Synametrics Technologies

WinSQL® Professional

User's Guide

Version 8.5

August 20, 2010

Publication date

August 2010

Product and version

WinSQL Professional 8.5

Reader comments

Any comments or suggestions regarding this publication are should be forwarded to the attention of:

666 Plainsboro Road
Suite 656
Plainsboro, NJ 08536

or sent by e-mail to: support@synametrics.com

Table of Contents

About This Document

Overview

WinSQL Professional is a utility designed to interact with many different types of databases. This WinSQL Professional User's Guide is designed to provide details instructions for installing, configuring, and using the program. It also includes information about more advanced features, such as data export, test data generation and schema diff.

Audience

The WinSQL Professional User's Guide is designed for use by database administrators, system administrators, programmers, and other technical staff.

Document Conventions

The following conventions are used throughout this document:

Convention	Description
Keyboard keys and function keys	Begin with an uppercase letter and appear in bold type, enclosed in brackets; for example, **[Enter]** or **[F1]**.
Key combinations	Are enclosed in brackets and appear in bold type If joined with a plus sign (+), press and hold the first and second key simultaneously; for example, Press **[Ctrl+B]**.
Execution icons	Begin with uppercase letters and appear in bold type, for example, **OK**.
Menu names and options	Begin with uppercase letters and appear in bold type; for example, On the **Edit** menu, click **Options**.
Window names	Begin with uppercase letters and appear in bold type; for example, the **Configuration** window is displayed.
Text variables	Are enclosed in angle brackets; for example, <file_name>.
Numeric variables	Are represented by a letter; for example, x.

In addition, the following special formats are used:

Format	Indicates…
Green text	a hyperlink to another section of this document or to a web site
`Courier`	text of a message displayed in a window
`Courier bold`	text that you must type in a window

Format	Indicates…
Bold	a reference to a window or to an object in a window, such as an icon, field, or column; also indicates emphasis on a critical instruction or step
Italics	a reference to another document; also indicates emphasis on certain words (Example: do *not* delete this file)

Chapter 1: Getting Started

Overview

Introduction

This chapter provides information about downloading, installing, and configuring WinSQL Professional.

In this chapter

This chapter contains the following topics:

Requirements

WinSQL Professional is a 32-bit program that runs only on Microsoft® Windows® XP, Windows 2000, Windows 2003 and Windows Vista. It uses ODBC to connect to any target database for which a driver is available. WinSQL Professional requires that you use ODBC Manager version 3.x or higher. Although version 3.x is required as the ODBC Manager, you can use a driver that is compliant with version 2.x. However, if you use an older driver, you may not be able to use all of the features of WinSQL Professional.

To confirm the version of ODBC Administrator currently installed, click **Help**, and then click **About** in the **ODBC Data Source Administrator window**. If version 3.x or higher is not installed, download the most current version from http://www.microsoft.com/data/odbc.

Installation and Set Up

Regardless of the version of WinSQL Professional you are using, you will always download and unzip (using WinZip™) the same executable file. These executable files are available in two different types of packaging, as described in the following table:

File Name	Description
WinSQL.zip	An installation program for WinSQL Professional. Extract the setup executable from the zipped file and follow instructions on the screen to install WinSQL Professional using this file. This is the recommended installation method.
WinSQLRaw.zip	This is a zipped file that contains all necessary executable files and ODBC drivers. However, this file does not contain an installer. Files must be manually copied to the directory of your choice, and the icons for the program must be manually created.
WinSQLUpgrade.zip	Use this file if you are upgrading from version 4.7 or 5.0. If you have version 4.6 or earlier installed, we recommend that you uninstall and reinstall version 5.5. This file does NOT contain every file – it only includes the main executable file and updated database plug-ins.

Obtaining a Serial Number

A serial number is required within the first 30 days of using WinSQL Professional. This serial number determines the edition for which you are licensed. You can obtain a serial number for WinSQL Professional Lite at no charge from our website (http://www.synametrics.com/WinSQLreg).

When you download WinSQL Professional from Synametrics Technologies' Web site, a serial number for WinSQL Professional Lite is delivered to you. If you decide to purchase either the WinSQL Professional Developer or WinSQL Professional version, you must contact Synametrics Technologies for a different serial number to convert your existing installation.

Registering WinSQL Professional ODBC Drivers

If using the ODBC drivers installed with WinSQL Professional, they must be registered prior to using WinSQL Professional. Registering the drivers configures them so that they appear in ODBC Manager in Windows Control Panel.

The first time you run WinSQL Professional, the following window is displayed:

The **WinSQL ODBC Installer** window displays all the available drivers installed with WinSQL Professional. The installation status is displayed next to the name of every driver. You may choose to register only the drivers that you want to use, or select all of them. Synametrics Technologies recommends that you register all the drivers.

Notes:

- To re-register or unregister a driver, click **Help**, and then click **Register ODBC Drivers** within WinSQL Professional.
- When drivers are unregistered, all associated data source names (DSNs) are also removed.
- WinSQL Professional works with all ODBC drivers, regardless of the manufacturer.

Therefore, you do not have to register these drivers if you are planning to use a different driver to connect to your database.

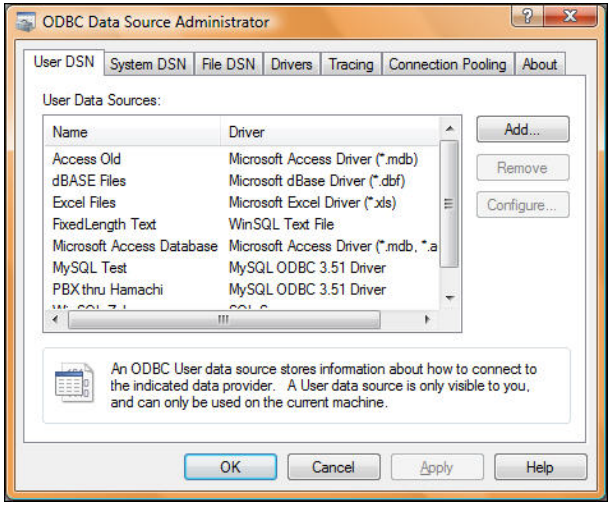
Configuring ODBC

Configuring ODBC and creating data sources is required before WinSQL Professional can be used for database management. The following topics describe these procedures.

Creating ODBC Connections

Perform the following steps to create ODBC connections:

1. From the **Start** menu, click **Control Panel**.
2. Click **Administrative Tools**.
3. Double-click the **ODBC Data Sources (32bit)** or the **Data Sources (ODBC)** icon. The **ODBC Data Source Administrator** window is displayed:



Three different types of DSNs can be created:

- User DSN - These data sources are local to a computer, and may only be used by the current user.
- System DSN - These data sources are local to a computer, rather than dedicated to a user. The system, or any user having privileges on the system, can use a data source set up with a system DSN.

- File DSN - These are file-based data sources that may be shared between all users that have the same drivers installed and who have access to the database. These data sources are not machine- or user-specific.

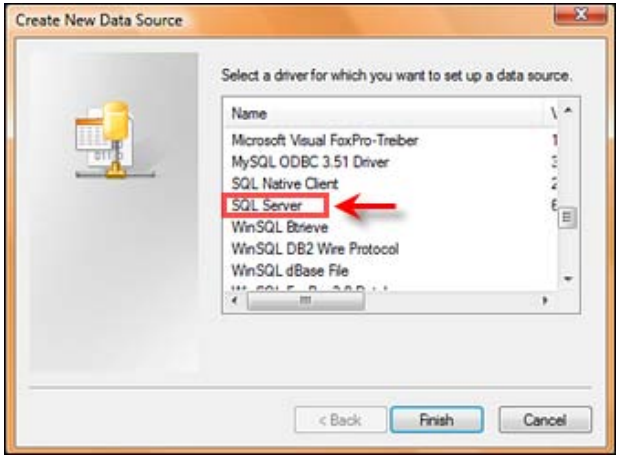
Determine the type of DNS you need. The following steps, as an example, describe how to create an MS-SQL Server database (System DSN) connection:

1 From the **ODBC Data Source Administrator** window, click the **System DSN** tab.

2 Click **Add**.

The window that is displayed lists all the ODBC drivers installed on the computer**.**

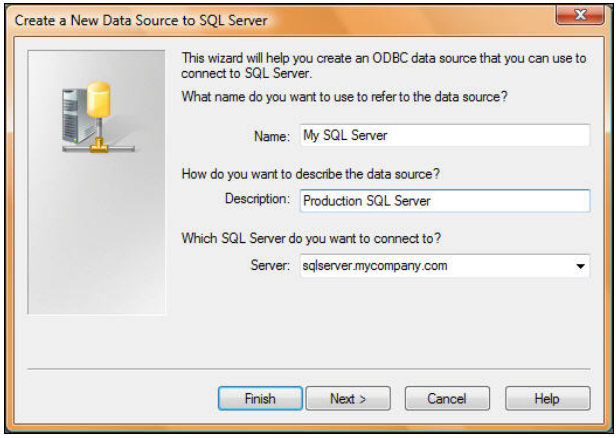
3 Click to select **SQL Server**:



4 Click **Finish**. The following window is displayed:

Note:

The following screen may look different if you are connecting to a database other than MS SQL Server.



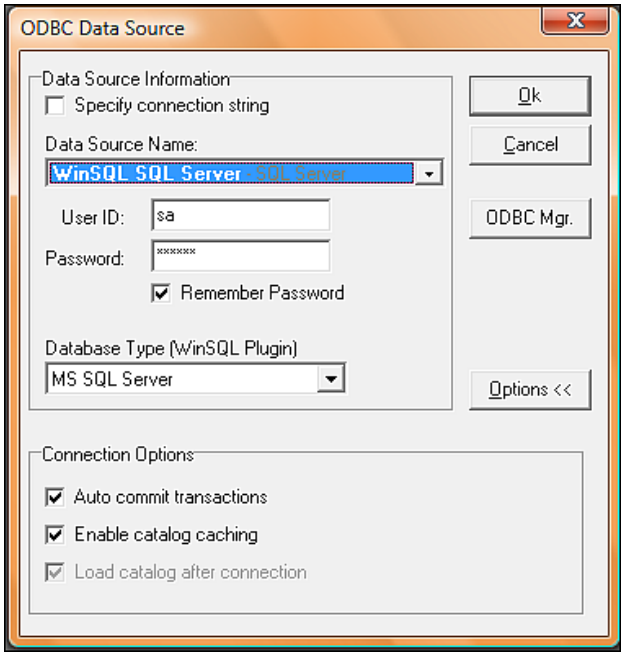
5. Enter the desired name in the **Name** field.
6. Enter an arbitrary description in the **Description** field.
7. Enter the server's host name or IP address in **the Which SQL Server do you want to connect to?** field.
8. Click **Next**.

Follow the instructions on the screen to complete this step. Depending upon the back-end database to which you are connecting, the screens may look different on your machine.

Establishing a Database Connection

Running queries against a database requires that a DSN be created. After performing the steps described in "Creating ODBC Connections," perform the following steps to connect to the database:

1. From the **File** menu, click **New Connection**.
2. Select the desired DSN in the **Data Source Name** drop-down list.
3. Enter the appropriate user ID in the **User ID** field.
4. Enter the appropriate password in the **Password** field. The **ODBC Data Source** window should be completed as follows:

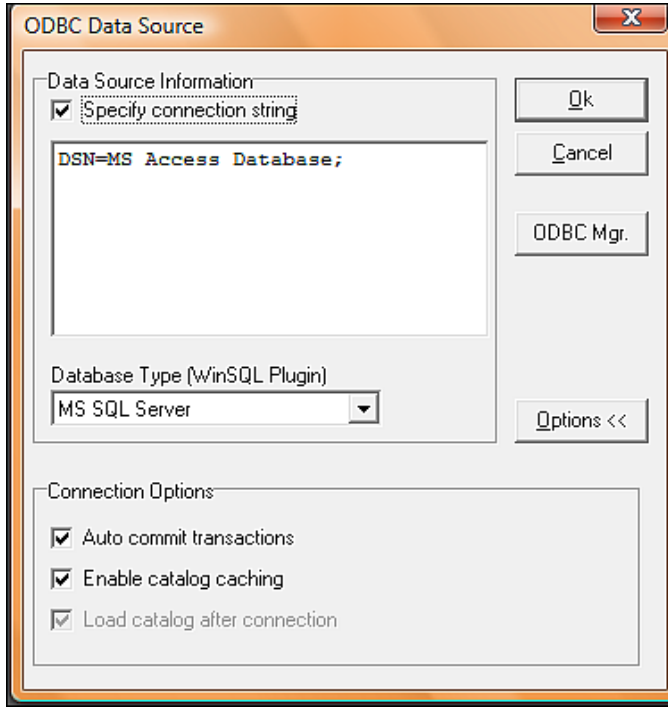


5. Click **OK** to connect to the database.

Connection Strings

A connection string is a set of connection parameters that tell WinSQL Professional how to connect to a remote database. If you want to use a connection string for connecting to a database, or if your application requires it, you must specify a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the DSN=, FILEDSN=, or the DRIVER= keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify attribute=value pairs in the connection string to override the default values stored in the data source.

Beginning with ODBC Administrator version 4.0, a DSN is not required to establish a database connection. To use a connection string rather than a DSN connection, click the **Specify connection string** checkbox. The following window is displayed:



Choose one of the following connection string types and use the accompanying format as an example for creating your entry:

- DSN connection, which uses the following format:

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

- File DSN connection, which uses the following format:

  FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]

- DSN-less connection, which uses a specific driver instead of a data source, and uses the following format:

  DRIVER=[{]driver_name[}][;attribute=value[;attribute=value]...]

If you are unsure about what to enter here, delete everything from the text box and click **OK**. The ODBC Driver manager window will display, and you can select the data source to which you wish to connect. WinSQL Professional captures the connection string that is used by the ODBC Driver manager and saves it for future use. The next time you establish connection, this text box will be completed for you.

Sample Connection Strings

The following is a list of connection string formats for the most common database types:

DBase / FoxPro

```
Driver={Microsoft dBASE Driver
(*.dbf)};Dbq=<c:\data>;

Where <c:\data> is the folder where
the files are stored
```

Firebird

```
DRIVER=Firebird/InterBase(r)
driver;uid=sysdba;pwd=secret;dbname
=<c:\data>\test.fdb
```

Where the database resides in the <c:\data> folder

Informix

Using WinSQL Professional Informix Wire Protocol
DRIVER={WinSQL Professional Informix Wire Protocol};uid=yourID;PWD=secret;host=yourServerIP;port=1526;DB=YourDatabase;srvr=ol_yourdb

Using Informix driver from IBM
Driver={INFORMIX 3.30 32 BIT};Host=hostname;Server=myserver;Service=ol_yourdb;Protocol=olsoctcp;Database=mydb;UID=username;PWD=myPwd

Where <ol_yourdb> represents the server name from sqlhosts file on the Informix server

IBM DB2

Using WinSQL Professional DB2 Wire Protocol Driver
Driver={WinSQL Professional DB2 Wire Protocol};Database=myDbName;IpAddress=myServerName;port=myPortNum;protocol=TCPIP;uid=myUserName;pwd=secret

Using IBM Driver

Driver={IBM DB2 ODBC DRIVER};Database=myDbName;hostname=myServerName;port=myPortNum;protocol=TCPIP;uid=myUserName;pwd=secret

Mimer

```
Driver={MIMER};Database=yourDatabas
e;uid=yourUserID;Pwd=secret;
```

MS Access

```
Standard Security
Driver={Microsoft Access Driver
(*.mdb)};Dbq=c:\mydb.mdb;Uid=Admin;
Pwd=;

Workgroup
Driver={Microsoft Access Driver
(*.mdb)};Dbq=C:\mydb.mdb;SystemDB=C
:\mydb.mdw;
```

MS Excel

```
Driver={Microsoft Excel Driver
(*.xls)};Dbq=c:\MyExcel.xls;Default
Dir=c:\mypath;
```

MS SQL Server

```
Using SQL Server Security
Driver={SQL
Server};Server=YourServerIP;Databas
e=YourDB;Uid=sa;Pwd=secret

Using Trusted Connection
Driver={SQL
Server};Server=YourServer;Database=
YourDBName;Trusted_Connection=yes

Using TCP/IP with on a non-standard
port
Driver={SQL
Server};Server=192.168.1.50,4321;ui
d=sa;pwd=secret;Database=YourDB;
```

MySQL

```
DRIVER={MySQL ODBC 3.51
Driver};server=YourServerIP;PORT=33
06;database=myDatabase;user=myUsern
ame;password=secret;option=3
```

Oracle

```
Using WinSQL Professional Wire
Protocol Driver
Driver={WinSQL Professional Oracle
Wire
Protocol};host=tarzan;port=1521;Uid
=scott;
Pwd=tiger;sid=orcl

Using WinSQL Professional Driver
DRIVER={WinSQL Professional
Oracle};uid=SCOTT;pwd=tiger;SRVR=or
cl

Using Microsoft Driver
Driver={Microsoft ODBC for
Oracle};UID=scott;PWD=tiger;SERVER=
ORCL;
```

In the previous two examples, the variable SERVER and SRVR refers to the name in TNSNAMES.ORA file on your machine. Note that WinSQL Professional Wire Protocol Driver, used in the first example, does not need TNSNAMES.ORA file.

If you are using Oracle's ODBC driver, the name of the driver is different on every machine. Refer to the ODBC Data Source Administrator to obtain the driver name.

Paradox

```
Driver={Microsoft Paradox Driver
(*.db
)};DBQ=c:\data\;DefaultDir=<c:\data
\>;

Where <c:\data> is the folder where
the files are stored.
```

PostgreSQL

```
Driver={PostgreSQL};Server=ipaddres
s;port=5432;Database=yourDBname;uid
=yourId;pwd=secret;
```

Sybase

```
Using WinSQL Professional Sybase
Wire Protocol
Driver={WinSQL Professional Sybase
Wire
Protocol};uid=sa;pwd=secret;NA=goof
y,5000;DB=YourDatabaseName

Using Sybase Open Client Driver
Driver={SYBASE ASE ODBC
Driver};Srvr=YourServerName;Uid=sa;
Pwd=secret
```

Text Files

```
Using WinSQL Professional Text File
driver
Driver={WinSQL Professional Text
File};DB=c:\data\

Using Microsoft Text Driver
Driver={Microsoft Text Driver
(*.txt;
*.csv)};Dbq=c:\data\;Extensions=asc
,csv,tab,txt;
```

Connection Options

Database Type / WinSQL Professional Plug-in

This connection option specifies the type of database to which you are connecting, and it also specifies a database plug-in. The names displayed in this list include the database plug-ins identified by WinSQL Professional when it is started. Choice of database type is only available in WinSQL Professional.

For more information about plug-ins, see "Database-Specific Plug-Ins" later in this document.

Important note:

If the database to which you are trying to connect is not displayed in the list, it does **not** mean you cannot connect to that database with WinSQL Professional. It means that WinSQL Professional will not be able to query specific database features, such as triggers and SQL scripts for views and stored procedures.

Auto commit transactions

When this option is checked, all transactions are automatically committed. Certain databases, such as Oracle, do not allow explicit transactions. Therefore, it may be useful to turn off this option, which will allow you to manually COMMIT and ROLLBACK any transaction.

When this option is turned off, a red indicator appears on the status bar. This indicator is a reminder to COMMIT or ROLLBACK before closing the connection.

Enable catalog caching

Selection this option at connection allows WinSQL to automatically create a cached version of your database catalog on your local machine. Caching your database catalog reduces the amount of time it takes to fetch metadata information from the back-end database, which greatly increases access time to your data.

When caching is enabled, WinSQL stores the contents of the **Catalog Details** window to a local file. The next time you connect to the same database, WinSQL loads the catalog information from this local file rather than sending queries to the database.

For more information, see the "Database Catalog Caching" topic later in this document.

Load catalog after connection

Selecting this option allows WinSQL to fetch the database catalog immediately after establishing connection.

Uninstalling WinSQL Professional

Prior to installing WinSQL Professional, you must unregister any registered ODBC drivers in WinSQL Professional. Failure to do so removes the files but does not remove registry entries.

Unregister ODBC Drivers

1. Start WinSQL Professional.
2. From the **Help** menu, click **Register ODBC drivers**.
3. Select all drivers, and then click **Unregister**.

Uninstall WinSQL Professional

1. From the **Start** menu, click **Control Panel**.
2. Double-click **Add/Remove Programs**.
3. Click **WinSQL Professional** in the list of programs that is displayed, and then click **Add/Remove**.
4. Click **Next**.
5. Click **Remove**, and then click **Next**.
6. Click **Remove**.

Note:

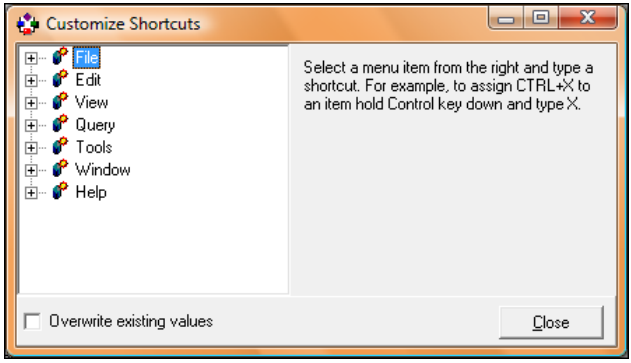
A reboot is NOT required after uninstalling WinSQL Professional.

Customizing Shortcut Keys

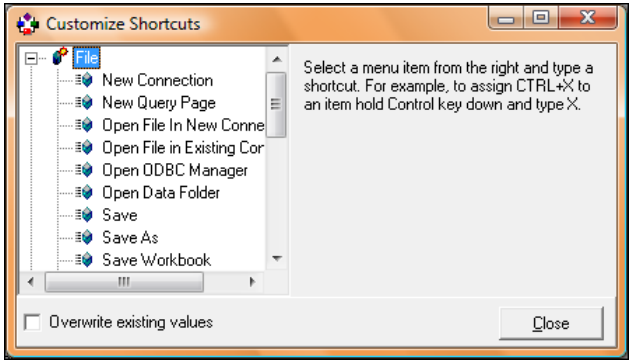
Keyboard shortcut keys can be used to access many of the features available in WinSQL Professional.

Perform the following steps to view and/or modify keyboard shortcut keys:

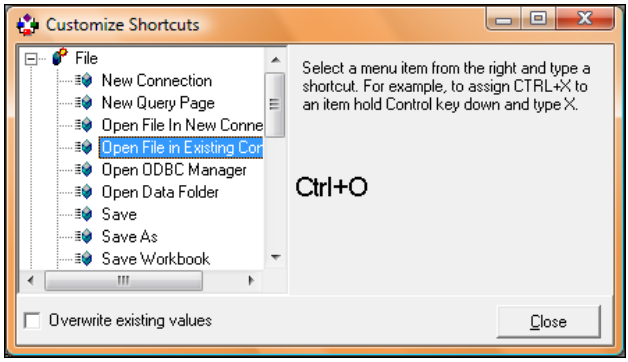
1 From the **Tools** menu, click **Customize Shortcuts**. The following window is displayed:



2 Click the **+** (plus sign) by each of the menu options to display a submenu, illustrated as follows:



3 Click to select an option. If a shortcut key exists for the option, it is displayed as follows:

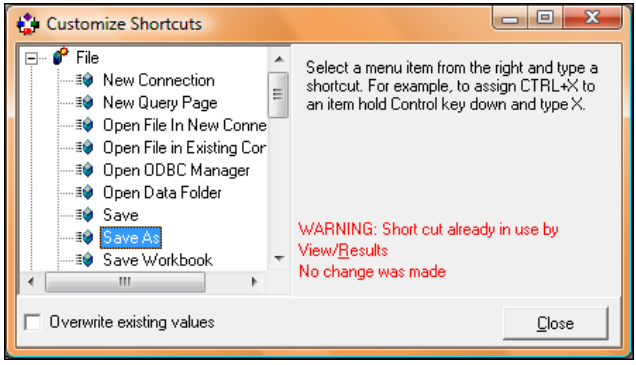


4 If a new keyboard shortcut key is desired, press the key combination, and then click to select **Overwrite existing values**

OR

if a keyboard shortcut key does not exist, press the key combination. If accepted and not in use by another option, the selected key combination is displayed on the **Customize Shortcuts** window.

If the select keyboard shortcut is already in use, the following information is displayed on the **Customize Shortcuts** window:



Continue to enter other key combinations until the value is accepted.

Note:

Any combination of control keys (**Shift**, **Alt**, and **Ctrl**), plus a letter or number, can be used. The only exception is **Ctrl+[space bar]**, which is reserved for use by Intelli Tips.

Chapter 2: Running Queries

Overview

Introduction

This chapter provides information about creating, running, viewing, saving, and printing queries within WinSQL Professional.

In this chapter

This chapter contains the following topics:

Screen Layout

WinSQL provides two different types of “views” (screen layouts) when viewing queries:

- Integrated View
- Classic View

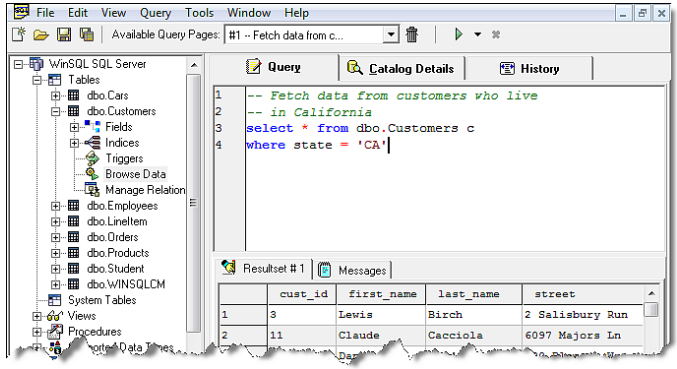
Important note:

For the purposes of this document, all procedural steps and screen shots refer to the Integrated View. However, Classic View screen shots are used in instances where they better serve to illustrate the function being performed.

Integrated View

When using the Integrated View, WinSQL displays part of the catalog screen next to the query window. This allows you to quickly and conveniently see the tables, views, and other objects in the database.

The Integrated View is depicted in the following screen shot:

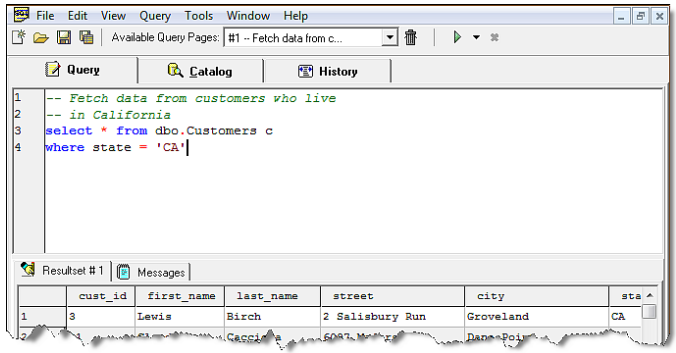


When you click a node that requires additional information, WinSQL automatically switches the current tab from “**Query**” to “**Catalog Details**.” For example, if you click the “Browse Data” or “Manage Relationship” nodes, WinSQL switches the current view to display the additional node information.

Classic View

Unlike the Integrated View, the Classic View does not display the catalog screen. When in Classic View, you must click the **Catalog** tab to view catalog details. An advantage to using this view is that it provides more room to display the query and results tabs.

The Classic View is depicted in the following screen shot:



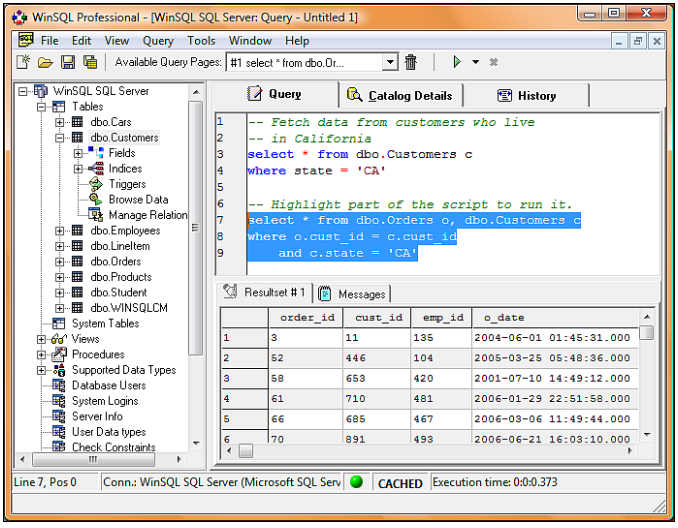
Switching Views

To switch between integrated and classic views, simply click **Switch to Integrated View** or **Switch to Classic View** from the **View** menu.

The text of the menu item changes based on the view you currently are using.

Writing SQL Scripts

Writing and submitting SQL queries to a back-end database is the most important and useful feature of WinSQL Professional. Once a database connection is established, any type of SQL script may be written to extract data from the database.



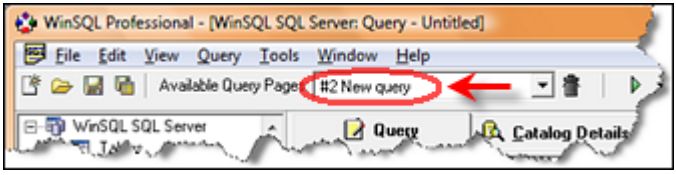
You can highlight part of the script to run, or not highlight anything to run the complete script. If there is more than one result set, WinSQL Professional will display them one after another in different tabs.

Query Pages

A query page is a group of SQL scripts displayed on the screen. You can have multiple query pages per window. Creating multiple query pages helps you manage queries when your SQL scripts get larger.

The first query page is created automatically when you connect to a database. Thereafter, you have an option to create as many query pages as you need. The number of query pages allowed is limited only by available memory on the machine.

To create a query page, point to the **File** menu, and then click **New Query Page**. The number of the current query page is displayed in the **Available Query Pages** field:



You can navigate to different query pages by selecting the desired query page from the **Available Query Pages** drop-down list.

Query Set Workbook

If you have more than one query page created in WinSQL Professional, they can all be saved in a query workbook.

To save all current queries in a workbook, point to the **File** menu, and then click **Save Workbook**.

Using Clipboard

Results from queries can be copied to the Windows Clipboard by pressing **[Ctrl+C]**. The information copied to the Clipboard can then be copied into a variety of other software applications, depending on the type of output desired.

Extended Copy - Code

WinSQL Professional allows you to copy SQL code into HTML or Rich Text format using the **Extended Copy** option in the Query window. You can also convert code using the **Convert to Code** option.

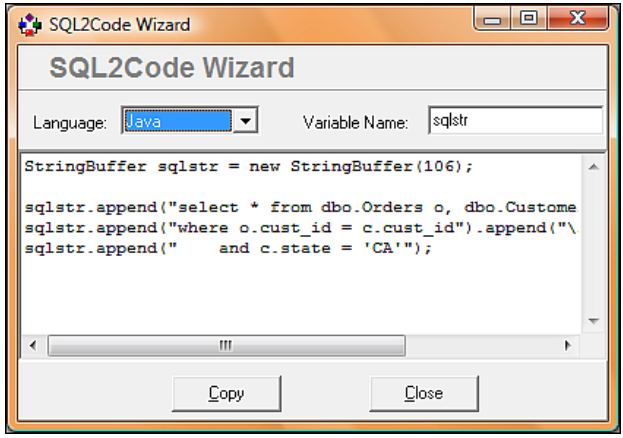
The following topics describe these options.

Convert to Code

This option allows you to write SQL code and convert that code into a third-generation language, such as C-Sharp, Visual Basic, or Java.

Perform the following steps to convert SQL code:

1. Right-click the desired script in the **Query** window, and then click **Extended Copy**.
2. Click **Convert to Code**. The **SQL2Code Wizard** window is displayed:



3. Select the desired language in the **Language** drop-down list.

4 If desired, change the default variable name in the **Variable Name** field.

5 Click **Copy** to copy the text to the clipboard.

Copy to Rich Text

This option copies SQL code to the Windows Clipboard in Rich Text format, which allows you to paste SQL code to different applications, such as a word processing application that uses Rich Text format.

To copy SQL code into Rich Text format, right-click the desired script in the **Query** windows, click **Extended Copy**, and then choose **Rich Text**.

Copy to HTML

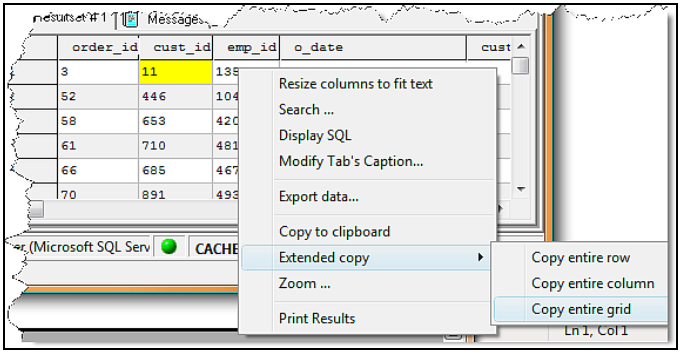
This option converts the SQL code into HTML, while retaining all syntax highlighting, and provides a convenient way of creating HTML documentation.

To copy SQL statements into HTML, right-click the desired script in the **Query** windows, click **Extended Copy**, and then choose **HTML**.

Extended Copy - Results

When query results are displayed in the **Query** window, you have the option to copy data from more than one cell.

To perform an extended copy, click in the results grid in the **Query** window, right-click, and then select the desired option.



Displaying Query Results

WinSQL Professional allows you to display query results in the following formats:

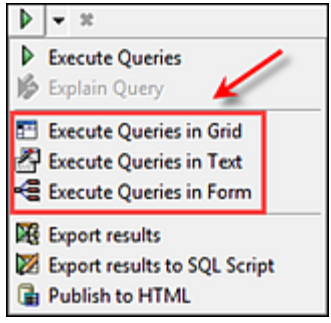
- Grid
- Text
- Form

It is recommended that you display results in Grid unless you are running a query that returns large amount of text containing new line characters. Managing result sets that are displayed in Grid is much easier than in Text. For example, you can save the contents of the results Grid to a CVS, tab-delimited, or spreadsheet file.

