE statements for user-defined types in the Database and DataWindow painters.

This means that using the Oracle native database interfaces in DataWindow Designer, you can:

Do this	In
Use Oracle syntax to create user-defined types	Database painter
Use Oracle syntax to create tables with columns that reference user-defined types	Database painter
View columns in Oracle tables that reference user-defined types	Database painter
Manipulate data in Oracle tables that have user-defined types	Database painter DataWindow painter DataWindow objects
Export Oracle table syntax containing user-defined types to a log file	Database painter
Invoke methods	DataWindow painter (Compute tab in SQL Toolbox)

Example

Here is a simple example that shows how you might create and use Oracle user-defined types in DataWindow Designer.

For more information about Oracle user-defined types, see your Oracle documentation.

❖ **To create and use Oracle user-defined types:**

1 In the ISQL view of the Database painter, create two Oracle user-defined types: ball_stats_type and player_type.

Here is the Oracle syntax to create ball_stats_type. Notice that the ball_stats object of type ball_stats_type has a method associated with it called get_avg.

```
CREATE OR REPLACE TYPE ball_stats_type AS OBJECT
(bat_avg NUMBER(4,3),rbi NUMBER(3),MEMBER FUNCTION
get_avg RETURN NUMBER,PRAGMA RESTRICT_REFERENCES
(get_avg,WNDS,RNPS,WNPS));
CREATE OR REPLACE TYPE BODY ball_stats_type ASMEMBER
```

```
FUNCTION get_avg RETURN NUMBER ISBEGINRETURN
SELF.bat_avg;
END;
END;
```

Here is the Oracle SQL syntax to create player_type. Player_type references the user-defined type ball_stats_type. DataWindow Designer supports such nesting graphically in the Database, DataWindow, and Table painters (see step 3).

```
CREATE TYPE player_type AS OBJECT (player_no
NUMBER(2),player_name VARCHAR2(30),ball_stats
ball_stats_type);
```

2 In the Database painter, create a table named lineup that references these user-defined types.

Here is the Oracle SQL syntax to create the lineup table and insert a row. Lineup references the player_type user-defined type.

```
CREATE TABLE lineup (position NUMBER(2) NOT NULL,
player player_type);
INSERT INTO lineup VALUES (1,player_type (34, 'David
Ortiz', ball_stats_type (0.321, 60)));
```

3 Display the lineup table in the Database or DataWindow painter.

DataWindow Designer uses the following structure->member notation to display the table:

```
lineup
======
position
player->player_no
player->player_name
player->ball_stats->bat_avg
player->ball_stats->rbi
```

4 To access the get_avg method of the object ball_stats contained in the object column player, use the following structure->member notation when defining a computed column for the DataWindow object. For example, when working in the DataWindow painter, you could use this notation on the Compute tab in the SQL Toolbox:

```
player->ball_stats->get_avg()
```

CHAPTER 10

Using DirectConnect

About this chapter

This chapter describes how to use the DirectConnect™ interface in DataWindow Designer.

Contents

Using the DirectConnect interface

The DirectConnect interface uses Sybase's Open Client CT-Library (CT-Lib) API to access a database through Sybase middleware data access products such as the DirectConnect for OS/390 component of MainFrame Connect and Open ServerConnect™.

Accessing Unicode data

DataWindow Designer can connect, save, and retrieve data in both ANSI/DBCS and Unicode databases. When character data or command text is sent to the database, DataWindow Designer sends a DBCS string if the UTF8 database parameter is set to 0 (the default). If UTF8 is set to 1, DataWindow Designer sends a UTF-8 string.

The database server must have the UTF-8 character set installed. See the description of the UTF-8 database parameter in the online Help for more information.

A Unicode database is a database whose character set is set to a Unicode format, such as UTF-8, UTF-16, UCS-2, or UCS-4. All data must be in Unicode format, and any data saved to the database must be converted to Unicode data implicitly or explicitly.

A database that uses ANSI (or DBCS) as its character set might use special datatypes to store Unicode data. Columns with these datatypes can store *only* Unicode data. Any data saved into such a column must be converted to Unicode explicitly. This conversion must be handled by the database server or client.

Connecting through the DirectConnect middleware product

Sybase DirectConnect is a data access server that provides a standardized middleware interface between your applications and your enterprise data sources. Data access services to a particular database are defined in a DirectConnect server. Since a DirectConnect server can support multiple access services, you can access multiple databases through a single server.

When you use the DirectConnect interface to connect to a particular database, your connection is routed through the access service for that database. An access service consists of a named set of configuration properties and a specific access service library.

To access DB2 data on an IBM mainframe through a DirectConnect server, you can use the DirectConnect interface to connect through either a DirectConnect for MVS access service or a DirectConnect Transaction Router Service (TRS).

TRS provides fast access to a DB2/MVS database by using remote stored procedures. The DirectConnect interface supports both versions of the TRS library: TRSLU62 and TRSTCP.

The DirectConnect server operates in two modes: SQL transformation and passthrough. The DirectConnect interface for DB2/MVS uses passthrough mode, which allows your DataWindow Designer application to have direct access to the capabilities of the DB2/MVS data source.

Connecting through the Open ServerConnect middleware product

Sybase's Open ServerConnect supports mainframe applications that retrieve and update data stored on the mainframe that Sybase client applications can execute. Client applications can connect directly to a DB2/MVS database through an Open ServerConnect application residing on the mainframe, eliminating the need for an intermediate gateway like DirectConnect. (This type of connection is also known as a *gateway-less* connection.) In addition, an Open ServerConnect application presents mainframe Remote Procedure Calls (RPCs) as database stored procedures to the client application.

To access DB2 data on an IBM mainframe through Open ServerConnect, you can use the DirectConnect interface to connect through Open ServerConnect for IMS and MVS.

Selecting the type of connection

To select how DataWindow Designer accesses the database, use the Choose Gateway drop-down list on the Connection tab of the DirectConnect Database Profile Setup dialog box and select one of the following:

- Access Service
- Gatewayless
- TRS

All the DBParm parameters defined for the DirectConnect interface are applicable to all three connections except the following:

- HostReqOwner applies to Access Service and Gatewayless only
- Request, ShowWarnings, and SystemOwner apply to Access Service only
- UseProcSyntax applies to Gatewayless only

See the online help for the complete list of DBParm parameters applicable to the DirectConnect interface.

Supported versions for the DirectConnect interface

The DirectConnect interface uses a DLL named *PBDIR110.DLL* to access a database through either DirectConnect or Open ServerConnect.

Required DirectConnect versions

To access a DB2/MVS database through the access service, it is strongly recommended that you use DirectConnect for MVS access service version 11.1.1p4 or later.

To access a DB2/MVS database through TRS, it is strongly recommended that you use DirectConnect TRS version 11.1.1p4 or later.

For information on DirectConnect for MVS and TRS, see your DirectConnect documentation.

Required Open ServerConnect versions

To access a DB2/MVS database through Open ServerConnect, it is strongly recommended that you use Open ServerConnect IMS and MVS version 4.0 or later.

For information on Open ServerConnect for MVS, see your Open ServerConnect documentation.

Supported DirectConnect interface datatypes

The DirectConnect interface supports the DataWindow Designer datatypes listed in Table 10-1 in DataWindow objects.

Table 10-1: Supported datatypes for DirectConnect

Char (fewer than 255 characters)	Long VarChar
Char for Bit Data	Real
Date	SmallInt
Decimal	Time
Double Precision	Timestamp (DateTime)
Float	VarChar
Integer	VarChar for Bit Data

Basic software components for the DirectConnect interface

Figure 10-1 shows the basic software components required to access a database using the DirectConnect interface and the DirectConnect middleware data access product.

Figure 10-1: Components of a DirectConnect connection using DirectConnect middleware