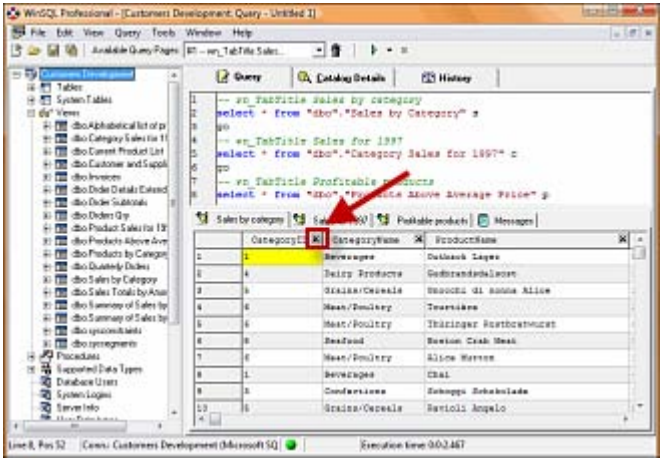
Note:

The result is limited to 16MB of characters per cell when the data is displayed in a Grid control, 4096 characters when displayed in the Text control and 32 KB when displayed in the Form control. Since it is not practical to display the entire 16 MB of data in one grid cell, WinSQL will display a magnifying glass icon if the cell data is greater than 1024 characters. You may lick the magnifying glass to zoom and view the entire data. In case of text field, data will get truncated if the length is greater than 4096 characters. This will **not** happen if you try to export the field to another database.

To select the method used to display query results, click the down arrow in the **Execute Queries** drop-down list, depicted as follows:



The resulting data display can be modified to include or exclude certain columns by click the X in the column header:



This functionality is also available for the Browse Data Grid option available in the Catalog Details tab.

Note: Right-click and select **Display all columns** to display formerly hidden columns.

Modifying results displayed in Grid

Double-click the desired row to modify the results displayed in a grid. The Update Wizard is invoked, allowing you to modify existing values.

Notes:

- You can only modify results sets that are generated from one table.
- Queries containing a JOIN statement are not editable.

Modifying results displayed in Text

Result sets in text are not editable.

Modifying results displayed in Form

To modify data, simply start typing the new value. After specifying the new values you can either press **[Enter]** or click **Save** to save the modified data to the database.

Several keyboard shortcuts can be used when the data is displayed in Form view. These shortcuts are described in the following table:

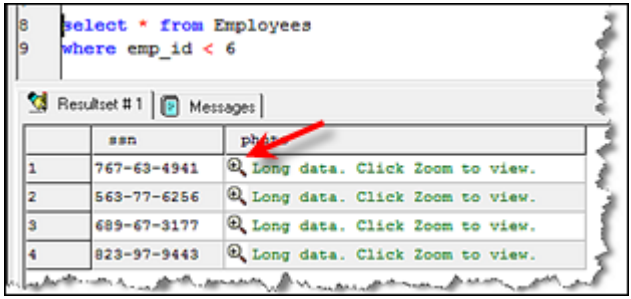
Key combination	Result
[Alt] + right arrow	Displays the next record
[Alt] + left arrow	Displays the previous record
[Shift] + **[Alt]** + right arrow	Saves any modified data and displays the next record
[Shift] + **[Alt]** + left arrow	Saves any modified data and displays the previous record
[Esc]	Discards any modified values and refreshes the window
[Enter]	Saves the newly-modified values. This shortcut only works in single-line edit controls. Pressing **[Enter]** in a multi-line control inserts a new line in the data.

Viewing BLOB/CLOB Data

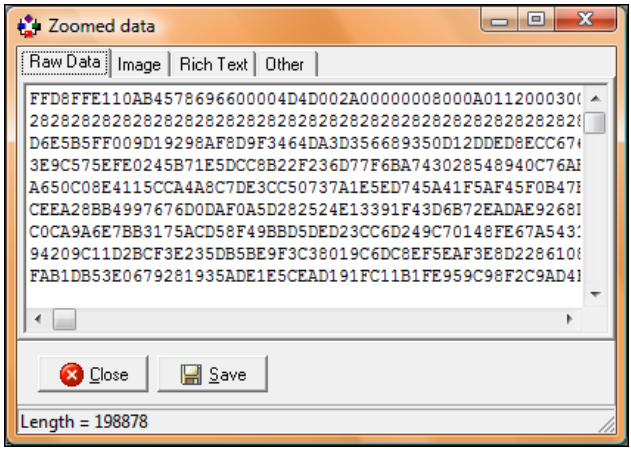
This feature provides a mechanism to view images, audio, video, or other binary format.

Perform the following steps to view BLOB/CLOB data:

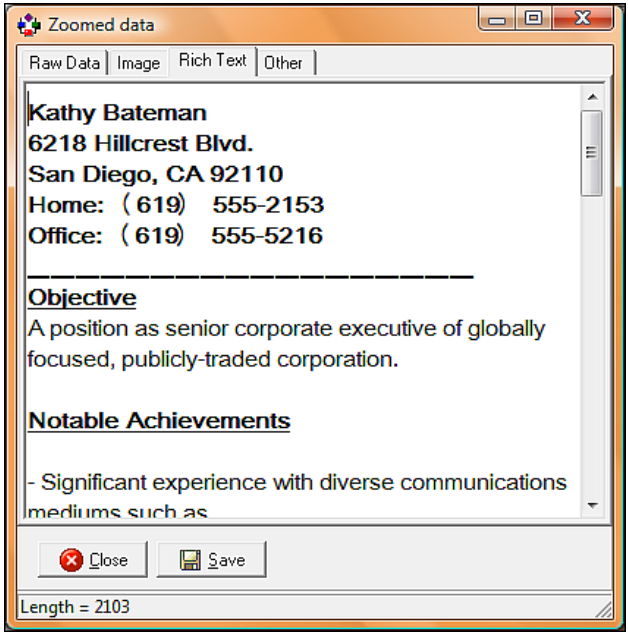
1. Establish a connection to the desired database. For more information, see "Establishing a Database Connection."
2. Click the **Query** tab.
3. Run an existing query, or create a new query that fetches a binary or large text from the database and click the play icon.
4. In the **Resultset** window locate the text in green (indicating long data), and click the zoom icon to display the object:



If the object is a text object, the following information is displayed in the **Zoomed data** window:



Click the **Rich Text** tab to view the formatted text object:

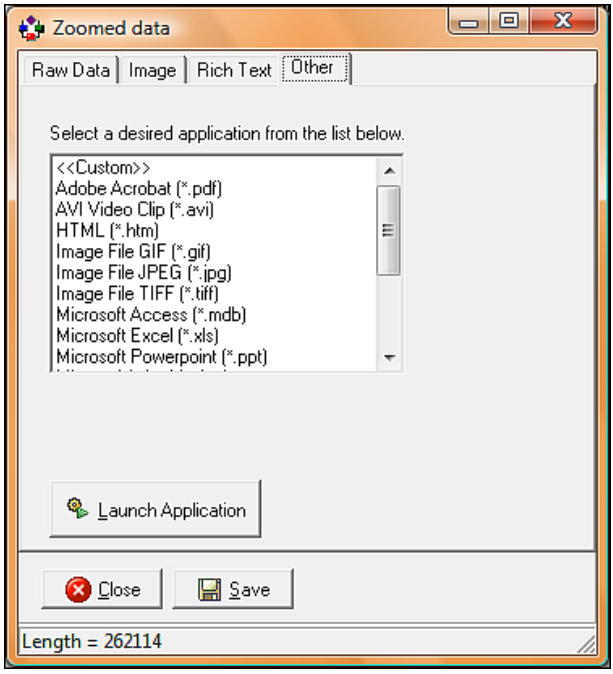


If the object is an image, click the **Image** tab to view the image:

5 Click **Close** to close the **Zoomed data** window.

Note:

If the binary object is not a rich text or image object and is, for example, a Microsoft® Word® document or an Microsoft® Excel® spreadsheet, click the **Other** tab to select the appropriate application to view the object, and then click **Launch Application**:

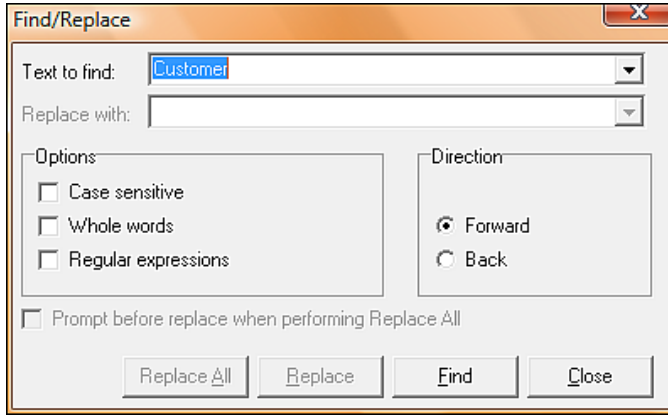


If your desired application is not listed, select **<<Custom>>** and specify the extension for your file. For example, if you have an AutoCad diagram saved in the database specify DWG as the file extension.

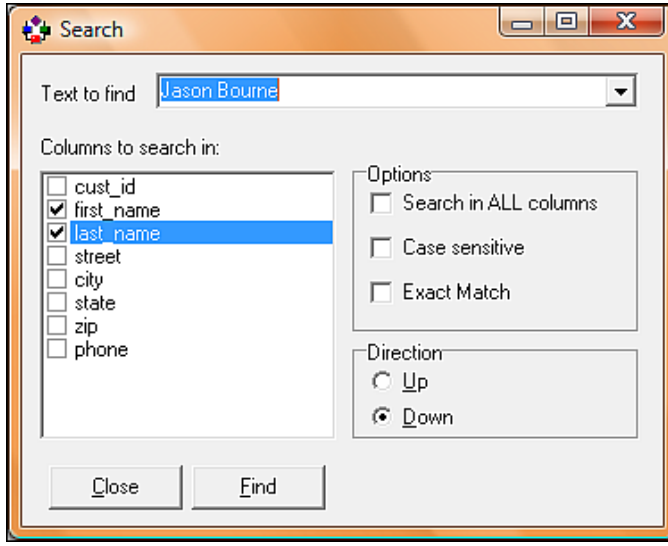
Searching for Text

Using the Find option in WinSQL Professional allows you to search for text within SQL scripts or query results.

To initiate a search, point to the **Edit** menu, and then click **Find**. The appropriate search window is displayed. For example, if the selected control is the **Query** window, the following window is displayed:



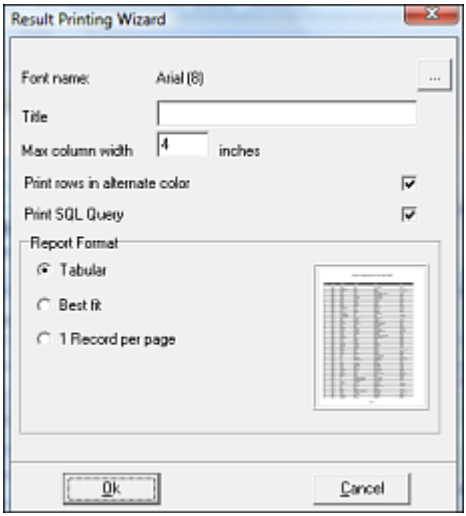
If the selected control is a results Grid, the following window is displayed:



This window displays the columns contained in the Grid. You can either select the desired columns, or you can select the **Search in ALL columns** option to search within all Grid columns.

Printing Results

Results displayed in a Grid can be printed in three different formats by right-clicking the displayed results, and then clicking **Print Results**. A window similar to the following is displayed:



After typing a title for your report in the **Title** field and selecting the desired column width and color option, select one of the available print format options, using the information in the following table as a guideline:

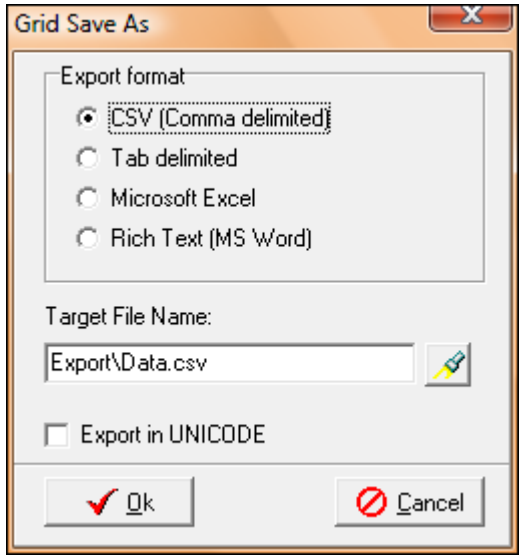
Option	Description
Tabular	This option prints the data in a format similar to a spreadsheet. Pages are added sideways for additional columns and downward for additional rows. This is the best option for printing large amounts of data using the least number of pages.
Best fit	This option allows a row of data in a record to wrap to additional lines, allowing multiple records to be printed on each page. This is the best option for viewing multiple records in one page.

Option	Description
1 Record per page	This option prints one record per page in a format similar to a printed form.

Click **OK** once the desired printing options have been selected.

Exporting Grid Content

Results displayed in a Grid can be exported by right-clicking the displayed results, and then clicking **Export data**, depicted as follows:



Select the desired export output, enter the desired file name in the **Target File Name** field, and then click **OK**.

Intelli Tips

The Intelli Tips feature facilitates the writing of SQL queries by displaying the catalog objects in a pop-up window as the user types his/her queries.

Prerequisites for Intelli Tips

The Intelli Tips feature will only work if:

- You are using WinSQL Professional
- You have fetched the database catalog prior to writing the SQL query

By default, WinSQL Professional fetches the database catalog after the connection is established. This feature can be disabled for use with large databases; however, the database catalog must be manually fetched prior to using Intelli Tips.

When Intelli Tips are enabled, Intelli Tips displays a window containing Table and Field as you type SQL script in the query window. You can also press **[Ctrl+spacebar]** to manually invoke the **Intelli Tips** window.

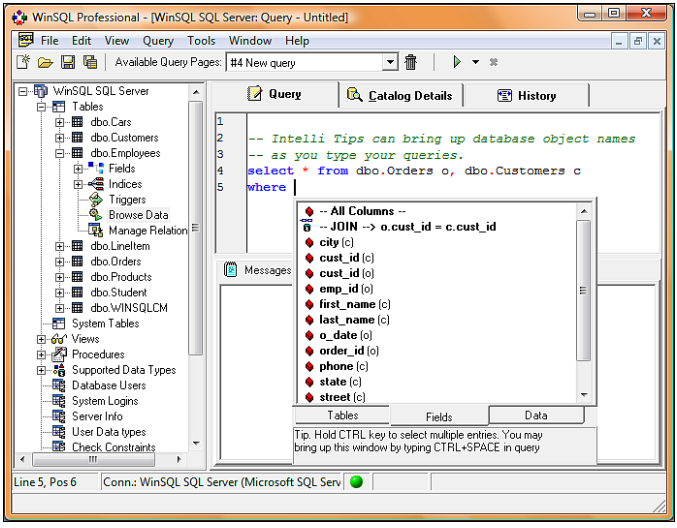
The data displayed in the **Intelli Tips** window is pulled from the **Catalog Details** window. Therefore, any filter you have applied in the **Catalog Details** window applies to the data in the **Intelli Tips** window.

The **Intelli Tips** window can display both tables and fields. WinSQL Professional parses the current query and displays the appropriate tab (tables or fields). To display columns for a particular table, click the **Tables** tab, select the desired table, and then click the **Fields** tab to display the fields within the selected table.

Table Joins

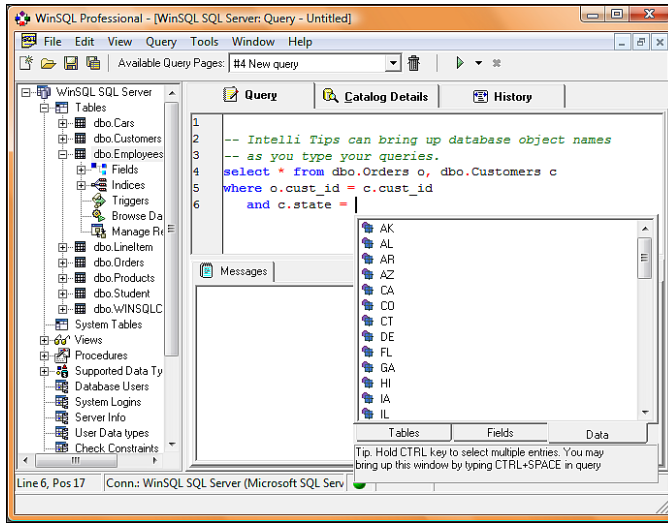
WinSQL Professional recognizes relationships that are specified on your back-end server using foreign key constraints. In addition to foreign keys, you can also create a local relationship among tables. For more information, see "Creating Local Relationships" later in this document.

When relationships are found among tables, the **Intelli Tips** window automatically displays the WHERE clause necessary for the JOIN statement, depicted as follows:



Data Lookup

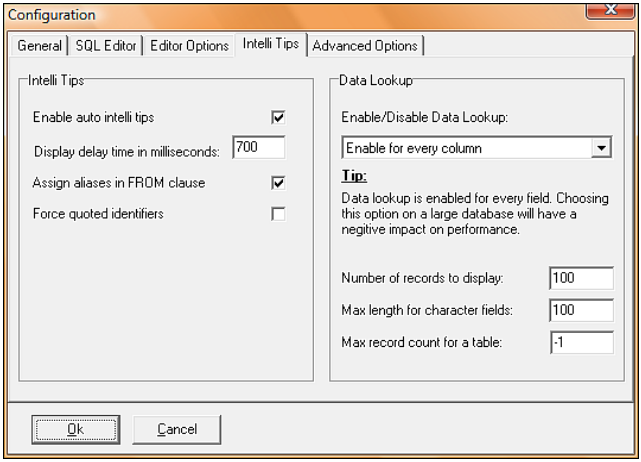
The Data Lookup feature in WinSQL Professional allows users to view the actual values in the tables when writing SQL queries. The value appears automatically when WinSQL Professional detects a WHERE clause in the query window, illustrated as follows:



Configuring Data Lookup Display Options

Perform the following steps to enable or disable and to configure the Data Lookup feature:

1. Start WinSQL Professional and connect to the desired database.
2. From the **Edit** menu, click **Options**.
3. Click the **Intelli Tips** tab. The following window is displayed:



4 Use the information in the following table to specify the desired parameters:

Field	Definition
Enable auto intelli tips	If checked, Intelli Tips will be displayed automatically as you type your SQL queries.
Display delay time in milliseconds	This is the amount of time WinSQL waits after a key is pressed before displaying the Intelli Tips window. This value is in milliseconds

Assign aliases in FROM clause	If checked, WinSQL will assign aliases for tables in the FROM clause. For example, if this NOT checked, the query will look like:
	`Select Customer.Name from Customer`
	When this is checked, the query will look like:
	`Select c.Name from customer c`

Force quoted identifiers	If checked, WinSQL will always enclose object names in quotes. The generated query will look like: `Select *`

Field	Definition
	`From "dbo"."customer"` When this is NOT checked, WinSQL will only put quotes on objects that have an embedded space in their name.
Enable/Disable Data Lookup	Select one of the following options: • **Disable data lookup**: Select this option if you wish to disable Data Lookup. Doing so will prevent the **Data** tab from appearing in the **Intelli Tip** window. • **Enable for every column**: Select this option to enable data lookup for every column whose data type is one of the following: 1) numeric, 2) Date/time, or 3) character (where the length is less than the specific character length). • **Enable for columns that I choose**: If this option is selected, you must specifically assign a SQL query to every column for which you wish to enable data lookup. To assign a SQL query, double-click that column in the **Catalog Details** window (see **Enable for every column**, above).
Number of records to display	The number of records to display in the **Intelli Tips** window. If the query returns more than the specified number of records, they will be ignored.
Max length for character fields	Data lookup is enabled for fields where the column length is less than this value. For example, if there is a remarks field in a table and its size is 255, Data Lookup is not enabled for this field. This value only applies to character

Field	Definition
	types.
Max record count for a table	For large databases, WinSQL Professional may take a long time to fetch records from a table. Therefore, if you specify a value other than -1, WinSQL Professional will only run data lookup queries for table having fewer numbers of rows than what you specify here.

5 Click **OK** when all options have been selected.

Modifying SQL Queries to Retrieve Data

When data is populated in the Intelli Tips window, WinSQL Professional runs a query in the background. This query retrieves necessary records for a particular column and displays them on the screen.

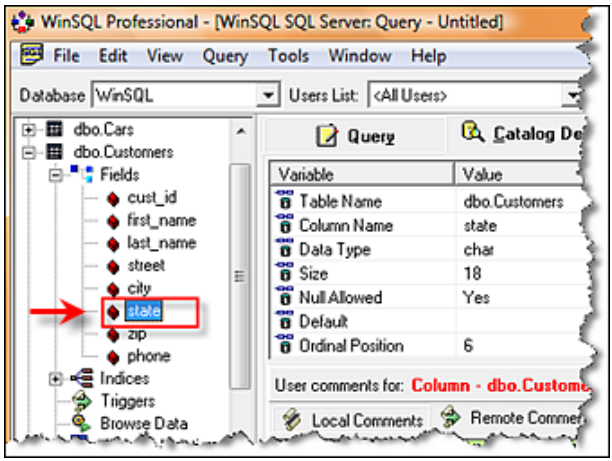
A SQL query can be associated with every column in the database.

Perform the following steps to associate a query with a column.

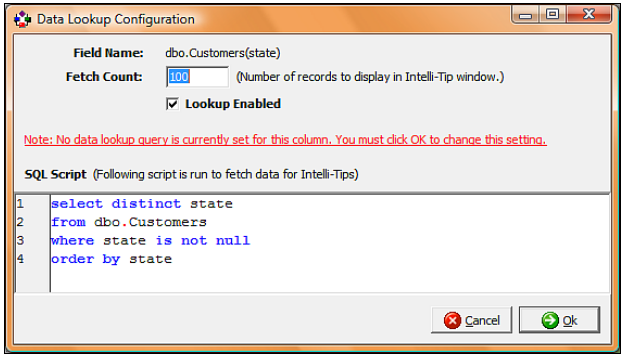
1 Start WinSQL Professional and connect to the desired database.

2 Click the **Catalog Details** tab.

3 Navigate to the desired column in the left pane of the catalog tree:



4 Double-click the desired field. A window similar to the following is displayed:



5 Modify the query as desired, and then click **OK**.

Drill-down Results

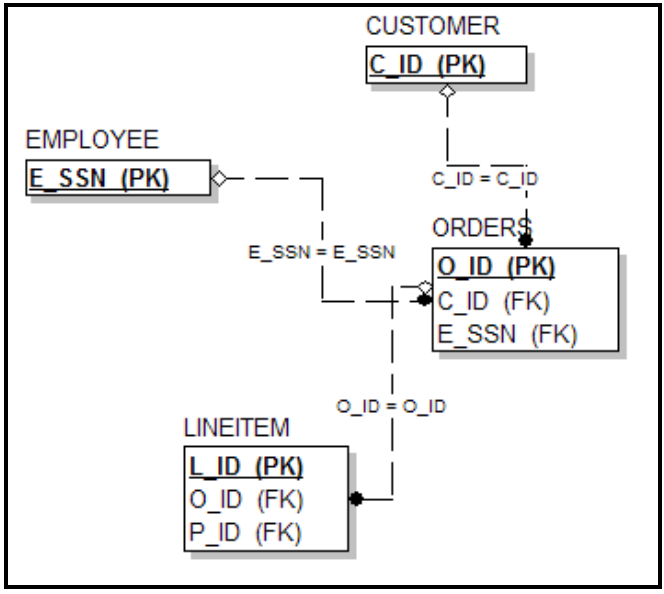
When you run a query from a single table that has relationship, WinSQL Professional allows you to drill down to related rows in the associated table.

Prerequisites for Drill-down

The Drill-down feature will only work when:

- You are using WinSQL Professional
- The FROM clause in the SELECT statement contains only one table
- You have fetched the database catalog prior to running your query
- Either foreign or local relationships are defined in the table

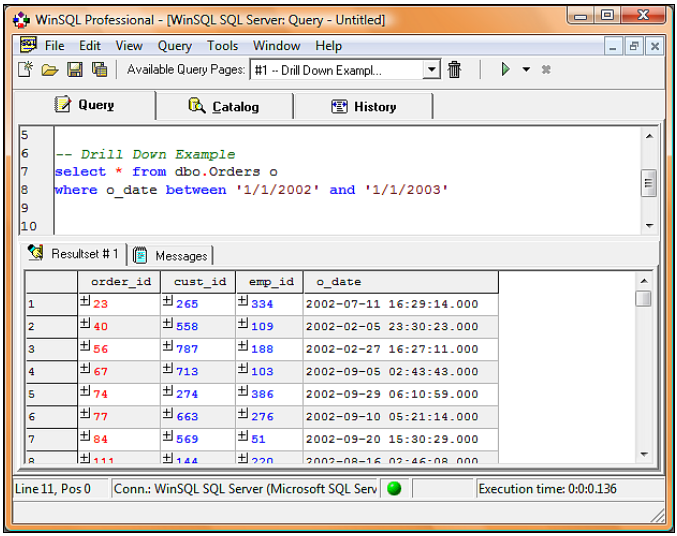
Consider the following scenario as an example, which contains a database with four tables. Their relationships are displayed in the figure below. Notice that the ORDERS table has two parents, EMPLOYEE and CUSTOMER, and it has one child, LINEITEM.



When you run the following query in WinSQL Professional, it displays the columns containing primary and foreign key in a different color with a **+** (plus sign) next to the value.

Note:

The following illustration is in Classic View.

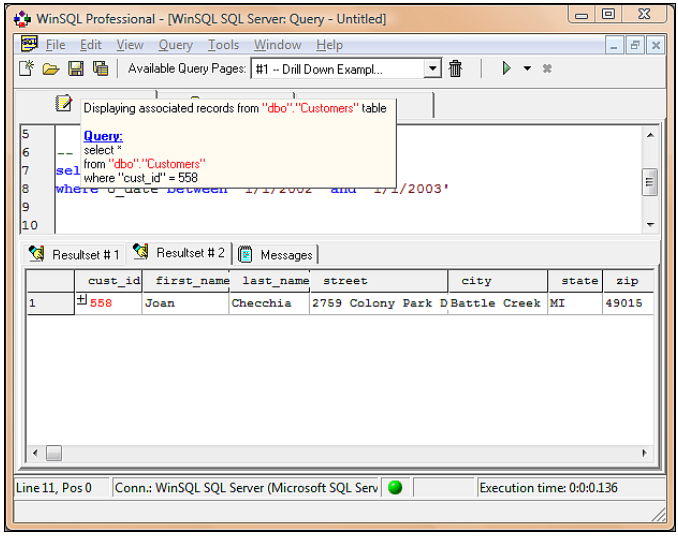


The first column in this example contains the primary key from the ORDERS table, which is displayed in red. The two subsequent columns contain data that map to the CUSTOMER and EMPLOYEE tables and is defined as a foreign key. Data for these columns is displayed in blue.

You can click the **+** to display related data from the associated table. For example if you click the E_SSN column where the value is 50, WinSQL Professional will run another query that will pull the employees whose E_SSN equal 50, depicted as follows:

Note:

The following illustration is in Classic View.



Query History

WinSQL Professional keeps a history of queries you have previously run. The default number of queries stored is 50; however, this value can be changed by clicking **Configure** on the **History** tab, and entering a new value in **Modify historic count** option. In addition, older queries are replaced by newer queries on a "first in, first out" basic.

Historic queries can also be copied to disk for backup purposes.

Comments with SQL Statements

WinSQL Professional recognizes two kinds of comments:

Comment type	Description
Line	A line beginning with two dashes is consider a comment line
Enclosed	Any text that is enclosed by /* /*

Beginning with WinSQL Professional version 3.5, you have the option of allowing WinSQL Professional to parse the comments before the SQL statement is submitted to the server. This option allows you to use comments that a back-end database may not support.

Synametrics recommends that you allow the DBMS to handle supported comments. This option can be configured by pointing to the **Edit** menu and then clicking **Options**.

Query Syntax and SQL Grammar

WinSQL Professional does not contain its own syntax. It inherits the SQL dialect from the back-end server and submits the query "as is" to the back-end database. Similarly, it displays the results data "as is" from the server. This feature allows users to submit queries that are specific to a particular database.

There are, however, exceptions to this rule. The submitted script is modified under the following conditions.

- When you use a WinSQL Professional specific parameter in the query. In this case, the parameter will be replaced before submitting the query

- If a query terminator is found within the script. The default value for query terminator is 'GO'.
- You use a code template. Code templates are explained in more detail in the "Database-Specific Plug-Ins" topic in this document. Templates replace pre-defined tokens into server specific SQL. For example, when a user types DESCRIBE <TABLENAME> in an Oracle database, WinSQL Professional replaces this command with a set of Oracle specific scripts necessary to pull table definition.

Parameterized Queries

Beginning with WinSQL Professional version 2.0, parameters can be accepted in queries. Any text that is preceded by two colons (::) is considered a parameter, and users are prompted to enter a value when the query is run. In addition, WinSQL Professional stores the values for previously-used parameters so that the user does not have to type them every time.

Notes:

- Once a value has been specified for the first parameter in a query, you can press **[Enter]** to move to the next parameter. WinSQL Professional attempts to determine the data type for the subsequent parameters; however, the default data type can be determined by selecting a type from the list.
- Parameter queries cannot be used to export data.

Generate Code

WinSQL's Generate Code function allows users to quickly generate one of the following SQL scripts:

- Select
- Insert
- Update
- Delete
- Create View
- Drop Table

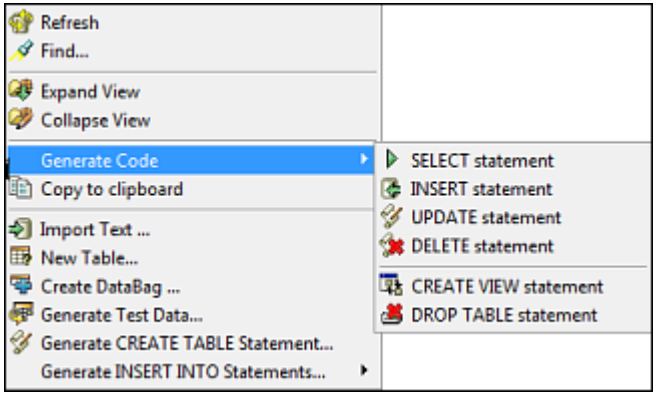
Once selected, the script displays in the **Query** window. The primary keys in the selected table are, by default, used as parameter fields in the script and are indicated by a double colon (**::**).

Note:

The parameter field indicator can be modified, if desired. For more information, see the "General Tab" topic in Chapter 3, Configuration Options.

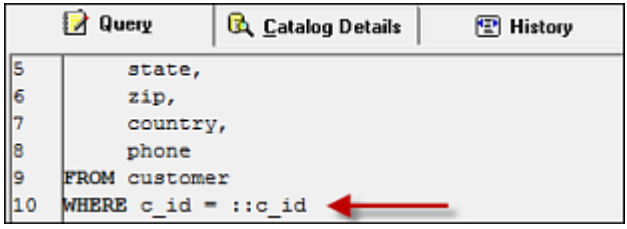
Perform the following steps to use the **Generate Code** function:

1. Start WinSQL Professional and connect to the desired database.
2. Click the plus (+) sign by Tables to view a list of tables in the catalog tree of your database.
3. Click to select the desired table, and then right-click to display the submenu.
4. Click **Generate Code**. A window similar to the following is displayed:



5 Click the desired script.

The content of the selected script is displayed in the **Query** window. In the following example, the **SELECT statement** was generated, and the primary key for the table (**c_id**) is the parameter field:



Notes:

- Parameter fields can be removed from the script as desired.
- When generating a new script, the highlighted content of the existing script displayed in the **Query** window is overwritten.

  If the cursor is positioned within the content of an existing script when a new script is generated, the new script is pasted at the current cursor position.

6 Click the **Execute queries** icon ( ) to run the script.

SQL Wizard

The SQL Wizard, available in WinSQL Professional, allows you to create SQL queries with the help of a wizard, rather than creating queries by entering code. It allows you to select the desired tables and fields, and to apply filter conditions, before creating the SQL code for you.

The SQL Wizard determines the referential integrity constraints specified against any table and draw appropriate relationships.

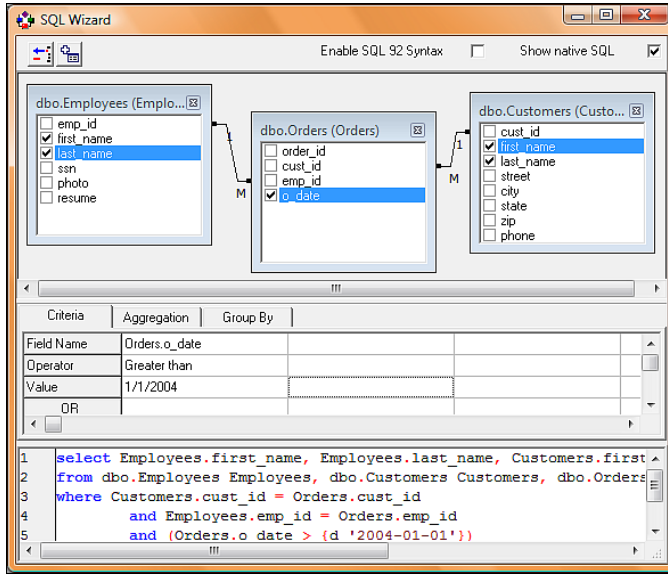
Note:

WinSQL Professional may not be able to determine referential integrity constraints in certain case where the ODBC driver does not provide necessary information, such as drivers for Microsoft Access.

To activate the SQL Wizard, point to the **Tools** menu, and then click **SQL Query Wizard**.

Show Native SQL

If this option is selected, WinSQL Professional converts a generic ODBC-specific SQL script into a database-specific syntax. This conversion is done through the ODBC driver and depends on the capabilities of the driver.



Scripting Wizard

WinSQL's scripting wizard allows users to select and execute multiple DML (INSERT, UPDATE, DELETE) or DDL (CREATE TABLE/VIEW/PROCEDURE) queries at one time, rather than executing them individually—an often time-consuming process

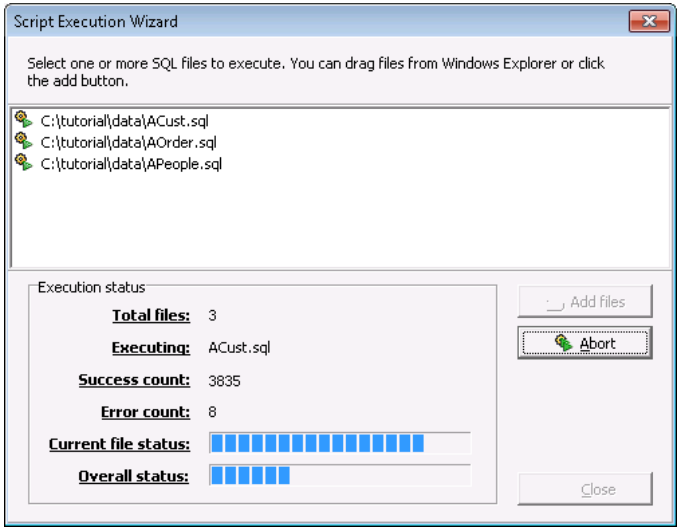
Important Note:

Do not run SELECT queries using the Scripting Wizard.

Perform the following steps to use the Scripting Wizard:

1 Start WinSQL Professional and connect to the desired database.

2 From the **Tools** menu, click **Scripting Wizard**. A window similar to the following is displayed:



3 Click **Add files** to navigate to and select the SQL script(s) you wish to run.

4 Click **Execute** to run the script(s).

The **Script Execution Wizard** window provides information about the execution status of the running scripts, along with information about successful and

unsuccessful query runs. This information is also written to a log file that can be saved and/or viewed using a standard text reader.

Chapter 3: Configuration Options

Overview

Introduction

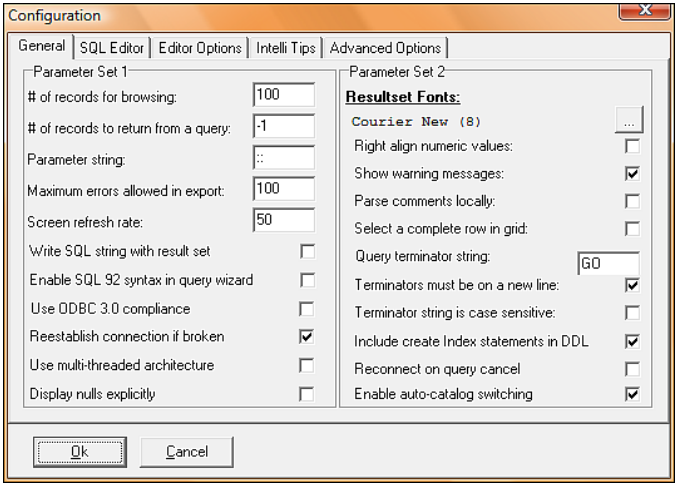
This chapter provides information about customizable configuration options available in WinSQL Professional.

In this chapter

This chapter contains the following topics:

General Tab

To access configuration options available on the **General** tab, point to the **Edit** menu, and then click **Options**. The **General** tab is displayed:



The information in the following table describes each of the options available on the **General** tab:

Option	Description
# of records for browsing	This is a numeric value corresponding to the number of records to display in the **Browse Data** feature available in the **Catalog Details** window. The default value is 100. **Caution**: If you set this value to a large number, WinSQL Professional will fetch a large result set whenever **Browse Data** is invoked. This can potentially slow down not only your machine but can also have negative affects on your network. Synametrics suggests that you instead apply filters on the records to limit the size of the returned results.

Option	Description
# of records to return from a query	This parameter limits the number of records that are returned from a SELECT statement. The default value is -1, which indicates no limit.
Parameter string	This token string is used to specify a parameter in the query window. The default value for this field is: Example: The following query contains FirstName as parameter `Select * from customer where fName = ::FirstName` When this query is run, WinSQL Professional prompts for a value in the **FirstName** parameter field.
Maximum errors allowed in export	This option only applies to WinSQL Professional and specifies the number of errors allowed before an export routine is terminated.
Screen refresh rate	This value indicates the number of records to fetch before the screen is refreshed. This option is only applicable when results are displayed in Text control.
Write SQL string with result set	If checked, the SQL statement is printed right before the result set. This is useful if you are displaying the result in the Text control.

Option	Description

Enable SQL 92 syntax in query wizard	If checked, the SQL Query Wizard will use SQL 92 syntax for JOIN.
	Example: The following query is written in traditional SQL
	`From Customers, Orders`
	`CWhere Customers.id = Orders.id`
	The same query can be written in SQL 92 syntax as follows:
	`From Customers INNER JOIN Orders ON Customers.id = Orders.id`

Use ODBC 3.0 compliance	If checked, WinSQL Professional will use ODBC 3.0 compliant calls to the ODBC manager. Synametrics recommends that you leave this option unchecked unless your driver specifically requires it.
Fetch catalog after connection	If checked, WinSQL Professional will fetch the database catalog as soon as a new connection is established. Even though this option may cause WinSQL Professional to take longer to connect, the Intelli Tips feature will work as soon as you start typing your queries. When this option is unchecked, Intelli Tips will not work unless you fetch the catalog first.
Reestablish connection if broken	An attempt is made to connect to the database if the connection is severed.
Use multithreaded architecture	If selected, queries are run in a second thread. It is strongly recommended that you leave this option checked (default).
Resultset Fonts	

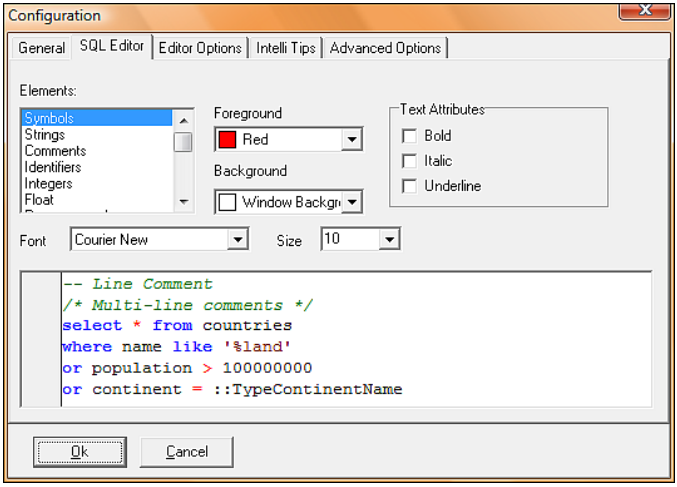
Option	Description
Courier New (8)	Click the ellipses icon to change the font of the text displayed in the **Result Set** window. **Note**: To change the font of the text displayed in the Query window, point to the **Edit** menu, click **Options**, click the **Editor Options** tab, and then select the desired font.
Right align numeric values	If checked, all numeric values in the result set will be right aligned. This option is only applicable when results are displayed in Text control.
Show warning messages	If checked, warning messages are displayed. This option is most helpful if the database is either Sybase or MS-SQL Server, but it can also be used by other databases. Synametrics recommends that you select this option.
Parse comments locally	If checked, WinSQL Professional will parse the query script and strip all the comments before submitting it to the back-end database.
Select a complete row in grid	If checked, individual cells within a result grid cannot be selected. All columns for a row are selected.
Query terminator string	This string breaks the script in multiple parts, and each part is sent separately to the database. The default value for this parameter is **GO**, and it works in conjunction with the **Terminator must be on a new line** and **Terminator string is case sensitive** configuration options.
Terminators must be on a new line	If checked, a terminator string is only treated as a terminator if it appears on a new line.

Option	Description
Terminator string is case sensitive	If checked, the terminator string is case sensitive.
Include create Index statements in DDL	If selected, WinSQL includes CREATE INDEX statements when reverse engineering a CREATE TABLE statements for a table.
Reconnect on query cancel	Selecting this option forces WinSQL to close the connection when **Cancel** is clicked while a query is running.
Enable auto-catalog switching	If selected, WinSQL checks if the database has changed after running a query.

SQL Editor Tab

Options on this tab manage syntax highlighting for SQL scripts. You can select different elements from list and define color and style for them.

To access configuration options available on the **SQL Editor** tab, point to the **Edit** menu, and then click **Options**. Click the **SQL Editor** tab:



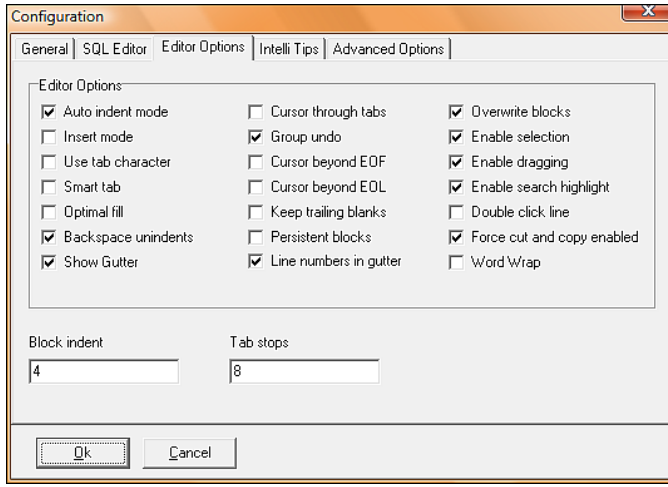
Note:

Font and Size apply to the entire SQL script and are not specific to any one element.

Make any desired changes, and then click **OK**.

Editor Options Tab

To access configuration options available on the **Editor Options** tab, point to the **Edit** menu, and then click **Options**. Click the **Editor Options** tab:



The information in the following table describes each of the options available on the **Editor Options** tab:

Option	Description
Auto indent mode	Positions the cursor under the first nonblank character of the preceding nonblank line when you press **[Enter]**.
Insert mode	Inserts text after any selection.
Use tab character	Inserts the tab character into the text when the user presses the **[Tab]** key. If this option is not selected, spaces are inserted instead.
Smart tab	Tabs to the first non-whitespace character in the preceding line.
Optimal fill	Begins every auto-indented line with the least number of characters possible, using tabs and spaces as necessary.

Option	Description
Backspace unindents	Aligns the insertion point to the previous indentation level (outdents it) when **[Backspace]** is pressed and when the cursor is on the first nonblank character of a line.
Show Gutter	Displays a non-editable area on the left hand side that displays line numbers.
Highlight matching brackets	Highlights matching brackets when the cursor is between two brackets.
Cursor through tabs	Enables the arrow keys to move the cursor to the logical spaces within each tab character.
Group undo	Undoes your last editing command, as well as any subsequent editing commands of the same type, when you press **[Alt]+[Backspace].**
Cursor beyond EOF	Allows the cursor to be positioned beyond end-of-file.
Cursor beyond EOL	Allows the cursor to be positioned beyond end-of-line.
Keep trailing blanks	Retains any blanks you might have at the end of a line.
Persistent blocks	Keeps marked blocks of text selected—even when the cursor is moved—until a new block of text is selected.
Line numbers in gutter	Displays line numbers in the gutter area.
Overwrite blocks	Overwrites selected text with new text.
Enable selection	Enables the selection of text when using the text editor.

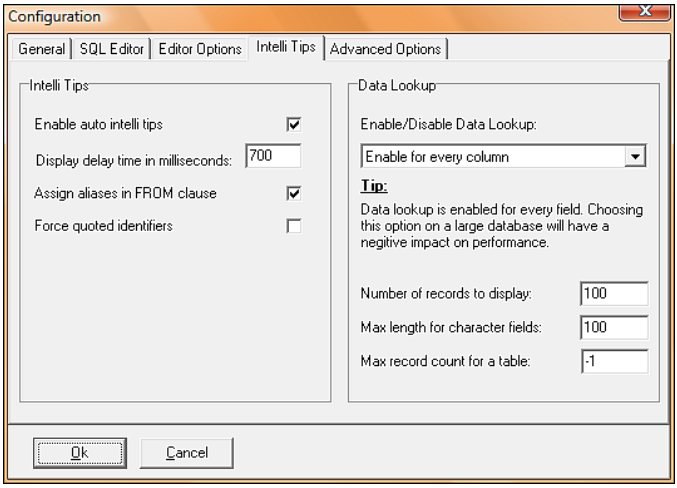
Option	Description
Enable dragging	Allows moving text by dragging it to a new position.
Enable search highlight	Highlights text when using the **Find and Replace** function.
Force cut and copy enabled	If selected, cut/copy operations are enabled even if there is no text currently selected, and it clears the clipboard if performing a cut/copy operation without having first selected text.
Word Wrap	Wraps the line to the next line if it is too long.

Once all selections have been made, click **OK** to save them and close this window.

Intelli Tips Tab

Options on this tab define how the Intelli Tips feature functions.

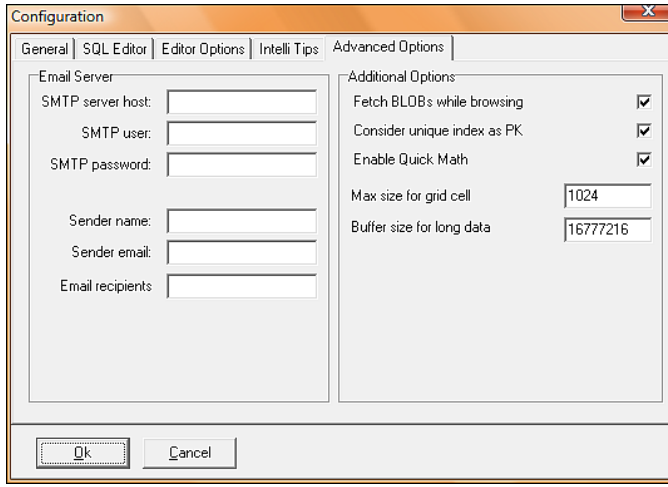
To access configuration options available on the **Intelli Tips** tab, point to the **Edit** menu, and then click **Options**. Click the **Intelli Tips** tab:



For detailed information about the options available on this tab, see the "Configuring Data Lookup Display Options" topic earlier in this document.

Advanced Options Tab

To access configuration options available on the **Advanced Options** tab, point to the **Edit** menu, and then click **Options**. Click the **Advanced Options** tab:



The information in the following table describes each of the options available on the Advanced Options tab:

Option	Description
SMTP server host	Host name for your SMTP server. This is used to send outbound email messages, which is used by the built-in scheduler.
SMTP user	Often SMTP servers require users to login before relaying an outbound email. Specify your user id in this field.
SMTP password	Specify your SMTP password.
Sender name	Outbound emails contain this value as the sender's name. This is typically your name.
Sender email	Outbound emails contain this value as the sender's email. This is typically your email address.

Option	Description
Email recipients	Specify the email address of the recipients who should get notifications when a scheduled task completes.
Fetch BLOBs while browsing	If selected, BLOB and CLOB data are fetched automatically while browsing data.
Consider unique index as PK	If selected, WinSQL treats a unique index as primary key if either one is not specified or the ODBC driver does not support this feature.
Enable Quick Math	If checked, Quick Math tip is displayed when you select more than one cell in the result grid that contains numeric values. The Quick Math tip displays the sum, maximum, minimum, and average values for the selected cells.
Max size for grid cell	This is maximum size for a grid cell before WinSQL will treat it as long data. If the content length of a cell exceeds this value, user must click the “Zoom” icon to view the entire data.
Buffer size for long data	This option affects how large data is inserted into the database when user tries to copy contents of a binary or large text file. Since large files are sent in chunks, this value specifies the size of one chunk. Most likely you should never have to change this value.

Chapter 4: Supported Functions

Overview

Introduction

This chapter provides information scalar functions and how to use them with WinSQL Professional.

In this chapter

This chapter contains the following topics:

Scalar Functions

In addition to supporting all functions supported by the back-end database, WinSQL Professional also supports ODBC scalar functions. In most cases, you will not need to use scalar functions since the back-end database usually has a richer set. However, in some cases, for instance when the source is a text file, you will find these functions extremely helpful.

Following are the types of scalar functions:

- String functions
- Numeric functions
- Time and date functions
- System functions

The syntax for using scalar functions is as follows:

```
{fn FUNCTION_NAME( PARAMETERS ) }
```

The string `{fn` MUST precede the function name and a `}` must be followed after the close parenthesis.

Example:

```
select {fn SUBSTRING(first_name, 1, 3)}
FirstThree, first_Name
from customers
where cust_id = 1049
```

This query displays the following result:

```
FirstThree  first_Name
----------  ------------------------
Den         Denzil
 1 Row(s) affected
```

The **FirstThree** field only displays the first 3 characters of the name.

Important note:

Your driver may not support all of the functions listed below. Check the driver help file for a detailed listing of supported functions.

String Functions

Function name	Description
ASCII(string_exp**)**	Returns the ASCII code value of the leftmost character of *string_exp* as an integer.
BIT_LENGTH(string_exp**)**	Returns the length in bits of the string expression.
CHAR(code**)**	Returns the character that has the ASCII code value specified by code. The value of code should be between 0 and 255; otherwise, the return value is data source-dependent.
CONCAT(string_exp1, string_exp2**)**	Returns a character string that is the result of concatenating *string_exp2* to *string_exp1*. The resulting string is DBMS-dependent. For example, if the column represents by *string_exp1* contained a NULL value, DB2 would return NULL, but SQL Server would return the non-NULL string.
DIFFERENCE(string_exp1, string_exp2**)**	Returns an integer value that indicates the difference between the values returned by the SOUNDEX function for *string_exp1* and *string_exp2.*
INSERT(string_exp1, start, length, string_exp2**)**	Returns a character string where *length* characters have been deleted from string_exp1 beginning at start and where *string_exp2* has been inserted into *string_exp,* beginning at the *start.*
LCASE(string_exp**)**	Returns a string equal to that in *string_exp* with all uppercase characters converted to lowercase.
LEFT(string_exp, count**)**	Returns the leftmost count characters of *string_exp.*
LENGTH(string_exp**)**	Returns the number of characters in *string_exp*, excluding trailing blanks.

Function name	Description
LOCATE(string_exp1, string_exp2, [start]**)**	Returns the starting position of the first occurrence of *string_exp1* within *string_exp2*. The search for the first occurrence of *string_exp1* begins with the first character position in *string_exp2* unless the optional argument, *start*, is specified. If *start* is specified, the search begins with the character position indicated by the value of *start*. The first character position in *string_exp2* is indicated by the value 1. If *string_exp1* is not found within *string_exp2*, the value 0 is returned.
LTRIM(string_exp**)**	Returns the characters of *string_exp*, with leading blanks removed.
REPEAT(string_exp, count**)**	Returns a character string composed of *string_exp* repeated *count* times.
REPLACE(string_exp1, string_exp2, string_exp3**)**	Search *string_exp1* for occurrences of *string_exp2* and replace with *string_exp3.*
RIGHT(string_exp, count**)**	Returns the rightmost *count* characters of *string_exp*.
RTIM(string_exp)	Returns the characters of *string_exp* with trailing blanks removed.
SOUNDEX(string_exp**)**	Returns a data source – dependent character string representing the sound of the words in *string_exp*. For example, SQL Server returns a 4-digit SOUNDEX code; Oracle returns a phonetic representation of each word.
SPACE(count**)**	Returns a character string consisting of *count* spaces.

Function name	Description
SUBSTRING(string_exp, start, length**)**	Returns a character string that is derived from *string_exp* beginning at the character position specified by start for length characters.
UCASE(string_exp**)**	Returns a string equal to that in *string_exp* with all lowercase characters converted to uppercase

Numeric Functions

Function name	Description
ABS(numeric_exp**)**	Returns the absolute value of *numeric_exp*
ACOS(float_exp)	Returns the arccosine of *float_exp* as an angle, expressed in radians.
ASIN(float_exp)	Returns the arcsine of *float_exp* as an angle, expressed in radians.
ATAN(float_exp)	Returns the arctangent of *float_exp* as an angle, expressed in radians.
ATAN2(float_exp1, float_exp2)	Returns the arctangent of the x and y coordinates, specified by *float_exp1* and *float_exp2.*
CEILING(numeric_exp**)**	Returns the smallest integer greater than or equal to numeric_exp.
COS(float_exp)	Returns the cosine of *float_exp*, where *float_exp* is an angle expressed in radians.
COT(float_exp**)**	Returns the cotangent of *float_exp*, where *float_exp* is an angle expressed in radians.
DEGREES(numeric_exp**)**	Returns the number of degrees converted from *numeric_exp* radians.

Function name	Description
EXP(float_exp**)**	Returns the exponential value of *float_exp.*
FLOOR(numeric_exp**)**	Returns the largest integer less than or equal to *numeric_exp*.
LOG(float_exp**)**	Returns the natural logarithm of float_exp.
LOG10(float_exp**)**	Returns the base 10 logarithm of *float_exp*.
MOD(integer_exp1, integer_exp2**)**	Returns the remainder (modulus) of *integer_exp1* divided by *integer_exp2*.
PI()	Returns the constant value of pi as a floating point value.
POWER(numeric_exp, integer_exp**)**	Returns the value of *numeric_exp* to the power of *integer_exp*.
RADIANS(numeric_exp**)**	Returns the number of radians converted from *numeric_exp* degrees.
RAND([integer_exp]**)**	Returns a random floating point value using *integer_exp* as the optional seed value.
ROUND(numeric_exp, integer_exp**)**	Returns *numeric_exp* rounded to *integer_exp* places right of the decimal point. If *integer_exp* is negative, *numeric_exp* is rounded to |*integer_exp*| places to the left of the decimal point.

Function name	Description
SIGN(numeric_exp**)**	Returns an indicator of the sign of *numeric_exp*. If *numeric_exp* is less than zero, –1 is returned. If *numeric_exp* equals zero, 0 is returned. If *numeric_exp* is greater than zero, 1 is returned.
SIN(float_exp**)**	Returns the sine of *float_exp*, where *float_exp* is an angle expressed in radians.
SQRT(float_exp**)**	Returns the square root of *float_exp*.
TAN(float_exp**)**	Returns the tangent of *float_exp*, where *float_exp* is an angle expressed in radians.
TRUNCATE(numeric_exp, integer_exp**)**	Returns *numeric_exp* truncated to *integer_exp* places right of the decimal point. If *integer_exp* is negative, *numeric_exp* is truncated to |*integer_exp*| places to the left of the decimal point.

Time, Date, and Interval Functions

Function name	Description
CURDATE()	Returns the current date.
CURTIME()	Returns the current local time.
DAYNAME(date_exp)	Returns a character string containing the data source – specific name of the day (for example, Sunday through Saturday or Sun. through Sat. for a data source that uses English, or Sonntag through Samstag for a data

Function name	Description
	source that uses German) for the day portion of *date_exp*.
DAYOFMONTH(date_exp)	Returns the day of the month based on the month field in date_exp as an integer value in the range of 1 – 31.
DAYOFWEEK(date_exp)	Returns the day of the week based on the week field in *date_exp* as an integer value in the range of 1 – 7, where 1 represents Sunday.
DAYOFYEAR(date_exp)	Returns the day of the year based on the year field in *date_exp* as an integer value in the range of 1 – 366.
HOUR(time_exp)	Returns the hour based on the hour field in *time_exp* as an integer value in the range of 0 – 23.
MINUTE(time_exp)	Returns the minute based on the minute field in *time_exp* as an integer value in the range of 0 – 59.
MONTH(date_exp)	Returns the month based on the month field in *date_exp* as an integer value in the range of 1 – 12.
MONTHNAME(date_exp)	Returns a character string containing the data source – specific name of the month (for example, January through December or Jan. through Dec. for a data source that uses English, or January through December for a data source that uses German) for the month portion of *date_exp*.
NOW()	Returns current date and time as a timestamp value.
QUARTER(date_exp)	Returns the quarter in *date_exp* as an integer value in the range of 1 – 4, where 1 represents January 1 through March 31.
SECOND(time_exp)	Returns the second based on the second field in *time_exp* as an integer

Function name	Description
	value in the range of 0 – 59.
TIMESTAMPDIFF(interval, timestamp_exp1, timestamp_exp2)	Returns the integer number of intervals of type interval by which *timestamp_exp2* is greater than *timestamp_exp1*. Valid values of interval are the following keywords: SQL_TSI_FRAC_SECOND SQL_TSI_SECOND SQL_TSI_MINUTE SQL_TSI_HOUR SQL_TSI_DAY SQL_TSI_WEEK SQL_TSI_MONTH SQL_TSI_QUARTER SQL_TSI_YEAR where fractional seconds are expressed in billionths of a second. For example, the following SQL statement returns the name of each employee and the number of years he or she has been employed: SELECT NAME, {fn TIMESTAMPDIFF(SQL_TSI_YEAR, {fn CURDATE()}, HIRE_DATE)} FROM EMPLOYEES If either timestamp expression is a time value and interval specifies days, weeks, months, quarters, or years, the date portion of that timestamp is set to the current date before calculating the difference between the timestamps. If either timestamp expression is a date value and interval specifies fractional seconds, seconds, minutes, or hours, the time portion of that timestamp is set to 0 before calculating the difference between the timestamps. **Important:** Not all drivers support all of the intervals.

Function name	Description
WEEK(date_exp**)**	Returns the week of the year based on the week field in *date_exp* as an integer value in the range of 1 – 53.
YEAR(date_exp**)**	Returns the year based on the year field in *date_exp* as an integer value. The range is data source – dependent.

System Functions

Function name	Description
DATABASE()	Returns the name of the database corresponding to the connection.
IFNULL(exp_value)	If *exp* is null, value is returned. If *exp* is not null, *exp* is returned.
USER()	Returns the user name in the DBMS.

Chapter 5: Database Catalog

Overview

Introduction

This chapter provides information about managing the database catalog, including viewing, printing, and applying filters.

In this chapter

This chapter contains the following topics:

About the Database Catalog

Before running any queries, it is important to know what objects are available in the back-end database. The **Catalog Details** tab in WinSQL provides a convenient way for you to quickly and efficiently glance at the elements contained within the database..

The **Catalog Details** tab presents its information to you in a tree view control, allowing easy access to the hierarchical nature of the metadata. The tree control window displays such information as:

- Tables
- Views
- Fields
- Indexes
- Stored Procedures
- Triggers
- Relationships between tables

Database Catalog Caching

By definition, “caching” means to use a temporary storage area to house frequently-accessed data, thereby decreasing access time. Upon connection to your database, WinSQL provides the option of automatically creating a cached version of your database catalog on your local machine. Caching your database catalog reduces the amount of time it takes to fetch metadata information from the back-end database, which greatly increases access time to your data.

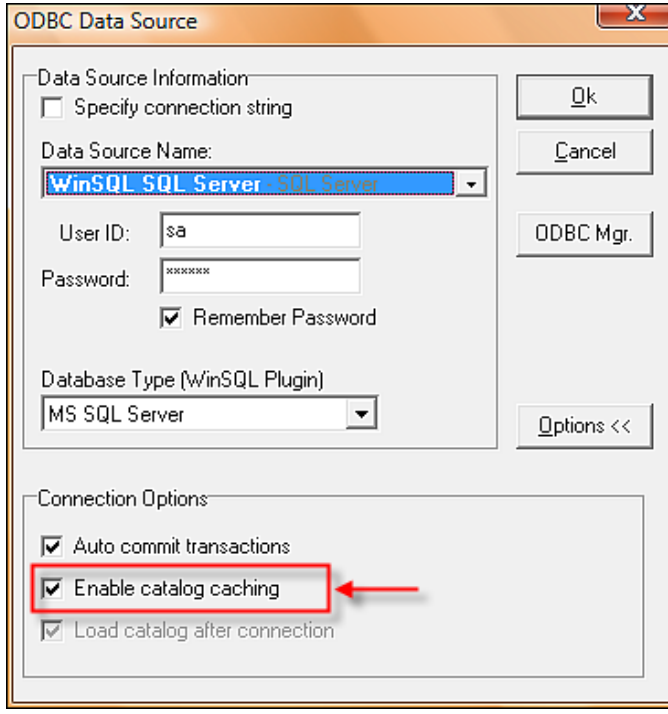
When caching is enabled, WinSQL stores the contents of the **Catalog Details** window to a local file. The next time you connect to the same database, WinSQL loads the catalog information from this local file rather than sending queries to the database.

Synametrics recommends that you use catalog caching for:

- Large databases
- Databases that are located on a slow network

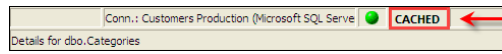
Enabling Catalog Caching

Click **Enable catalog caching** at the time of connection:

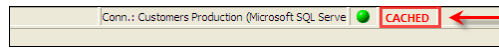


Notes:

- If **Enable catalog caching** is selected at connection, WinSQL will load the catalog immediately after connecting. Therefore, the **Load catalog after connection** option on this window is disabled.
- When the catalog is loaded from cache, the word **CACHED** displays in the WinSQL status bar:



- If the loaded cache is over 15 days old, the color of this word turns red (**CACHED**):



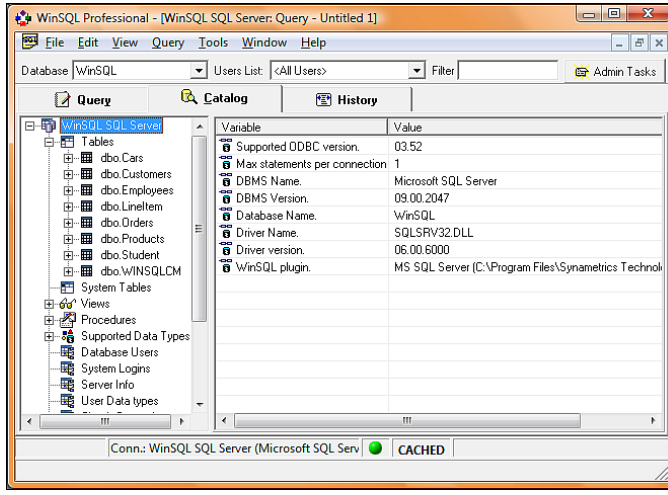
This is a reminder that you should refresh the catalog. To refresh the catalog, you can either press **F5** or click **Refresh Catalog** from the **View** menu. You can also click this indicator to determine how old this cache is.

Viewing the Database Catalog

Viewing the database catalog is very useful when writing SQL statements. WinSQL Professional displays this information in a hierarchical format, depicted as follows:

Note:

The following illustration is in Classic View.



The objects in the **Catalog Details** window are divided into two categories:

- Common objects to all databases
- RDBMS specific object

The first category is available for all databases and is available in all editions of WinSQL Professional. However, the second category is only available in WinSQL Professional, and the contents depend upon the back-end database.

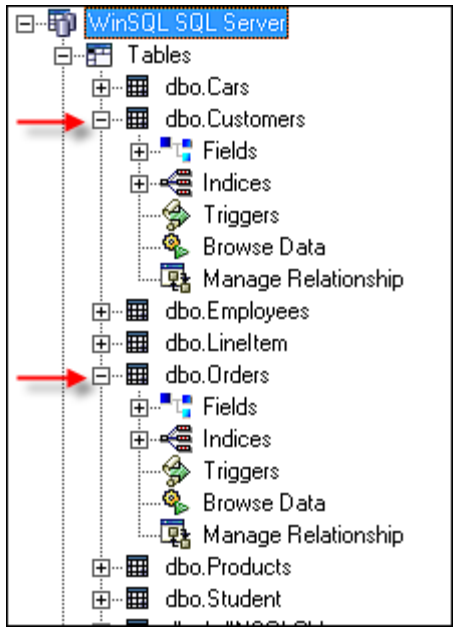
Notes:

- The screen shot above displays a connection to MS-SQL Server and contains the database-specific features you see if the database type is MS-SQL Server.
- Catalog information depends heavily on the capability of the ODBC driver. WinSQL Professional queries these drivers and

determines which information is available. For example, the CLI drivers associated with the Informix client setup cannot display the parameters for a stored procedure.

Exploring Objects in the Catalog

The entire database catalog is available through a Tree control. Click on the **+** (plus sign) to open a detailed section for a particular object. For example, to see a list of available fields for a table, click the **+** for the desired table, and then select **Fields**.



Providing Filters

In some situations where the number of tables is very large, it may take a long time to display all the tables and their fields. The **Catalog Details** window provides two filtering parameters that can restrict the total number of objects displayed. These filters are:

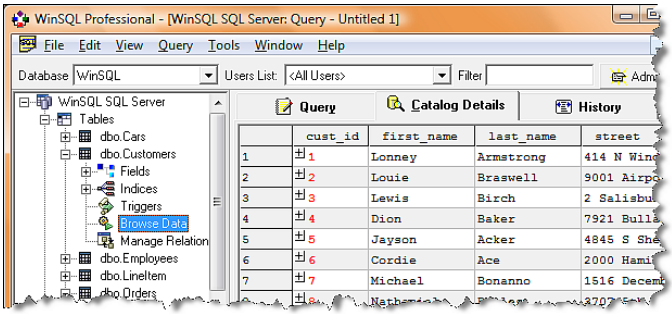
- **By user name** - displays objects owned by this user
- **By object name** - displays objects where the name starts with the specified value

Browsing Data

Click the **Browse Data** node to browse the first few records in a table or view. The default number of records displayed is 100; however, this value can be changed in the WinSQL Professional configuration options.