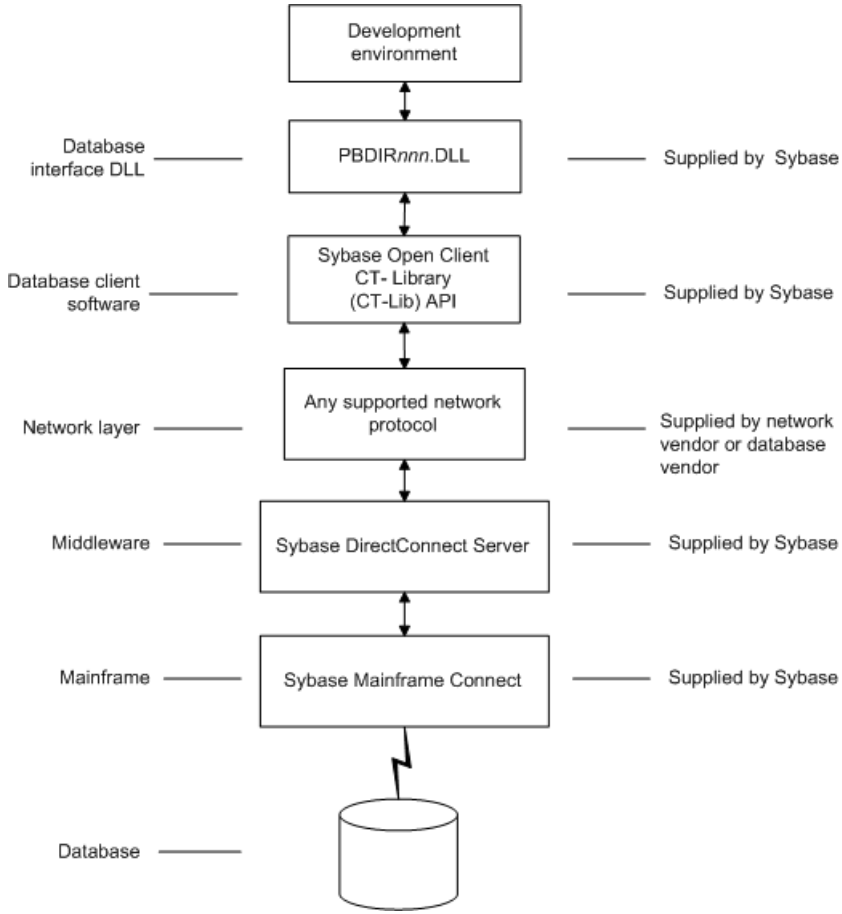
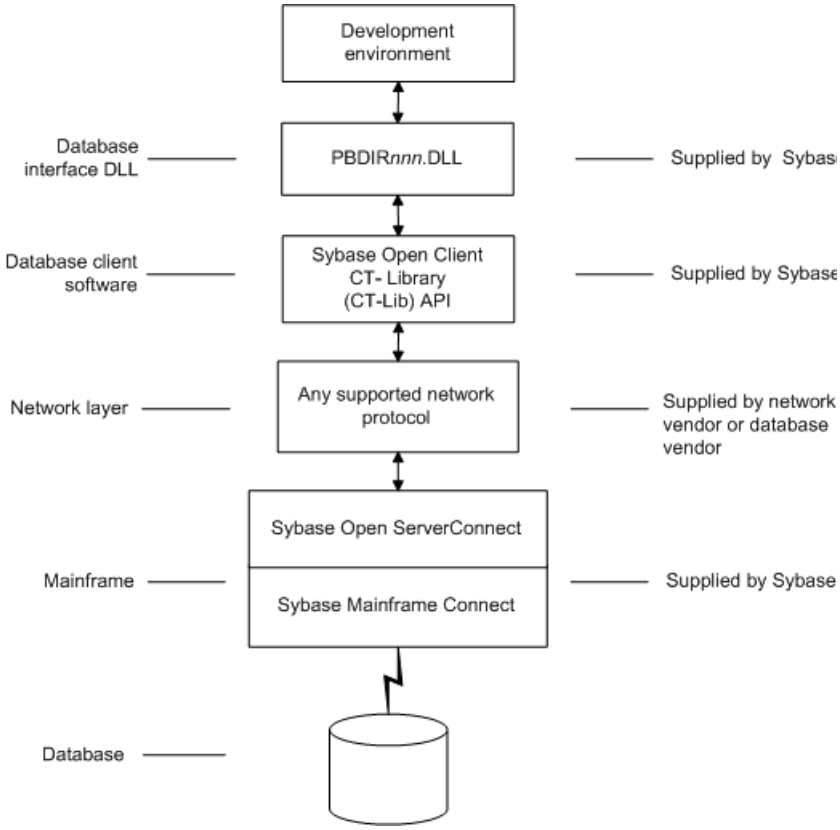


Figure 10-2 shows the basic software components required to access a database using the DirectConnect interface and the Open ServerConnect middleware data access product.

Figure 10-2: Components of a DirectConnect connection using Open ServerConnect middleware



Preparing to use the database with DirectConnect

Before you define the interface and connect to a database through the DirectConnect interface, follow these steps to prepare the database for use:

1 Install and configure the Sybase middleware data access products, network, and client software.

2 Install the DirectConnect interface.

3 Verify that you can connect to your middleware product and your database outside DataWindow Designer.

4 Create the extended attribute system tables outside DataWindow Designer.

Step 1: Install and configure the Sybase middleware product

You must install and configure the Sybase middleware data access product, network, and client software.

❖ **To install and configure the Sybase middleware data access product, network, and client software:**

1 Make sure the appropriate database software is installed and running on its server.

You must obtain the database server software from your database vendor.

For installation instructions, see your database vendor's documentation.

2 Make sure the appropriate DirectConnect access service software is installed and running on the DirectConnect server specified in your database profile
or
Make sure the appropriate Open ServerConnect software is installed and running on the mainframe specified in your database profile.

3 Make sure the required network software (such as TCP/IP) is installed and running on your computer and is properly configured so you that can connect to the DirectConnect server or mainframe at your site.

You must install the network communication driver that supports the network protocol and operating system platform you are using.

For installation and configuration instructions, see your network or database administrator.

4 Install the required Open Client CT-Library (CT-Lib) software on each client computer on which DataWindow Designer is installed.

You must obtain the Open Client software from Sybase. Make sure the version of Open Client you install supports *both* of the following:

The operating system running on the client computer
The version of DataWindow Designer that you are running

Required Open Client versions
To use the DirectConnect interface, you must install Open Client.

For information about Open Client, see your Open Client documentation.

5 Make sure the Open Client software is properly configured so you can connect to the middleware data access product at your site.

Installing the Open Client software places the *SQL.INI* configuration file in the SQL Server directory on your computer. *SQL.INI* provides information that SQL Server uses to find and connect to the middleware product at your site. You can enter and modify information in *SQL.INI* with the configuration utility or editor that comes with the Open Client software.

For information about editing the *SQL.INI* file, see "Editing the SQL.INI file" on page 113. For more information about setting up *SQL.INI* or any other required configuration file, see your SQL Server documentation.

6 If required by your operating system, make sure the directory containing the Open Client software is in your system path.

7 Make sure only one copy of each of the following files is installed on your client computer:

- DirectConnect interface DLL
- Network communication DLL (such as *NLWNSCK.DLL* for Windows Sockets-compliant TCP/IP)
- Open Client DLLs (such as *LIBCT.DLL* and *LIBCS.DLL*)

Step 2: Install the interface

In the DataWindow Designer Setup program, select the Typical install, or select the Custom install and select the Direct Connect Interface (DIR).

Step 3: Verify the connection

Make sure you can connect to your middleware product and your database and log in to the database you want to access from outside DataWindow Designer.

Some possible ways to verify the connection are by running the following tools:

- **Accessing the database server** Tools such as the Open Client/Open Server Configuration utility (or any Ping utility) check whether you can reach the database server from your computer.
- **Accessing the database** Tools such as ISQL or SQL Advantage (interactive SQL utilities) check whether you can log in to the database and perform database operations. It is a good idea to specify the same connection parameters you plan to use in your DataWindow Designer database profile to access the database.

Editing the SQL.INI file

Make sure the *SQL.INI* file provides an entry about either the access service being used and the DirectConnect server on which it resides or the Open ServerConnect program being used and the mainframe on which it resides.

For the server object name, you need to provide the exact access service name as it is defined in the access service library configuration file on the DirectConnect server. You must also specify the network communication DLL being used, the TCP/IP address or alias used for the DirectConnect server on which the access service resides, and the port on which the DirectConnect server listens for requests:

```
[access_service_name]
query=network_dll,server_alias,server_port_no
```

DataWindow Designer users must also specify the access service name in the SQLCA.ServerName property of the Transaction object.

Defining the DirectConnect interface

To define a connection through the DirectConnect interface, you must create a database profile by supplying values for at least the basic connection parameters in the Database Profile Setup - DirectConnect dialog box. You can then select this profile anytime to connect to your database in the plug-in.

For information on how to define a database profile, see “Using database profiles” on page 6.

PART 4

Working with Database Connections

This part describes how to establish, manage, and troubleshoot database connections.

CHAPTER 11

Managing Database Connections

About this chapter

After you install the necessary database software and define the database interface, you can connect to the database from DataWindow Designer. Once you connect to the database, you can work with the tables and views stored in that database.

This chapter describes how to connect to a database in DataWindow Designer, maintain database profiles, and share database profiles.

Contents

Terminology

In this chapter, the term **database** refers to *both* of the following unless otherwise specified:

- A database or DBMS that you access with a standard database interface and appropriate driver
- A database or DBMS that you access with the appropriate native database interface

About database connections

This section gives an overview of when database connections occur in DataWindow Designer. It also explains why you should use database profiles to manage your database connections.

When database connections occur

Connections in DataWindow Designer

DataWindow Designer connects to the database profile that is set as the active connection in the Database painter when you:

- Create or open a DataWindow object
- Open the Database painter

If you install the EAS Demo database and the ODBC interface when you install the DataWindow Designer plug-in, the EAS Demo DB is set as the active connection when you first use the plug-in. When you create or connect to a different profile in the Database painter, that profile becomes the new active connection and is written to the Windows registry.

An active connection is required
You need to establish an active connection in the Database painter before you use a DataWindow wizard to create a new DataWindow object.

What's in this book

This book describes how to connect to your database when you are working in the DataWindow Designer plug-in.

Using database profiles

What is a database profile?

A **database profile** is a named set of parameters stored in the registry that defines a connection to a particular database in the DataWindow Designer plug-in.

Why use database profiles?

Creating and using database profiles is the easiest way to manage your database connections in DataWindow Designer because you can:

- Select a database profile to establish or change database connections. You can easily connect to another database anytime during a DataWindow Designer session. This is particularly useful if you often switch between different database connections.
- Edit a database profile to modify or supply additional connection parameters.
- Use the Preview tab page to test a connection and copy the connection syntax to your application code.
- Delete a database profile if you no longer need to access that data.
- Import and export profiles.

Because database profiles are created when you define your data and are stored in the registry, they have the following benefits:

- They are always available to you.
- Connection parameters supplied in a database profile are saved until you edit or delete the database profile.

Connecting to a database

To establish or change a database connection in DataWindow Designer, use a database profile. You can select the database profile for the database you want to access in the Objects view in the Database painter. For how to create a database profile, see "Creating a database profile" on page 8.