Tip:

Double-click the header of any row to sort the data by that row.



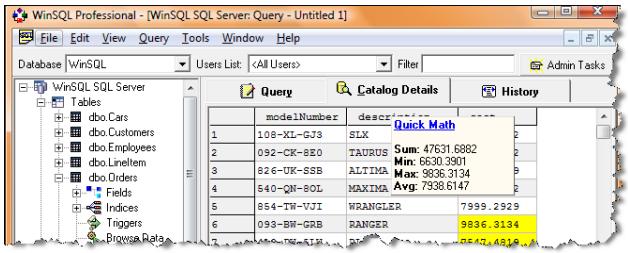
Using the Quick Math Feature

The Quick Math feature allows you to quickly perform basic math calculations on numeric fields displayed in a query result grid.

Perform the follow steps to use the Quick Math function:

1. Log into WinSQL and establish a database connection. For more information, see "Establishing a Database Connection."
2. Click the **Catalog Details** tab.
3. Click the plus sign by **Tables**.
4. Click the plus sign by the desired table.
5. Click **Browse Data**.
6. Select at least two rows from the result grid that represent numeric data.

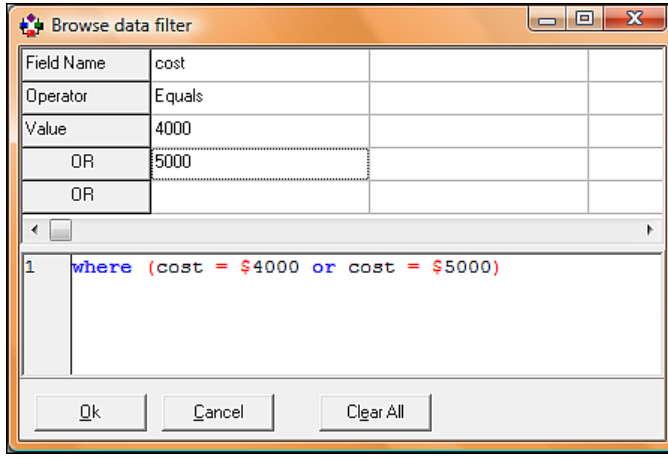
   WinSQL displays a **Quick Math** popup window displaying a sum, minimum, maximum and average values for the selected rows.



The **Quick Math** window displays for approximately eight seconds. Repeat step 6 to display the **Quick Math** window again.

Applying Filters

You can restrict the records that appear in the **Browse Data** window by right-clicking and selecting **Apply Filter** on the data grid. The following window is displayed:



Enter the desired options, and then click **OK**.

Adding User Comments

This feature provides a mechanism for entering comments about elements within a database. The comments can either be viewed only by the user who entered the comment (Local Comment), or by all users of the database (Shared Comment).

Comments can be entered for any of the following database elements:

- Databases
- Tables
- Fields
- Indices
- Views
- Stored Procedures

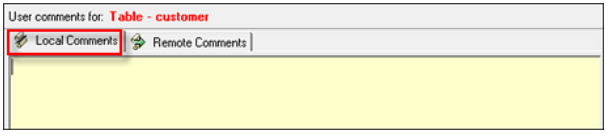
Creating Local and Shared Comments

A Local Comment is available for viewing/editing only by the user who entered the comment and is not shared by other users of the database. These comments are stored in a local file on the user's machine.

A Remote Comment is available for viewing/editing by other users of the database and is stored in a table within the database.

Creating a local comment

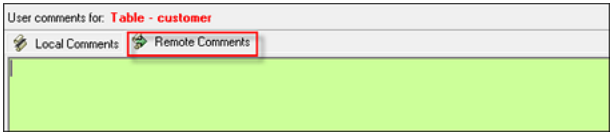
1. Log into WinSQL and establish a database connection. For more information, see "Establishing a Database Connection."
2. Open the element to which you wish to add a comment.
3. Ensure the **Local Comments** tab is selected:



4. Enter the comment in the text area.

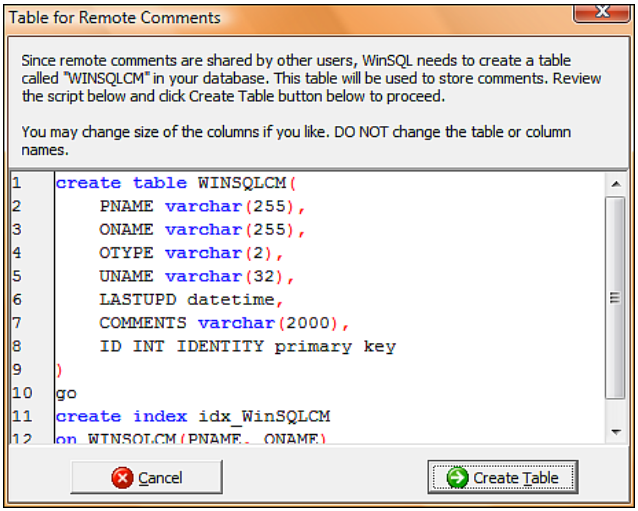
Creating a remote comment

1. Log into WinSQL and establish a database connection. For more information, see "Establishing a Database Connection."
2. Open the element to which you wish to add a comment.
3. Ensure the **Remote Comments** tab is selected:



4 Enter the comment in the text area.

5 If the **WINSQLCM** table does not yet exist, the following window is displayed, and you must click **Create Table** to continue:



If the **WINSQLCM** table exists, the comment is entered.

Important note:

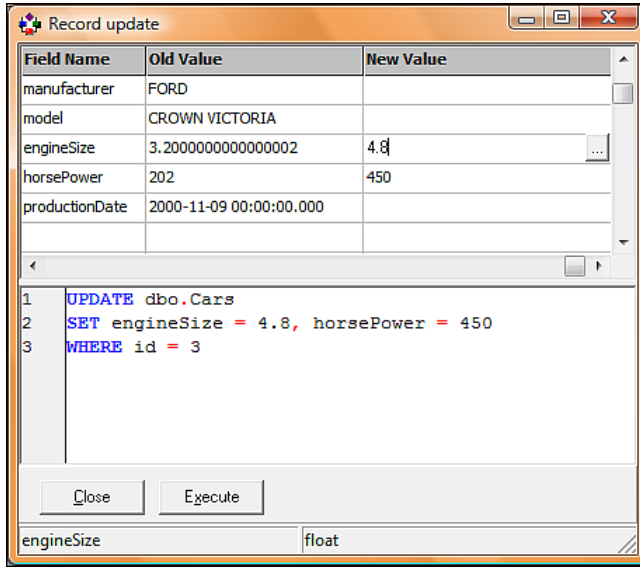
Do not change the table or column names in the **Table for Remote Comments** window. Doing so will prevent the remote comments feature from functioning correctly.

Using the Insert/Update Record Wizard

To invoke the Update wizard, double-click an existing row in the **Browse Data** window. To invoke the Insert wizard, double-click an empty row in the **Browse Data** window. You can also invoke the Update wizard by right-clicking an existing row, and then clicking **Update record**.

These wizards allow you to input the desired values in the grid and generate the appropriate SQL script.

The following window illustrates the record update feature:



Click **Execute** to perform the insert or update, or click **Close** to close the record without making any changes.

Using Strings and Dates

WinSQL Professional automatically encloses the values entered in the insert or update windows with the appropriate prefixes and suffix characters. Therefore, when typing the values in the grid, you **do not** enclose string values in quotes. Similarly, dates can be entered in the format that is understood by your system. This format is specified in the **Regional Settings** available in the Windows **Control Panel**.

Example:

Column name	Data type	Value entered
FirstName	VARCHAR	Sarah
LastName	VARCHAR	$$NULL
ID	INT	430
CustSince	DATETIME	4/25/1963 3:00 PM

Based on these values, WinSQL Professional generates the following query:

```
INSERT INTO customers(first_name, last_name, id,
custsince)
VALUES ('Sarah', NULL, '430',  DATETIME (1963-
04-25 15:00:00.) YEAR TO SECOND)
```

Notice that WinSQL Professional appropriately encloses the string value and formats the **DATETIME** field.

Note:

This example assumes that the database is Informix.

Ignoring Fields and NULL Values

Double-click any row to ignore it. When a row is ignore, text in all the columns get crossed out. In case of an INSERT, when a field is ignored, it will either put NULL or the default specified in the table.

Type **$$NULL** in the grid to explicitly set the value to null.

Adding BLOB and CLOB Data

This feature provides a mechanism adding images, audio, video, or other binary format files

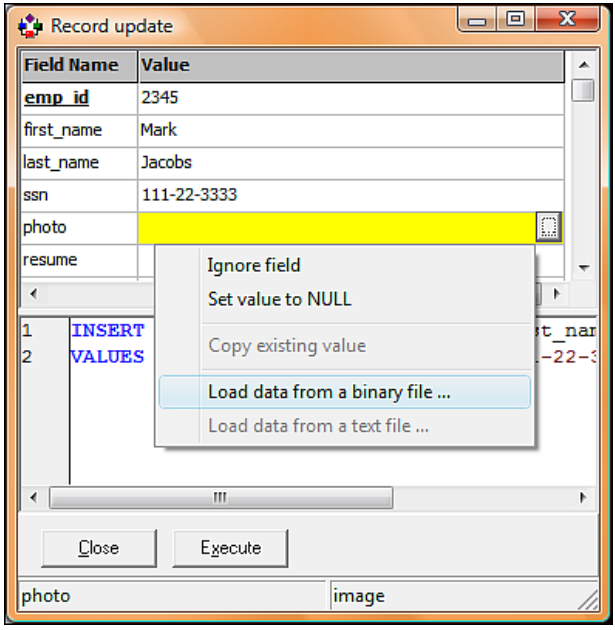
Perform the following steps to add BLOB/CLOB data:

1. Log into WinSQL and establish a database connection. For more information, see "Establishing a Database Connection."
2. Click the **Catalog Details** tab.
3. Click the plus sign by **Tables**.

4 Click the plus sign by the desired table.

5 Click **Browse Data**.

6 Right-click in the data window, and then click **Insert new record**. The following window is displayed:



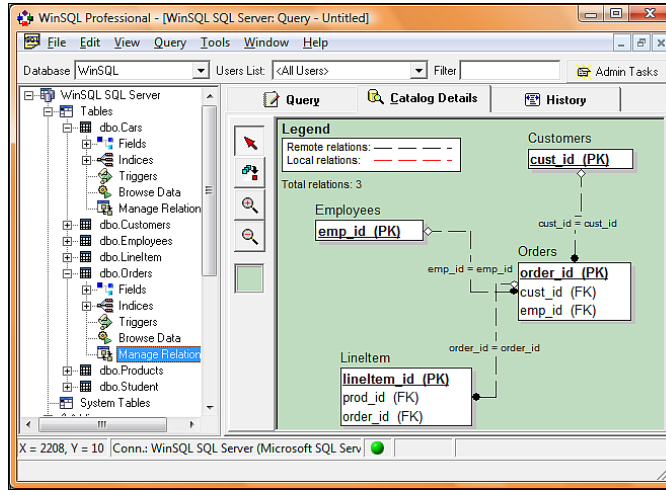
Note:

If any of the fields takes binary or long character data, click the icon in the cell and select either **Load data from binary file** or **Load data from text file**.

Managing Relationships

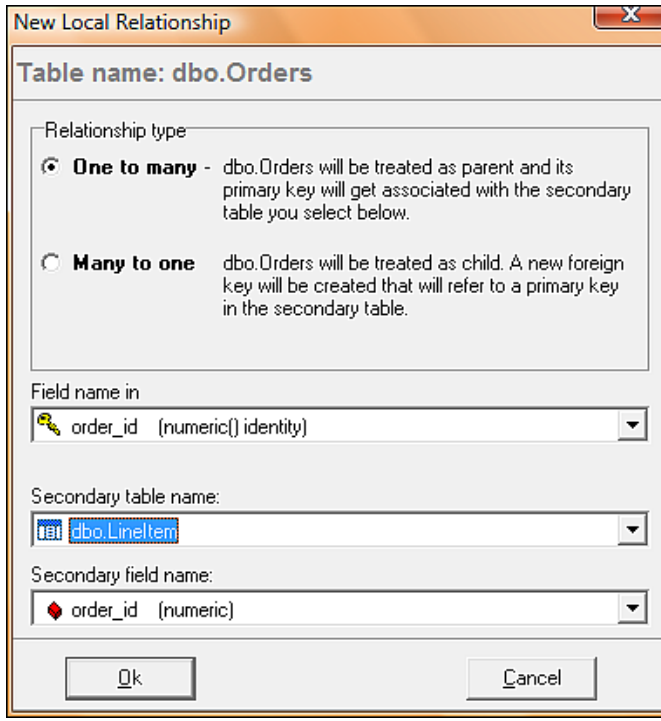
WinSQL Professional exploits the foreign key constraints defined between tables in the back-end database. Several features, such as Intelli-tips Tips and Drill Down results, depend on this information.

To manage relationships click on the **Manage Relationship** node in the **Catalog Details** window, depicted as follows:



Creating Local Relationships

To create a local relationship click on the **Add Relationship**, displayed when **Manage Relationship** is selected. The **New Local Relationship** window is displayed:



Use this window to define a new **One to many** or **Many to one** relationship between two tables. If you select One-to-many, it will create a new relationship where your selected table will become the parent. To create the selected table as a child, select **Many to one**.

If the parent table does not contain any primary keys, WinSQL Professional will create a primary key locally – meaning your back-end database will not be aware of this change.

If WinSQL Professional does not acknowledge a relationship, the reason could be one of the following:

- The database designer has purposely chosen not to define foreign keys. Since foreign key constraints affect the performance of the database at run time, database designers often choose not to use them
- Either the database or the ODBC driver does not support this functionality. In this case, you can define a local relationship, which is local to WinSQL

Professional and does not depend on or affect the database.

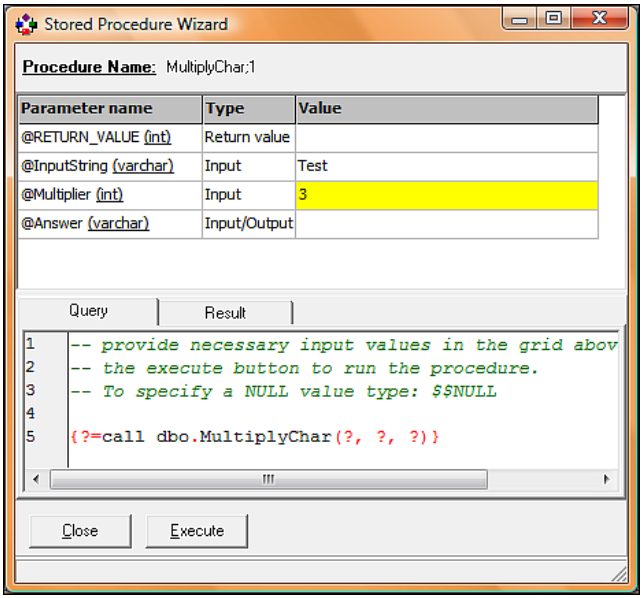
Deleting Relationships

To delete a local relationship, access the **Manage Relationship** window, right-click on a relationship line, and then click **Delete Relation**.

To delete a remote relationship, you must write an ALTER TABLE statement against your database.

Executing Stored Procedures

Using WinSQL Professional, you can execute and capture the results of a stored procedure using the Stored Procedure Wizard. Select the stored procedure from the **Catalog Details** window, right-click it, and then select **Execute Procedure**. A window similar to the following is displayed:



Complete the appropriate input fields in the grid, and then click **Execute**. After execution, the output parameters appear in the grid control in red.

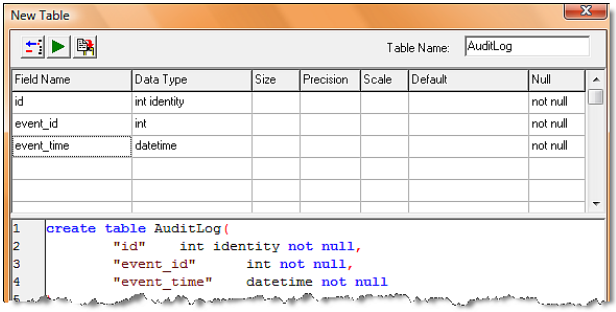
Using the New Table Wizard

The New Table wizard is used to create new tables. It performs such functions as allowing you to enter new fields, select field data types, and generate SQL statements. Once information is entered into all the fields, the SQL statement can be modified to include additional information, such as primary key, foreign key, or tablespace information.

Perform the following steps to access this wizard:

1. Click the **Catalog Details** tab.
2. Click **Admin Tasks**:

3. Click **Create New Table**. A window similar to the following is displayed:



4. Type a name for the table in the **Table Name** field.
5. In the table grid, type the field names and select the desired data type and other options for each field.

As you type field values, WinSQL generates the CREATE TABLE statement for your new table.

Reverse Engineering a Table Design

You can generate a CREATE TABLE statement based on the schema. Please note that this string is limited to the capability of the ODBC driver and may not include all the properties of the table. For example, ODBC does not provide the information about tablespaces and therefore, this information is not generated.

To generate a CREATE TABLE statement, right-click in the **Catalog Details** window, and then click **Generate CREATE TABLE Statement**.

Printing

The catalog information can be printed through the printing wizard in WinSQL Professional. By answering just a few questions, you can print the complete schema of a database in matter of minutes.

Database Search Wizard

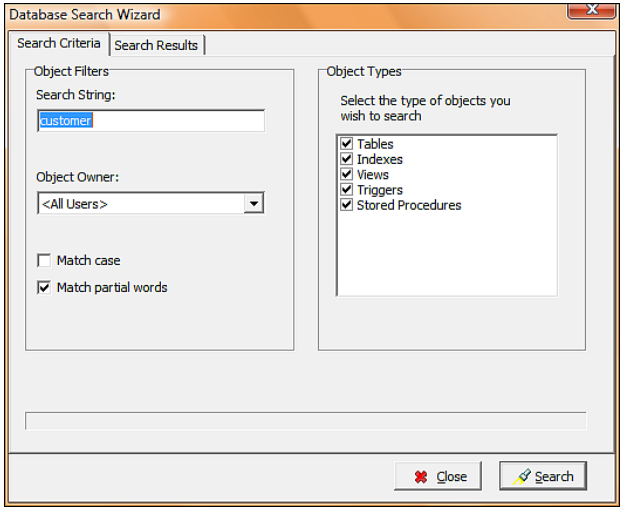
The Database Search Wizard is used to help you find instances of a string in a database.

The following topics describe how to use the Database Search Wizard.

Using the Database Search Wizard

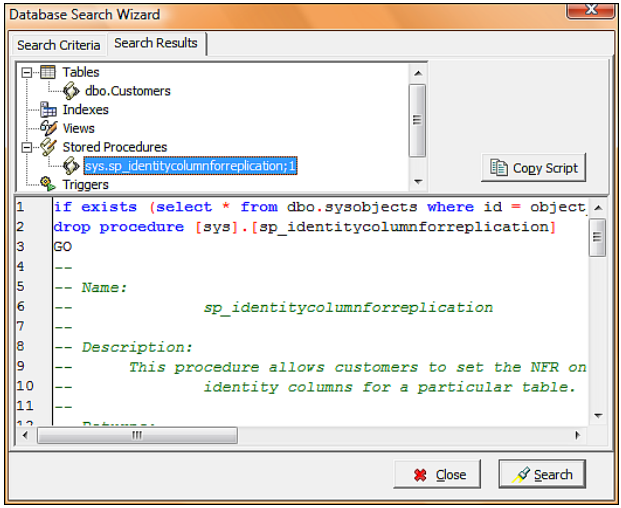
Perform the following steps to start the Database Search Wizard:

1. From the **Tools** menu, click **Database Search**. The following window is displayed:



2. Enter the search parameters and types of objects to search, and then click **Search**.

3. Click the **Search Results** tab to view the results of your search:



By default, the search is performed only on tables, columns, and their indexes. However, if you are connected to a database for which a plug-in is available, WinSQL Professional allows you to search views, stored procedures, triggers, and any other searching objects.

For more information about WinSQL Professional plug-ins, see "Chapter 12: Database-Specific Plug-Ins."

4 Click **Close** to close the **Database Search Wizard** window.

Chapter 6: Using the Database Task Scheduler

Overview

Introduction

This chapter provides information using the Task Scheduler to help automate routine database tasks.

In this chapter

This chapter contains the following topics:

About the Task Scheduler

The Task Scheduler in WinSQL Professional allows users to schedule tasks that can be run on a user-defined schedule. This Task Scheduler is tightly integrated with the Microsoft Windows operating system.

Types of Tasks

Following type of tasks can be scheduled:

- Running SQL scripts
- Exporting data from one source to another target
- Exporting data to HTML files
- Running Data Diff and Schema Diff wizards

Creating a Task to Run a SQL Script

Running SQL scripts based on a schedule is the most powerful and useful feature of the Task Scheduler.

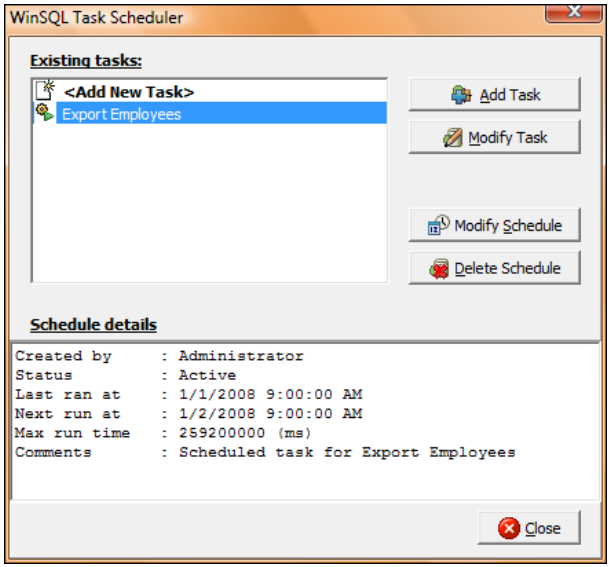
Perform the following steps to create a task to run an SQL script:

1 Start WinSQL Professional and connect to the desired database.

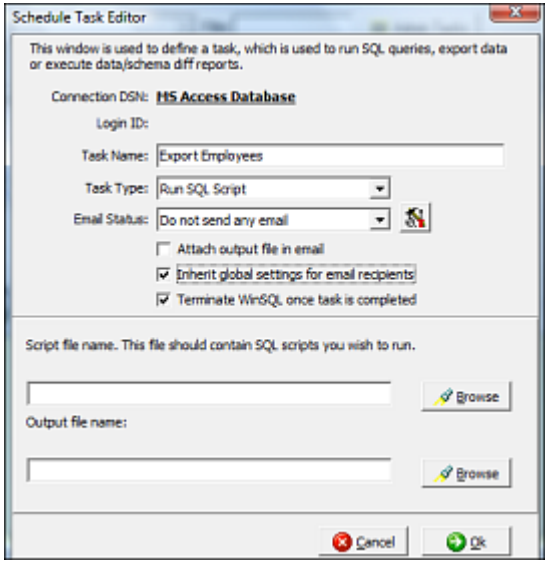
2 Create a query and save it with an .SQL extension.

For more information, see "Writing SQL Scripts."

3 From the **Tools** menu, click **Scheduler**. The following window is displayed:



4 Double-click **Add New Task** in the **Existing tasks** pane, OR click **Add Task**. The following window is displayed:



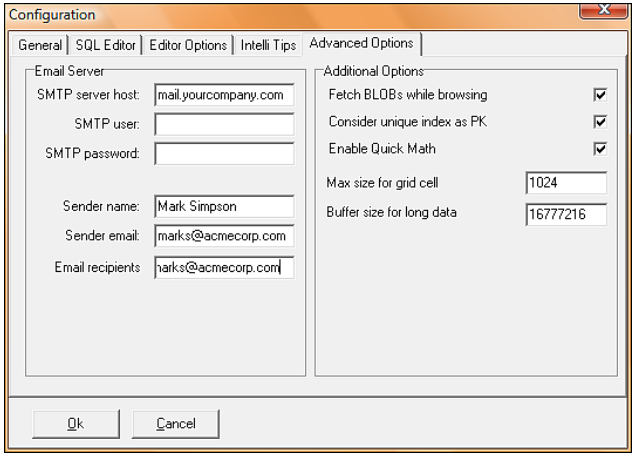
5 Use the following information to specify the desired parameters:

Field	Description
Task Name	Name for this task
Task Type	Type of task to run
Email Status	Select an option from the drop down menu for the type of e-mail status message you wish to receive. **Note:** If you elect to send a status email, you must configure the outbound e-mail parameters by clicking **Configure email server** ( ). For more information, see "Configuring the email server."

Field	Description
Attach output file in email	Click to select this option if you want to attach the output file in the status e-mail.
Inherit global settings for email recipients	Select this option to deliver a task status e-mail to all recipients that have been globally configured in WinSQL OR Clear this check box to send a task status e-mail to a select group of recipients. If this option is selected, a window displays that allows you to enter specific e-mail addresses.
Terminate WinSQL Professional once task is completed	Click to select if you want WinSQL Professional to terminate once the task has run.
Script file name	Name of the script file created and saved in step 2.
Output file name	The name of the file that contains the results (and any warnings or error messages) of the script.

Configuring the email server

1 Click the **Configure email serve**r () icon to the right of the **Email Status** field. The following window is displayed:



2 Complete the fields on the **Email Server** section of this window, and then click **OK** to close the window and return to the **Schedule Task Editor** window, where the newly-created task is displayed in the **Existing tasks** pane.

Note:

SMTP server host and **Email recipient** are **required** fields. All others are optional.

If you are unsure how to complete these fields, contact your information technology (IT) department.

3 Ensure the desired task is selected, and then click **Modify Schedule** to assign a schedule to the task.

Important note:

WinSQL Professional is tightly integrated with the Windows operating system for task scheduling. Therefore, your login ID *must* have sufficient rights to run the desired task.

4 Click the **Task**, **Schedule**, and **Settings** tab to assign the desired options to the task.

Notes:

- Consult the Windows operating system documentation or online help for further information about scheduling tasks.

- Confirm the system date and time are correct on your computer. Scheduled tasks rely on this information to execute tasks on a specified date and at a specified time.

Chapter 7: Publishing HTML Pages

Overview

Introduction

This chapter provides information about publishing SQL query results to HTML pages.

Note:

You must be using WinSQL Professional version 3.0 or higher to use this functionality.

In this chapter

This chapter contains the following topics:

Using Export Templates

Export template files can be created with WinSQL Professional. Once these files are created, they can be stored for future use. In addition, these files are needed if you wish to run export routines from a command line.

Note:

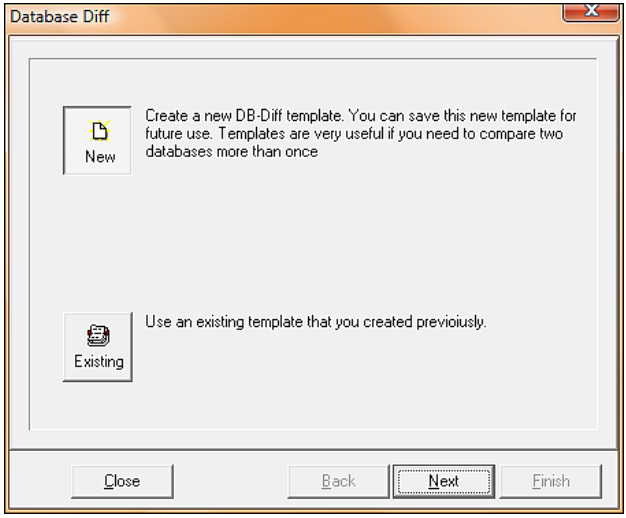
The HTML Publishing feature contains default values for all required fields. This allows the user to click **Next** until the **Summary** window is visible.

Creating a New HTML Template

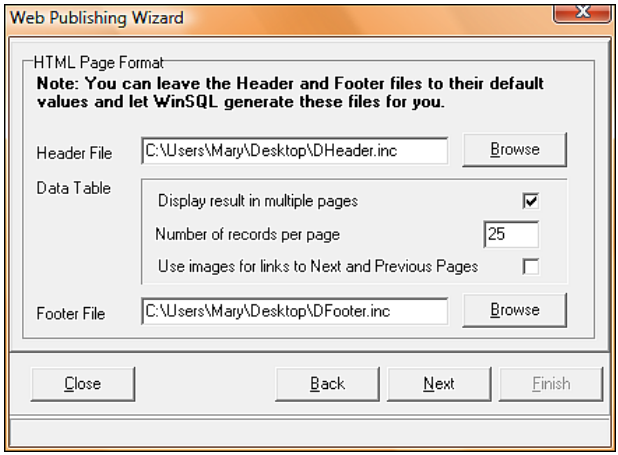
WinSQL Professional exports data based on an SQL query. Therefore, you must select a query in the **Query** window before clicking the export icon. You cannot use parameterized queries to publish HTML.

Perform the following steps to execute the Web Publishing Wizard to generate a new HTML template:

1 From the **Query** menu, click **Publish as HTML**. The following window is displayed:



2 Click **New**, and then click **Next**. The following window is displayed:



Header and Footer Files

WinSQL Professional uses two files, header and footer, to enclose the exported data. This allows you to format the output to suit your needs.

The following describes rules for the header file:

- The header file must have the beginning HTML tags including <html>, <title> <body> and others.
- The resultant data gets displayed into a table. Therefore, the header file must end with a <table> tag.

The following describes rules for the footer file:

- Footer file must have the ending tags of an HTML page.
- It must begin with a </table> tag.

The following is an example of a header file:

```
<html>
<head>
<title>SQL query result</title>
</head>
<body bgcolor="#FFFFFF">
<!—You can add more data here -->
```

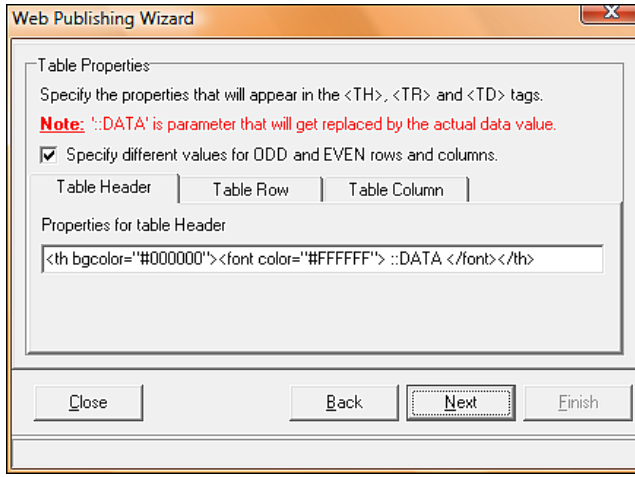
```
<div align="center"><center>
<table border="0" cellpadding="2">
```

The following is an example of a footer file:

```
</table></center></div></body></html>
```

Table header, rows and columns properties

After clicking **Next**, the following window is displayed:



This window allows you to specify the properties for the table header, table rows and table columns. The values in these fields must conform to HTML rules and **must** have a **::DATA** parameter. **::DATA** is replaced by the actual value when the export routine is run.

Example Table Header:

```
<th bgcolor="#000000"><font color="#FFFFFF">
::DATA </font></th>
```

Example Table Row:

```
<tr BGCOLOR="#FFFFCC"> ::DATA </tr>
```

Example Table Column:

```
<td> ::DATA </td>
```

Once all the necessary information has been entered, click the **Next** until the summary screen is displayed. Click **Finish** to publish the results.

Using an Existing HTML Template

Perform the following steps to use an existing HTML template:

1. From the **Query** menu, click **Publish as HTML**.
2. Click **New**, and then click **Existing**:

3. Click **Browse** to navigate to the location of the existing HTML export template (.HET file), and then click **Open**.
4. Click **Next** under the summary window is displayed, and then click **Finish** to publish the results.

Chapter 8: Using Database Diff

Overview

Introduction

This chapter provides information about using the database diff functionality to compare databases.

In this chapter

This chapter contains the following topics:

Database Diff Wizards

Beginning with WinSQL Professional version 4.5, two types of database diff wizards are available:

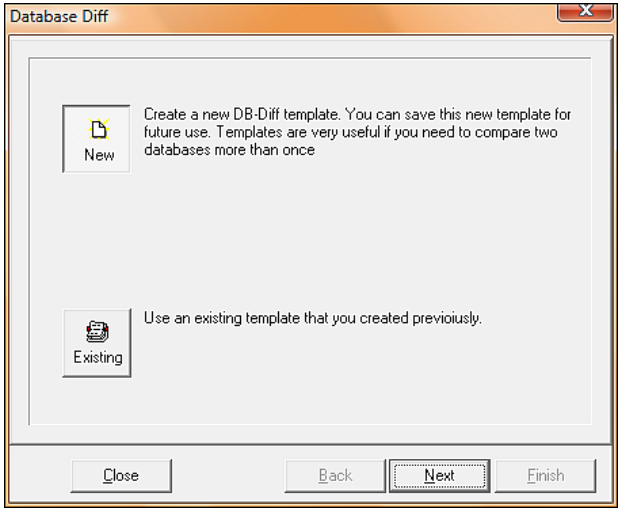
- Schema Diff
- Data Diff

Schema Diff Wizard

The Schema Diff feature provides you a mechanism to compare schemas of two similar databases. For instance, you can use WinSQL Professional to compare the schema of a database used in a development environment to a similar database in production environment.

Perform the following steps to invoke the Schema Diff wizard:

1 From the **Tools** menu, click **Database Diff**, and then click **Schema Diff**. The following window is displayed:



2 If this is the first time you are invoking the wizard, click **New**

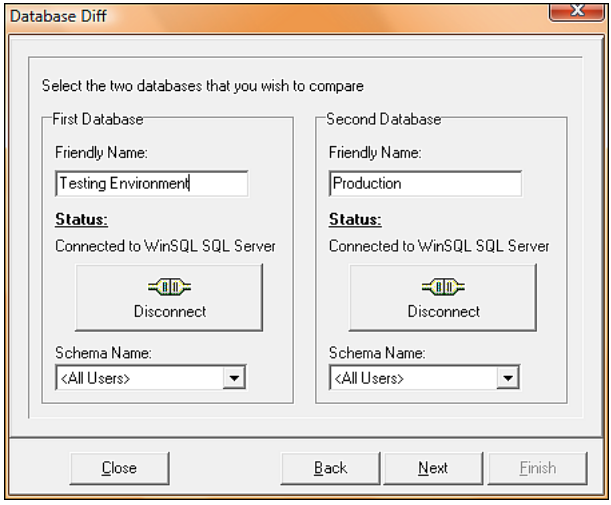
Or

If you wish to use a previously saved template, click **Existing**.

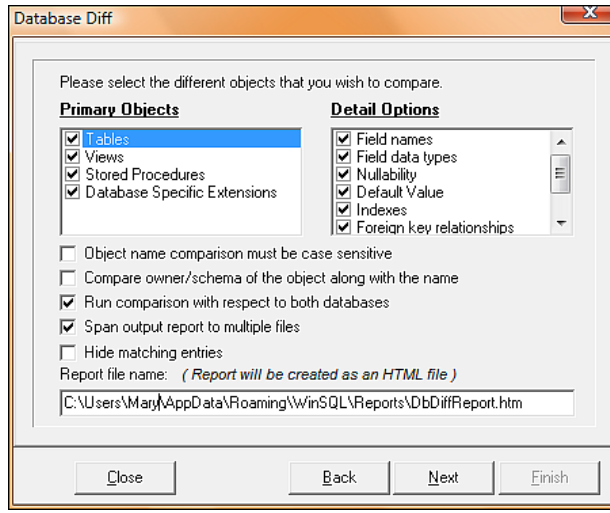
3 Click **Next**.

4 Select the first and second databases. Click the icon in the center of the screen to establish connection to both databases.

Once the connection is established, the **Connection** icon changes to **Disconnect**, depicted as follows:



5 Click **Next.** The following window is displayed:



This window is used to select the objects you wish to compare. The **Primary Objects** section of this window is used to display the primary objects contained within the database, such as:

- Tables
- Views
- Stored Procedures
- Database Specific Extensions

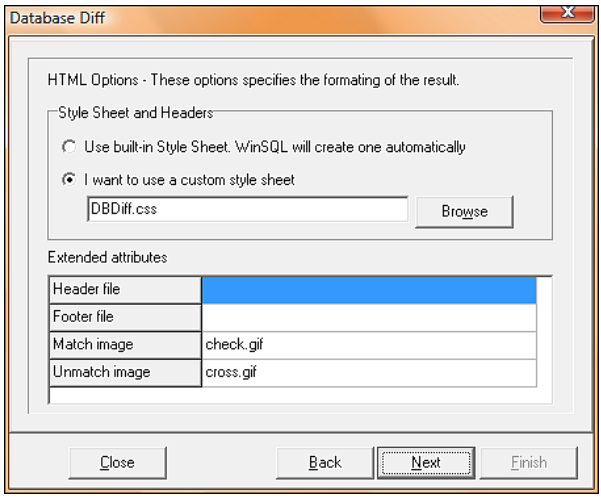
The **Detail Options** section of this window displays additional detail for the option selected in the **Primary Objects** section. The contents of this section changes based on the selection made in the **Primary Objects** section. For example, if you select "Stored Procedures" in the **Primary Objects** section, the **Detail Options** section will contain items to verify if you want to match the procedure script.

The following additional options available on this window, and a description of each, is included in the following table:

Option	Description
Object name comparison	If selected, comparisons

Option	Description
must be case sensitive	are case sensitive.
Compare owner/schema of the object along with the name	If selected, the object owner is also matched.
Run comparison with respect to both databases	If selected, a comparison is run twice – once for each database. When the comparison is run, WinSQL considers one database as the reference and finds out which objects are missing or different in the other database.
Span output report to multiple files	If selected, multiple HTML files are created for output reports. It is highly recommended that you leave this option selected, particularly for large databases.
Hide matching entries	If selected, matched objects do not appear in the report. Selecting this option greatly reduces the report size.

6 Once all options are selected, click **Next**. The following window is displayed:



Reports are generated as HTML files, and they are displayed using the HTML browser configured on your system.

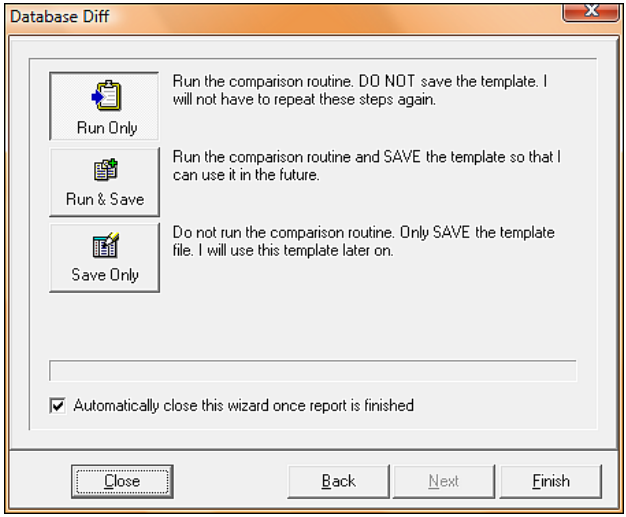
The following is a description of the options available for generating reports:

Use built-in style sheet – If this option is selected, the generated report will use a built-in style sheet for HTML rendering.

Header/Footer file – If specified, WinSQL Professional uses the text contained in the specified file as the header and/or footer for the report. This is a convenient way to customize the look and feel of the generated report

Match/Unmatch image – This is the URL for the images displayed in the report.

7 Once all options are selected, click **Next**. The following window is displayed:



8 Select the appropriate option, and then click **Finish.**

Comparing Objects Using Schema Diff

When comparing two databases, WinSQL Professional picks the first database and treats it as a pivot database, meaning it will examine all the objects in this database and will try to match it will the non-pivot database.

If **Run comparison with respect to both databases** is selected, WinSQL Professional will run the comparison twice – once for each database as the pivot database.

Primary Objects and Detail Options

The content of this list contains all the objects that can be compared. On the right side are the individual options for the object selected on the left. For example when you select **Stored Procedures** on the left, you see **Parameters**, **Parameter type**, **Procedure script**, and "**Show DDL if doesn't match**" on the right.

Important note:

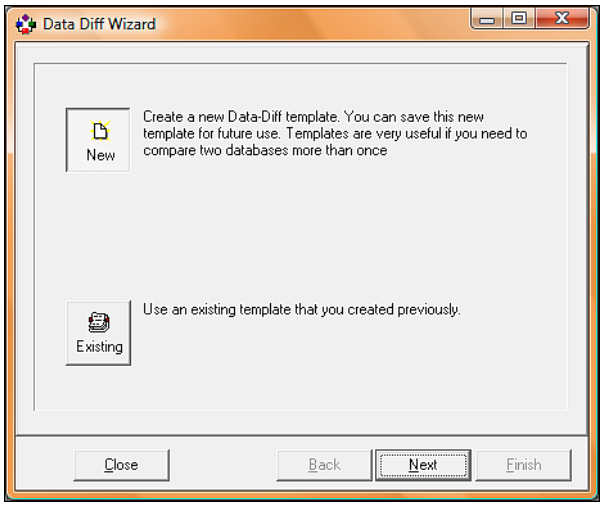
You must select an appropriate database plug-in at the time of connection in order for the Database Diff to work correctly.

Data Diff Wizard

The Data Diff wizard compares data between two tables.

Perform the following steps to invoke the Data Diff wizard:

1. From the **Tools** menu, click **Database Diff**, and then click **Data Diff**. The following window is displayed:



2. If this is the first time you are invoking the wizard, click **New**.

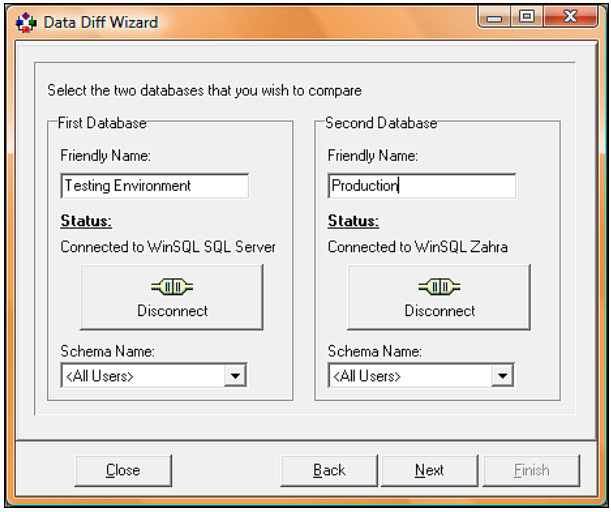
   Or

   If you wish to use a previously saved template, click **Existing**.

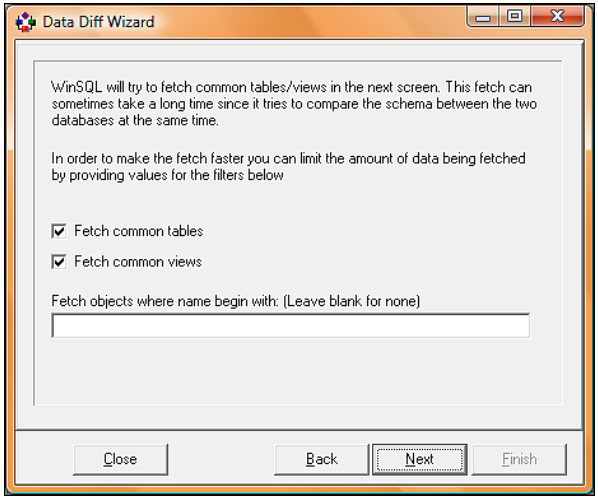
3. Click **Next**.

4. Select the first and second databases by clicking the icons in the center of the screen.

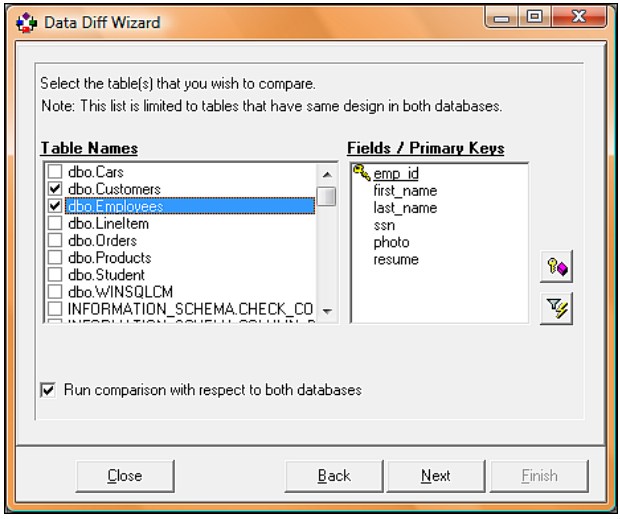
   Once the connection is established, the **Connection** icon changes to **Disconnect**, depicted as follows:



5 Click **Next**. The following window is displayed:



6 If you do not wish to specify any filter, leave the fields blank and every table will be selected. Click **Next**. The following window is displayed:



WinSQL Professional displays tables and views on the left, while fields for the selected tables are displayed on the right. If a field has a primary key defined, WinSQL Professional will automatically put a key next to the name.

In certain cases, such as when you select a particular view or your ODBC driver does not support primary keys, WinSQL Professional will not be able to automatically detect a primary key. In this case you must manually create one.

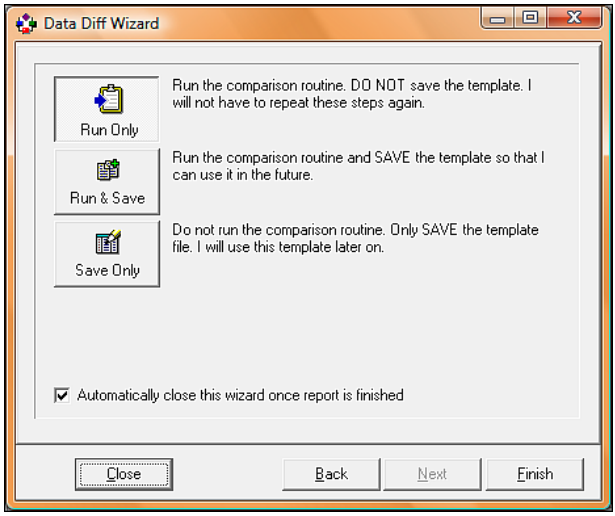
To create a primary key you need to:

- Select the appropriate field
- Click the **Toggle Primary Key** icon  .

Notes:

- Creating primary keys in this window does not have any affect on the underlying design of the table.
- If you wish to exclude any field from comparison select the field and either double-click it or click the **Toggle field selection** icon  .

7 Once all options are selected, click **Next**. The following window is displayed:



8 Select the appropriate option, and then click **Finish.**

Comparing Objects Using Data Diff

In order to compare data between two tables every table must meet the following requirements:

- The design of the two tables MUST be similar. You cannot compare tables if their designs do not match.
- There MUST be at least one field as primary key in both tables. If a primary key is not defined, you can specify one using the wizard.

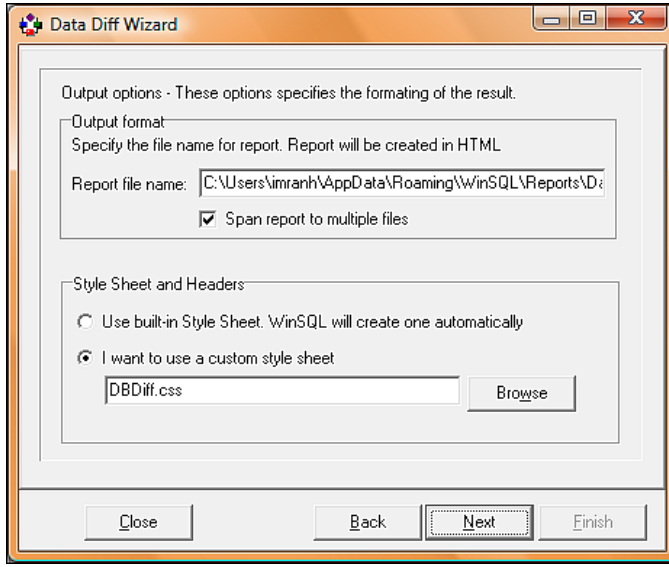
When comparing records, WinSQL Professional selects all records from the first table,, and then issues a SELECT query with appropriate WHERE clause in the second table. If a record is found, it then checks to see if the data in every field is identical in both tables.

Since WinSQL Professional compares the design of every table in both databases while it is fetching, it can take a long time to fetch every object in the database. Therefore, it is recommended that you specify a filter if there are a large number of tables/views in your database.

Running Database Comparisons

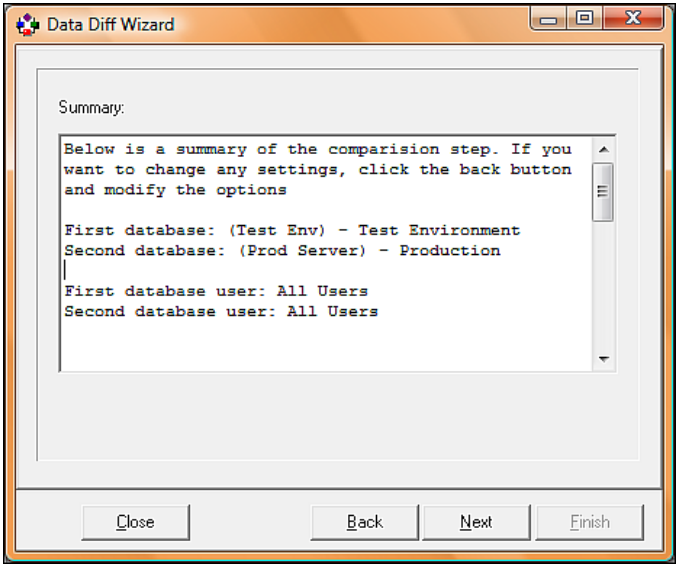
During comparison, WinSQL Professional select all records from a table in the first database and compares them with records in the corresponding table in the second database. This comparison only detects rows that match, partially match, or do not exist in the second table. It does not detect rows that are present in the second table but not in first. Therefore, it is important to switch the pivot database once the first comparison is completed.

Select **Run comparison with respect to both databases** if you want WinSQL Professional to switch the pivot table after the first comparison has completed.



Next, type the location where you want the output generated in **Report file name** field.

Finally, read the summary and confirm all settings are correct and then click **Next**.



As with any other Wizard in WinSQL Professional, you can either save the template, or run the wizard without saving it.

Chapter 9: Generating Test Data

Overview

Introduction

This chapter provides information about creating a test environment by generating test data.

In this chapter

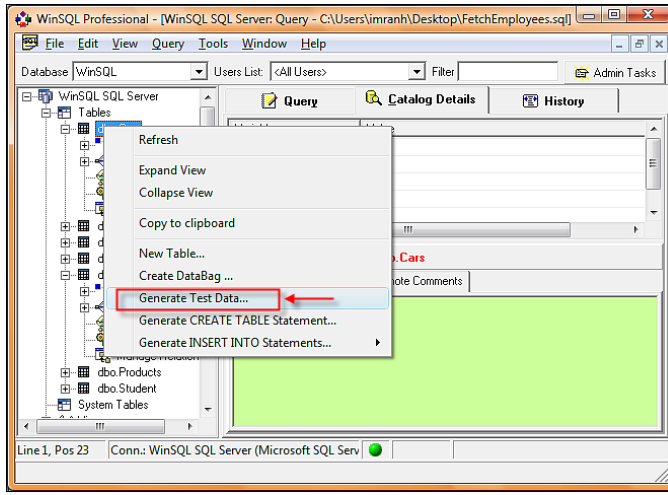
This chapter contains the following topics:

Creating a Test Environment

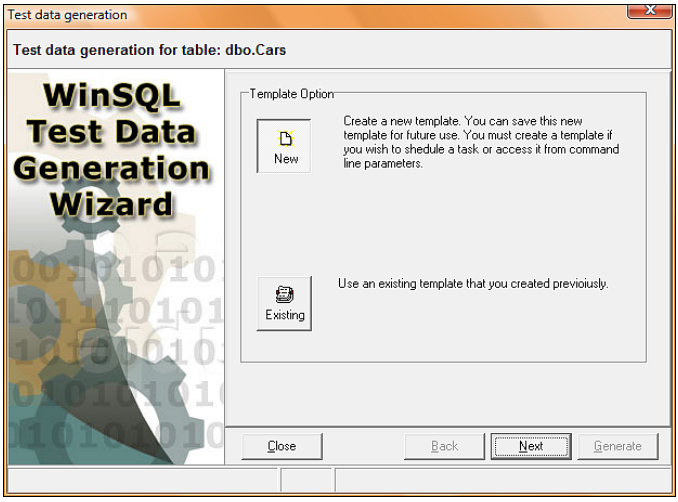
Beginning with WinSQL Professional version 3.5, test environments can be created to perform testing prior to moving to a production environment. This feature can be very useful when random data is required during the development and testing phase of a project. The Test Data Generation wizard allows you to create millions of rows of meaningful data within a database.

Starting the Test Data Generation Wizard

Before invoking the Test Data Generation wizard, you must decide which table to use to generate the data. After this determination has been made, right-click the desired table in the **Catalog Details** window and then click **Generate test data**.



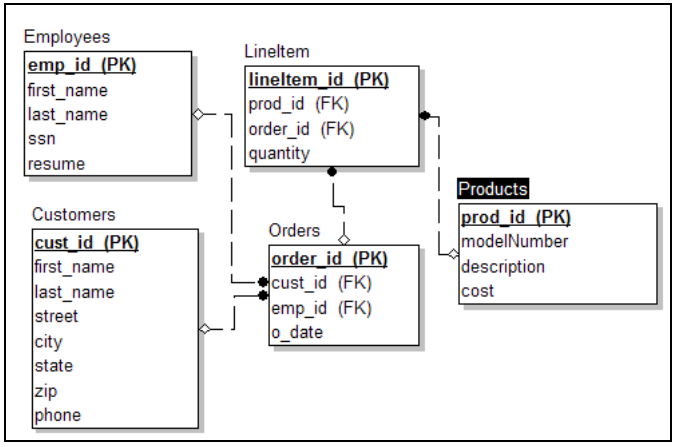
The following window is displayed:



Example scenario

In order to understand all the steps required to create test data, the following example is provided, including the steps used during the process.

The example scenario uses the following E-R diagram of a database table:



In this example, Microsoft SQL Server 7.0 is used, and all the data types reflected are for this type of database.

Deciding which tables to populate first

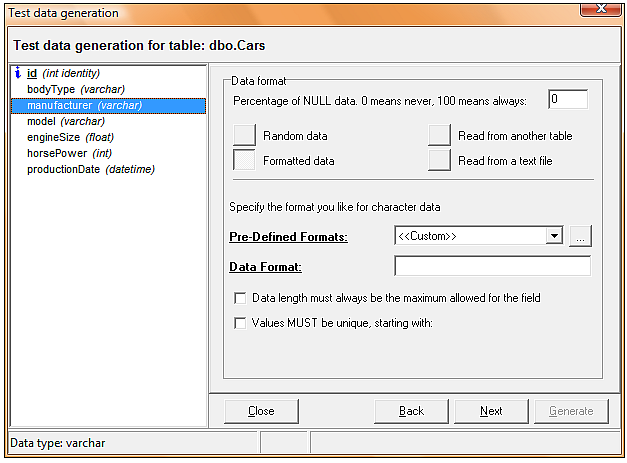
When you have multiple tables in your database, it is important that you populate data for the primary tables first. Primary tables are tables that do not have any foreign key constraints defined. Therefore, data can be generated without depending on other entities. In our example, the Customers, Employees and Products tables fall into this category.

The following steps describe the test data generation process:

1. From the **Catalog Details** window, select the **Customers** table.
2. Right-click this table and then click **Generate test data**.
3. Click **New** to create a new template.

   Note:

   Default values are supplied for all fields. If you click **Next** until completion, WinSQL Professional generates about 1000 table rows.
4. Click **Next**. The following window is displayed:



The options on this window allow you to select fields and the data formats for the fields you wish to generate. Field names are displayed on the left side of the window.

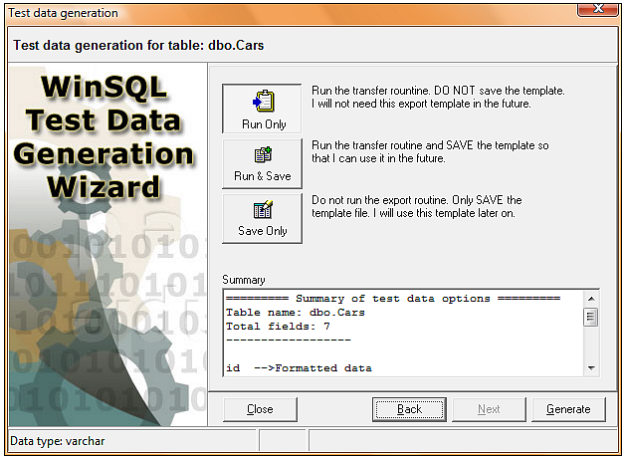
Four data sources can be used:

Data source	Description
Random data	This option generates completely random data. Although the text will not make any sense, this is the fastest method to generate data. This is the default value for all the fields except if there is a unique index on the field, in which case unique numbers will be generated.
Formatted data	This option allows you to select the format of the data. Depending on the data type of the field, available options can change. For example, if the data type is DATETIME, you can specify the starting and ending date. Additionally, you can specify if you wish to include time along with date. For detailed information about how to use formatted data, see the "Specifying Formatted Data" topic.
Read from another table	This option is useful if you have a foreign key constraint defined that references another table. WinSQL Professional queries the ODBC driver to determine if any foreign key constraints are defined. If found, the program automatically selects this option with appropriate values for the fields. This feature depends heavily on the capabilities of the ODBC driver. For example, the ODBC driver for Microsoft Access does not return this information to the application, and users manually have to specify these values.

Data source	Description
Read from a text file	Use this option to read data from a text file. This text file MUST have one row per line and can have multiple columns separated by a | (pipe) character. You can either sequentially read the file or assign records randomly. In the case of a sequential read, there MUST be enough records in the input file. **Tokens** Data in one text file can be a source for multiple columns. Consider the following scenario: You want to insert city, state & zip information in a table. If these three fields come from three different text files, there would be no relation between city, state and zip. This can cause New York City to become part of California with a zip code of Orlando, Florida. To avoid this situation, WinSQL Professional can read multiple fields from a text file. These fields are separated by a | (pipe) character and are called **tokens**. A sample text file can look like: New York City|New York|10001 Woodbridge|Virginia|22192 Palo Alto|California|94301

5 After your selections have been made, click **Next**.

6 Based on your data source selection, different additional options as presented to you. Make sure selections, and then click **Next**. A summary window similar to the following is displayed:



7 Select one of the following options:

- Run Only
- Run & Save
- Save Only

8 Click **Generate**. WinSQL begins to generate records in your back-end database.

Specifying Formatted Data

Different format characters are used to specify formatted data. The information in this section defines these formats.

Note:

The information in this section applies only if you selected Formatted Data as your data source in the previous section.

String Formats

The following table specifies the rules for specifying format for strings:

Keyword	Description
/	Escape character
#	Numeric values
$	Uppercase characters only
^	Lowercase characters only
@	Alphanumeric with uppercase letters
%	Alphanumeric with lowercase letters
~	Alphanumeric with mixed case letters
*	This character MUST appear at the end, and it means that one or more characters can appear on the right side of the generated string

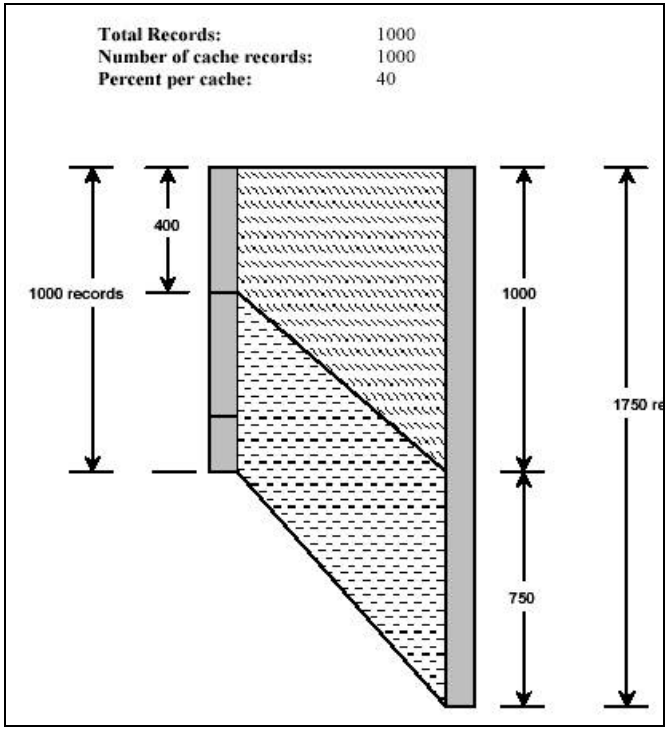
Cached records

When records are read from either a table or text file, WinSQL Professional reads a certain amount of rows in the memory. These records are called cached records. The default number of records in cache is 1000. Once enough records are generated, the program discards all the rows and refreshes the cache.

Percentage per cache

This is the percentage of total records that should be assigned from a batch of cached records.

The following figure depicts the concept graphically:



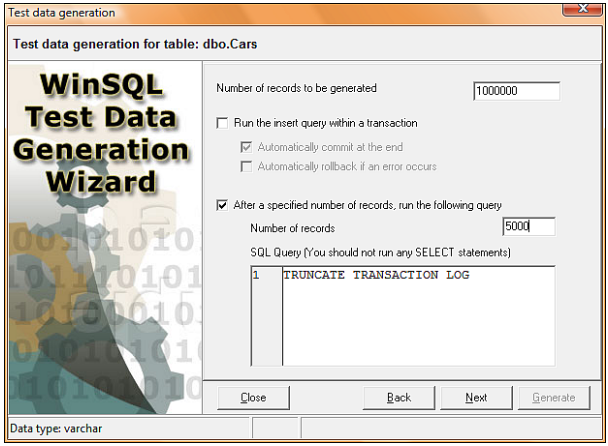
In the above example, we have to insert 1000 records in the target table, and there are 1750 records in the input file/table. The specified percent cache is 40%, which means that WinSQL Professional will read 1000 rows from the table or text file and randomly pick records 400 values. Then, WinSQL Professional discards the cache data and fills it again with new values. However, the second time this operation is performed, there are only 750 records left in the input source. Therefore, the remaining 60% of the data will be created using the 750 records that are in the cache.

Notice that the first field in the example, **`cust_id`**, is an AGN (automatically generated number), for which the database is responsible for assigning values. Synametrics recommends that you not change the value for this field.

For the purposes of this example, a sample data file (SDF) is used as the data source. A few sample SDF files come with

WinSQL Professional and are located in the folder where WinSQL Professional is installed.

1. Click the **first_name** field.
2. On the right side of the window, click **Read from a text file**.
3. Click **Browse** to select **FirstName.sdf**.
4. Repeat steps 5-7 for the **last_name** field, selecting **LastName.sdf** as the file name.
5. Click the **city** field.
6. On the right side of the window, click **Read from a text file**.
7. Click **Browse** to select **citiState.sdf**.
8. Repeat steps 9-11 for the **state** and **zip** fields, using 2 and 3 for token numbers. Select the **citiState.sdf** file for the **state** and **zip** fields. This file contains fields for 3 columns.
9. Click the **phone** field.
10. On the right side of the window, click **Formatted data**.
11. Select **US Telephone Numbers** from the list that is displayed. If additional formatting options are desired, select them from the list that is displayed.
12. Click **Next**. The following window is displayed:



13 In the **Number of records to be generated** field, enter the number of records you wish to generate. There is no record number limit.

Note:

If the input source is a text file for any field and includes a sequential flag, you **must** have enough records in the SDF, or WinSQL Professional will restart from the beginning of the last batch and will create duplicate records.

Running insert query within a transaction

Data can be generated within a transaction. This allows you to ROLLBACK in case of any errors. By default, this option is disabled. Before enabling this option, you must refer to your database documentation regarding transaction support. Certain databases, such as Informix, allow you to temporarily disable transactions, in which case you will not be able use this feature.

After a specified number of records, run the following query

This feature allows you to run a query after a certain number of records are generated. This could be useful if you need to backup the transaction log so that more records can be inserted. Leave this field blank if you do not wish to run any query.

Repeat steps 1-17 for the employee and products tables.

Populating secondary tables

Once all the primary tables are populated, data to satisfy foreign keys relationships in the secondary tables is now present. Follow the same steps to generate data for these tables, noting that you can now select a table as an input source. If your ODBC driver supports foreign key functionality, WinSQL Professional automatically selects the appropriate input source that will point to a primary key in another table.

In our example, **lineItem** is the last table in the dependence-hierarchy since it depends on the orders table. Therefore, the next candidate for data generation should be the orders table.

Ignoring a field

If you need WinSQL Professional to ignore a field for any particular reason, simply double-click the column name in the list box. This will cross out the field name and display it in gray text.

Although ignoring a field is almost like setting the NULL percentage to 100, there can be situations where they can differ. Consider the following example:

You have a table that is defined as:

```
create table myTable(
id     autoNumber primary key,
 name  varchar(30),
 phone varchar(15)
)
```

In the case where the name field is ignored, WinSQL Professional will generate the following code:

```
insert into myTable(phone)
values("555-1212")
```

However, if the field is set to produce 100% NULL values, the generated SQL will look like:

```
insert into myTable(name, phone)
values(NULL, "555-1212")
```

In the above example, it does not matter if the name field is ignored or set to NULL. The output is the same in either case. However, if the table definition looked like following, there would be a difference:

```
create table myTable(
                    id    autoNumber  primary  key,
                          name  varchar(30),
                          phone varchar(15)
                    default             '555-1212'
```

Chapter 10: Entity-Relationship Diagrams

Overview

Introduction

This chapter provides information about creating entity relationship diagrams.

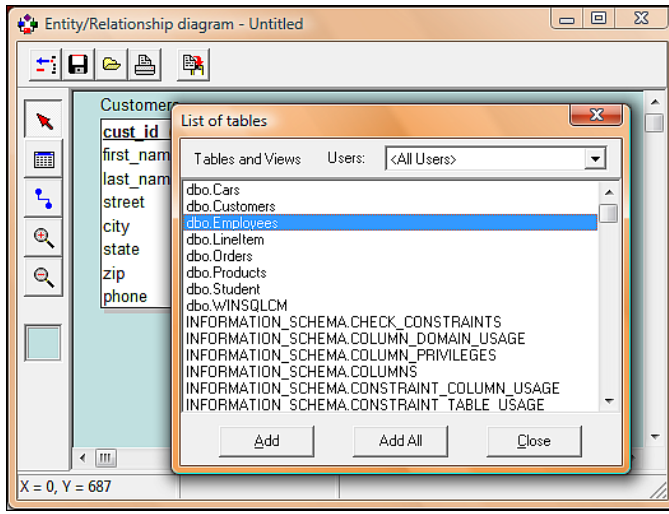
In this chapter

This chapter contains the following topic:

Creating an E/R Diagram

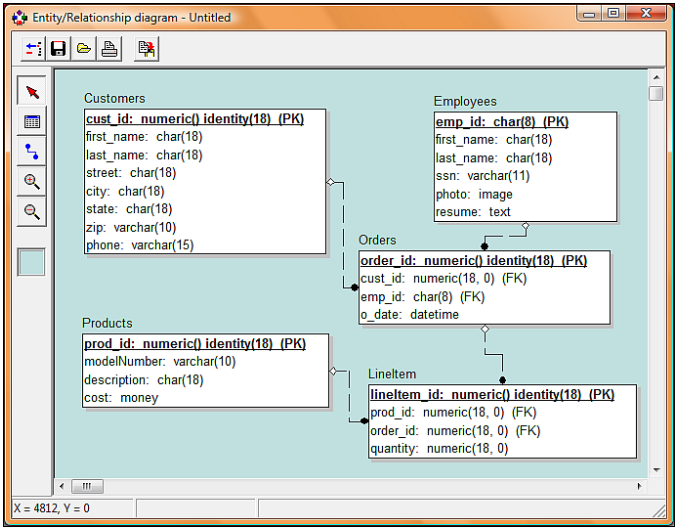
WinSQL Professional allows you to create an E/R Diagram based on an existing database. You can selectively pick the tables that are logically grouped with each other or pick all tables in the entire database to draw the diagram.