Selecting a database profile

Database painter Objects view

You can select a database profile from the Database painter Objects view.

❖ **To connect to a database:**

1 Select View>Database Painter from the Visual Studio menu bar.

The Database painter displays. The Objects view lists your installed database interfaces.

Where the interface list comes from
When you run the Setup program, it updates the Vendors list in the registry with the interfaces you install. The Database painter Objects view displays the same interfaces that appear in the Vendors list.

2 Click the plus sign (+) to the left of the interface you are using or double-click the name.

The list expands to display the database profiles defined for your interface.

3 Select the name of the database profile you want to access and click the Connect button, or display the pop-up menu for a database profile and select Connect.

What happens when you connect

When you connect to a database by selecting its database profile, DataWindow Designer writes the profile name and its connection parameters to the registry key *HKEY_CURRENT_USER\Software\Sybase\DataWindowDesignerPlugin\2.5\DatabaseProfiles\PowerBuilder.*

Each time you connect to a different database, DataWindow Designer overwrites the "most-recently used" profile name in the registry with the name for the new database connection.

Specifying passwords in database profiles

Your password does *not* display when you specify it in the Database Profile Setup dialog box.

However, when DataWindow Designer stores the values for this profile in the registry, the actual password *does* display, in encrypted form, in the DatabasePassword or LogPassword field.

Suppressing display in the profile registry entry

To suppress password display in the profile registry entry, do the following when you create a database profile.

❖ **To suppress password display in the profile registry entry:**

1. Select the Prompt For Database Information check box on the Connection tab in the Database Profile Setup dialog box.

   This tells DataWindow Designer to prompt for any missing information when you select this profile to connect to the database.

2. Leave the Password box blank. Instead, specify the password in the dialog box that displays to prompt you for additional information when you connect to the database.

What happens

When you specify the password in response to a prompt instead of in the Database Profile Setup dialog box, the password does not display in the registry entry for this profile.

For example, if you do not supply a password in the Database Profile Setup - Adaptive Server Enterprise dialog box when creating a database profile, the Client Library Login dialog box displays to prompt you for the missing information.

Maintaining database profiles

You can easily edit or delete an existing database profile in DataWindow Designer.

You can edit a database profile to change one or more of its connection parameters. You can delete a database profile when you no longer need to access its data. You can also change a profile using the Database painter.

What happens

When you edit or delete a database profile, DataWindow Designer either updates the database profile entry in the registry or removes it.

Deleting a profile for an ODBC data source
If you delete a database profile that connects to an ODBC data source, DataWindow Designer does *not* delete the corresponding data source definition from the ODBC initialization file. This lets you re-create the database profile later if necessary without having to redefine the data source.

Importing and exporting database profiles

Each database interface provides an Import Profile(s) and an Export Profile(s) option. You can use the Import option to import a previously defined profile for use with an installed database interface. Conversely, you can use the Export option to export a defined profile for use by another user.

The ability to import and export profiles provides a way to move profiles easily between developers. It also means you no longer have to maintain a shared file to maintain profiles. It is ideal for mobile development when you cannot rely on connecting to a network to share a file.

❖ **To import a profile:**

1 Highlight a database interface and select Import Profile(s) from the pop-up menu. (In the Database painter, select Import Profile(s) from the File or pop-up menu.)

2 From the Select Profile File dialog box, select the file whose profiles you want to import and click Save.

3 Select the profile(s) you want to import from the Import Profile(s) dialog box and click OK.

The profiles are copied into your registry. If a profile with the same name already exists, you are asked if you want to overwrite it.

❖ **To export a profile:**

1 Highlight a database interface and select Export Profile(s) from the pop-up menu. (In the Database painter, select Export Profile(s) from the File or pop-up menu.)

2 Select the profile(s) you want to export from the Export Profile(s) dialog box and click OK.

The Export Profile(s) dialog box lists all profiles defined in your registry regardless of the database interface for which they were defined. By default, the profiles defined for the selected database interface are marked for export.

3 From the Select Profile File dialog box, select a directory and a file in which to save the exported profile(s) and click Save.

The exported profiles can be saved to a new or existing file. If saved to an existing file, the profile(s) are added to the existing profiles. If a profile with the same name already exists, you are asked if you want to overwrite it.

About the DataWindow Designer extended attribute system tables

DataWindow Designer uses a collection of five system tables to store extended attribute information (such as display formats, validation rules, and font information) about tables and columns in your database. You can also define extended attributes when you create or modify a table in DataWindow Designer.

This section tells you how to:

- Make sure the DataWindow Designer extended attribute system tables are created with the proper access rights when you log in to your database for the first time
- Display and open a DataWindow Designer extended attribute system table

- Understand the kind of information stored in the DataWindow Designer extended attribute system tables
- Control extended attribute system table access

Logging in to your database for the first time

By default, DataWindow Designer creates the extended attribute system tables the first time you connect to a database.

To ensure that DataWindow Designer creates the extended attribute system tables with the proper access rights to make them available to all users, the first person to connect to the database with DataWindow Designer must log in with the proper authority.

❖ **To ensure proper creation of the DataWindow Designer extended attribute system tables:**

- Make sure the first person to connect to the database with DataWindow Designer has sufficient authority to create tables and grant permissions to PUBLIC.

  This means that the first person to connect to the database should log in as the database owner, database administrator, system user, system administrator, or system owner, as specified by your DBMS.

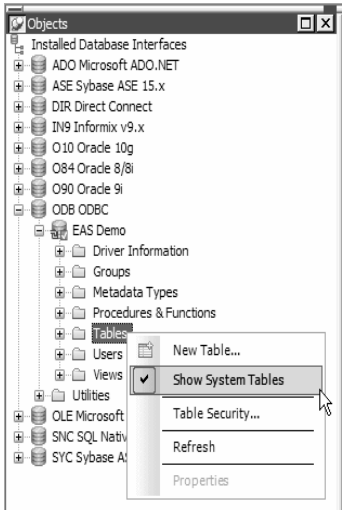
Displaying the DataWindow Designer extended attribute system tables

DataWindow Designer updates the extended attribute system tables automatically whenever you change the information for a table or column. The DataWindow Designer extended attribute system tables are different from the system tables provided by your DBMS.

You can display and open DataWindow Designer extended attribute system tables in the Database painter just like other tables.

❖ **To display the DataWindow Designer extended attribute system tables:**

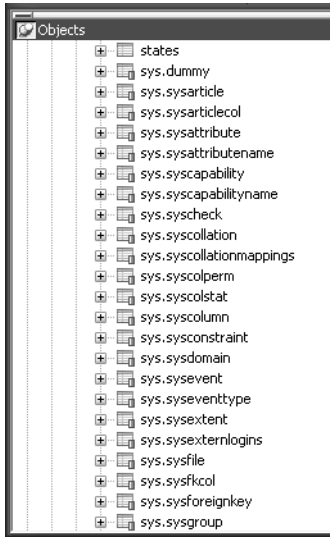
1 In the Database painter, highlight Tables in the list of database objects for the active connection and select Show System Tables from the pop-up menu.



2 The DataWindow Designer extended attribute system tables and DBMS system tables display in the tables list, as follows:

- **DataWindow Designer system tables** The five system tables are: pbcatcol, pbcatedt, pbcatfmt, pbcattbl, and pbcatvld.

- **DBMS system tables** The system tables supplied by the DBMS usually have a DBMS-specific prefix (such as *sys* or *dbo*).



3 Display the contents of a DataWindow Designer system table in the Object Layout, Object Details, and/or Columns views.

For instructions, see the *User's Guide*.

Do not edit the extended attribute system tables
Do not change the values in the DataWindow Designer extended attribute system tables.

Contents of the extended attribute system tables

DataWindow Designer stores five types of extended attribute information in the system tables as described in Table 11-1.

Table 11-1: Extended attribute system tables

System table	Information about	Attributes
pbcatcol	Columns	Names, comments, headers, labels, case, initial value, and justification
pbcatedt	Edit styles	Edit style names and definitions
pbcatfmt	Display formats	Display format names and definitions
pbcattbl	Tables	Name, owner, default fonts (for data, headings and labels), and comments
pbcatvld	Validation rules	Validation rule names and definitions

For more about the DataWindow Designer system tables, see the Appendix in the *User's Guide*.

Prefixes in system table names

For some databases, DataWindow Designer precedes the name of the system table with a default DBMS-specific prefix. For example, the names of DataWindow Designer system tables have the prefix DBO in a SQL Server database (such as DBO.pbcatcol), or SYSTEM in an Oracle database (such as SYSTEM.pbcatfmt).

The preceding table gives the base name of each system table without the DBMS-specific prefix.

Controlling system table access

To control access to the DataWindow Designer system tables at your site, you can specify that DataWindow Designer not create or update the system tables or that the system tables be accessible only to certain users or groups.

You can control system table access by doing any of the following:

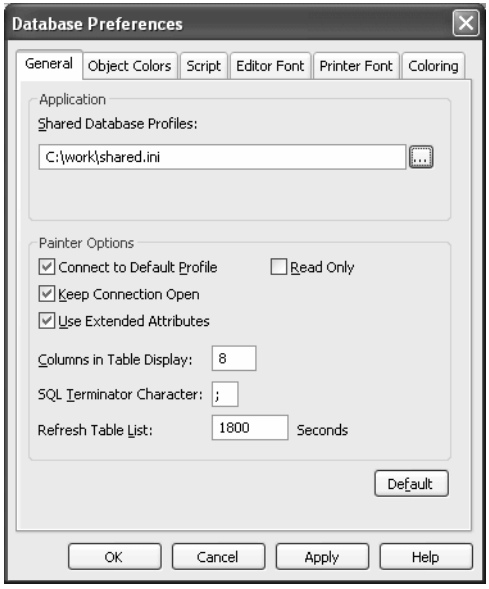
- **Setting Use Extended Attributes** Set the Use Extended Attributes database preference in the Database Preferences dialog box in the Database painter.

- **Setting Read Only** Set the Read Only database preference in the Database Preferences dialog box in the Database painter.
- **Granting permissions on the system tables** Grant explicit permissions on the system tables to users or groups at your site.

Setting Use Extended Attributes or Read Only to control access

❖ **To control system table access by setting Use Extended Attributes or Read Only:**

1 Select Design>Options from the menu bar to display the Database Preferences dialog box.



2 On the General page, set values for Use Extended Attributes or Read Only as follows:

Preference	What you do	Effect
Use Extended Attributes	Clear the check box	Does not create the DataWindow Designer system tables if they do not exist. Instead, the painter uses the appropriate default values for extended attributes (such as headers, labels, and text color). If the DataWindow Designer system tables already exist, DataWindow Designer does not use them when you create a new DataWindow object.
Read Only	Select the check box	If the DataWindow Designer system tables already exist, DataWindow Designer uses them when you create a new DataWindow object, *but does not update them.* You *cannot* modify (update) information in the system tables or any other database tables in the DataWindow painter when the Read Only check box is selected.

3 Click OK.

DataWindow Designer applies the preference settings to the current connection and all future connections and saves them in the registry.

Granting permissions on system tables to control access

If your DBMS supports SQL GRANT and REVOKE statements, you can control access to the DataWindow Designer system tables. The default authorization for each repository table is:

```
GRANT SELECT, UPDATE, INSERT, DELETE ON table TO PUBLIC
```

After the system tables are created, you can (for example) control access to them by granting SELECT authority to end users and SELECT, UPDATE, INSERT, and DELETE authority to developers. This technique offers security and flexibility that is enforced by the DBMS itself.

CHAPTER 12

Setting Additional Connection Parameters

About this chapter

To fine-tune your database connection and take advantage of DBMS-specific features that your interface supports, you can set additional connection parameters at any time. These additional connection parameters include:

- Database parameters
- Database preferences

These connection parameters are described in the Database Connectivity section in the online Help.

This chapter describes how to set database parameters and database preferences in DataWindow Designer.

Contents

Basic steps for setting connection parameters

This section gives basic steps for setting database parameters and database preferences in DataWindow Designer.

❖ **To set database parameters:**

1 Learn how to set database parameters in the plug-in or in code.

See “Setting database parameters” on page 131.

2 Determine the database parameters you can set for your database interface.

For a table listing each supported database interface and the database parameters you can use with that interface, see "Database parameters and supported database interfaces" in the online Help.

3 Read the description of the database parameter you want to set in the online Help.

4 Set the database parameter for your database connection.

❖ **To set database preferences:**

1 Learn how to set database preferences in the plug-in.

See "Setting database preferences" on page 133.

2 Determine the database preferences you can set for your DBMS.

For a table listing each supported database interface and the database preferences you can use with that interface, see "Database parameters and supported database interfaces" in the online Help.

3 Read the description of the database preference you want to set in the online Help.

4 Set the database preference for your database connection.

About the Database Profile Setup dialog box

The interface-specific Database Profile Setup dialog box makes it easy to set additional connection parameters in the plug-in or in code. You can:

- Supply values for connection options supported by your database interface

  Each database interface has its own Database Profile Setup dialog box that includes settings only for those connection parameters supported by the interface. Similar parameters are grouped on the same tab page. The Database Profile Setup dialog box for *all* interfaces includes the Connection tab and Preview tab. Depending on the requirements and features of your interface, one or more other tab pages might also display.

- Easily set additional connection parameters in the plug-in

  You can specify additional connection parameters with easy-to-use check boxes, drop-down lists, and text boxes. DataWindow Designer generates the proper syntax automatically when it saves your database profile in the system registry.

- Generate connection syntax for use in your code

  As you complete the Database Profile Setup dialog box in DataWindow Designer, the correct connection syntax for each selected option is generated on the Preview tab. DataWindow Designer assigns the corresponding database parameter or transaction object property name to each option and inserts quotation marks, commas, semicolons, and other characters where needed. You can copy the syntax you want from the Preview tab into your code.

Setting database parameters

In DataWindow Designer, you can set database parameters by editing the Database Profile Setup dialog box for your connection. You can also specify them in your code in DataWindow .NET.

Setting database parameters in the plug-in

Editing database profiles

To set database parameters for a database connection in the DataWindow Designer plug-in, you must edit the database profile for that connection.

Character limit for strings

Strings containing database parameters that you specify in the Database Profile Setup dialog box for your connection can be up to 999 characters in length.

This limit applies only to database parameters that you set in a database profile in the development environment. Database strings specified in code as properties of the Transaction object are *not* limited to a specified length.

Setting database parameters in code

If you are developing an application that connects to a database, you must specify the required connection parameters in the appropriate code as properties of the Transaction object.

One of the connection parameters you might want to specify in code is DbParameter. You can do this by copying DbParameter syntax from the Preview tab in the Database Profile Setup dialog box into your code.

Copying DbParameter syntax from the Preview tab

The easiest way to specify DbParameter parameters in code is to copy the DbParameter syntax from the Preview tab in the Database Profile Setup dialog box into your code, modifying the default Transaction object name (SQLCA) if necessary.

As you set parameters in the Database Profile Setup dialog box in the plug-in, DataWindow Designer generates the correct connection syntax on the Preview tab. Therefore, copying the syntax directly from the Preview tab ensures that you use the correct DbParameter syntax in your code.

❖ **To copy DBParm syntax from the Preview tab into your code:**

1 On one or more tab pages in the Database Profile Setup dialog box for your connection, supply values for any parameters you want to set.

For instructions, see “Setting database parameters in the plug-in” on page 131.

For information about the parameters for your interface and the values to supply, click Help.

2 Click Apply to save your changes to the current tab without closing the Database Profile Setup dialog box.

3 Click the Preview tab.

The correct DbParameter syntax for each selected option displays in the Database Connection Syntax box.

4 Select one or more lines of text in the Database Connection Syntax box and click Copy.

DataWindow Designer copies the selected text to the clipboard.

5 Click OK to close the Database Profile Setup dialog box.

6 Paste the selected text from the Preview tab into your code, modifying the default Transaction object name (SQLCA) if necessary.

Setting database preferences

How to set

The way you set connection-related database preferences in DataWindow Designer varies, as summarized in the following table.

Table 12-1: Database preferences and where they can be set

Database preference	Set in plug-in by editing
AutoCommit	Database Profile Setup dialog box for your connection
Lock	Database Profile Setup dialog box for your connection
Shared Database Profiles	Database Preferences dialog box
Connect DB at Startup	Database Preferences dialog box
Connect to Default Profile	Database Preferences dialog box
Read Only	Database Preferences dialog box
Keep Connection Open	Database Preferences dialog box
Use Extended Attributes	Database Preferences dialog box
SQL Terminator Character	Database Preferences dialog box

The following sections give the steps for setting database preferences in the plug-in.

For more information

For information about using a specific database preference, see its description in the online Help.

Setting database preferences in the plug-in

There are two ways to set database preferences in the DataWindow Designer plug-in on *all* supported development platforms, depending on the preference you want to set:

- Set AutoCommit and Lock (Isolation Level) in the Database Profile Setup dialog box for your connection

ADO.NET
For ADO.NET, Isolation is a database parameter.

- Set all other database preferences in the Database Preferences dialog box in the Database painter

Setting AutoCommit and Lock in the database profile

The AutoCommit and Lock (Isolation Level) preferences are properties of the default Transaction object, SQLCA. For AutoCommit and Lock to take effect in the DataWindow Designer plug-in, you must specify them *before* you connect to a database. Changes to these preferences after the connection occurs have no effect on the current connection.

To set AutoCommit and Lock before DataWindow Designer connects to your database, you specify their values in the Database Profile Setup dialog box for your connection.

❖ **To set AutoCommit and Lock (Isolation Level) in a database profile:**

1 Display the Database Profiles dialog box.

2 Click the plus sign (+) to the left of the interface you are using
or
Double-click the interface name.

The list expands to display the database profiles defined for your interface.

3 Select the name of the profile you want and click Edit.

The Database Profile Setup dialog box for the selected profile displays.

4 On the Connection tab page, supply values for one or both of the following:

- **Isolation Level** If your database supports the use of locking and isolation levels, select the isolation level you want to use for this connection from the Isolation Level drop-down list. (The Isolation Level drop-down list contains valid lock values for your interface.)
- **AutoCommit Mode** The setting of AutoCommit controls whether DataWindow Designer issues SQL statements outside (True) or inside (False) the scope of a transaction. *If your database supports it*, select the AutoCommit Mode check box to set AutoCommit to True or clear the AutoCommit Mode check box (the default) to set AutoCommit to False.

For example, in addition to values for basic connection parameters (Server, Login ID, Password, and Database), the Connection tab page for the following Sybase Adaptive Server Enterprise profile named Sales shows nondefault settings for Isolation Level and AutoCommit Mode.

5 (Optional) In DataWindow Designer, click the Preview tab if you want to see the connection syntax generated for Lock and AutoCommit.

DataWindow Designer generates correct connection syntax for each option you set and for your preferred language in the Database Profile Setup dialog box. You can copy this syntax directly into your code.

6 Click OK to close the Database Profile Setup dialog box.

DataWindow Designer saves your settings in the database profile entry in the registry.