From the **Tools** menu, click **Draw E/R Diagram**. The following window is displayed which allows you to select table(s) that you wish to include in this diagram.



You can add a table either by double-clicking a table name or selecting it and then clicking **Add**. Click **Add All** to include every table in your database.

The following screen shows the E/R diagram after all desired tables are included:



The following table describes the actions available in the **Entity/Relationship diagram** window:

Icon	Description
	**Add New Table** - Click to display a list of Tables and Views in your database. In the list that is displayed, select and double-click a table to add it in the diagram.
	**Add New Relation** – Click to create a Master-Detail relationship between two tables. After you click this icon, the cursor changes to a hand. You MUST select the master table first and then the detail table.
	**Zoom In** - Click to enlarge the size of the image on the screen, as well as the printed output.
	**Zoom Out** - Click to reduce the size of the image on the screen, as well as the printed output.

Saving the Diagram

Click **Save** to save the diagram to disk. The default extension for the file is .WER which stands for WinSQL Professional E/R

Diagram. Similarly, use the **Open file** icon to read a previously-created diagram.

Chapter 11: Exporting and Importing Data

Overview

Introduction

This chapter provides information about exporting data to different data sources.

In this chapter

This chapter contains the following topics:

Exporting Data

WinSQL offers several ways to move data from once source to another. For example, you can:

- Copy contents of one table to another table in a different database
- Export data from table to a text file
- Import data from a text file to a database
- Generate INSERT INTO statements for an existing data. These INSERT statements can later be run against an empty table to insert data rows.

This section talks about different ways of importing and exporting data from one source to another target in WinSQL.

Export templates

Beginning with WinSQL Professional version 2.0, you can create export templates files to use at a later date. Additionally, these files are needed if you wish to run the export routines from a command line.

Exporting Data to Another Data Source

Three types of export options exist for exporting data to another data source:

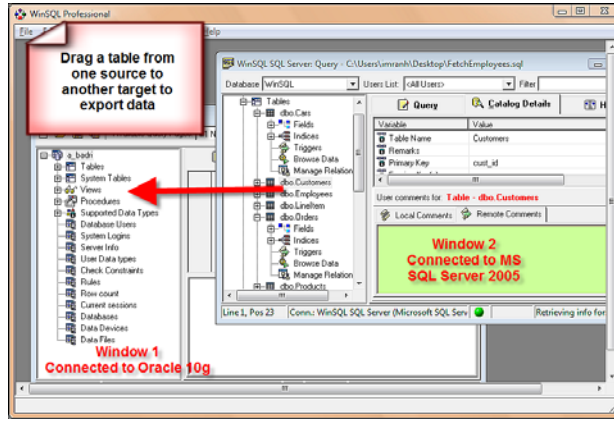
- **Connected**: users have access to both source and target databases at the time of export. Records are inserted in the target database as they are read from the source. No data is stored locally.
- **Disconnected**: the source and target databases are not connected at the time of export. This feature uses “DataBags” to store both the data and database schema in a file which can then be imported into another database. The DataBag functionality is superior to the typical export to a text file in that data size is not limited and binary and unprintable characters (such as tabs) correctly interpreted and exported.
- Text file: data is exported from the source database to a text file and visa-versa

Connected

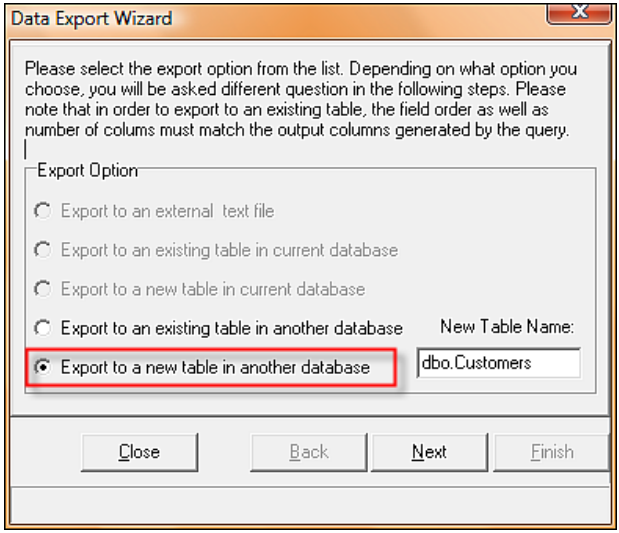
Data can be exported from one table in the source database to a target table or from an SQL query against the source database to a target table.

The section below describes how to use Drag-N-Drop to move contents of table from a source to a target:

1. Establish a connection between two databases. For more information, see "Establishing a Database Connection."
2. Click the **Catalog Details** tab.
3. Click the plus sign by **Tables** in the source database.
4. Click to select the desired table in the source database.
5. While holding the left mouse button, drag the table to the target database. This action is illustrated as follows:



6. When the mouse button is released after dragging the table into the target database, the **Data Export Wizard** window is displayed:



The **Export to a new table in another database** option is selected.

By default, the table name is the same as in the source database but can be modified.

Note:

If the table exists in the target database, WinSQL automatically selects the **Export to an existing table in another database**.

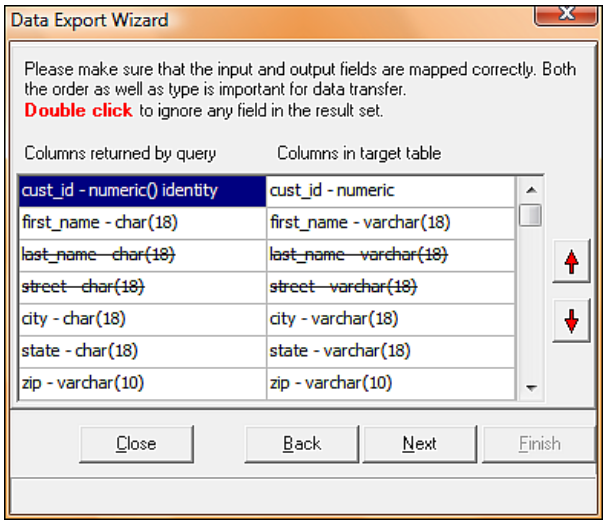
7 Click **Next**. The following window is displayed:

8 Ensure the displayed create table statement is correct, and then click **Next**.

Note:

WinSQL will convert, as necessary, data types between the two databases. If a particular data type is not supported in the target database, “not supported” is displayed in this window but can be modified to specify a data type that most closely matches.

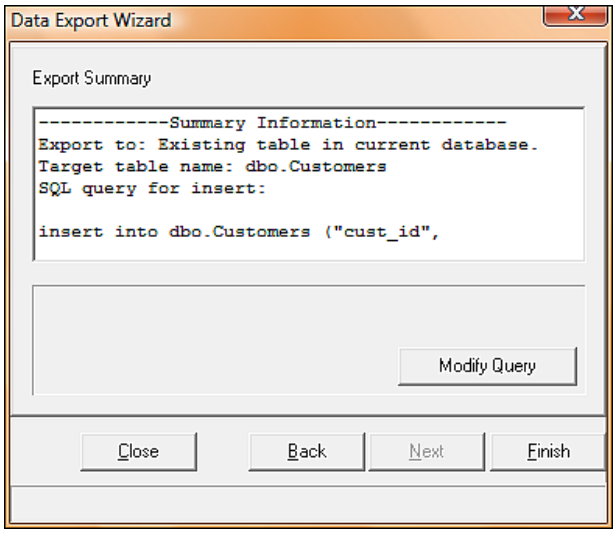
The following window is displayed:



Note:

In the previous illustration, the rows that have lines through them indicate that the columns be ignored. In this example, the user does not wish to export data for the “last_name” and “street” columns.

9 Ensure the data mapping between the two databases is correct, and then click **Next**. The following window is displayed:

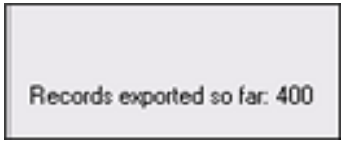


10 Ensure the export summary information is correct, and then click **Finish**.

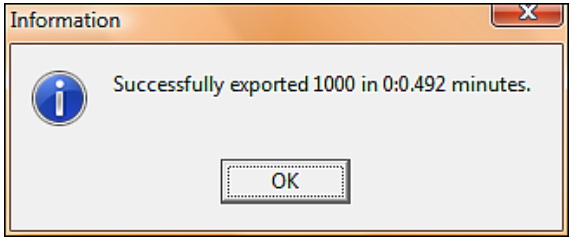
Note:

If the export summary information is incorrect, click **Modify Query** to make the necessary changes.

The data export begins, and status information is displayed during the process:



When the export has completed, the following window is displayed:



11 Click **OK** to continue. The data export is completed.

Disconnected

The disconnected export option uses DataBags to complete the export. DataBags contain a local representation of schema and data from a table in a relational database. Using DataBags, WinSQL can recreate a table along with its data. Therefore, DataBags can be used for either database backup or to move data from one source to another in a disconnected mode.

There are two ways to create DataBags: using table drag-and-drop, or using WinSQL menu options. These methods are described in the following sections.

Once the DataBag is created, perform the steps described in this section to import the contents of the DataBag into the target database.

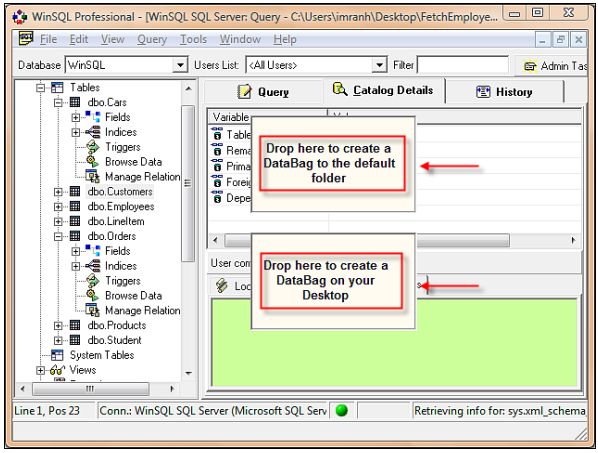
Disconnected Export: Drag-and-Drop

1 Establish a connection with the source database. For more information, see "Establishing a Database Connection."

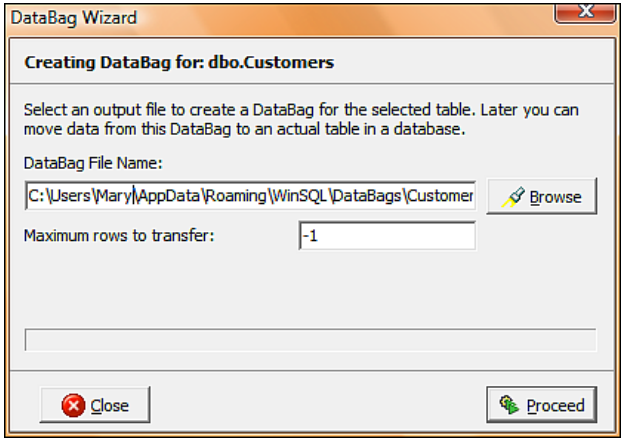
2 Click the **Catalog Details** tab.

3 Click the plus sign by **Tables** in the source database.

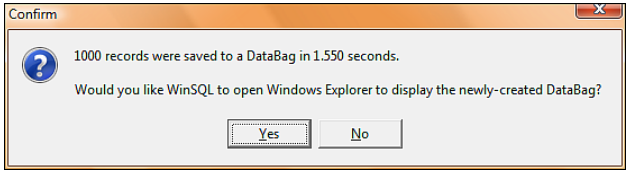
4 Select the desired table, and then drag it to the right side of the window. When the table is dragged to the right side of the window, the following options are displayed:



5 Continue dragging the table to the desired option. If **Drop here to create a DataBag to the default folder** is selected, the following window is displayed:



6 Accept the default folder location by clicking **Proceed**. The following window is displayed:



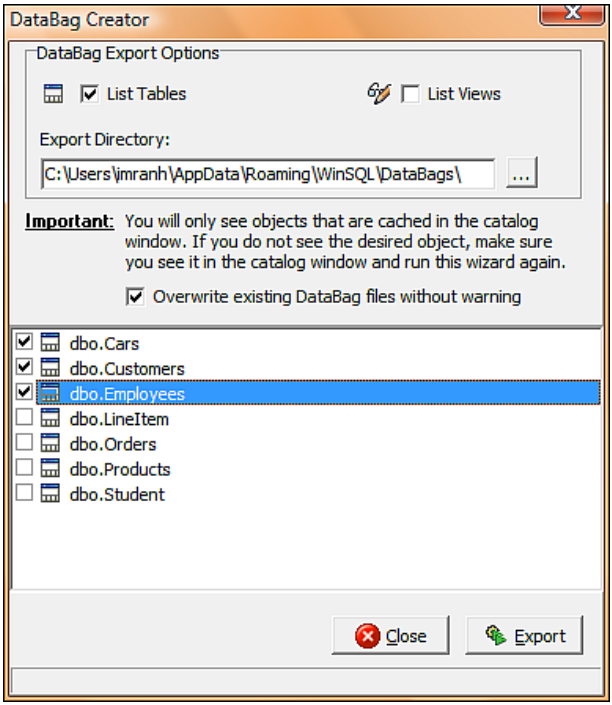
7 If you wish to view the contents of the DataBag, click **Yes**. If not, click **No**.

The data export is completed.

Disconnected Export: WinSQL Menu Options

1 Establish a connection with the source database. For more information, see "Establishing a Database Connection."

2 From the **Tools** menu, point to **Data To-Go**, and then click **Create DataBags**. The following window is displayed:



3 Select the tables for which you wish to create the DataBag, and then click **Export**.

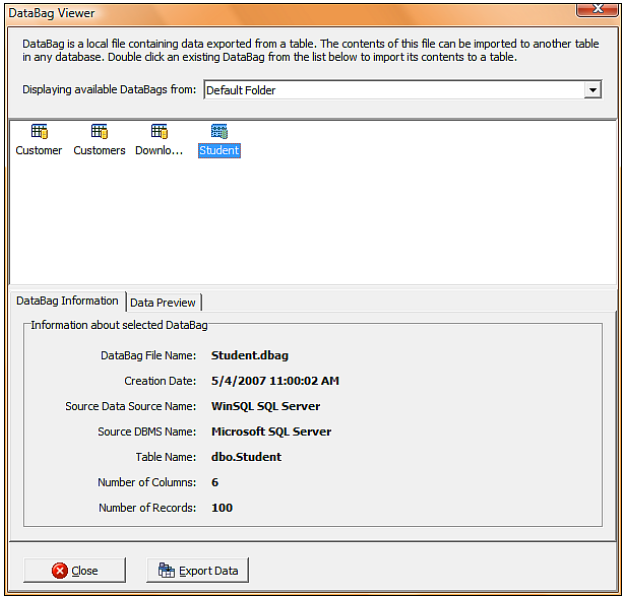
Importing a DataBag into a Target Database

1 Disconnect from the current database.

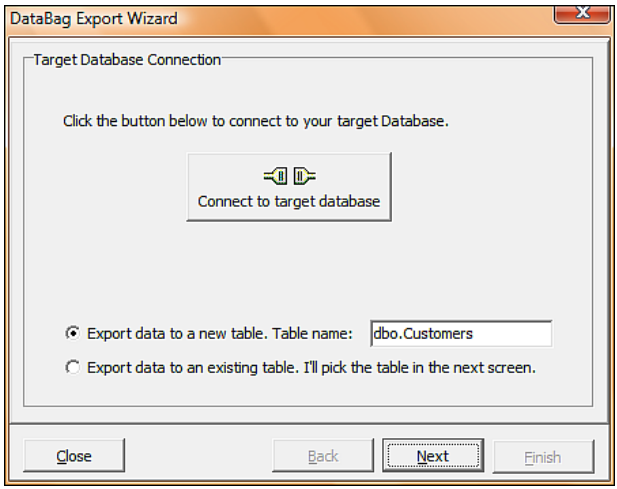
Note:

It is not necessary at this point to connect to the target database; the DataBag Export Wizard will guide you through establishing the connection during the data transfer.

2 From the **Tools** menu, point to **Data To-Go**, and then click **DataBags Viewer**. The following window is displayed:



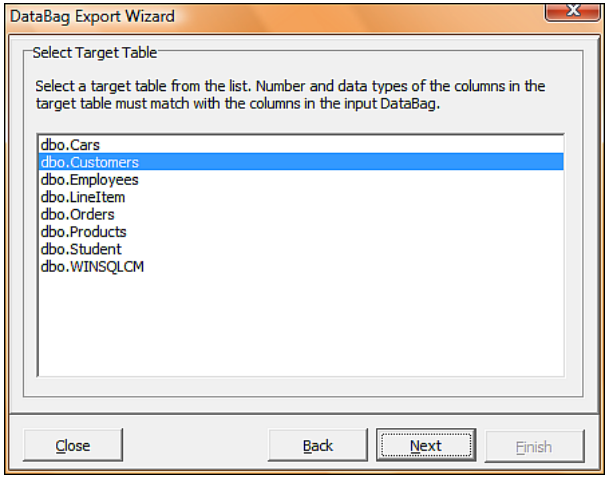
3 Select the desired table, and then click **Export Data**. Double-clicking a table will also invoke the wizard. The following window is displayed:



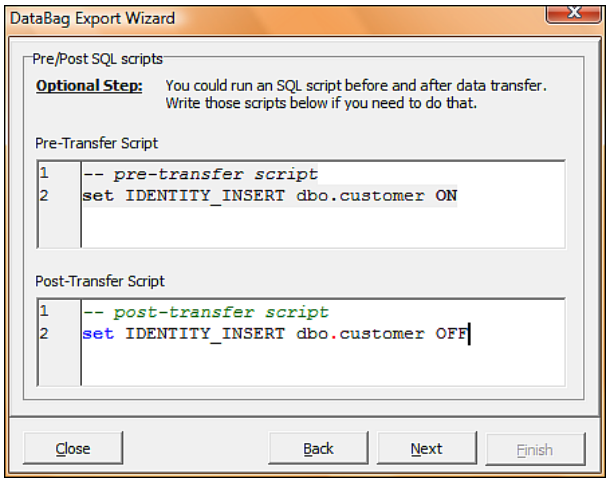
4 Click **Connect to target database**.

5 Select the desired database connection, and then click **OK**.

6 Click **Next** to continue. The following window is displayed:



7 Click to select the desired table, and then click **Next**. The following window is displayed:



8 If you wish to run a SQL script either before or after the data transfer, enter the script in either/or the **Pre-Transfer** or **Post-Transfer** text areas.

9 Click **Next** to continue.

10 If the information displayed in the **Summary** window is correct, click **Finish** to complete the data transfer.

11 A message is displayed indicated that the data transfer completed successfully. Click **OK**.

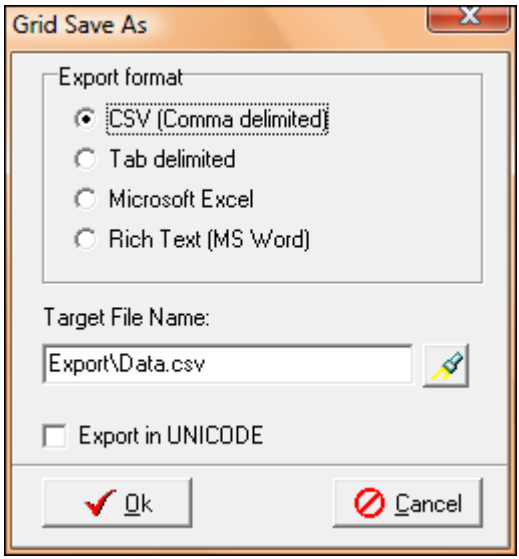
Quick Transfer to a Text File

1 Establish a connection with the source database. For more information, see "Establishing a Database Connection."

2 Click the **Query** tab.

3 Select and run the desired query to display the data.

4 Right-click a data row **ResultSet** window, and then click **Export data**. The following window is displayed:



5 Select the desired export format.

6 Accept the default target file name, or modify it as desired, and then click **OK**. The data transfer is processed and completed.

Enhanced Transfer to a Text File

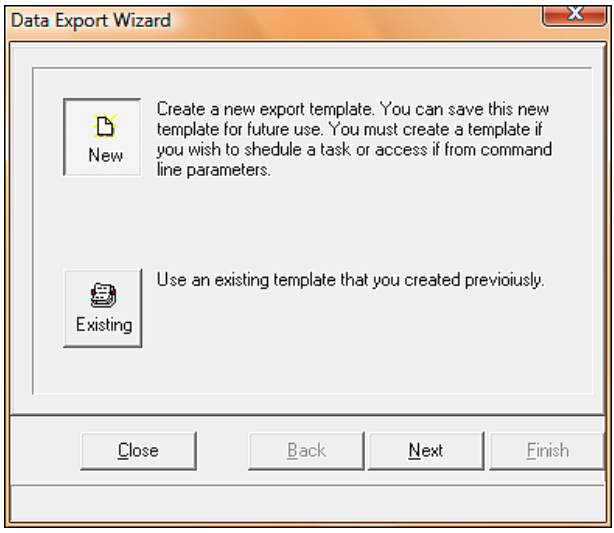
The following steps describe how to create an ODBC data source that connects to a text driver. This option provides more flexibility and allows you to run SQL queries on the text file.

1 Establish a connection with the source database. For more information, see "Establishing a Database Connection."

2 Click the **Query** tab.

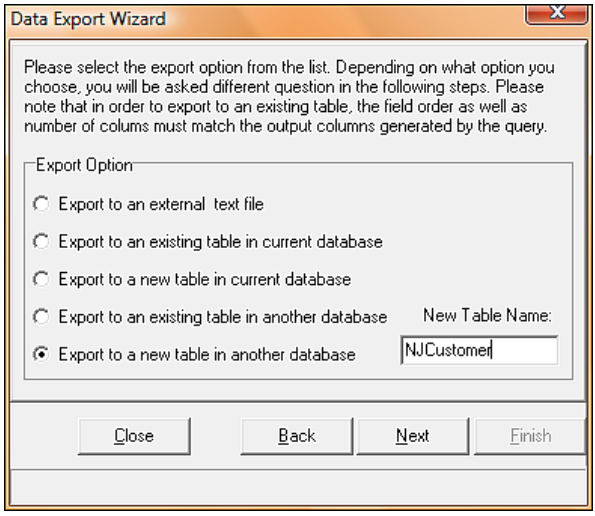
3 Click the down arrow by the play icon.

4 Click **Export results**. The following window is displayed:



5 Click **Next**.

6 Click **Export to a new table in another database**:



Note:

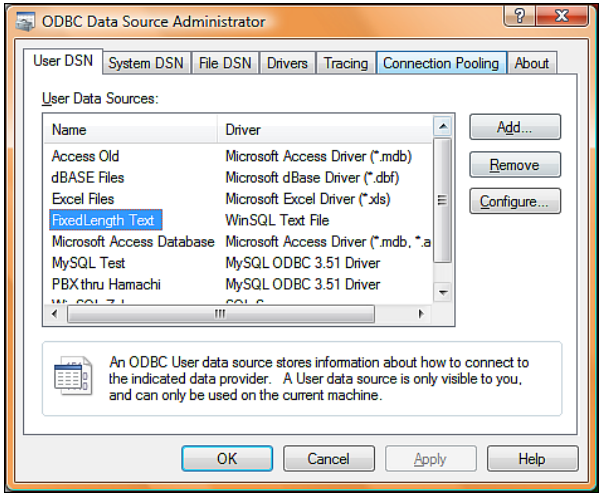
Seleting **Export to a new table in another database** will prompt for the creation of an ODBC Data Source pointing to your text file. Creating a data source, rather than simply

exporting to a text file, allows you to treat the text file as a database table and allows you to run SQL queries on the text.

7 Click **Next**.

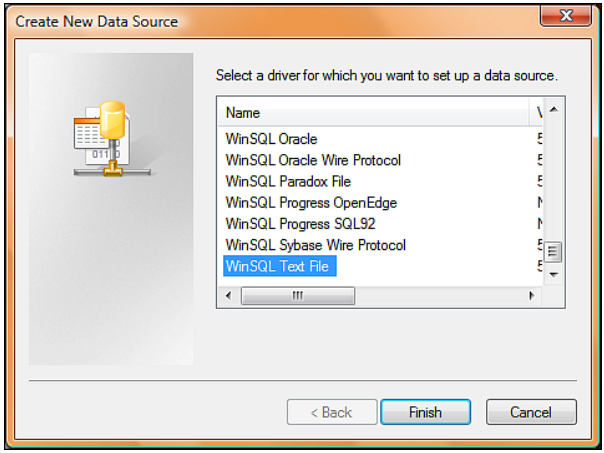
8 Enter a name for the new table, and then click **Next**. The **ODBC Data Source** window is displayed.

9 If you do not have a pre-created DSN for text file, click **ODBC Mgr.** to create a text driver. The following window is displayed:



10 Click **Add**.

11 Scroll through the list that is displayed, and select **WinSQL text file**.



12 Click **Finish**.

13 Complete the fields as desired in the **ODBC Text Driver Setup** window, and then click **OK**.

14 When the **ODBC Data Source** window is displayed, click to select the newly-created text driver.

15 Click **OK**. The **Data Export Wizard** window is displayed.

16 Modify the query as desired, and then click **Next**.

17 Modify the data mapping options as desired, and then click **Next**.

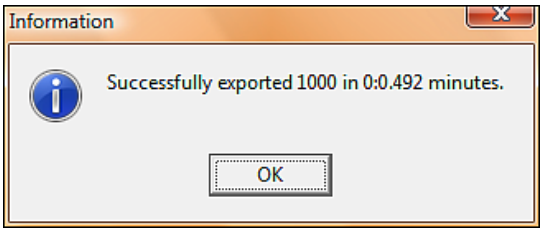
18 Ensure the export summary information is correct, and then click **Finish**.

Note:

If the export summary information is incorrect, click **Modify Query** to make the necessary changes.

The data export begins, and status information is displayed during the process.

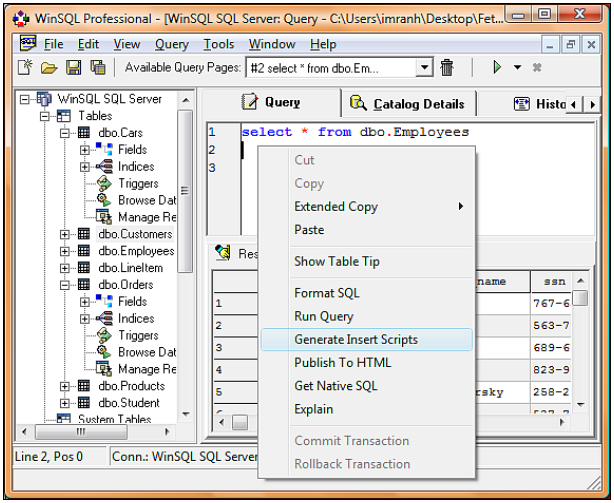
When the export has completed, the following window is displayed:



19 Click **OK** to continue. The data export is completed.

Exporting Data From an SQL Query

1 Write a SELECT statement in the **Query** window, right-click the query, and then click **Generate Insert Scripts**, depicted as follows:



The SELECT statement can be a simple query fetching data from one table, or it can have multiple JOINS combining several tables with complicated WHERE clause.

Notice that the only difference when exporting the result of a query verses exporting from a table is that in the latter case WinSQL Professional generates the SQL statement for you. This statement is a simple SELECT statement fetching all columns from

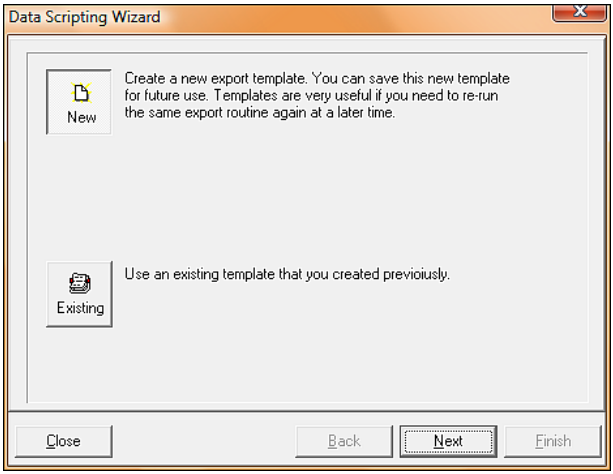
the specified table. Therefore, the screens look very similar in both cases

Note:

Synametrics strongly recommends running the query and checking for syntax errors prior to invoking this wizard.

As with any other WinSQL Professional wizard, you have a choice of creating a new template or use an existing template.

The following window is displayed:

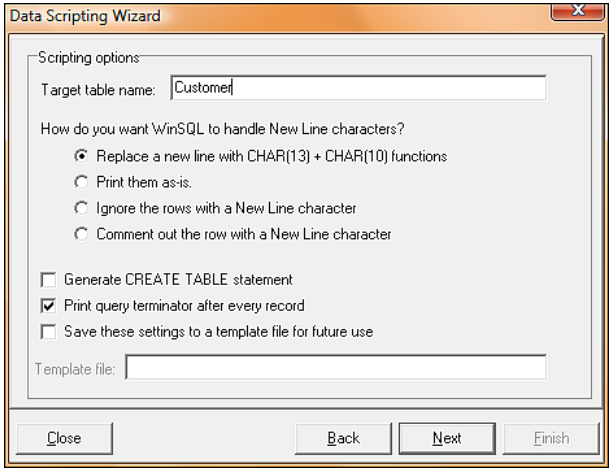


2 If this is the first time you are running this wizard, select **Create a new export template**. An export template stores all necessary information required to move data from one source to another target and, if saved, can be used for future exports.

OR

Select **Use an existing template that you previously created**.

3 Click **Next** to continue. The following window is displayed:



4 Use the following information to complete the options displayed on this window:

Target table name

If you create scripts for a table, the **Target table name** field is already populated. However, in the case of a query, you need to specify a table name. This is the name of the table used in the **CREATE TABLE** and **INSERT INTO** statements.

Handling New Line characters

Database rows can often contain embedded new line characters. Since a new line (CHAR (13) + CHAR (10)) character has a special meaning in SQL, they must be handled differently when such characters appear in the actual data.

The following table describes each of these options:

Option	Description
Replace with CHAR(13) + CHAR(10)	This is the default option and will replace all new line characters with an ODBC compliant function call. Example: Let's say a field contains the following data. 1 Main Street Suite 130

Option	Description
	This data will be converted to '1 Main Street' + {fn CHAR(13)} + {fn CHAR(10)} + 'Suite 130' Notice the {fn CHAR(13)} statement, which represents an ODBC escape sequence for representing ASCII 13 and 10 characters.
Print as-is	Choosing this option will print the new line characters as-is, meaning the actual INSERT statement may appear in multiple lines.
Ignore	This option will ignore every row that has a new line character.
Comment out	This option will write the rows to the output, but these rows will be commented out. You can later uncomment them and run them manually.

Generate CREATE TABLE statement

When this option is selected, a CREATE TABLE statement is generated before the actual INSERT statements. Note that this CREATE TABLE statement is generated based on the result of a SQL Query. Even if you are exporting every rows from a table, the wizard is going to issue a "**select * from tableName**" statement and run the wizard. Therefore, the generated CREATE TABLE statement will not have information regarding primary keys, foreign keys, or any index the table may have.

Print query terminator after every record

This option prints a query terminate after the INSERT statement.

5 After selecting all options, click **Next**, and then click **Finish** to allow WinSQL Professional to generate the INSERT statements. Once the statements are created, you can either save them to a file or copy them to the Windows Clipboard.

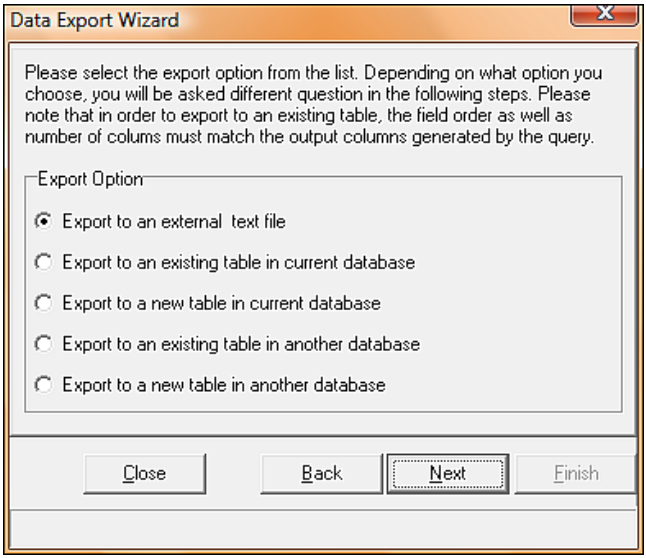
Exporting to an External Database

WinSQL Professional exports data based on an SQL query. To create an export, you must select a query on the **Query** tab before beginning the export.