Setting preferences in the Database Preferences dialog box

To set the following connection-related database preferences, complete the Database Preferences dialog box in the DataWindow Designer Database painter:

- Shared Database Profiles
- Connect to Default Profile
- Read Only
- Keep Connection Open
- Use Extended Attributes
- SQL Terminator Character

Other database preferences
The Database Preferences dialog box also lets you set other database preferences that affect the behavior of the Database painter itself. For information about the other preferences you can set in the Database Preferences dialog box, see the *User's Guide*.

❖ **To set connection-related preferences in the Database Preferences dialog box:**

1 Open the Database painter.

2 Select Design>Options from the menu bar.

The Database Preferences dialog box displays. If necessary, click the General tab to display the General property page.

3 Specify values for one or more of the connection-related database preferences in the following table.

Table 12-2: Connection-related database preferences

Preference	Description	For details, see
Shared Database Profiles	Specifies the pathname of the file containing the database profiles you want to share. You can type the pathname or click Browse to display it.	Not applicable in the DataWindow Designer plug-in
Connect to Default Profile	Controls whether the Database painter establishes a connection to a database using a default profile when the painter is invoked. If not selected, the Database painter opens without establishing a connection to a database.	Connect to Default Profile in online Help
Read Only	Specifies whether DataWindow Designer should update the extended attribute system tables and any other tables in your database. Select or clear the Read Only check box as follows: • **Select the check box** Does not update the extended attribute system tables or any other tables in your database. You *cannot* modify (update) information in the extended attribute system tables or any other database tables from the DataWindow painter when the Read Only check box is selected. • **Clear the check box** (Default) Updates the extended attribute system tables and any other tables in your database.	Read Only in the online Help
Keep Connection Open	When you connect to a database in DataWindow Designer without using a database profile, specifies when DataWindow Designer closes the connection. Select or clear the Keep Connection Open check box as follows: • **Select the check box** (Default) Stays connected to the database throughout your session and closes the connection when you exit • **Clear the check box** Opens the connection only when a painter requests it and closes the connection when you close a painter or finish compiling a script	Keep Connection Open in the online Help
Use Extended Attributes	Specifies whether DataWindow Designer should create and use the extended attribute system tables. Select or clear the Use Extended Attributes check box as follows: • **Select the check box** (Default) Creates and uses the extended attribute system tables • **Clear the check box** Does *not* create the extended attribute system tables	Use Extended Attributes in the online Help
Columns in Table Display	Specify the number of table columns to be displayed when InfoMaker displays a table graphically. The default is eight.	

4 Do one of the following:

- Click Apply to apply the preference settings to the current connection without closing the Database Preferences dialog box.
- Click OK to apply the preference settings to the current connection and close the Database Preferences dialog box.

DataWindow Designer saves your preference settings in the database section of *DW.INI*.

CHAPTER 13

Troubleshooting Your Connection

About this chapter

This chapter describes how to troubleshoot your database connection in DataWindow Designer by using the following tools:

- Database Trace
- ODBC Driver Manager Trace

Contents

Using the Database Trace tool

This section describes how to use the Database Trace tool.

About the Database Trace tool

The Database Trace tool records the internal commands that DataWindow Designer executes while accessing a database. You can trace a database connection in the plug-in.

DataWindow Designer writes the output of Database Trace to a log file named *DBTRACE.LOG* (by default) or to a nondefault log file that you specify. When you enable database tracing for the first time, DataWindow Designer creates the log file on your computer. Tracing continues until you disconnect from the database.

Using the Database Trace tool with one connection
You can use the Database Trace tool for only one DBMS at a time and for one database connection at a time.

For example, if your application connects to both an ODBC data source and an Adaptive Server Enterprise database, you can trace either the ODBC connection or the Adaptive Server Enterprise connection, but not both connections at the same time.

How you can use the Database Trace tool

You can use information from the Database Trace tool to understand what DataWindow Designer is doing *internally* when you work with your database. Examining the information in the log file can help you:

- Understand how DataWindow Designer interacts with your database
- Identify and resolve problems with your database connection
- Provide useful information to Technical Support if you call them for help with your database connection

If you are familiar with DataWindow Designer and your DBMS, you can use the information in the log to help troubleshoot connection problems on your own. If you are less experienced or need help, run the Database Trace tool *before* you call Technical Support. You can then report or send the results of the trace to the Technical Support representative who takes your call.

Contents of the Database Trace log

Default contents of the trace file

By default, the Database Trace tool records the following information in the log file when you trace a database connection:

- Parameters used to connect to the database
- Time to perform each database operation (in microseconds)
- The internal commands executed to retrieve and display table and column information from your database. Examples include:
  - Preparing and executing SQL statements such as SELECT, INSERT, UPDATE, and DELETE
  - Getting column descriptions
  - Fetching table rows

  - Binding user-supplied values to columns (if your database supports bind variables)
  - Committing and rolling back database changes
- Disconnecting from the database
- Shutting down the database interface

You can opt to include the names of DBI commands and the time elapsed from the last database connection to the completion of processing for each log entry. You can exclude binding and timing information as well as the data from all fetch requests.

Database Trace dialog box selections

The Database Trace dialog box lets you select the following items for inclusion in or exclusion from a database trace file:

- **Bind variables** Metadata about the result set columns obtained from the database
- **Fetch buffers** Data values returned from each fetch request
- **DBI names** Database interface commands that are processed
- **Time to implement request** Time required to process DBI commands; the interval is measured in thousandths of milliseconds (microseconds)
- **Cumulative time** Cumulative total of timings since the database connection began; the timing measurement is in thousandths of milliseconds

Registry settings for DBTrace

The selections made in the Database Trace dialog box are saved to the registry. Windows registry settings for the database trace utility configuration are stored under the *HKEY_CURRENT_USER\Software\Sybase\DataWindow Designer\2.0\DBTrace* key. Registry strings under this key are: ShowBindings, FetchBuffers, ShowDBINames, Timing, SumTiming, LogFileName, and ShowDialog. Except for the LogFileName string to which you can assign a full file name for the trace output file, all strings can be set to either 0 or 1.

The ShowDialog registry string can be set to prevent display of the Database Trace dialog box when a database connection is made with tracing enabled. This is the only one of the trace registry strings that you cannot change from the Database Trace dialog box. You must set ShowDialog to 0 in the registry to keep the configuration dialog box from displaying.

INI file settings for DBTrace

If you do not have access to the registry, you can use PB.INI to store trace file settings. Add a [DbTrace] section to the INI file with at least one of the following values set, then restart DataWindow Designer:

```
[DbTrace]
```

```
ShowDBINames=0
FetchBuffers=1
ShowBindings=1
SumTiming=1
Timing=1
ShowDialog=1
LogFileName=dbtrace.log
```

The keywords are the same as in the registry and have the same meaning. When you connect to the database again, the initial settings are taken from the INI file, and when you modify them, the changes are written to the INI file.

If the file name for LogFileName does not include an absolute path, the log file is written to the following path, where <username> is your login ID: *Documents and Settings*\<*username*>*Application Data*\DataWindow Designer2.5.

If there are no DbTrace settings in the INI file, the registry settings are used.

Error messages

If the database trace utility cannot open the trace output file with write access, an error message lets you know that the specified trace file could not be created or opened. If the trace utility driver cannot be loaded successfully, a message box informs you that the selected Trace DBMS is not supported in your current installation.

Format of the Database Trace log

The specific content of the Database Trace log file depends on the database you are accessing and the operations you are performing. However, the log uses the following basic format to display output:

COMMAND: (*time*)

{*additional_information*}

Parameter	Description
COMMAND	The internal command that DataWindow Designer executes to perform the database operation.
time	The number of microseconds it takes DataWindow Designer to perform the database operation. The precision used depends on your operating system's timing mechanism.
additional_information	(Optional) Additional information about the command. The information provided depends on the database operation.

Example

The following portion of the log file shows the commands DataWindow Designer executes to fetch two rows from a SQL Anywhere database table:

```
FETCH NEXT: (0.479 MS)
 COLUMN=400 COLUMN=Marketing COLUMN=Evans
FETCH NEXT: (0.001 MS)
 COLUMN=500 COLUMN=Shipping COLUMN=Martinez
```

If you opt to include DBI Names and Sum Time information in the trace log file, the log for the same two rows might look like this:

```
FETCH NEXT:(DBI_FETCHNEXT) (1.459 MS / 3858.556 MS)
 COLUMN=400 COLUMN=Marketing COLUMN=Evans
FETCH NEXT:(DBI_FETCHNEXT) (0.001 MS / 3858.557 MS)
 COLUMN=500 COLUMN=Shipping COLUMN=Martinez
```

For a more complete example of Database Trace output, see “Sample Database Trace output” on page 146.

Starting the Database Trace tool

By default, the Database Trace tool is turned off in DataWindow Designer. You can start it to trace your database connection.

❖ **To start the Database Trace tool:**

1 Open the Database Profile Setup dialog box for the connection you want to trace.

2 On the Connection tab, select the Generate Trace check box and click OK or Apply. (The Generate Trace check box is located on the System tab in the OLE DB Database Profile Setup dialog box.)

The Database Profiles dialog box displays with the name of the edited profile highlighted.