Note:

Parameter queries cannot be used to export data.

Once you are satisfied with the SQL script, point to the **Query** menu, click **Export**, and then choose one of the following options:



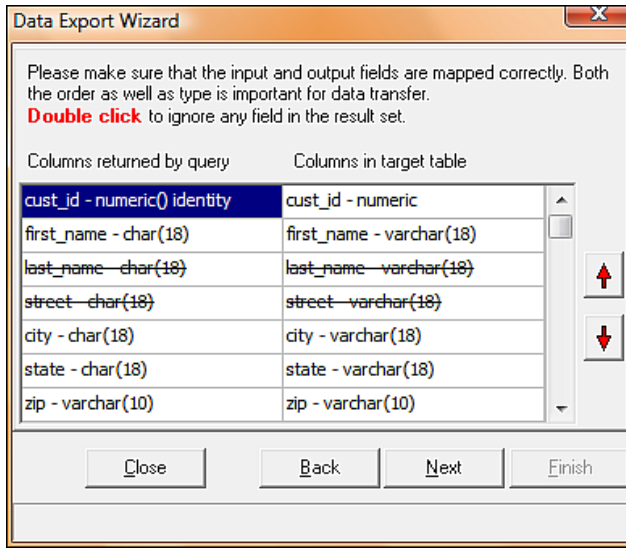
The following table describes each of the export options available in the **Data Export Wizard** window:

Export option	Description
Export to an external text file	You can customize the export format in the following steps. These format options include delimiters and headers. This option does not use ODBC to create the text file. IMPORTANT: Synametrics recommends that you use this option if the data type of the columns in the table are either numeric, date time, or characters less than 512 in length. To save data containing binary fields or long character fields consider creating an ODBC DSN for Text and let WinSQL treat your text file as an table.
Export to an existing table in	Use this option if the table already exists

Export option	Description
current database	in the target database. If you are trying to export the data in the same database, consider using an SQL UPDATE statement rather than using the Export Wizard.
Export to a new table in current database	Use this option if the table does not exist in the target database. WinSQL Professional will create a CREATE TABLE statement, matching the data types as closely as possible. You can manipulate the generated script to add constraints or change the data types. **Notes:** • If you decide to export the data into a new table, WinSQL Professional generates the SQL statement for you. • It is important that you not change the name of the target table in the script. If you need to change the name, click **Back** and replace the table name.
Export to an existing table in another database	Use this option if the table already exists in the target database.
Export to a new table in another database	This is similar to exporting to a new table in current database, except you are exporting to a different database.

Matching the source and target fields

It is very important that the order of source and the target fields have a one-to-one correspondence, depicted as follows:



Use the arrow icons to move a target field up or down until it matches with the corresponding source field. If the fields don't match, either the data will get exported to incorrect columns or the export process will fail. By double-clicking a row you can exclude a column from the data transfer.

Saving the export template

Once you are satisfied with all the options and settings, you can save the template file for future use. If you specify a file name that already exists, WinSQL Professional overwrites the file with the new format.

Using Existing Templates

Stored export templates can by used by clicking the desired template file, clicking licking **Next**, and then **Finish**.

Note:

If you are using a stored template that was created on another computer, ensure that both the source and target DSNs exist.

Export to INSERT Scripts

In addition to exporting to another data source, you can also use WinSQL Professional to generate INSERT scripts representing a query result. These INSERT scripts are very useful when moving data from one source to another, or for backing up your existing data.

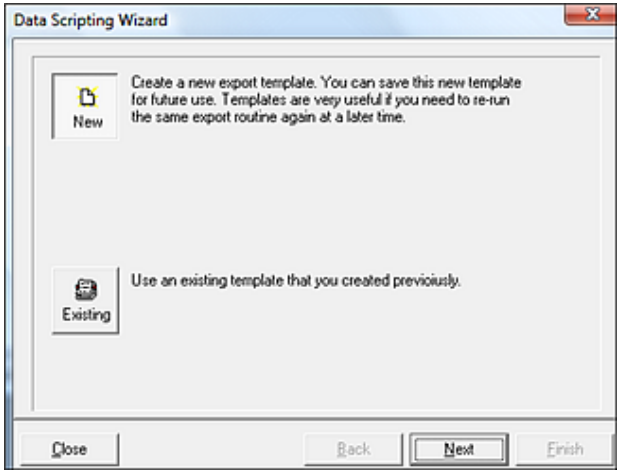
There are two methods to export data to INSERT script:

- Export data from a table
- Export data from a SELECT query

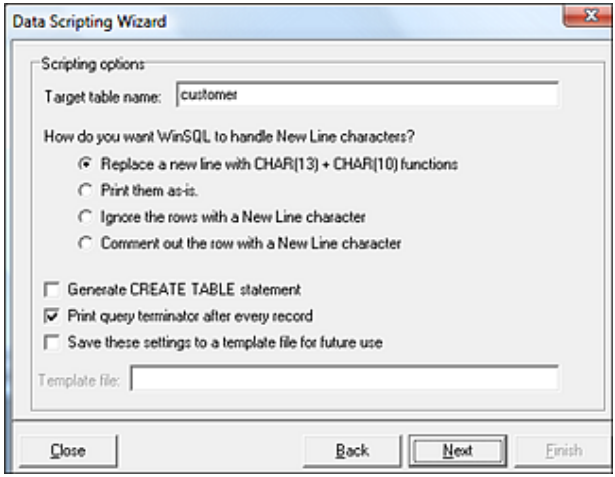
Exporting Data From a Table

1 In the **Catalog Details** window, select the desired table, right-click, and then click **Generate INSERT INTO Statements**.

2 Click **Selected Table**. The following window is displayed:



3 Accept **New**, and then click **Next**. A window similar to the following is displayed:



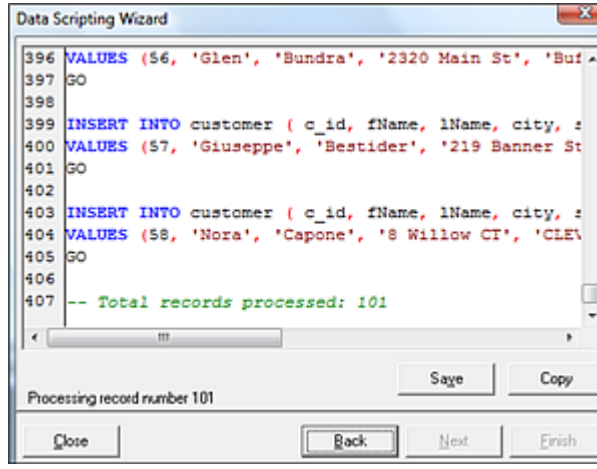
4 Select the desired options on this window, using the information in the following table as a guideline:

Field	Description
How do you want WinSQL to handle New Line characters? **Note**: These options define how "NEW LINE" characters appear in your text file. Since every record in the database appears in a line by itself, it is important to handle scenarios where a new line character appears in the data itself. These options allow you to choose how to handle them.	**Replace a new line with CHAR(13) _ CHAR(10 functions** A new line character in the data will be replaced by CHAR(13) + CHAR(10). For example, if the data in the database is: "It is a nice day today" It will be converted to: "It is a nice " + CHAR(13) + CHAR(10) + " day today"
	Print them as is No change will be made. You will see data in two lines.
	Ignore the rows with a New Line character Do not export rows that contain a new line character

Field	Description
	Comment out the row with a New Line character Export the row but comment out the query. You can later correct the problem and re-run it.
Generate CREATE TABLE statement	If selected, a CREATE TABLE statement will also be printed beside the INSERT INTO statements.
Print query terminator after every record	If selected, WinSQL's query terminator prints after every query. The default value for this option is 'GO.'
Save these settings to a template file for future use	Select this option to create a template file. If selected, you must specify a file name in the "Template File" field.

5 Once the desired options have been selected, click **Next**.

6 Click **Finish** to generate the script. A window similar to the following is displayed:



7 Perform one of the following options:

- Click **Save** to save the new script. You will be prompted to enter a new file name.
- Click **Copy** to copy the script to the Windows Clipboard.
- Click **Close** to close the **Data Scripting Wizard** window.

Exporting to an Executable File

Traditionally, sharing data between users requires exporting data into a common format, such as a text file or a spreadsheet file, and then importing the file on the receiving side.

WinSQL provides the ability to share data SQL-generated query results with other users in the form of an executable (EXE) file. With a few keystrokes, you can:

- Save the results of any SELECT statement to an EXE file. This executable can either be:
    - **Static**- always displays the same data
    - **Dynamic**- refreshes data from the server each time it is run
- Share the EXE file with other users, with no specific application requirements other than use of the Windows XP, Windows 2000, Windows 2003, Windows 2008, Windows Vista, or Windows 7 operating systems.
- Create an EXE file containing an unlimited number of rows of data that can easily be searched, sorted, and printed. This data can be copied and pasted into other applications, such as Microsoft Excel, for further manipulation and analysis.
- When this EXE file is run, it will allow users to
    - Print a report based on the data
    - Search data
    - Copy data and paste into applications like Microsoft Excel
    - Double-click the column header to sort the data
    - Show or hide desired columns

No other file is required to run the executable if it is static. An ODBC driver is required if the executable is created as dynamic.

Prerequisites

No prerequisites exist for static executable files. WinSQL creates a self-contained file that runs on any machine without installation and does not need any other file in order to run.

When you create a dynamic executable, the generated file must connect to the database at run time. This means you must have the ODBC driver installed on the target machine. Due to installation complexities and licensing issues, the generated file does not include the ODBC driver.

Additionally, you must specify a valid connection string that is used to establish connection to the database. There are two types of connection strings:

- Contains a data source name (DSN)
- Contains the ODBC driver name

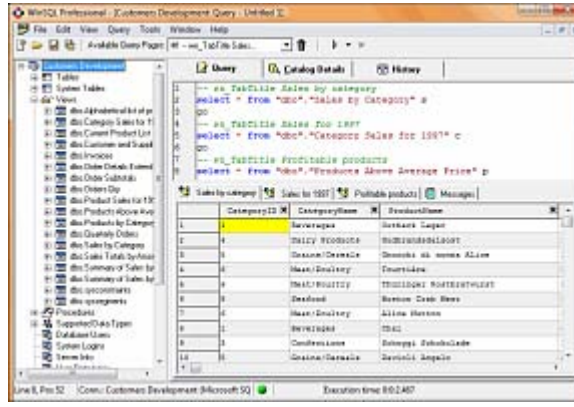
If you expect the same DSN on the target machine, you can have WinSQL copy the parameters of your existing connection to the generated executable.

If the same DSN does not exist on the target machine, you must specify a connection string containing the driver name. A connection string contains parameters as name-value pairs. For more information, see the “Connection Strings” topic in this document.

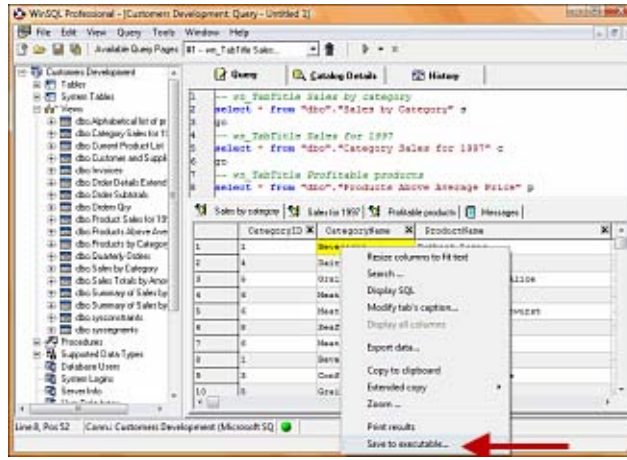
Perform the following steps to create an EXE file:

1 Select a query, and display the query results in grid form. For more information, see Displaying Query Results.

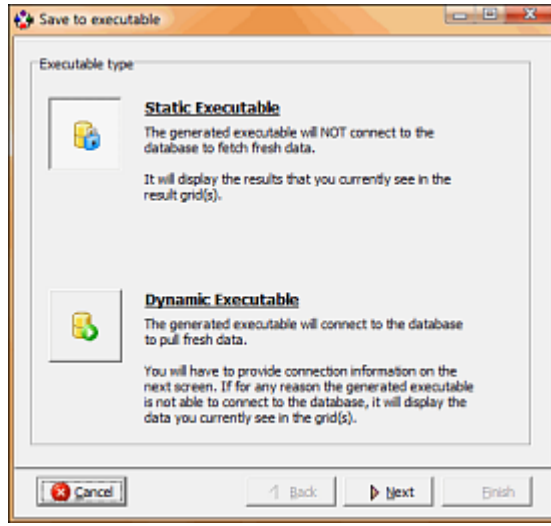
A window similar to the following is displayed:



2 Right-click within the results grid in this window, and then click **Save to executable**:

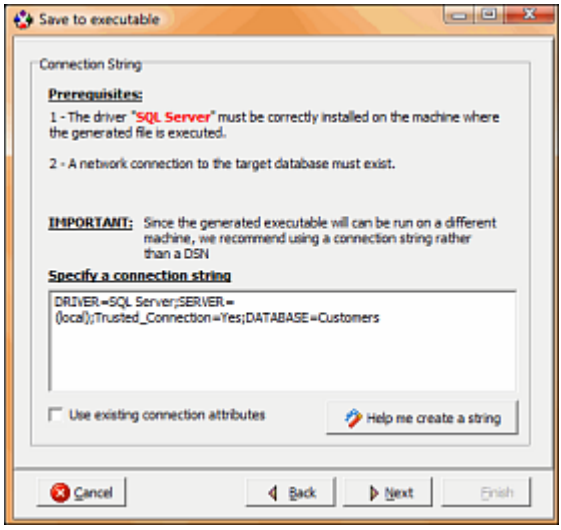


The **Save to executable** window is displayed:



3 Select the desired executable type, and then click **Next**.

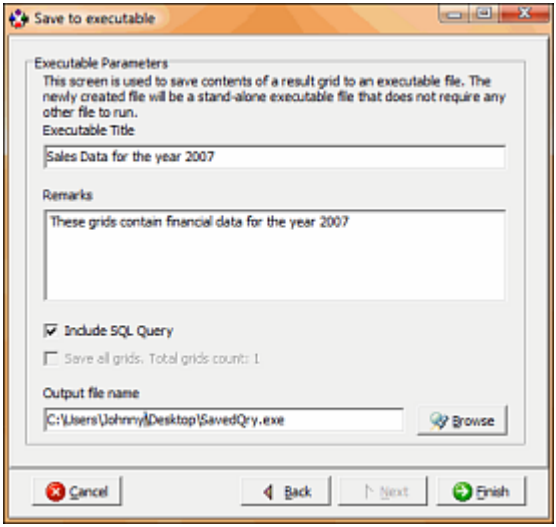
If **Dynamic Executable** is selected, a window similar to the following is displayed:



4 In the **Specify a connection string** field, type the connection string you wish to use to connect to the database.

If you expect the same DSN to be available on the target machine, you can select the **Use existing connection attributes** check box.

5 Click **Next**. A window similar to the following is displayed:



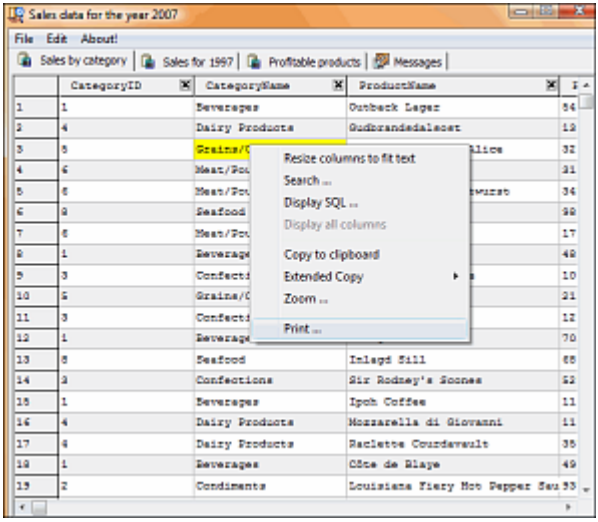
6 Complete the fields on this window, using the information in the following table as a guideline:

Option	Description
Executable Title	Type a name to use as the title for this EXE file. This name displays when the EXE file is generated.
Remarks	Use this field (optional) to type comments about the EXE file. These remarks display on the "Messages" tab in the generated EXE file.
Include SQL Query	Select this option to include the SQL query in the EXE file.
Save all grids	Select this option to save all grids in the query in the resulting EXE file.
Output file name	Specify the name of the EXE file name. If you do not wish to save the EXE file in the default location, click **Browse** to navigate to another location.

7 Once all options have been selected or entered, click **OK** to create the EXE file.

The file is created in the specified location and can now be shared with other users.

The following screenshot depicts a generated EXE file:



Notice that the screen shot above looks similar to WinSQL's result grid. In reality, it belongs to the generated executable. Similar to WinSQL, multiple result sets are displayed in different tabs, allowing the user to select the desired results.

If the generated executable is dynamic, the data is refreshed when the executable is run. For any reason, if this executable is not able to connect to the database to fetch fresh data, it will act as if it were a static file, displaying stale records.

Generating Parameterized Executables

For users who do not have access to WinSQL but who wish to run database queries using specific parameters, “parameterized” executables files can be created by a WinSQL user and delivered in the form of an executable file. These executables are dynamic, meaning the data is refreshed, and the results displayed are current each time they are run.

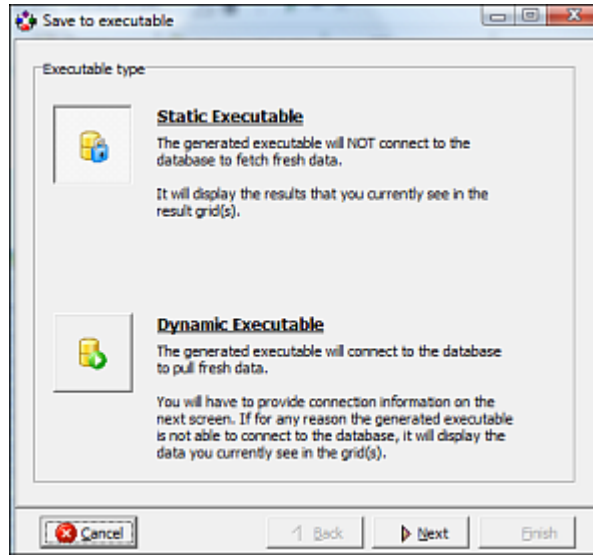
Perform the following steps to create a parameterized executable:

1 Use the information in the “Parameter Queries” topic provided in this document to create your parameter query. The following is an example of a query using ::EnterRegion as a parameter that will prompt the user to supply a value when the query is run:

```
SELECT * FROM Customers
WHERE region = ::EnterRegion
```

2 Execute the query.

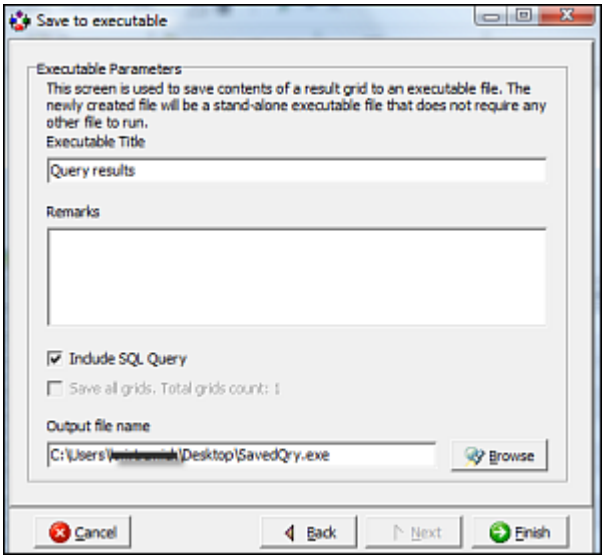
3 With the query results displayed, right-click within the **Resultset** window, and then click **Save to executable**. A window similar to the following is displayed:



4 Click the **Dynamic Executable** icon, and then click **Next**.

5 A window displays providing information about the perquisites and the connection string required to run the executable. If you are unsure what to specify for the connection string, click **Help me create a connection string**.

6 Click **Next**. A window similar to the following is displayed:



7 Type the desired name for this executable in the **Executable Title** field.

8 If necessary, click **Browse** to navigate to and select the location where you wish to save your query.

9 Click **Finish**. The executable is now available for delivery

Importing Text Files

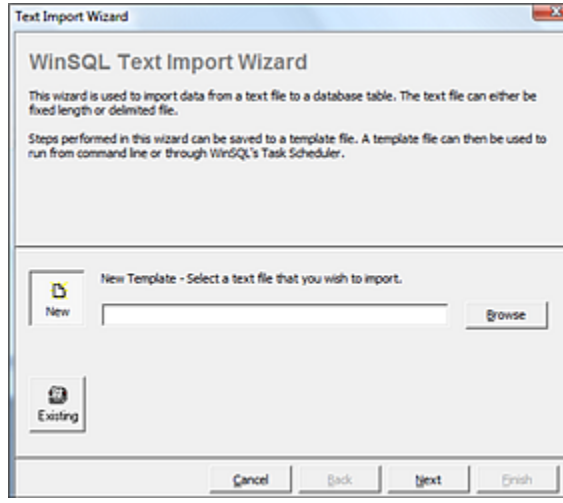
WinSQL provides a Text Import Wizard to use when importing data from a text file into an existing table or a new table in your database.

The topics in this section describe how to use the Text Import Wizard to import a delimited text file and a fixed width text file.

Importing a Delimited Text File

Perform the following steps to use the Text Import Wizard to import a delimited text file:

1. From the **Query** menu, point to **Import/Export**, and then click **Import From Text File**. The following window is displayed:



Note:

You can also invoke the Text Import Wizard by opening Windows Explorer, navigating to and selecting the text file you wish to import, and then dragging it to the WinSQL **Catalog** window.

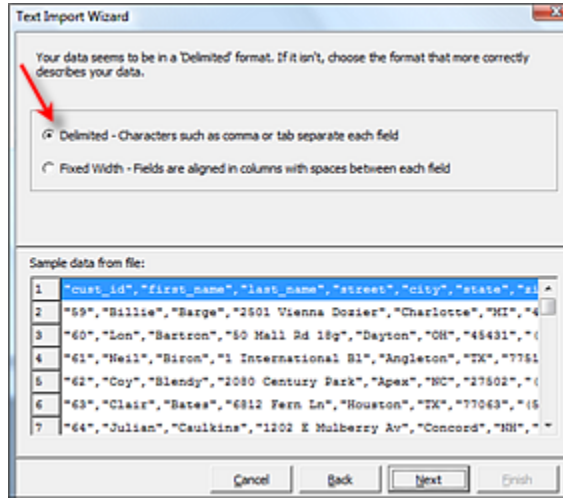
2. Click **New**.
3. Click **Browse**.

4 Navigate to the location containing the text file you wish to import, select the file, and then click **Open**.

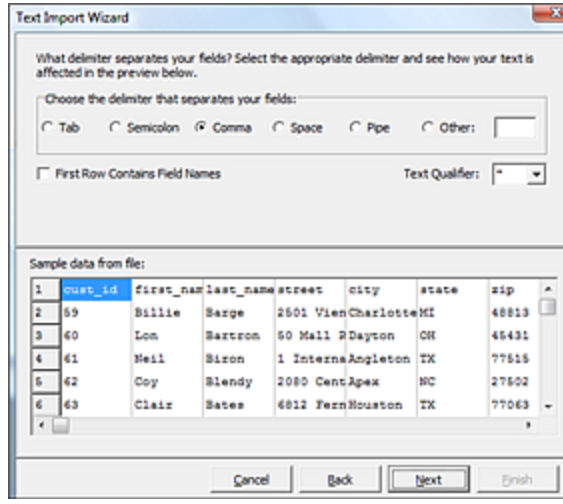
5 Click **Open**.

6 Click **Next**.

7 Ensure the **Delimited** format is selected:



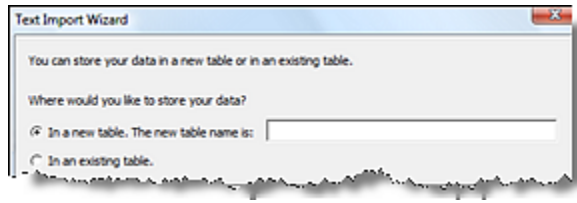
8 Click **Next**. The following window is displayed:



9 Choose the appropriate delimiter that separates the text in your file.

10 If your file contains field names or column headings, click to select the **First Row Contains Field Names** check box.

11 Once all options have been selected, click **Next**. The following window is displayed:

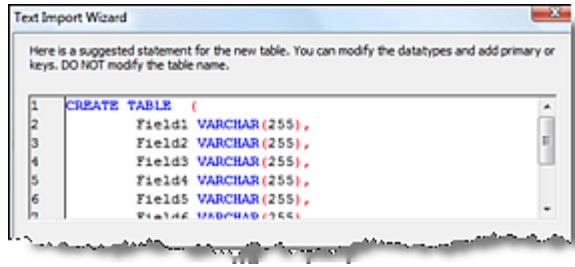


12 Select **In a new table** if you wish to import the data into a new table in your database. If this option is selected, you must type a name for the new table in the blank field.

OR

Select **In an existing table** if you wish to import the data into an existing table in your database. If this option is selected, you will be prompted to select the table in which you wish to import the data.

13 Click **Next**. A window similar to the following displayed:

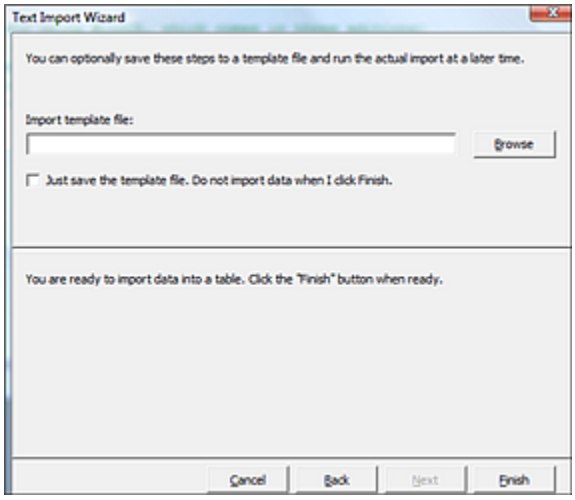


This window displays a CREATE TABLE statement that defines the design of the new table, allowing you to make modifications, if desired.

Note:

You may make changes to the field types or lengths, but **do not** modify the table name.

14 If all field types and lengths are correct, click **Next**. The following window is displayed:

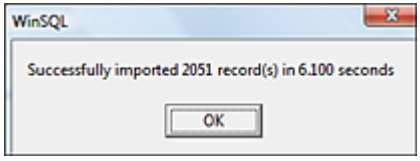


15 Click **Finish.**

Note:

If you wish to save this import as a template, either browse to or select an existing template file, and then click **Finish**.

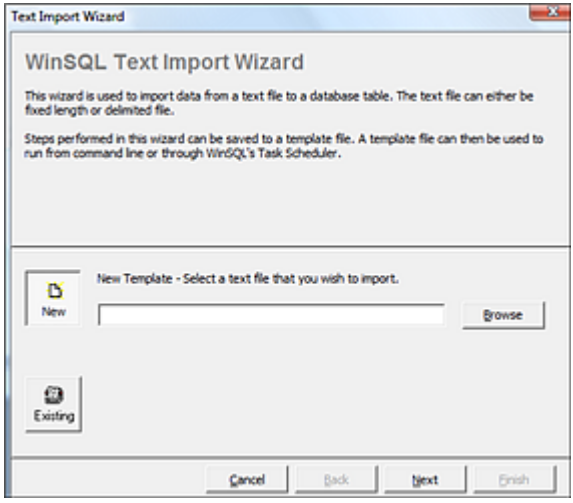
The data is imported, and a message similar to the following is displayed:



Importing a Fixed Width Text File

Perform the following steps to use the Text Import Wizard to import a fixed length text file:

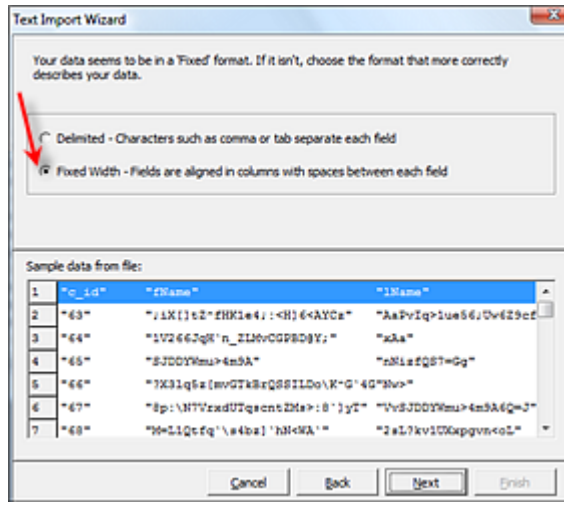
1. From the **Query** menu, point to **Import/Export**, and then click **Import From Text File**. The following window is displayed:



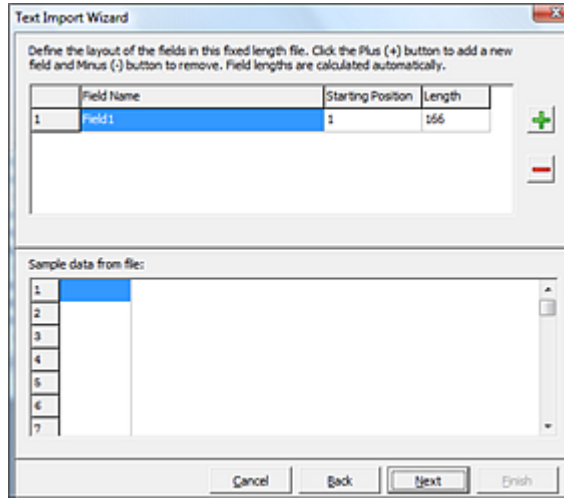
Note:

You can also invoke the Text Import Wizard by pointing opening Windows Explorer, navigating to and selecting the text file you wish to import, and then dragging it to the WinSQL **Catalog** window.

2. Click **New**.
3. Click **Browse**.
4. Navigate to the location containing the text file you wish to import, select the file, and then click **Open**.
5. Click **Open**.
6. Click **Next**.
7. Ensure the **Fixed Width** format is selected:

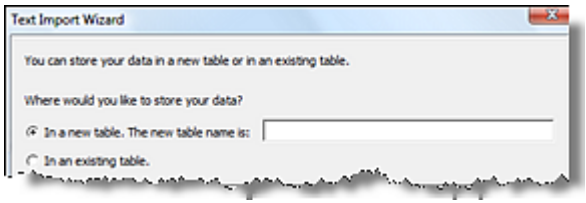


8 Click **Next**. The following window is displayed:



9 Click the green plug (+) icon to add fields from your import file, and then specify the starting position for each field added. Field lengths are automatically calculated.

10 Once all fields have been selected, click **Next**. The following window is displayed:

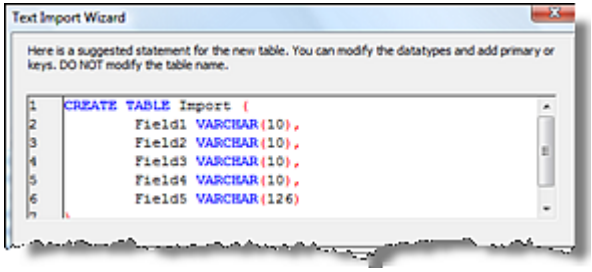


11 Select **In a new table** if you wish to import the data into a new table in your database. If this option is selected, you must type a name for the new table in the blank field.

OR

Select **In an existing table** if you wish to import the data into an existing table in your database. If this option is selected, you will be prompted to select the table in which you wish to import the data.

12 Click **Next**. A window similar to the following displayed:

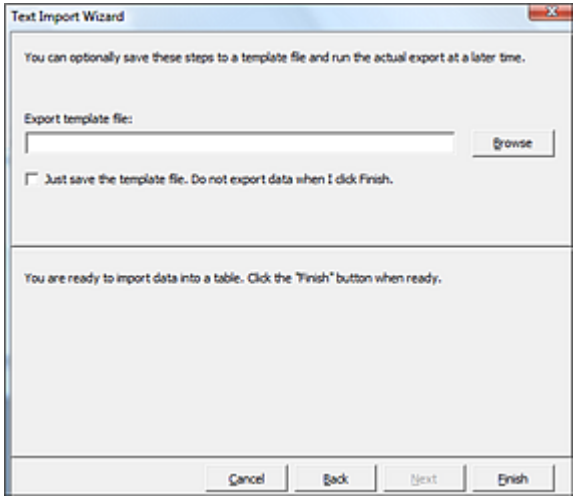


13 If all field types and lengths are correct, click **Next**.

Note:

You may make changes to the field types or lengths, but **do not** modify the table name.

The following window is displayed:

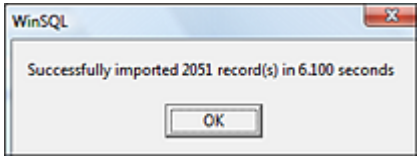


14 Click **Finish.**

Note:

If you wish to save this import as a template, either browse to or select an existing template file, and then click **Finish**.