For example, here is the relevant portion of a database profile entry for Adaptive Server 12.5 Test. The setting that starts Database Trace is DBMS:

```
[Default]      [value not set]
AutoCommit     "FALSE"
Database       "qadata"
DatabasePassword  "00"
DBMS           "TRACE SYC Adaptive Server Enterprise"
DbParm         "Release='12.5'"
Lock           ""
```

```
LogId          "qalogin"
LogPassword    "00171717171717"
Prompt         "FALSE"
ServerName     "Host125"
UserID         ""
```

3 Click Connect in the Database Profiles dialog box to connect to the database.

The Database Trace dialog box displays, indicating that database tracing is enabled. You can enter the file location where DataWindow Designer writes the trace output. By default, DataWindow Designer writes Database Trace output to a log file named *DBTRACE.LOG*. You can change the log file name and location in the Database Trace dialog box.

The Database Trace dialog box also lets you select the level of trace information that you want in the database trace file.

4 Select the types of items you want to include in the trace file and click OK.

DataWindow Designer connects to the database and starts tracing the connection.

Stopping the Database Trace tool

Once you start tracing a particular database connection, DataWindow Designer continues sending trace output to the log until you do one of the following:

- Reconnect to the same database with tracing stopped
- Connect to another database for which you have not enabled tracing

❖ **To stop the Database Trace tool:**

1 In the Database Profile Setup dialog box for the database you are tracing, clear the Generate Trace check box on the Connection tab.

2 Click OK in the Database Profile Setup dialog box.

The Database Profiles dialog box displays with the name of the edited profile highlighted.

3 Right-click on the connected database and select Re-connect from the drop-down menu in the Database Profiles dialog box.

DataWindow Designer connects to the database and stops tracing the connection.

Using the Database Trace log

DataWindow Designer writes the output of the Database Trace tool to a file named *DBTRACE.LOG* (by default) or to a nondefault log file that you specify. To use the trace log, you can do the following anytime:

- View the Database Trace log with any text editor
- Annotate the Database Trace log with your own comments
- Delete the Database Trace log or clear its contents when it becomes too large

Viewing the Database Trace log

You can display the contents of the log file anytime during a DataWindow Designer session.

❖ **To view the contents of the log file:**

- Open the log file in one of the following ways:
  - Use the File Editor in DataWindow Designer. (For instructions, see the *User's Guide*.)
  - Use any text editor outside DataWindow Designer.

Leaving the log file open
If you leave the log file open as you work in DataWindow Designer, the Database Trace tool *does not update* the log.

Annotating the Database Trace log

When you use the Database Trace log as a troubleshooting tool, it might be helpful to add your own comments or notes to the file. For example, you can specify the date and time of a particular connection, the versions of database server and client software you used, or any other useful information.

❖ **To annotate the log file:**

1 Open the *DBTRACE.LOG* file in one of the following ways:
  - Use the File Editor in DataWindow Designer. (For instructions, see the *User's Guide*.)
  - Use any text editor outside DataWindow Designer.

2 Edit the log file with your comments.

3 Save your changes to the log file.

Deleting or clearing the Database Trace log

Each time you connect to a database with tracing enabled, DataWindow Designer appends the trace output of your connection to the existing log. As a result, the log file can become very large over time, especially if you frequently enable tracing when connected to a database.

❖ **To keep the size of the log file manageable:**

- Do either of the following periodically:
  - Open the log file, clear its contents, and save the empty file.

    Provided that you use the default *DBTRACE.LOG* or the same nondefault file the next time you connect to a database with tracing enabled, DataWindow Designer will write to this empty file.
  - Delete the log file.

    DataWindow Designer will automatically create a new log file the next time you connect to a database with tracing enabled.

Sample Database Trace output

This section gives an example of Database Trace output that you might see in the log file and briefly explains each portion of the output.

The example traces a connection with Sum Timing enabled. The output was generated while running a DataWindow Designer application that displays information about authors in a publications database. The SELECT statement shown retrieves information from the Author table.

The precision (for example, microseconds) used when Database Trace records internal commands depends on your operating system's timing mechanism. Therefore, the timing precision in your Database Trace log might vary from this example.

Connect to database

```
CONNECT TO TRACE SYC Adaptive Server Enterprise:
DATABASE=pubs2
LOGID=bob
SERVER=HOST12
DPPARM=Release='12.5.2',StaticBind=0
```

Prepare SELECT statement

```
PREPARE:
SELECT authors.au_id, authors.au_lname, authors.state
FROM authors
WHERE ( authors.state not in ( 'CA' ) )
ORDER BY authors.au_lname ASC (3.386 MS / 20.349 MS)
```

Get column descriptions	`DESCRIBE: (0.021 MS / 20.370 MS)`
	`name=au_id,len=12,type=CHAR,pbt=1,dbt=1,ct=0,prec=0,`
	`  scale=0`
	`name=au_lname,len=41,type=CHAR,pbt=1,dbt=1,ct=0,`
	`  prec=0,scale=0`
	`name=state,len=3,type=CHAR,pbt=1,dbt=1,ct=0,prec=0,`
	`  scale=0`

Bind memory buffers to columns	`BIND SELECT OUTPUT BUFFER (DataWindow):`
	`  (0.007 MS / 20.377 MS)`
	`name=au_id,len=12,type=CHAR,pbt=1,dbt=1,ct=0,prec=0,`
	`  scale=0`
	`name=au_lname,len=41,type=CHAR,pbt=1,dbt=1,ct=0,`
	`  prec=0,scale=0`
	`name=state,len=3,type=CHAR,pbt=1,dbt=1,ct=0,prec=0,`
	`  scale=0`

Execute SELECT statement	`EXECUTE: (0.001 MS / 20.378 MS)`

Fetch rows from result set	`FETCH NEXT: (0.028 MS / 20.406 MS)`
	` au_id=648-92-1872 au_lname=Blotchet-Hall state=OR`
	`FETCH NEXT: (0.012 MS / 20.418 MS)`
	` au_id=722-51-5454 au_lname=DeFrance state=IN`
	`...`
	`FETCH NEXT: (0.010 MS / 20.478 MS)`
	`au_id=341-22-1782 au_lname=Smith state=KS`
	`FETCH NEXT: (0.025 MS / 20.503 MS)`
	`*** DBI_FETCHEND *** (rc 100)`

Update and commit database changes	`PREPARE:`
	`UPDATE authors SET state = 'NM'`
	`WHERE au_id = '648-92-1872' AND au_lname = 'Blotchet-`
	`Halls' AND state = 'OR'  (3.284 MS / 23.787 MS)`
	`EXECUTE: (0.001 MS / 23.788 MS)`
	`GET AFFECTED ROWS: (0.001 MS / 23.789 MS)`
	`^ 1 Rows Affected`
	`COMMIT: (1.259 MS / 25.048 MS)`

Disconnect from database	`DISCONNECT: (0.764 MS / 25.812 MS)`
Shut down database interface	`SHUTDOWN DATABASE INTERFACE: (0.001 MS / 25.813 MS)`

Using the ODBC Driver Manager Trace tool

This section describes how to use the ODBC Driver Manager Trace tool.

About ODBC Driver Manager Trace

You can use the ODBC Driver Manager Trace tool to trace a connection to any ODBC data source that you access in DataWindow Designer through the ODBC interface.

Unlike the Database Trace tool, the ODBC Driver Manager Trace tool *cannot* trace connections through one of the native database interfaces.

What this tool does

ODBC Driver Manager Trace records information about ODBC API calls (such as SQLDriverConnect, SQLGetInfo, and SQLFetch) made by DataWindow Designer while connected to an ODBC data source. It writes this information to a default log file named *SQL.LOG* or to a log file that you specify.

What both tools do

The information from ODBC Driver Manager Trace, like Database Trace, can help you:

- Understand what DataWindow Designer is doing *internally* while connected to an ODBC data source
- Identify and resolve problems with your ODBC connection
- Provide useful information to Technical Support if you call them for help with your database connection

When to use this tool

Use ODBC Driver Manager Trace *instead* of the Database Trace tool if you want more detailed information about the ODBC API calls made by DataWindow Designer.

Performance considerations

Turning on ODBC Driver Manager Trace can slow your performance while working in DataWindow Designer. Therefore, use ODBC Driver Manager Trace for debugging purposes only and keep it turned off when you are not debugging.

SQL.LOG file

DataWindow Designer writes ODBC Driver Manager Trace output to a default log file named *SQL.LOG* or to a log file that you specify. The default location of *SQL.LOG* is in your root directory.

Starting ODBC Driver Manager Trace

To start ODBC Driver Manager Trace in order to trace your ODBC connection, you must edit your database profile.

Starting ODBC Driver Manager Trace

To start ODBC Driver Manager Trace, edit the database profile for the connection you want to trace, as described in the following procedure.

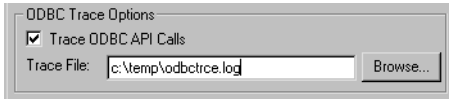
❖ **To start ODBC Driver Manager Trace:**

1 Open the Database Profile Setup-ODBC dialog box for the ODBC connection you want to trace.

2 On the Options tab, select the Trace ODBC API Calls check box.

3 (Optional) To specify a log file where you want DataWindow Designer to write the output of ODBC Driver Manager Trace, type the path name in the Trace File box
or
(Optional) Click Browse to display the pathname of an existing log file in the Trace File box.

By default, if the Trace ODBC API Calls check box is selected and no trace file is specified, DataWindow Designer sends ODBC Driver Manager Trace output to the default *SQL.LOG* file.



4 Click OK or Apply
or
Right-click on the connected database and select Re-connect from the drop-down menu in the Database Profiles dialog box.

The Database Profiles dialog box displays with the name of the edited profile highlighted.

DataWindow Designer saves your settings in the database profile entry in the registry in the *HKEY_CURRENT_USER\Software\Sybase* DataWindow Designer*\2.0\DatabaseProfiles* key.

For example, here is the relevant portion of a database profile entry for an ODBC data source named Employee. The settings that start ODBC Driver Manager Trace (corresponding to the ConnectOption DBParm parameter) are emphasized.

```
DBMS     "ODBC"
...
DbParm   "ConnectString='DSN=Emloyee;UID=dba;
PWD=00c61737',ConnectOption='SQL_OPT_TRACE,SQL_OPT_
TRACE_ON;SQL_OPT_TRACEFILE,C:\Temp\odbctrce.log'"
```

5 Click Connect in the Database Profiles dialog box to connect to the database
or
Right-click on the connected database and select Re-connect from the drop-down menu in the Database Profiles dialog box.

DataWindow Designer connects to the database, starts tracing the ODBC connection, and writes output to the log file you specified.

Stopping ODBC Driver Manager Trace

Once you start tracing an ODBC connection with ODBC Driver Manager Trace, DataWindow Designer continues sending trace output to the log file until you stop tracing. After you stop tracing as described in the following sections, you must reconnect to have the changes take effect.

Stopping ODBC Driver Manager Trace

❖ **To stop ODBC Driver Manager Trace:**

1 Open the Database Profile Setup - ODBC dialog box for the connection you are tracing.

For instructions, see “Starting ODBC Driver Manager Trace” on page 150.

2 On the Options tab, clear the Trace ODBC API Calls check box.

If you supplied the pathname of a log file in the Trace File box, you can leave it specified in case you want to restart tracing later.

3 Click OK in the Database Profile Setup - ODBC dialog box.

The Database Profiles dialog box displays, with the name of the edited profile highlighted.

4 Click Connect in the Database Profiles dialog box
or
Right-click on the connected database and select Re-connect from the drop-down menu in the Database Profiles dialog box.

DataWindow Designer connects to the database and stops tracing the connection.

Viewing the ODBC Driver Manager Trace log

You can display the contents of the ODBC Driver Manager Trace log file anytime during a DataWindow Designer session.

❖ **To view the contents of the log file:**

- Open SQL.LOG or the log file you specified in one of the following ways:
  - Use the File Editor in DataWindow Designer. (For instructions, see the *User's Guide*.)
  - Use any text editor outside DataWindow Designer.

Leaving the log file open
If you leave the log file open as you work in DataWindow Designer, ODBC Driver Manager Trace *does not update it.*

Sample ODBC Driver Manager Trace output

This section shows a partial example of output from ODBC Driver Manager Trace to give you an idea of the information it provides. The example is part of the trace on an ODBC connection to the EAS Demo DB.

For more about a particular ODBC API call, see your ODBC documentation.

```
PB110 179:192   EXIT  SQLSetConnectOption  with return
code 0 (SQL_SUCCESS)
      HDBC 0x036e1300
      UWORD    104 <SQL_OPT_TRACE>
      UDWORD 1

PB110 179:192   ENTER SQLSetConnectOption
   HDBC 0x036e1300
   UWORD 110 <SQL_OPT_TRACEFILE>
   UDWORD 160694373
PB110 179:192   EXIT  SQLSetConnectOption  with return
```

```
code 0 (SQL_SUCCESS)
   HDBC 0x036e1300
   UWORD 110 <SQL_OPT_TRACEFILE>
   UDWORD 160694373
PB110 179:192   ENTER SQLDriverConnectW
   HDBC 0x036e1300
   HWND 0x004607fa
   WCHAR * 0x1f4be068 [      -3] "******\ 0"
   SWORD -3
   WCHAR * 0x1f4be068
   SWORD 8
   SWORD * 0x00000000
   UWORD 1 <SQL_DRIVER_COMPLETE>
PB110 179:192   EXIT  SQLDriverConnectW  with return
code 0 (SQL_SUCCESS)
   HDBC 0x036e1300
   HWND 0x004607fa
   WCHAR * 0x1f4be068 [      -3] "******\ 0"
   SWORD -3
   WCHAR * 0x1f4be068
   SWORD 8
   SWORD * 0x00000000
      UWORD 1 <SQL_DRIVER_COMPLETE>
PB110 179:192   ENTER SQLGetInfoW
      HDBC 0x036e1300
      UWORD 6 <SQL_DRIVER_NAME>
      PTR 0x036e2098
      SWORD 6
      SWORD * 0x0012cd30
PB110 179:192   EXIT  SQLGetInfoW  with return code 1
(SQL_SUCCESS_WITH_INFO)
      HDBC 0x036e1300
      UWORD 6 <SQL_DRIVER_NAME>
      PTR 0x036e2098 [       6] "DB\ 0"
      SWORD 6
      SWORD * 0x0012cd30 (22)
   DIAG [01104] [Sybase][ODBC Driver]Data truncated (0)
PB110 179:192   ENTER SQLGetInfoW
      HDBC 0x036e1300
      UWORD 10 <SQL_ODBC_VER>
      PTR 0x036e39f8
      SWORD 100
      SWORD * 0x0012cd38
PB110 179:192   EXIT  SQLGetInfoW  with return code 0
(SQL_SUCCESS)
      HDBC 0x036e1300
```

```
      UWORD 10 <SQL_ODBC_VER>
      PTR 0x036e39f8 [      20] "03.51.0000"
      SWORD 100
      SWORD * 0x0012cd38 (20)
PB110 179:192   ENTER SQLGetInfoW
      HDBC 0x036e1300
      UWORD 2 <SQL_DATA_SOURCE_NAME>
      PTR 0x036e3c88
      SWORD 512
      SWORD * 0x0012cc32
PB110 179:192   EXIT  SQLGetInfoW  with return code 0
(SQL_SUCCESS)
      HDBC 0x036e1300
      UWORD 2 <SQL_DATA_SOURCE_NAME>
      PTR 0x036e3c88 [      28] "EAS Demo DB"
      SWORD 512
      SWORD * 0x0012cc32 (28)
PB110 179:192   ENTER SQLGetInfoW
      HDBC 0x036e1300
      UWORD 16 <SQL_DATABASE_NAME>
      PTR 0x036e3c88
      SWORD 512
      SWORD * 0x0012cc32
PB110 179:192   EXIT  SQLGetInfoW  with return code 0
(SQL_SUCCESS)
      HDBC 0x036e1300
      UWORD 16 <SQL_DATABASE_NAME>
      PTR 0x036e3c88 [      16] "easdemodb"
      SWORD 512
      SWORD * 0x0012cc32 (16)
PB110 179:192   ENTER SQLGetInfoW
      HDBC 0x036e1300
      UWORD 25 <SQL_DATA_SOURCE_READ_ONLY>
      PTR 0x036e3c88
      SWORD 512
      SWORD * 0x0012cc32
PB110 179:192   EXIT  SQLGetInfoW  with return code 0
(SQL_SUCCESS)
      HDBC 0x036e1300
      UWORD 25 <SQL_DATA_SOURCE_READ_ONLY>
      PTR 0x036e3c88 [       2] "N"
      SWORD 512
      SWORD * 0x0012cc32 (2)
PB110 179:192   ENTER SQLGetInfoW
      HDBC 0x036e1300
      UWORD 13 <SQL_SERVER_NAME>
```

```
      PTR 0x036e3c88
      SWORD 512
      SWORD * 0x0012cc32
PB110 179:192   EXIT  SQLGetInfoW  with return code 0
(SQL_SUCCESS)
      HDBC 0x036e1300
      UWORD 13 <SQL_SERVER_NAME>
      PTR 0x036e3c88 [      16] "easdemodb"
      SWORD 512
      SWORD * 0x0012cc32 (16)
PB110 179:192   ENTER SQLGetInfoW
      HDBC 0x036e1300
      UWORD 17 <SQL_DBMS_NAME>
      PTR 0x036e3c88
      SWORD 512
      SWORD * 0x0012cab6
PB110 179:192   EXIT  SQLGetInfoW  with return code 0
(SQL_SUCCESS)
      HDBC 0x036e1300
      UWORD 17 <SQL_DBMS_NAME>
      PTR 0x036e3c88 [  48] "SQL Anywhere"
      SWORD 512
      SWORD * 0x0012cab6 (48)
PB110 179:192   ENTER SQLGetInfoW
      HDBC 0x036e1300
      UWORD 6 <SQL_DRIVER_NAME>
      PTR 0x036e1a10
      SWORD 550
      SWORD * 0x0012cbbc
PB110 179:192   EXIT  SQLGetInfoW  with return code 0
(SQL_SUCCESS)
      HDBC 0x036e1300
      UWORD 6 <SQL_DRIVER_NAME>
      PTR 0x036e1a10 [      22] "DBODBC9.DLL"
      SWORD 550
      SWORD * 0x0012cbbc (22)
PB110 179:192   ENTER SQLAllocStmt
      HDBC 0x036e1300
      HSTMT * 0x0012d0b4
PB110 179:192   EXIT  SQLAllocStmt  with return code 0
(SQL_SUCCESS)
      HDBC 0x036e1300
      HSTMT * 0x0012d0b4 ( 0x036e1c48)
PB110 179:192   ENTER SQLGetTypeInfo
      HSTMT 0x036e1c48
      SWORD 0 <SQL_ALL_TYPES>
```

PART 5

Appendix

The Appendix describes how to modify the PBODB110 initialization file.

APPENDIX

Adding Functions to the PBODB110 Initialization File

About this appendix

In general, *you do not need to modify the PBODB110 initialization file*. In certain situations, however, you might need to add functions to the PBODB110 initialization file for connections to your back-end DBMS through the ODBC or OLE DB interface in DataWindow Designer.

This appendix describes how to add functions to the PBODB110 initialization file if necessary.

Contents

About the PBODB110 initialization file

What is the PBODB110 initialization file?