The data is imported, and a message similar to the following is displayed:

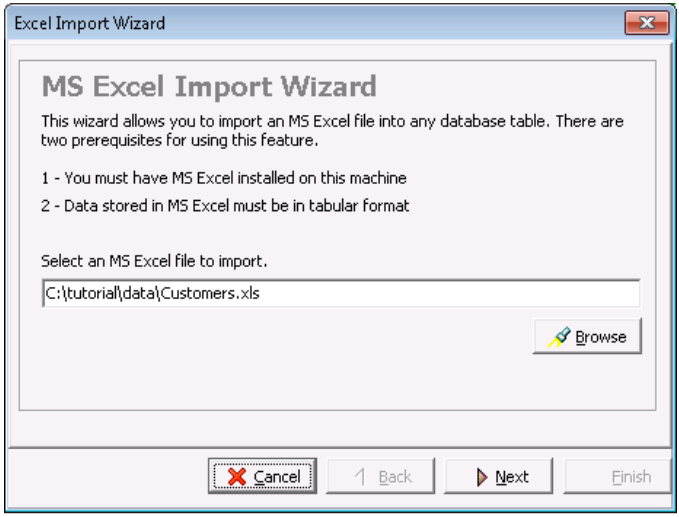


Importing from Microsoft Excel

WinSQL provides a Microsoft® Excel® Import Wizard to use when importing data from an Excel spreadsheet into an existing table or a new table in your database.

Perform the following steps to use the Text Import Wizard to import an Excel worksheet:

1 From the **Query** menu, point to **Import/Export**, and then click **Import From MS Excel**. The following window is displayed:



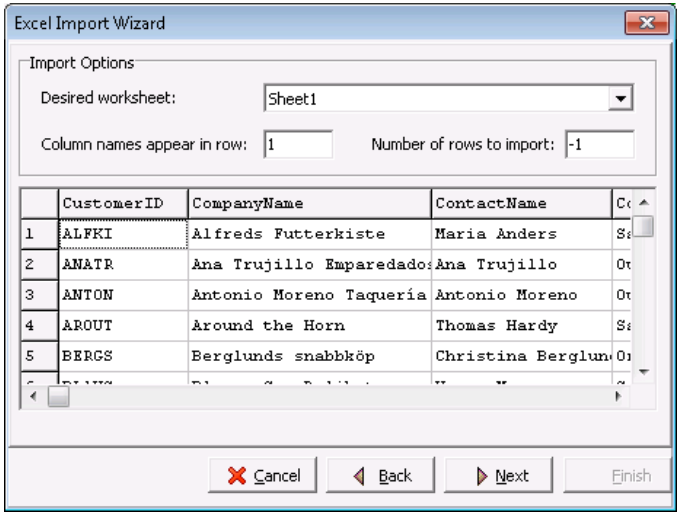
Note:

You can also invoke the Import Wizard by opening Windows Explorer, navigating to and selecting the Excel file you wish to import, and then dragging it to the WinSQL **Catalog** window.

2 Click **Browse** to navigate to the Excel workbook.

3 Click to select the file, and then click **Open**.

4 Click **Next**. A window similar to the following, containing a preview of the Excel data, is displayed:



5 If your workbook contains multiple worksheets, click the down arrow in the **Desired worksheet** field to select a worksheet.

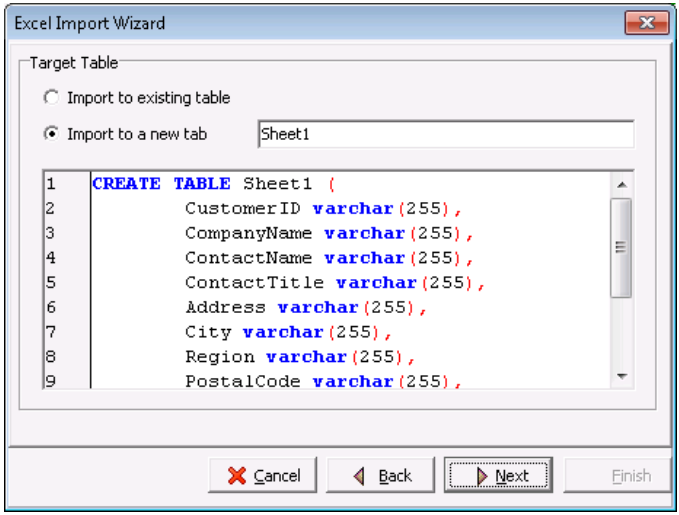
6 Click **Next**

7 If you wish to import your Excel data into an existing table, select the **Import to existing table** option, and then click the down arrow to select the desired table

OR

Accept the default option, **Import to a new table**, to import your Excel data into a new table. If this option is selected, WinSQL will suggest a CREATE TABLE statement that you can modify, illustrated as follows:

.



Important Note:

If you wish to change the name of the new table, type the desired name in the text field next to the **Import to a new table** option. Do not modify the table name in SQL script.

8 Click **Next**, and then click **Finish**.

WinSQL will read the Excel data and import it into the selected or new table in the database.

Chapter 12: Offline Backup/Restore

Overview

Introduction

This chapter provides information about performing a database backup and restore.

In this chapter

This chapter contains the following topics:

Backing Up a Database

WinSQL offers an offline backup utility that can be used in place of your native backup utility. One of the primary advantages of using WinSQL's backup utility is that it offers the flexibility of backing up one type of database and, if desired, restoring to a different type of database.

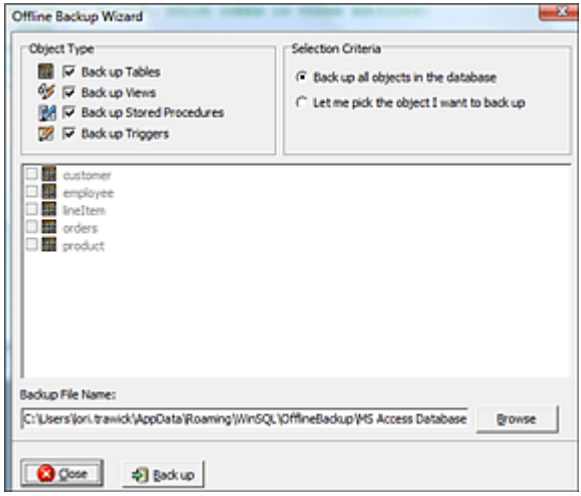
Additionally, WinSQL's backup utility is always available and is not dependent on outside resources, such as an ISP backup service whose resources are shared by many users.

Important note:

WinSQL's backup utility is limited to backing up only certain objects in the database, such as tables, views, stored procedures, and triggers. Other database-specific objects, such as user logins, user defined data types, and sequences are not eligible for back up.

Perform the following steps to back up a database:

1. Connect to the database you wish to back up. For more information, see "Establishing a Database Connection."
2. From the **Tools** menu, point to **Offline Backup**, and then click **Backup**. A window similar to the following is displayed:



3 Accept the default option to back up all objects in the database

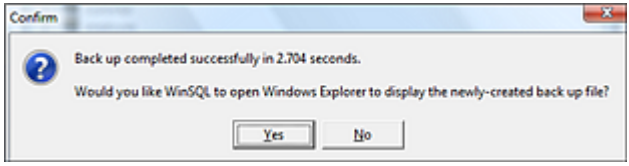
OR

Click the **Let me pick the object I want to back up** option. If this option is selected, you must click to select the check box by each object you wish to back up.

4 Accept the default backup file name and location, or click browse to select another location and/or file name.

5 Click **Back up**.

When the backup has completed, a window similar to the following is displayed:



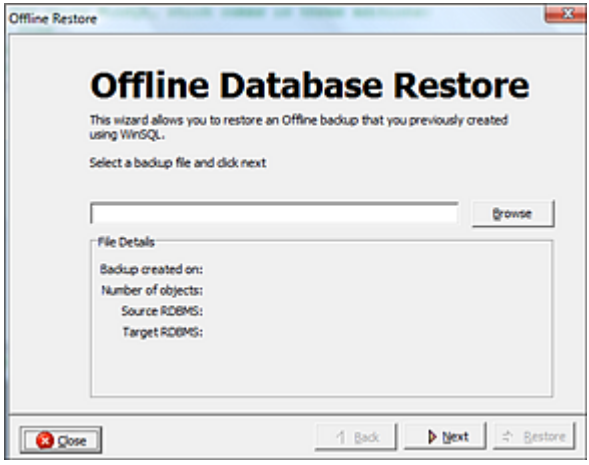
6 If you wish to view the backup file, click **Yes**. Otherwise, click **No** to close this window.

Restoring a Database

Perform the following steps restore a database:

1. Connect to the database you wish to back up. For more information, see "Establishing a Database Connection."

2. From the **Tools** menu, point to **Offline Backup**, and then click **Backup**. A window similar to the following is displayed:

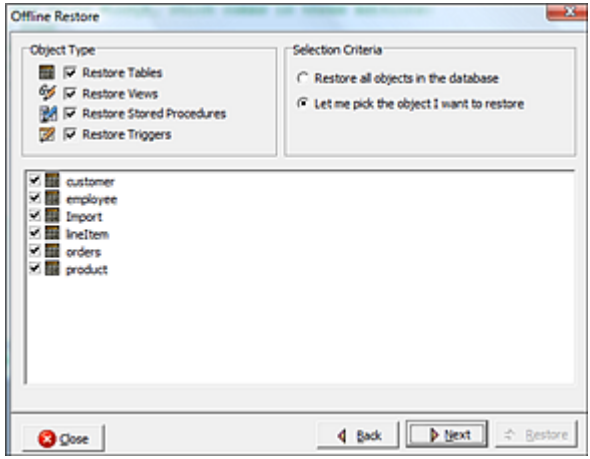


3. Click **Browse** to navigate to the location where your backup file is located, click to select the file, and then click **Open**.

   Note:

   If the source database type is different from the target database, WinSQL displays a warning message that only data will be restored. In such cases, you cannot restore views, stored procedures or triggers.

4. Click **Next**. A window similar to the following is displayed:

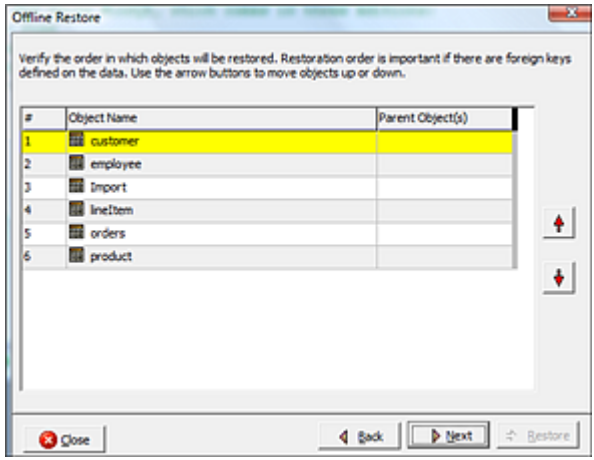


5 Click **Restore all objects in the database**

OR

Click the **Let me pick the object I want to restore** option. If this option is selected, you must click to select the check box by each object you wish to restore.

6 Once all options have been selected, click **Next**. A window similar to the following is displayed:

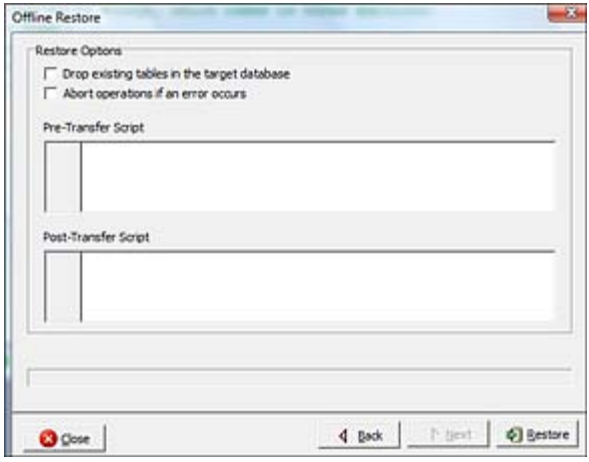


7 Verify the order in which the objects will be restored. If necessary, click the Up and Down arrows to rearrange the objects.

Note:

The restoration order is important in cases where tables are related to each other. For example, assume that the “Orders” table contains a field that refers to the primary key in the “Customer” table. In this case, you must restore the parent table prior to restoring the children tables.

8 Click **Next**. A window similar to the following is displayed:



9 Click **Drop existing tables in the target database** to allow the restore process to create new tables in your database

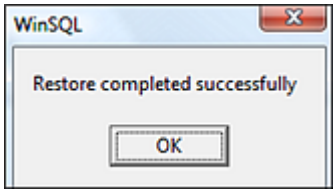
OR

Do **not** select this option to allow the restore to append data from the backup file to the existing database tables.

10 Click **Abort operations if an error occurs** to prevent the restore from continuing to run if an error is encountered during the restore.

11 Click **Restore**.

When the restore has completed, the following window is displayed:



Chapter 13: Database-Specific Plug-Ins

Overview

Introduction

This chapter provides information about using database plug-ins.

In this chapter

This chapter contains the following topic:

Overview of Database Plug-Ins

Database plug-ins enhance the capability of WinSQL Professional. Since ODBC is a generic API, it does not support database-specific functions. Most of this functionality is achieved by querying system tables (also known as the system catalog) in the back-end database. Plug-ins provide the scripts for these queries, which are in plain text and XML formats.

XML is becoming the standard for configuration files and for storing small documents. Many programmers and system administrators are already familiar with its semantics and syntax. Therefore, WinSQL Professional uses XML to create plug-ins, which makes its' editing very easy.

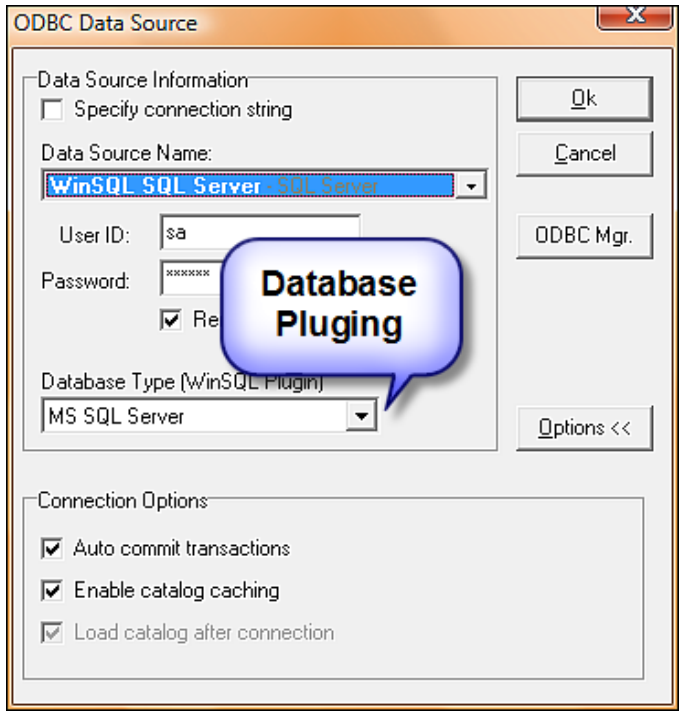
What do plug-ins provide?

The following features are provided by database plug-ins:

- Retrieving scripts for Views, Triggers, Stored procedures
- Generating DDL scripts for tables
- Database specific objects such as sequences, functions, rules, and check constraints
- Database specific syntax for creating indexes
- Configuration information from the server

How to select a plug-in

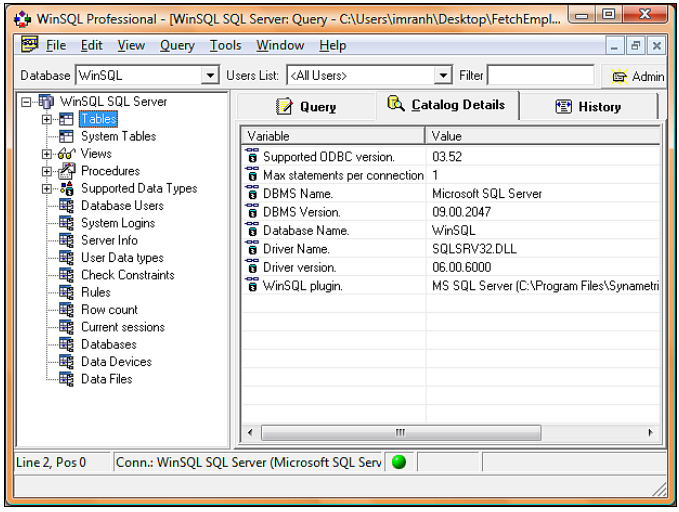
You select a plug-in when you first connect to the database, as depicted in the following image:



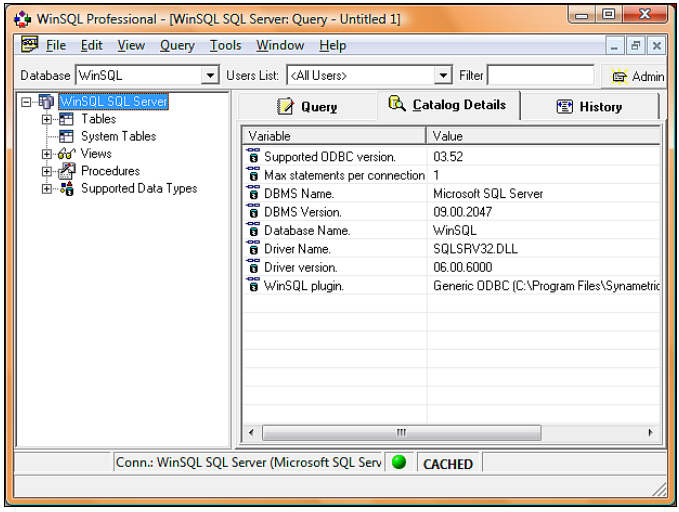
What happens when you do not select a plug-in?

Although selecting a plug-in is not required to run queries, selecting them significantly enhances the functionality of WinSQL Professional.

Consider the following images. The first image depicts WinSQL Professional connected to a Microsoft SQL Server database and using a plug-in:



The second image shows how the nodes in the tree are reduced when a generic plug-in is used:



Notice that you do not see any other node after **Supported Data Types**. This is because information for all other nodes comes in from the plug-in.

What if a plug-in is not available for your database?

If a plug-in is not available for your database, it does not mean that you cannot connect to a database. It means that some of the functionality will not be available.

There are two reasons why a plug-in is not available:

- You are connecting to a database for which a plug-in cannot be written. Examples of such databases are Microsoft Access, Microsoft Excel and text files.
- You are using a database that is either new or our engineers have not attempted to write a plug-in for it. Please note that our engineers are constantly working on creating new plug-ins for different databases. If you think a new plug-in should be created for the database you are working with, contact our support team and they will work with you to create a new plug-in

Chapter 14: Administrative Tasks

Overview

Introduction

This chapter provides information about running routine administrative tasks, such as creating or rebuilding indexes.

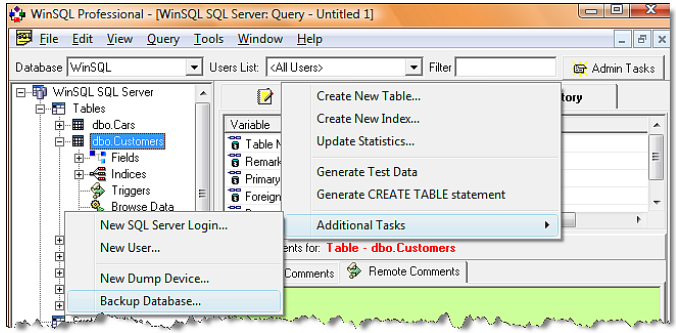
In this chapter

This chapter contains the following topics:

Wizards Used For Administrative Tasks

WinSQL Professional provides several wizards to perform different administrative tasks. The database specific plug-in selected during connection determines the type of wizards available at run time.

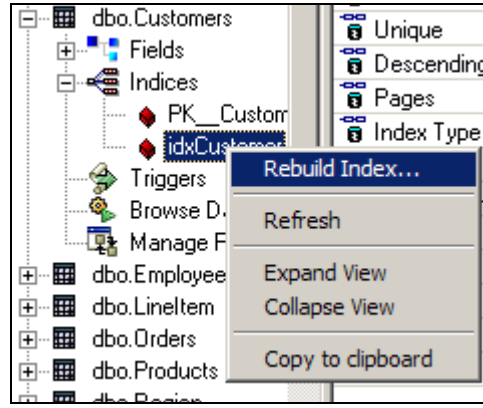
The wizards are available on the **Catalog Details** tab by clicking the **Additional Tasks** option, depicted as follows:

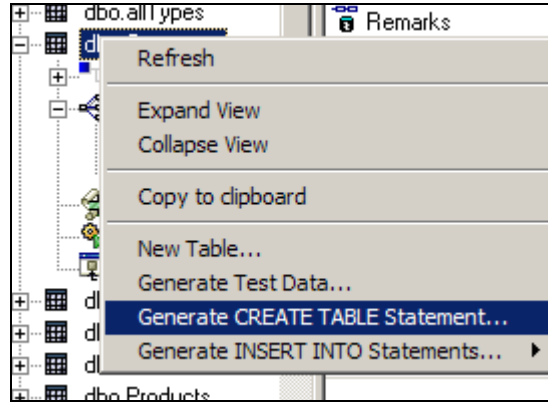
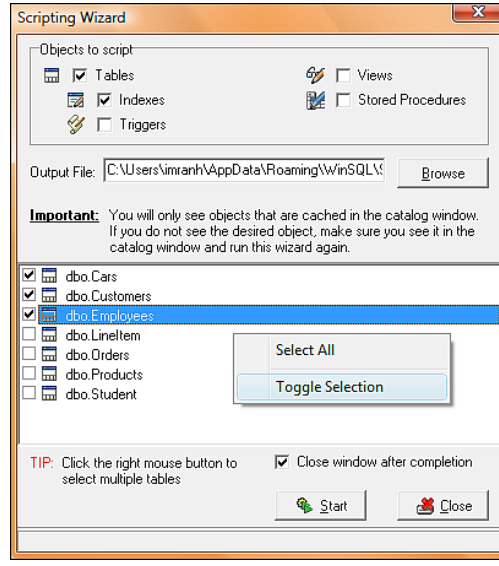


Most of these wizards depend upon the functionality supported by the back-end database, and certain wizards may not be available for certain databases. For example, the TEXT driver does not support index creation. Therefore, running the **Create New Index** wizard for a TEXT data source would result in errors.

Available Wizards

The following table describes some of the wizards available in WinSQL Professional:

Wizard Name	Description
Create New Index	This wizard is invoked by selecting **Create New Index** from the drop down menu after clicking **Admin Tasks**. Although this option is available for most data sources, some back-end sources, for example, TEXT driver, may not support it. Plug-in enabled databases, such as Oracle, MS-SQL Server, DB2, Sybase, Informix, Mimer, PostgreSQL, MySQL and others use a database-specific syntax for CREATE INDEX and support extensions that are not available in other RDBMS. For example, if you are connected to an MS-SQL Server database, you can create a CLUSTERED index.
Rebuild Index	This wizard is invoked by selecting **Rebuild Index** from the menu when an index is selected.  You can only rebuild an index if it is NOT associated with a primary key. Rebuilding any index requires dropping and recreating the same index again. Most RDBMS do not allow dropping an index if it is associated with a primary key.
Generating DDL scripts	WinSQL Professional allows you to generate DDL scripts for tables, views, triggers and stored procedures. You can either select one object at a

Wizard Name	Description
	time, or you can use the DDL Scripting wizard to selectively pick the object. To pick one table at a time, select the table from **Catalog Details** window, and then click **Generate CREATE TABLE Statement**.  To use the DDL Scripting wizard, point to the **Tools** menu, and then click **Generate DDL Scripts**. Select the desired type of objects from this window, specify a file name where the script will be saved, and click **Start**. 

Chapter 15: Command Line Processing

Overview

Introduction

This chapter provides information about command line options and how to use this feature, and how to write and run WinSQL scripts.

In this chapter

This chapter contains the following topics:

Command Line Options

Beginning with WinSQL Professional version 2.0, using command line options is available. You can run both a stored SQL script and an export task. This allows WinSQL Professional to be executed from a MS-DOS batch file or a scheduler that can run external tasks.

If your back-end server supports external commands, you can even call WinSQL Professional from a stored procedure or trigger. For example, you can use the **xp_cmdshell** stored procedure in Sybase to launch WinSQL Professional to export data to an Informix database.

Command Line Syntax

The syntax for running command line options is described as follows:

`WinSQL <filename> | <option-Value>`

- **`filename`**: This is the name of a text file holding the SQL script. This will just open the text file in the editor but will not run it.
- **`option-Value`**: Using these pair of option-value, you can run an SQL query against the database. Refer the following table for a complete listing of the options.

The following table describes of the available options:

Option	Value	Description
-a	NULL	Close WinSQL Professional automatically after the task is complete.
-d	DSN Name	Contains the name of the ODBC DSN. If there is a space in the name, it must be enclosed in quotes.
-g	NULL	Runs WinSQL Professional in debug mode. Will create a debug.log file that can be used to pin point any potential problem.

Option	Value	Description
-o	Output file	This file is used to store any error messages or data returned from the server as a result of a query execution. If this option is not specified, there will be no output generated.
-q	Query File	Name of the file that holds the SQL query.
-p	Password	User password for database connection
-u	User Name	User name for database connection
-x	Export template	Name of the export template file.
-h	Publish HTML	Name of the HTML template file.
-?	NULL	Displays a help dialog box displaying all the command line options

Command Line Examples

Example 1

The following example shows how to run any SQL query from the command line. It will open a file named **updateStudents.sql** and connect to the database using specified DSN, User Name, and Password. All error messages or any output data will go to **output.txt** file, and WinSQL Professional will close once the query has completed.

```
WinSQL "-dStudent DSN" –uAdmin –pAdmin –
qc:\updateStudents.sql –oc:\output.txt -a
```

Example 2

The following example opens a file in the editor but does not run it.

```
WinSQL c:\student.sql
```

Example 3

The following example exports the data based on a template file called *StudentTemplate.wet*.

```
WinSQL –xStudentTemplate.wet -a
```

Notice that you do not have to provide a user id, password and DSN name when specifying a template file because this information is embedded in the template file. If you need to change any parameters in the template file, use the **Export Template File Editor** on the **Tools** menu.

Important notes for all examples

- There should not be any space between the option and its value.
- You can run only one SQL statement from the command line. If you have multiple statements separated by a terminator, errors may generate and the query will not run.

Running Scripts From a Command Line

Scripting allow users to write special functions in a plain text file and pass the name of this file as a command line argument to WinSQL. WinSQL runs these functions in a sequential order.

The following rules apply when using a script file:

- The file extension of a script file must be one of the two values:
  - **.wxf**, (*MyScriptFile.wxf*)
  - **.winsqlscript** (*MyScriptFile.winsqlscript*)
- Lines starting with two dashes are considered comments and are ignored at run time.
- Empty lines are ignored.
- Function names are case-sensitive and must include open and close parenthesis, even if parameters are not required.
- Parameters must be enclosed in double quote characters.

To run the simply pass the file name containing a script as a command line argument to WinSQL, for example:

winsql MyScriptFile.wxf

Script Functions

The following is a list of available functions you can write in a script.

Note that each function begins with *wnsFunc_*.

Function	Name/Description/Parameters
Function Name	**wnsFunc_AutoLogin**
Description	Establishes a connection to a database. This should be the first function in your script This function takes three parameters in the following order:
Parameters	1. **DSN name** - This is either a DSN name or a

Function	Name/Description/Parameters
	connection string. 2. **User ID** - Login ID for the user. Set this value to a -1 if the first parameter is a connection string., 3. **Password** - Set this value to -1 if the first parameter is a connection string.
Function Name	**wnsFunc_AutoTerminate**
Description	Terminates WinSQL's process. This should be the last function in your script. Any script appearing after this function is ignored. If you skip this function, WinSQL will run the script and will display an empty window at the end.
Parameters	There are no parameters for this function.
Function Name	**wnsFunc_DataDiff**
Description	Runs the Data Diff wizard for the given template file. You must use the Data Diff Wizard in WinSQL and create a template before using this function. For more information, see the "Data Diff Wizard" topic in this document.
Parameters	Uses the name of the Data Diff template file name as the parameter.
Function Name	**wnsFunc_ExportData**
Description	Runs an export template. You must create an export template using WinSQL's GUI. For more information, see the "Using Export Templates" topic in this document.
Parameters	Expects one parameter, which is the name of a template file. For more information, see the "Using Export Templates" topic in this document.
Function Name	**wnsFunc_ImportText**
Description	Runs a text import template. You must create an import template using WInSQL's GUI.
Parameters	Expects one parameter, which is the name of a template file.

Function	Name/Description/Parameters
	For more information, see the "Importing Text Files" topic in this document.
Function Name	**wnsFunc_HtmlExport**
Description	Exports to HTML file(s) based on a template. You must create an HTML export template using WinSQL's GUI. For more information, see the "Using Export Templates" topic in this document.
Parameters	Expects one parameter, which is the name of a template file.
Function Name	**wnsFunc_RunScript**
Description	Runs any SQL script saved in a file. Multiple queries can be used, separated by a query separator. Uses two parameters in the following order.
Parameters	• **SQL file name** - Name of the file containing SQL queries. • **Output file name** - This file will contain results from a SELECT queries and/or record affected count for DML and DDL queries.
Function Name	**wnsFunc_SchemaDiff**
Description	Runs schema diff wizard for the given template. You must create a Schema Diff template using WinSQL's GUI. Fore more information, see the "Schema Diff Wizard" topic in this document.
Parameters	Expects one parameter, which is the name of a template file.
Function Name	**wnsFunc_SendStatusEmail**
Description	Sends a status e-mail. This function is overloaded, meaning it can take either 2 parameters or 5 parameters, described as follows:
Parameters	Two Parameter Version:

Function	Name/Description/Parameters
	1. **E-mail option** - This can either be 1, 2, or 3. If this value is 1, an e-mail is only sent when the status of every previous task is successful. If this value is 2, an e-mail is only sent when any of the previous tasks generates an error. When this value is 3, e-mail is sent regardless of the error status. 2. **Attachments** - Name of the file(s) to be attached. Multiple file names can be attached, separated by three pipe characters (|||). You can use $FILE_ATTACHMENTS string as the name of a file if the file name should be determined from the previous script. For example, when you run an Export template to a text file, the name of the output file is read at run time. Refer to the script samples following this table. **Note**: This function obtains the recipient and sender e-mail addresses from the WinSQL global configuration, which can be set by selecting **Options** from the **Edit** menu in WinSQL. Five Parameter Version: 1. **E-mail option** - same as above 2. **Attachment** - same as above 3. **Recipient** – E-mail address of the recipient. Multiple addresses can be added, separated by a comma. 4. **Sender name** - Name of the sender 5. **Sender e-mail** – E-mail address of the sender **Note**: In either parameter version, the body of the e-mail is generated by WinSQL and is based on the tasks that are run prior to calling this function.
Function Name	**wnsFunc_ExportDataBag**
Description	Exports data stored in a Databag to a table, This method takes 5 parameters in the following

Function	Name/Description/Parameters
	order.
Parameters	1. Databag File name 2. Target DSN 3. User ID for the connection 4. Password for the connection 5. Target table name
Function Name	**wnsFunc_SwitchDatabase**
Description	Switches the connection context to a different database. This function is only meaningful when your back-end database supports this concept. For example, Microsoft SQL Server and Sybase support switching to a different database, but Oracle does not.
Parameters	Takes one parameter containing the desired database name.

Sample Scripts

Example 1

This script run SQL scripts saved in a file named *BackupDB.sql*.

```
-- Establishes connection to a database.The DSN
name is 'testData'
-- User id is scott and tiger is the password
wnsFunc_AutoLogin("testData", "scott", "tiger")


-- Switch the database to a different value
wnsFunc_SwitchDatabase("Northwind")

-- Following function will run the scripts in
BackupDB.sql file
-- Output logs are saved to BackupDB.out file
wnsFunc_RunScript("C:\data\BackupDB.sql",
"C:\data\BackupDB.out")

-- Send status email if the script fails or
succeeds
wnsFunc_SendStatusEmail("3",
```

```
"C:\data\BackupDB.out")

-- Terminate WinSQL
wnsFunc_AutoTerminate()
```

Assuming the above script in saved in a file named *TestScript.wxf*, the following command will execute the script:

c:\Program Files\Synametrics Technologies\WinSQL\winsql "c:\data\TestScript.wxf"

Example 2

This script runs an export template that exports data from a table to a text file. The name of the generated text file is automatically read from the export template and is attached to the e-mail. This is done by specifying $FILE_ATTACHMENTS as a file name in *wnsFunc_SendStatusEmail* function.

Of important note in this example is the fact that you do not have to call the *wnsFunc_AutoLogin* function. This is because the connection parameters are specified in the export template file, which is 24Hour.wet in this case.

```
-- Execute export routine
wnsFunc_ExportData("C:\data\24Hour.wet")

-- Send status email
wnsFunc_SendStatusEmail("3",
"$FILE_ATTACHMENTS")

-- Finally terminate WinSQL
wnsFunc_AutoTerminate()
```

Troubleshooting Common Problems

The syntax used for creating and running queries is very sensitive. Often, a query will not run because a particular text object is either used incorrectly or not specified.

The following examples depict incorrectly written scripts, along with an explanation of why they are incorrect:

Example 1

```
WinSQL –xStudentTemplate.wet –dStudentDSN –uAdmin –
pAdmin
```

You cannot specify any other parameter along with –x. The template file stores all the necessary connection parameters.

Example 2

```
WinSQL –dStudentDSN –uAdmin –qc:\updateStudents.sql
–oc:\output.txt -a
```

Here, the –p option for password is not specified. If WinSQL Professional does not see a –p option, it will prompt the user for a password. If the password is blank or your database does not need a password, you must still provide this parameter with no value specified.

Index

Revision History

The following changes have been made to this document since the August 2009 release (v8.0):

Topic	Change
Scripting Wizard	New topic added to Chapter 2
Generate Code	New topic added to Chapter 2
Importing from Microsoft Excel	New topic added to Chapter 11
Generating Parameterized Executables	New topic added to Chapter 11