When you access data through the ODBC interface, DataWindow Designer uses the PBODB110 initialization file (*PBODB110.INI*) to maintain access to extended functionality in the back-end DBMS for which ODBC does not provide an API call. Examples of extended functionality are SQL syntax or function calls specific to a particular DBMS.

Editing PBODB110.INI

In most cases, you do *not* need to modify *PBODB110.INI*. Changes to this file can adversely affect DataWindow Designer. Change *PBODB110.INI* only if you are asked to do so by a Technical Support representative.

However, you *can* edit *PBODB110.INI* if you need to add functions for your back-end DBMS.

If you modify *PBODB110.INI*, first make a copy of the existing file. Then keep a record of all changes you make. If you call Technical Support after modifying *PBODB110.INI*, tell the representative that you changed the file and describe the changes you made.

Adding functions to PBODB110.INI

PBODB110.INI lists the functions for certain DBMSs that have ODBC drivers. If you need to add a function to *PBODB110.INI* for use with your back-end DBMS, you can do either of the following:

- **Existing sections** Add the function to the Functions section for your back-end database if this section exists in *PBODB110.INI*.
- **New sections** Create new sections for your back-end DBMS in *PBODB110.INI* and add the function to the newly created Functions section.

Adding functions to an existing section in the file

If sections for your back-end DBMS *already exist* in *PBODB110.INI*, use the following procedure to add new functions.

❖ **To add functions to an existing section in PBODB110.INI:**

1 Open *PBODB110.INI* in one of the following ways:

- Use the File Editor in DataWindow Designer. (For instructions, see the *User's Guide*.)
- Use any text editor outside DataWindow Designer.

2 Locate the entry for your back-end DBMS in the DBMS Driver/DBMS Settings section of *PBODB110.INI*.

For example, here is the*PBODB110.INI* entry for SQL Anywhere:

```
;*****************************************
;DBMS Driver/DBMS Settings see comments at end
;of file
;*****************************************
...
[SQL Anywhere]
PBSyntax='WATCOM50_SYNTAX'
PBDateTime='STANDARD_DATETIME'
PBFunctions='ASA_FUNCTIONS'
PBDefaultValues='autoincrement,current date,
   current time,current timestamp,timestamp,
   null,user'
PBDefaultCreate='YES'
PBDefaultAlter='YES'
PBDefaultExpressions='YES'
```

```
DelimitIdentifier='YES'
PBDateTimeInvalidInSearch='NO'
PBTimeInvalidInSearch='YES'
PBQualifierIsOwner='NO'
PBSpecialDataTypes='WATCOM_SPECIALDATATYPES'
IdentifierQuoteChar='"'
PBSystemOwner='sys,dbo'
PBUseProcOwner='YES'
SQLSrvrTSName='YES'
SQLSrvrTSQuote='YES'
SQLSrvrTSDelimit='YES'
ForeignKeyDeleteRule='Disallow if Dependent Rows
   Exist (RESTRICT),Delete any Dependent Rows
   (CASCADE),Set Dependent Columns to NULL
   (SET NULL)'
TableListType='GLOBAL TEMPORARY'
```

3 Find the name of the section in *PBODB110.INI* that contains function information for your back-end DBMS.

To find this section, look for a line similar to the following in the DBMS Driver/DBMS Settings entry:

```
PBFunctions='section_name'
```

For example, the following line in the DBMS Driver/DBMS Settings entry for SQL Anywhere indicates that the name of the Functions section is ASA_FUNCTIONS:

```
PBFunctions='ASA_FUNCTIONS'
```

4 Find the Functions section for your back-end DBMS in *PBODB110.INI*.

For example, here is the Functions section for SQL Anywhere:

```
;*******************************************
;Functions
;*******************************************
[ASA_FUNCTIONS]
AggrFuncs=avg(x),avg(distinct x),count(x),
   count(distinct x),count(*),list(x),
   list(distinct x),max(x),max(distinct x),
   min(x),min(distinct x),sum(x),sum(distinct x)
Functions=abs(x),acos(x),asin(x),atan(x),
   atan2(x,y),ceiling(x),cos(x),cot(x),degrees(x),
   exp(x),floor(x),log(x),log10(x),
   mod(dividend,divisor),pi(*),power(x,y),
   radians(x),rand(),rand(x),
   remainder(dividend,divisor),round(x,y),
```

```
sign(x),sin(x),sqrt(x),tan(x),
"truncate"(x,y),ascii(x),byte_length(x),
byte_substr(x,y,z),char(x),char_length(x),
charindex(x,y),difference(x,y)insertstr(x,y,z),
lcase(x),left(x,y),length(x), locate(x,y,z),
lower(x),ltrim(x),patindex('x',y),repeat(x,y),
replicate(x,y),right(x,y),rtrim(x),
similar(x,y),soundex(x),space(x),str(x,y,z),
string(x,...),stuff(w,x,y,z),substr(x,y,z),
trim(x),ucase(x),upper(x),date(x),
dateformat(x,y),datename(x,y),day(x),
dayname(x),days(x),dow(x),hour(x),hours(x),
minute(x),minutes(x),minutes(x,y),month(x),
monthname(x),months(x),months(x,y),now(*),
quarter(x),second(x),seconds(x),seconds(x,y),
today(*),weeks(x),weeks(x,y),year(x),years(x),
years(x,y),ymd(x,y,z),dateadd(x,y,z),
datediff(x,y,z),datename(x,y),datepart(x,y),
getdate(),cast(x as y),convert(x,y,z),
hextoint(x),inttohex(x),
connection_property(x,...),datalength(x),
db_id(x),db_name(x),db_property(x),
next_connection(x),next_database(x),
property(x),property_name(x),
property_number(x),property_description(x),
argn(x,y,...),coalesce(x,...),
estimate(x,y,z),estimate_source(x,y,z),
experience_estimate(x,y,z),ifnull(x,y,z),
index_estimate(x,y,z),isnull(x,...),
number(*),plan(x),traceback(*)
```

5 To add a new function, type a comma followed by the function name at the end of the appropriate function list, as follows:

- **Aggregate functions** Add aggregate functions to the end of the AggrFuncs list.
- **All other functions** Add all other functions to the end of the Functions list.

Case sensitivity
If the back-end DBMS you are using is case sensitive, be sure to use the required case when you add the function name.

The following example shows a new function for SQL Anywhere added at the end of the Functions list:

```
;*****************************************
;Functions
;*****************************************
[ASA_FUNCTIONS]
AggrFuncs=avg(x),avg(distinct x),count(x),
   count(distinct x),count(*),list(x),
   list(distinct x),max(x),max(distinct x),
   min(x),min(distinct x),sum(x),sum(distinct x)
Functions=abs(x),acos(x),asin(x),atan(x),
   atan2(x,y),ceiling(x),cos(x),cot(x),degrees(x),
   exp(x),floor(x),log(x),log10(x),
   mod(dividend,divisor),pi(*),power(x,y),
   radians(x),rand(),rand(x),
   ...
   number(*),plan(x),traceback(*),newfunction()
```

6 Save your changes to *PBODB110.INI*.

Adding functions to a new section in the file

If entries for your back-end DBMS *do not exist* in *PBODB110.INI*, use the following procedure to create the required sections and add the appropriate functions.

Before you start
For more about the settings to supply for your back-end DBMS in *PBODB110.INI*, read the comments at the end of the file.

❖ **To add functions to a new section in PBODB110.INI:**

1 Open *PBODB110.INI* in one of the following ways:

- Use the File Editor in DataWindow Designer. (For instructions, see the *User's Guide*.)
- Use any text editor outside DataWindow Designer.

2 Edit the DBMS Driver/DBMS Settings section of the PBODB110 initialization file to add an entry for your back-end DBMS.

Finding the name
The name required to identify the entry for your back-end DBMS in the DBMS Driver/DBMS Settings section is in *PBODB110.INI*.

Make sure that you:

- Follow the instructions in the comments at the end of *PBODB110.INI*.
- Use the same syntax as existing entries in the DBMS Driver/DBMS Settings section of *PBODB110.INI*.
- Include a section name for PBFunctions.

For example, here is the relevant portion of an entry for a DB2/2 database:

```
;************************************************
;DBMS Driver/DBMS Settings
;************************************************
[DB2/2]
...
PBFunctions='DB22_FUNCTIONS'
...
```

3 Edit the Functions section of *PBODB110.INI* to add an entry for your back-end DBMS.

Make sure that you:

- Follow the instructions in the comments at the end of *PBODB110.INI*.
- Use the same syntax as existing entries in the Functions section of *PBODB110.INI*.
- Give the Functions section the name that you specified for PBFunctions in the DBMS Driver/DBMS Settings entry.

For example:

```
;************************************************
;Functions
;************************************************
[DB22_FUNCTIONS]
AggrFuncs=avg(),count(),list(),max(),min(),sum()
Functions=curdate(),curtime(),hour(), ...
```

4 Type a comma followed by the function name at the end of the appropriate function list, as follows:

- **Aggregate functions** Add aggregate functions to the end of the AggrFuncs list.
- **All other functions** Add all other functions to the end of the Functions list.

Case sensitivity
If the back-end DBMS you are using is case sensitive, be sure to use the required case when you add the function name.

The following example shows (in bold) a new DB2/2 function named substr() added at the end of the Functions list:

```
;*********************************************
;Functions
;*********************************************
[DB22_FUNCTIONS]
AggrFuncs=avg(),count(),list(),max(),min(),sum()
Functions=curdate(),curtime(),hour(), substr()
```

5 Save your changes to *PBODB110.INI*.

Index

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V